Expresiones de Referencia y Figuras Retóricas para la Distinción y Descripción de Entidades en Discursos Generados Automáticamente

Tesis doctoral

Presentada por
Raquel Hervás Ballesteros
para optar al grado de Doctor en Informática

Dirigida por el
Prof. Dr. D. Pablo Gervás Gómez-Navarro

Departamento de Ingeniería del Software e Inteligencia Artificial
Facultad de Informática
Universidad Complutense de Madrid
Madrid, Abril de 2009

Agradecimientos

En primer lugar quiero agradecer a mi tutor, el doctor Pablo Gervás, su apoyo y guía en el proceso de realización del trabajo aquí presentado. A pesar de lo perdida que me encontrado en distintas fases de la realización de esta tesis, él siempre ha confiado en mí y ha sabido guiarme con paciencia infinita para que todo llegara a buen puerto.

Gracias también a la inestimable ayuda que ha supuesto estar rodeada de gente tan estupenda como son los miembros del grupo de investigación NIL. Siempre habéis estado ahí para dar vuestro apoyo y consejos, sobre todo en este final que se ha hecho tan largo. Gracias a todos por preocuparos y por ayudarme a seguir adelante.

También agradecer al grupo GAIA dirigido por Pedro González Calero su acogida en los primeros años de andadura en la universidad. Y por supuesto al Departamento de Ingeniería del Software e Inteligencia Artificial, que gracias a gente como Carmen Fernández Chamizo y Luis Hernández Yáñez, entre otros, ha resultado ser el lugar perfecto para realizar un trabajo como éste.

Además me gustaría dar las gracias a los miembros del Centre for Informatics and Systems of the University of Coimbra (CISUC) por la buena acogida que me dieron durante todas mis estancias allí, y especialmente a Francisco Pereira y Amílcar Cardoso por su guía y colaboración en el trabajo realizado.

Muchas gracias también a Pablo Moreno, Virginia Francisco y Susana Bautista por su ayuda en la revisión de este documento y sus acertados consejos que sin duda han servido para mejorar el resultado final.

De fuera de nuestras fronteras quisiera agradecer la colaboración y consejos de Rafael Pérez y Pérez (Universidad Nacional Autónoma de México), Birte Lonneker (Universidad de Hamburgo, Alemania) y Charles Callaway (Universidad de Edinburgo, Reino Unido). Todos ellos han proporcionado una perspectiva más amplia a este trabajo.

De manera más personal tengo que agradecer su apoyo a todos aquellos que me han acompañado incondicionalmente durante estos años. Gracias a Dácil y Ana por ser unas estupendas amigas y haber sabido escuchar siempre que lo he necesitado. Sin el ánimo y empuje de Luis, Ana, Juanan, Guille y Pedro Pablo este camino se habría hecho mucho más duro. Aunque desde la distancia, Sonia, Verónica, Cecilia, Amalia, Paula y Cristina siempre me han apoyado y creído que podría con ello. Gracias a Virginia por haber sido apoyo en tantos trabajos, viajes y problemas que han ido surgiendo en el camino. Y a todos aquellos que me habéis acompañado de una manera u otra, gracias por haber estado ahí.

Esta tesis está dedicada a mi familia, y especialmente a mis padres. Ellos han dedicado su vida a darnos todo aquello que ellos no pudieron tener, y

gracias a esa dedicación he llegado hasta aquí. Espero que estén orgullosos de lo que he conseguido, porque se lo debo a ellos.

Y por último, a ti, Pablo, por compartir tu vida conmigo. Tu paciencia y apoyo en las largas noches de trabajo y los fines de semana sin descanso es algo que nunca podré agradecerte lo suficiente. Simplemente, gracias por estar ahí.

I can't wait for the rest of my life...

Resumen

El campo de la interacción hombre-máquina ha ido evolucionando rápidamente en los últimos años, convirtiéndose en un elemento fundamental de cualquier sistema informático. Si un sistema es capaz de comunicarse con un ser humano mediante interacciones que a éste le resulten naturales y amigables (voz, imágenes, etc.), el usuario será mucho más perceptivo hacia la información que está recibiendo y tendrá más confianza en la aplicación.

En este sentido, un campo fundamental dentro de la interacción hombre-máquina es el de la Generación de Lenguaje Natural (GLN), un subcampo de la Inteligencia Artificial y la Lingüística Computacional que se encarga del diseño e implementación de sistemas que producen textos comprensibles en lenguajes humanos. Entre todos los problemas que se deben resolver para que el proceso completo de generación resulte satisfactorio, se encuentra el de decidir cómo habrá que referirse a las entidades o elementos que aparecerán en el texto.

La tarea de Generación de Expresiones de Referencia se encarga de resolver este problema concreto. Las diferentes menciones al mismo elemento en un texto deben ser reemplazadas por la forma específica en que referirse a ellas o *referencias*. A la hora de planificar las referencias de un texto se deben tener en cuenta dos propósitos. En primer lugar, una referencia a un elemento del discurso debe permitir al lector u oyente distinguir a este elemento de cualquier otro presente en el contexto con el que se pudiera confundir. Por otro lado, en ocasiones las referencias contendrán información que más allá de la función de distinguir pretendan además describir las entidades a las que se refieren, presentando información relevante sobre las mismas previamente desconocida.

De estas dos funciones (distintiva y descriptiva), sólo la primera ha sido ampliamente estudiada en la literatura. Se pueden encontrar numerosos trabajos que se encargan de abarcar el problema de la generación de expresiones de referencia con función distintiva estudiando aspectos como la minimalidad de una expresión, la semejanza de la expresión a las utilizadas por los seres humanos, la no ambigüedad de la referencia generada, etc.

En cuanto a la descripción de entidades, aunque existen trabajos sobre generación de lenguaje natural basada en descripciones, se ha realizado poco trabajo desde el punto de vista del enriquecimiento del discurso con expresiones descriptivas que además realcen cierta información considerada importante, o sobre la relación de este proceso con la función distintiva de las referencias.

En este trabajo se aborda la generación de referencias de dos maneras diferentes. En primer lugar se proponen soluciones alternativas y mejoras a los algoritmos clásicos de generación de expresiones de referencia básica con función distintiva. Se aborda el problema desde tres frentes diferentes: cómo adecuar el nivel de abstracción al que se están nombrando las referencias

según el contexto de la situación, qué estrategia de búsqueda usar para la elección de los atributos que permitan distinguir a un concepto, y qué palabras o expresiones resultan más adecuadas para expresar una referencia en lenguaje natural. Para cada uno de estos frentes se presentan soluciones basadas en técnicas y recursos clásicos de la Inteligencia Artificial como son los algoritmos evolutivos, el razonamiento basado en casos, o las ontologías. Además se evaluarán las diferentes soluciones presentadas teniendo en cuenta las métricas clásicas en este campo.

En segundo lugar se explora el enriquecimiento de un discurso dado aportando información descriptiva utilizando figuras retóricas basadas en similitudes entre dominios como la comparación y la analogía. Para que sea posible utilizar este tipo de figuras en un sistema de generación de lenguaje natural se deben resolver problemas de arquitectura, fuentes de conocimiento, determinación de las analogías y comparaciones, etc. En este trabajo se estudian estos problemas y se propone un marco general para abordar la generación de este tipo de referencias.

Los resultados obtenidos para las soluciones propuestas en este trabajo dan lugar a una discusión sobre aspectos a mejorar en trabajo futuro y limitaciones de los algoritmos implementados. También se discute la relación de la generación de expresiones de referencia, desde el punto de vista de sus funciones distintiva y descriptiva, con el resto del proceso de generación de lenguaje natural.

Finalmente se presentan las conclusiones de esta investigación, así como líneas abiertas para trabajo futuro y campos de aplicación de las soluciones y resultados obtenidos.

Abstract

The field of human-computer interaction has evolved rapidly in recent years, becoming a key element of any computer system. If a system is capable of communicating with a human being through interactions that result natural and friendly for him or her (voice, images, etc.), the user will be much more perceptive to the transmitted information and will have more trust on the application and its results.

In this regard, a key area within the human-computer interaction field is Natural Language Generation (NLG), a subfield of Artificial Intelligence and Computational Linguistics. The field of Natural Language Generation is responsible for the design and implementation of systems that produce understandable texts in human languages from an initial non-linguistic representation of information. Within this field, one of the problems to be solved in order to generate satisfactory results is to decide how to refer to entities or elements that appear in the text.

The task of Referring Expression Generation deals with this specific problem. The different references to the same element in a text should be replaced by specific ways in which to refer to them or *references*. The process of referring expression generation should take into account two objectives. First, a reference to an element in the discourse should allow the reader or listener to distinguish it from any other element in the context with which it could be confused. In addition, sometimes the references may contain additional information intended to describe the corresponding entities beyond the function of distinguishing.

Of these two functions (distinctive and descriptive), only the former has been widely studied in the literature. Numerous works can be found dealing with the problem of distinguishing references, confronting issues such as minimality of an expression, similarity of a expression with the ones used by human beings, absence of ambiguity in the generated reference, etc.

However, although there is some work related to the generation of natural language descriptions, there are fewer works focused on enhancing a discourse with certain expressions that highlight descriptive information considered important, or on its relationship with the generation of distinguishing references.

This work addresses the complete problem of reference planning in two different ways. Firstly, several solutions and improvements to classical referring expression generation are proposed for references that attempt to distinguish the referents from other entities in context. The problem is addressed from three fronts: how to adjust the level of abstraction employed to name the reference according to the situation, which strategy to use for choosing the attributes that distinguish a concept, and what words or expressions are more appropriate to express a reference in natural language. For each of these points we present solutions based on classical techniques

and methodologies of Artificial Intelligence, such as evolutionary algorithms, case-based reasoning, or ontologies. The results obtained from the different solutions are also evaluated using classical metrics from this field.

Secondly, this work explores the enhancement of a given speech by providing descriptive information using figures of speech based on similarities between domains, such as comparison and analogy. In order to use such figures in a natural language generation system, it is necessary to address issues related to managing sources of knowledge, determining the appropriate figures, and defining an architecture to implement such systems. This work studies these issues and proposes a general framework to generate this kind of references.

The results obtained by the solutions proposed in this work lead to a discussion on the shortcomings of each approach, identifying aspects that could be improved in future work. The relationship between the generation of referring expressions (both distinctive and descriptive) and the complete process of natural language generation is also discussed.

Finally, the conclusions derived from these lines of research are presented, along with the identification of possible lines for future work and areas of application for the solutions and results presented in this work.

Índice general

Agradecimientos III
Resumen V
Abstract VII

1. Introducción **1**
1.1. Generación de Expresiones de Referencia 2
1.1.1. Referentes y Referencias 3
1.1.2. Rango de Elección 4
1.1.3. Objetivo Comunicativo de las Referencias 5
1.1.4. Criterios de Elección 7
1.2. Referencias Básicas usando Frases Nominales 8
1.2.1. Nivel de Abstracción para Referirse a un Concepto . . 8
1.2.2. Selección de Atributos 9
1.2.3. Lexicalización 9
1.3. Referencias basadas en Similitudes entre Conceptos 10
1.4. Evaluación de Expresiones de Referencia 11
1.5. Planteamiento del Trabajo y Objetivos 11
1.6. Estructura de la Memoria 13

2. Generación de Lenguaje Natural y Figuras Retóricas **15**
2.1. Generación de Lenguaje Natural 16
2.2. Generación de Expresiones de Referencia 19
2.2.1. Tipos de expresiones de referencia 20
2.2.2. Normas Conversacionales 21
2.2.3. Evitando Falsas Inferencias 24
2.2.4. Otras Consideraciones 25
2.2.5. Algoritmos para la Generación de Expresiones de Referencia 26
2.2.6. Sistemas con Generación de Información Descriptiva sobre Entidades 28
2.3. Evaluación para la Generación de Lenguaje Natural 29

2.3.1. Comparación con Resultados Humanos 30
2.3.2. Evaluación Orientada a Tarea 34
2.3.3. Evaluación Competitiva para Generación de Expresiones de Referencia 35
2.4. TAP, un *Pipeline* para Aplicaciones de GLN 43
2.4.1. Determinación de Contenido 44
2.4.2. Generación de Expresiones de Referencia 44
2.5. Recursos Léxico-Conceptuales para GLN 44
2.5.1. Uso de Ontologías para Procesamiento Lingüístico . . 46
2.5.2. WordNet 47
2.6. Figuras Retóricas basadas en Similitudes entre Dominios . . . 52
2.6.1. Metáfora, Analogía y Alineamiento Estructural 53
2.6.2. Comparaciones 56
Resumen y Conclusiones 57

3. Algoritmos Evolutivos, CBR y Agentes 59
3.1. Algoritmos Evolutivos 60
3.1.1. Estructura de los Algoritmos Evolutivos 60
3.1.2. Ventajas de los Algoritmos Evolutivos 61
3.1.3. Limitaciones de los Algoritmos Evolutivos 63
3.1.4. Aplicación de los Algoritmos Evolutivos a la Generación de Lenguaje Natural 65
3.2. Razonamiento Basado en Casos 65
3.2.1. El Ciclo CBR 66
3.2.2. Case Retrieval Nets 68
3.2.3. Razonamiento Basado en Casos en el Procesamiento de Lenguaje Natural 70
3.3. Agentes Inteligentes 71
3.3.1. Sistemas Multiagente 72
3.3.2. Open Agent Architecture, una Arquitectura para Sistemas Multiagente 73
Resumen y Conclusiones 74

4. Generación de Expresiones de Referencia Básicas 75
4.1. Elección de Nivel de Abstracción de las Referencias 76
4.1.1. Generación de Referencias Usando Ontologías para Determinar el Nivel de Abstracción 77
4.2. Selección de Atributos como Problema de Búsqueda 83
4.2.1. Selección de Atributos basada en Agrupamientos Relativos de los Mismos 84
4.2.2. Selección de Atributos dependiendo del Valor más Frecuente 88
4.2.3. Búsqueda de los Atributos usando Algoritmos Evolutivos 92

4.3. Sinonimia y Preferencias Personales para Lexicalización . . . 95
4.3.1. Elección Léxica como Parte de la Lexicalización Usando WordNet . . . 96
4.3.2. Lexicalización Escogiendo la Opción Más Frecuente . 100
4.3.3. Razonamiento Basado en Casos para Reproducir el Estilo de las Referencias . . . 103
4.4. Evaluación en las Tareas Competitivas . . . 111
4.4.1. Resultados de Evaluación en la Primera Tarea Competitiva para la Generación de Expresiones de Referencia 111
4.4.2. Resultados de Evaluación en la Segunda Tarea Competitiva para la Generación de Expresiones de Referencia . . . 115
4.4.3. Resultados de Evaluación en la Tercera Tarea Competitiva para la Generación de Expresiones de Referencia 120
Resumen y Conclusiones . . . 123

5. Figuras Retóricas para Describir Entidades 125
5.1. Análisis Teórico de Analogías y Comparaciones en Texto . . . 126
5.1.1. Propuesta para la Introducción de Analogías y Comparaciones en Texto . . . 126
5.1.2. Comparaciones . . . 129
5.1.3. Analogías . . . 131
5.2. Sistema Multiagente para Analogías y Comparaciones . . . 135
5.2.1. Conocimiento General . . . 135
5.2.2. Arquitectura Multiagente para la Creación e Inserción de Figuras Retóricas en Texto . . . 136
5.2.3. Comparaciones . . . 138
5.2.4. Analogías . . . 144
Resumen y Conclusiones . . . 151

6. Discusión y Propuestas 153
6.1. Generación y Evaluación de Expresiones de Referencia Básicas 153
6.1.1. Elección del Nivel de Abstracción de las Referencias . 154
6.1.2. Soluciones Evolutivas para la Búsqueda de Conjuntos de Atributos . . . 155
6.1.3. Elección léxica con WordNet . . . 156
6.1.4. Lexicalización de Referencias basada en Razonamiento Basado en Casos . . . 157
6.1.5. Evaluación de las Referencias Generadas . . . 157
6.2. Generación y Evaluación de Referencias Usando Figuras Retóricas . . . 158
6.2.1. Flujo de Control para la Generación de Figuras Retóricas . . . 158

6.2.2. Conocimiento General Necesario para la Generación de Figuras Retóricas 158
6.2.3. Generación de Comparaciones 159
6.2.4. Generación de Analogías 159
6.2.5. Evaluación de las Referencias Generadas 161
6.3. Interacción entre las Comparaciones y Analogías y la Generación Automática de Texto 161
6.3.1. Relación de las Comparaciones con el Resto del Proceso de Generación 162
6.3.2. Relación de las Analogías con el Resto del Proceso de Generación 162
6.3.3. Propuesta de Implementación sobre un Generador de Lenguaje Natural 163
6.4. Referencias Distintivas y Descriptivas y su Relación con el Resto del Proceso GLN 165
6.4.1. Interacciones en el Flujo de Control del Proceso de Generación 165
6.4.2. Propuesta de Implementación en una Arquitectura GLN 167
6.4.3. Discusión 169
Resumen y Conclusiones 170

7. Conclusiones y Trabajo Futuro 171
7.1. Distinción y Descripción de Entidades 171
7.2. Valoración de la Generación de Referencias con Función Distintiva 172
7.3. Valoración de la Generación de Referencias usando Figuras Retóricas basadas en Similitudes entre Dominios 175
7.4. Evaluación de las Referencias Generadas 176
7.5. Interacciones con el Proceso Completo de Generación 177
7.6. Trabajo futuro 178

Bibliografía 180

A. Publicaciones I
A.1. Generación de Lenguaje Natural I
A.2. Generación de Expresiones de Referencia II
A.3. Figuras Retóricas III
A.4. Tareas Competitivas para la Generación de Referencias . . . III

B. Características del Corpus TUNA V
B.1. Descripción del Formato de las Instancias del Corpus V
B.1.1. El Nodo `TRIAL` V
B.1.2. El Nodo `DOMAIN` VI
B.1.3. El Nodo `WORD-STRING` VII

B.1.4. El Nodo `ATTRIBUTE-SET` vii
B.1.5. El Nodo `ANNOTATED-WORD-STRING` viii
B.1.6. `ATTRIBUTE`s que no aparecen en la representación `DOMAIN` ix

C. Conocimiento General de los Dominios de Sapper xi

Índice de Tablas

2.1. Ejemplo de dominio de entrada 37
2.2. El concepto de matriz léxica en WordNet 49

4.1. Ejemplos de instancias de vinos 82
4.2. Resultados obtenidos usando el algoritmo para la mínima referencia sobre los datos de entrenamiento 87
4.3. Mejores resultados obtenidos sobre los datos de entrenamiento 88
4.4. Probabilidades de aparición para los valores de los atributos . 90
4.5. Resultados de evaluación para los datos de entrenamiento . . 92
4.6. Resultados de evaluación del algoritmo evolutivo para los datos de entrenamiento 95
4.7. Resultados de evaluación para los datos de entrenamiento . . 103
4.8. Resultados de evaluación del módulo basado en casos sobre los datos de entrenamiento 110
4.9. Resultados de evaluación del módulo basado en casos sobre los datos de desarrollo 110
4.10. Resultados obtenidos sobre los datos de desarrollo del algoritmo basado en agrupamientos de atributos 112
4.11. Resultados sobre los datos de desarrollo usando el algoritmo adaptado para generar referencias mínimas 112
4.12. Proporción de referencias mínimas por sistema 113
4.13. Resultados del coeficiente Dice de los sistemas participantes sobre los datos de test. Los sistemas que no comparten una letra son significativamente diferentes 113
4.14. Tiempos medios de identificación para los sistemas evaluados. Los sistemas que no comparten una letra son significativamente diferentes 114
4.15. Resultados de evaluación para los datos de desarrollo del algoritmo NIL-UCM-MFVF 115

4.16. Resultados de evaluación de los sistemas participantes sobre los datos de test 116
4.17. Comparación de los sistemas participantes para los coeficientes Dice y MASI 116
4.18. Resultados de evaluación para los datos de desarrollo del algoritmo NIL-UCM-BSC 117
4.19. Resultados de evaluación de los sistemas participantes sobre los datos de test 117
4.20. Resultados de evaluación para los datos de desarrollo del algoritmo NIL-UCM-FVBS 118
4.21. Resultados de evaluación intrínseca de los sistemas participantes sobre los datos de test 119
4.22. Resultados de evaluación extrínseca de los sistemas participantes sobre los datos de test 119
4.23. Comparación de los sistemas participantes para las medidas de evaluación extrínsecas 119
4.24. Resultados sobre los datos de desarrollo para los tres sistemas 121
4.25. Resultados de evaluación intrínseca de los sistemas participantes sobre los datos de test 122
4.26. Resultados de evaluación extrínseca de los sistemas participantes sobre los datos de test 122

5.1. Significado considerado dependiendo del valor de la intensidad 136
5.2. Ejemplos de relaciones para algunos conceptos de los dos dominios 149
5.3. Correspondencia resultante entre La Guerra de las Galaxias y la saga del Rey Arturo 149

Índice de Figuras

2.1. Esquema del *pipeline* propuesto por Reiter 19
2.2. Algoritmo Incremental de Reiter y Dale 28
2.3. Ejemplos de las situaciones planteadas en el experimento del corpus GRE3D3 32
2.4. Ejemplo de situación presentada a los evaluadores en el proyecto TUNA 33
2.5. Parte del grafo alrededor del concepto *butcher* 56

3.1. Esquema básico de un algoritmo evolutivo 62
3.2. Ciclo del Razonamiento Basado en Casos 67
3.3. Ejemplo de una Case Retrieval Net en el dominio de agencias de viajes 69
3.4. Ejemplo de un sistema multi-agente implementado usando OAA 74

4.1. Taxonomía que corresponde al ejemplo 76
4.2. Extracto de la ontología sobre vinos 78
4.3. Parte de la información disponible para el vino `Corton Montrachet White Burgundy` 79
4.4. Algoritmo para la elección del nivel de abstracción 81
4.5. Esquema de funcionamiento del algoritmo basado en el valor más frecuente 91
4.6. Ejemplo de caso del corpus 105
4.7. Similitudes entre las distintas orientaciones 105

5.1. La sociedad de agentes vista en el monitor de OAA 137

6.1. Parte de la jerarquía sobre vinos de WordNet 155

B.1. Formato de las instancias del corpus VI
B.2. Atributos y valores usados en el corpus TUNA VII

Capítulo 1

Introducción

"Mama always said life was like a box of chocolates.
You never know what you're gonna get."
— Forrest Gump (1994)

En las últimas décadas, con la llegada de los ordenadores personales al gran público, la interacción hombre-máquina ha ido evolucionando hasta convertirse en un elemento fundamental de cualquier sistema informático. A medida que la complejidad de los sistemas aumenta, la importancia de facilitar la interacción con el usuario se vuelve más importante. Por la propia naturaleza del ser humano, si un sistema está preparado para comunicar la información que debe presentar al usuario en lenguaje natural, éste será más receptivo a la misma y tendrá más confianza en la aplicación.

De esta tarea se encarga la Generación de Lenguaje Natural (GLN), un subcampo de la Inteligencia Artificial y la Lingüística Computacional. Esta ciencia cubre del diseño y la construcción de sistemas que producen textos comprensibles en lenguajes humanos partiendo de información no lingüística y utilizando conocimiento sobre el lenguaje y el dominio de la aplicación.

Donde un sistema de procesamiento de lenguaje natural tiene que tratar con la ambigüedad existente entre las diferentes interpretaciones de una entrada dada, la generación de lenguaje natural tiene que decidir entre las distintas posibilidades de decir la misma cosa. El gran reto de la GLN es, por tanto, uno de elección y no de resolución de la ambigüedad. Los sistemas de generación existentes tienden a centrarse en la generación de textos técnicos, donde es más fácil identificar la manera "correcta" de decir algo. Sin embargo, en los últimos años, la GLN está empezando a considerar otros dominios de aplicación donde el rango de elección es mayor. Aplicaciones como la generación de poesía (Manurung, 2003) o cuentos fantásticos (Callaway y Lester, 2001b) presentan mayores posibilidades de elección durante el proceso de generación que los diagnósticos médicos (Portet et al., 2007) o los partes meteorológicos (Goldberg et al., 1994).

El proceso de la generación de texto (Reiter y Dale, 2000) está dividido en varias etapas, durante las cuales una ENTRADA CONCEPTUAL es refinada progresivamente según se le añade la información necesaria para dar lugar al texto final. Esta entrada conceptual será completamente dependiente del dominio de aplicación. Durante las primeras etapas se escoge la información que aparecerá en el texto final (DETERMINACIÓN DE CONTENIDO), se la organiza según cierto orden y estructura en MENSAJES, que son en realidad representaciones semánticas de lo que posteriormente serán las frases en lenguaje natural (PLANIFICACIÓN DEL DISCURSO), y se selecciona la manera de describir o mencionar cada elemento que aparecerá en el texto (GENERACIÓN DE EXPRESIONES DE REFERENCIA). El resultado es un PLAN DEL DISCURSO donde el contenido, la estructura, y el nivel de detalle de cada concepto que será reflejado en el texto final está fijado. La fase de LEXICALIZACIÓN que sigue se encarga de escoger las palabras y frases concretas que serán utilizadas para expresar los conceptos del dominio y las relaciones que aparecen en los mensajes. Una fase final de REALIZACIÓN SUPERFICIAL une finalmente todas las piezas relevantes en un texto lingüística y tipográficamente correcto.

Este trabajo de tesis se engloba dentro de este campo de la Generación de Lenguaje Natural, y más concretamente en la tarea de Generación de Expresiones de Referencia, centrándose en algunos de los problemas específicos que se pueden encontrar dentro de ella. En las secciones que siguen se expone la motivación general de esta tesis como respuesta a las dificultades que hay que tener en cuenta a la hora de escoger las referencias a utilizar en un texto. Se presentan también los objetivos concretos que se pretenden cubrir con el trabajo aquí presentado, y finalmente se resume la estructura general del presente documento.

1.1. Generación de Expresiones de Referencia

Los datos de entrada con los que trabaja la Generación de Expresiones de Referencia son planes de discurso del texto final donde las etapas de Determinación de Contenido y Planificación del Discurso ya se han encargado de decidir qué información se va a transmitir en el texto y cómo se va a estructurar esa información. La estructura de este esquema estará formada por mensajes que serán transformados posteriormente en oraciones sintácticas correctas para el idioma de trabajo.

Los identificadores simbólicos con los que se representan las entidades que aparecen en el texto en cada uno de los mensajes tendrán que ser sustituidos por distintos tipos de referencias que sean suficientes para que el oyente o lector identifique dichas entidades unívocamente. De esto se encarga la Generación de Expresiones de Referencia, que tiene que decidir si para referirse a una entidad concreta se usará un pronombre (*he*, *she*, *they*, etc.),

un nombre propio (*John*, *The Caledonian Express*, etc.), una frase nominal simple (*the train*), acompañada de modificadores o relaciones (*the Aberdeen train* o *the train on platform 12*). En cada caso, es necesario trabajar con información semántica de las entidades a las que se desea referirse.

Veremos en esta sección diversas cuestiones relacionadas con la Generación de Expresiones de Referencia, como las elecciones que hay que realizar en el proceso y los objetivos que se pueden cubrir con las referencias generadas, entre otros.

1.1.1. Referentes y Referencias

Jurafsky y Martin (2000) definen la siguiente terminología en el campo de las expresiones de referencia. Nos referiremos a las expresiones lingüísticas que denotan un elemento o un individuo en un texto como EXPRESIONES DE REFERENCIA, REFERENCIAS O MENCIONES, y al elemento que es referido como REFERENTE O ENTIDAD. El resto de entidades en el contexto con las que se podría confundir el referente buscado se denominan DISTRACTORES y se dice que forman el CONJUNTO DE CONTRASTE o *contrast set*. Dos expresiones de referencia que se usan para referirse a la misma entidad o referente se dice que CORREFIEREN y se denominan CORREFERENCIAS. Existe también un término para una expresión de referencia que permite el uso de otra, en la forma que el uso de *John* permite referirse a John posteriormente como *he*. En este caso se dice que *John* es el ANTECEDENTE de *he*. La referencia a una entidad que ha sido introducida previamente en el discurso se denomina ANÁFORA, y la expresión de referencia usada en este caso se denomina ANAFÓRICA.

A la hora de generar referencias automáticamente, es importante poder distinguir entre la primera aparición de una entidad en el discurso y las posteriores, ya que la forma en que serán realizadas en el texto final será muy diferente. La función de una REFERENCIA INICIAL es introducir a cierta entidad o referente en el discurso. En general, las entidades deben introducirse en el discurso haciendo referencia a aquellas de sus propiedades que serán útiles más adelante, preparando al lector para lo que está por venir. Sin embargo, parece que esta tarea es dependiente del dominio, por lo que desde un punto de vista computacional la mejor estrategia puede ser examinar textos del dominio y ver qué tipo de referencias iniciales se usan y se consideran útiles. Las REFERENCIAS SUBSIGUIENTES, al contrario, se utilizan para referirse a entidades que han sido previamente mencionadas en el discurso. Estas referencias son anafóricas, como ya hemos visto previamente, y para que puedan ser correctamente identificadas por el lector tiene que ser posible que éste sea capaz de establecer cuál es el antecedente de las mismas.

1.1.2. Rango de Elección

La opción más común a la hora de generar una expresión de referencia para una entidad que aparece en el discurso es el uso de una frase nominal, ya sea su nombre propio o su tipo acompañado de los modificadores que puedan ser necesarios. En el caso de que la entidad a la que referirse tenga nombre propio, en la mayoría de las ocasiones su uso garantiza que la referencia generada permita identificar unívocamente el elemento al que se está refiriendo (siempre que quien va a oír o leer la referencia conozca el nombre propio de la entidad). En caso contrario, el uso del tipo para referirse al elemento es también una buena opción, siempre que se pueda acompañar a éste de modificadores que ayuden a la identificación si es necesario. Si por ejemplo aparecen en el contexto más entidades del mismo tipo que la que queremos referenciar, será necesario añadir a la referencia propiedades o cualidades que la distingan de las demás en el contexto. Si no se pudieran encontrar estas propiedades, todavía sería posible usar relaciones (*the table next to the door*), diversos adjetivos demostrativos, como por ejemplo *this* y *that* para indicar proximidad o lejanía, o *one* y *the other one* para mencionar los referentes secuencialmente. Por ejemplo, la referencia *the table* es suficiente para identificar una mesa si ésta es la única que existen en el contexto. Si hubiera otra sería necesario utilizar modificadores para distinguirlas (podría ser *the red table* frente a *the blue table*). Si no existieran modificadores que permitieran esta distinción, se podría recurrir a los adjetivos demostrativos antes mencionados.

Sin embargo, cuando los seres humanos hablan o escriben, no se limitan a escoger una expresión de referencia adecuada para una entidad y a utilizarla cada vez que se refieren a ella, sino que adecúan el uso de referencias al contexto del discurso y de la situación en cada momento para no resultar repetitivos. Uno de los tipos posibles de expresiones de referencia que se usan para evitar la repetición en el discurso son los pronombres (Callaway y Lester, 2001c). Las referencias pronominales no pueden ser usadas en cualquier lugar del discurso, sino que deben cumplir ciertas condiciones para que sean inteligibles. La heurística más generalizada es que se puede usar un pronombre cuando el elemento al que nos vamos a referir ya ha sido mencionado y además permanece en el foco de atención. Otras cuestiones deben ser tenidas en cuenta también, como por ejemplo la estructura del texto, ya que existen trabajos que demuestran que en los cambios de tiempo verbal o de punto de vista de los textos se "reinicia" el uso de los pronombres (McCoy y Strube, 1999).

Finalmente, existen también otras opciones menos utilizadas pero que deben ser tenidas en cuenta a la hora de generar expresiones de referencia. Dependiendo de la situación y el contexto, en ocasiones será necesario utilizar otros tipos de referencias como pueden ser los plurales o los superla-

tivos. Cada una de estas nuevas opciones lleva asociada una cierta dificultad y problemática a la hora de tomar decisiones sobre su uso.

1.1.3. Objetivo Comunicativo de las Referencias

El uso apropiado de expresiones de referencia que puedan competir con las que aparecen en los textos generados por humanos presenta cierta dificultad. Una expresión de referencia debe comunicar suficiente información para identificar unívocamente un referente dentro del contexto del discurso actual, pero siempre evitando modificadores redundantes o innecesarios.

La mayor parte de las expresiones de referencia que aparecen en cualquier tipo de discurso se utilizan básicamente para referirse a entidades del contexto que deben ser unívocamente distinguidas del resto que las rodean. Este tipo de referencias cubren una FUNCIÓN DISTINTIVA. Un ejemplo de este tipo de referencias sería el (1.1), donde el hablante indica al oyente apoyándose en una relación que coloque la caja sobre la mesa que está junto a la puerta y no sobre ninguna otra.

(1.1) *Put the box on* <u>*the table next to the door*</u>

Sin embargo, en ocasiones una referencia se utiliza para aportar información extra que es considerada como importante por el hablante o escritor, pero que no es necesaria para identificar al referente dentro del contexto. Estas referencias están cumpliendo una FUNCIÓN DESCRIPTIVA O INFORMATIVA. En el ejemplo (1.2), extraído del cuento "The Princess and the Pea", sólo hay un personaje que sea rey, pero se aporta la información extra de que es viejo al lector.

(1.2) *Suddenly a knocking was heard at the palace gate, and* <u>*the old king*</u> *went to open it*

En general, las función descriptiva de las referencias se utiliza para enfatizar ciertas propiedades de una entidad que pueden ser importantes más adelante en el discurso, o para informar sobre características que se salen de lo común. En el ejemplo (1.3) se especifica que los zapatitos de Cenicienta son de cristal aunque no se mencionan más zapatos en todo el cuento. En este caso, se pretende recalcar el hecho de que están hechos de cristal, asumiendo que el lector sabe que ése no es un material común para hacer unos zapatos.

(1.3) *Her godmother gave her a pair of* <u>*glass slippers*</u>*, the prettiest in the whole world*

La DESCRIPCIÓN DE ENTIDADES es una tarea indispensable en la generación automática de texto para cualquier tipo de discurso. Desde los sistemas

de diálogo hasta los textos narrativos, todo texto generado necesita referirse a elementos del mundo con distintos fines, y la base de estas referencias serán las descripciones que se hayan hecho de dichos elementos anteriormente. En ocasiones, las entidades referidas necesitarán ser distinguidas de otras del mismo tipo o con características comunes, y para ello sus características deberán ser expuestas de manera que puedan ser útiles para tal fin. Otras veces la información con que son descritas ciertas entidades será importante para una comprensión completa del discurso.

La importancia de estas descripciones es mayor cuando el tipo de discurso hace que la única información disponible para el lector u oyente sea la dada en el propio discurso. En el contexto de un diálogo entre dos personas, el oyente dispone tanto de la información del entorno que les rodea como de las ideas intercambiadas entre ambos. Por ello, una referencia como *la mesa* podría ser entendida sin necesidad de mayores explicaciones. Imaginemos una habitación en la que sólo hay una mesa entre otros objetos, y que el hablante ordena al oyente la frase (1.4). El oyente entenderá que si el hablante ha especificado que la mesa es de madera será porque esa información es relevante de alguna manera, como por ejemplo porque luego le va a hablar de qué tipo de madera es o que la fabricó él mismo.

(1.4) *Look at* <u>*the wooden table*</u>

Sin embargo, en un discurso donde la comunicación es exclusivamente lingüística y no hay ningún contexto visual, la única información de la que dispondrá el lector u oyente es la que se ha mostrado en el texto, y por ello las descripciones de las entidades del discurso serán cruciales para que el texto transmita la información deseada.

Cheng (1998) divide los constituyentes de una expresión de referencia en dos partes basadas en sus objetivos comunicativos y en la forma en que deben ser generadas. Son la PARTE DISTINTIVA (*referring part*), que tiene como objetivo distinguir a un objeto entre otros con los que se puede confundir, y la PARTE DESCRIPTIVA (*non-referring part*), que pretende aportar información adicional sobre el objeto. De esta manera, las referencias no tienen por qué ser sólo de un tipo u otro, sino que la misma referencia puede tener como objetivo tanto distinguir como aportar información extra.

Cheng argumenta que, aunque la principal función de las expresiones de referencia es la distinción de entidades, es posible que en una misma referencia que tiene información distintiva se desee añadir además información extra para cumplir con una función descriptiva. En estos casos deben cumplirse dos principios para que la referencia generada sea correcta y cumpla ambas funciones:

1. La parte descriptiva de la referencia no debe confundir al oyente o lector sobre el referente al que se refiere la parte distintiva. Es decir, si

la referencia puede identificar unívocamente la entidad a referenciar, el lector u oyente no debe ser confundido al añadir la parte descriptiva a la referencia.

2. La parte descriptiva de la referencia no debe disminuir la legibilidad del texto. Por ejemplo, la expresión de referencia resultante no debe resultar demasiado complicada de leer al añadir la parte descriptiva a la misma.

A pesar de que lo más común es encontrar referencias que tienen tanto parte distintiva como descriptiva, en la mayor parte de la literatura se estudian estas dos funciones de las expresiones de referencia por separado por cuestiones de simplicidad.

Desde el punto de vista de cómo y dónde tomar la decisión de qué información incluir en las referencias dependiendo del objetivo a cubrir, la introducción de información con intención descriptiva ocurre en función de criterios específicos del dominio, pero la introducción de atributos con intención distintiva ocurre en función del contexto del discurso.

1.1.4. Criterios de Elección

Como hemos visto, el número de combinaciones posibles que se pueden usar a la hora de generar una referencia es extremadamente alto, y para seleccionar entre ellas se tienen en cuenta diversas cuestiones. Por ejemplo, el proceso de seleccionar qué información usar al referirse a un elemento debe tener en cuenta principios básicos de economía del discurso como sería no usar más información de la que es necesaria en cada caso para identificar correctamente al elemento referenciado. Para ello se debe dar la suficiente información para que el elemento no sea confundido con los elementos que pueda tener alrededor, mientras que al mismo tiempo no se debe incluir información innecesaria que pueda dar lugar a falsos razonamientos o implicaciones. Para tomar estas decisiones será muy importante tener en cuenta las funciones que se pretenden cubrir con la referencia: descriptiva, distintiva, o ambas al mismo tiempo.

Aparte de estas consideraciones sobre la función de las referencias se debe tener en cuenta una razón más por la que los seres humanos usan un tipo u otro de referencias, o bien unos u otros modificadores. Cada persona tiene un ESTILO diferente de expresión dependiendo de su posición social, personalidad, situación, y otros muchos factores externos al lenguaje. Sin embargo, este estilo puede hacer que unas personas sean más dadas al uso de pronombres, por ejemplo. O incluso que prefieran unos modificadores sobre otros dependiendo de su profesión. Parece por tanto lógico pensar que un pintor se concentrará más en los colores de las entidades a las que se está refiriendo mientras que un escultor lo hará en las formas.

1.2. Referencias Básicas usando Frases Nominales

Las frases nominales (generalmente en la forma nombre + adjetivos) son una de las formas más comunes en las que puede expresarse una referencia a una entidad, y al mismo tiempo es también una de las más complejas debido al rango de posibilidades entre las que se puede escoger. Generalmente en estas referencias el nombre corresponderá al tipo o clase al que pertenece el referente, y los adjetivos a los modificadores que corresponden a sus propiedades. Estos modificadores podrían ser tanto atributos como relaciones, aunque en este trabajo sólo se estudiará al generación de expresiones de referencia utilizando los primeros. Donde un pronombre sólo debe mantener concordancia en número y género, una referencia en forma de frase nominal con función distintiva tiene que enfrentarse a otros problemas fundamentales.

En ocasiones puede ser que el tipo establecido para generar la referencia no sea el más adecuado dado el contexto en el que se está desarrollando el discurso. En los casos en que se disponga de información sobre las entidades del discurso en forma de taxonomía o jerarquía de conceptos, es posible buscar el nivel al que está discurriendo el discurso y por tanto el nivel al que se debe mencionar una referencia. Será por tanto una búsqueda del NIVEL DE ABSTRACCIÓN PARA REFERIRSE A UN CONCEPTO.

Una vez decidido el tipo del elemento al que nos vamos a referir, será necesario decidir qué conjunto de modificadores aplicables a la entidad la distinguen unívocamente de todas las demás con las que puede confundirse. A este proceso se le denomina SELECCIÓN DE ATRIBUTOS.

Una vez que se ha decidido la información que se va a expresar en la referencia (tipo + atributos), es necesario escoger de qué manera se va expresar esta información en texto. Para ello habrá que escoger qué palabras o expresiones son más adecuadas en cada momento tanto para los atributos como para los tipos. De esto se encargará la LEXICALIZACIÓN en la etapa siguiente del proceso de generación, aunque esta tarea estará muy ligada con la generación de referencias.

1.2.1. Nivel de Abstracción para Referirse a un Concepto

En un contexto en el que los posibles referentes están organizados en términos de una taxonomía y pueden ser además diferenciados usando sus atributos, se puede utilizar este conocimiento adicional para encontrar el nivel de abstracción más apropiado para generar las referencias.

Para ello habría que identificar a qué nivel de la taxonomía se debe escoger la referencia de manera que señale unívocamente al referente buscado, quedando el resto de distractores en ramas diferentes de la jerarquía. Imaginemos por ejemplo que se dispone de una taxonomía de perros y gatos, con información específica sobre sus razas. Si nos encontramos en una situación

en la que hay que perros de varias razas y nos tenemos que referir a uno de ellos, lo más adecuado sería utilizar el nombre de su raza (por ejemplo *doberman*) para referirnos a él. Sin embargo, si sólo hay un doberman y hay que distinguirlo de un conjunto de gatos, no tiene sentido ser específico (ya que puede dar lugar a confusiones) y es mejor utilizar simplemente la referencia *el perro*. De esta manera estamos escogiendo el nivel de abstracción en el que se va a desarrollar la referencia.

Una vez escogido el nivel de abstracción apropiado, si es necesario se podría recurrir a mencionar atributos adicionales de la entidad a la que nos estamos refiriendo de manera que ésta sea unívocamente distinguida de otros distractores que comparten con ella la misma rama de la taxonomía. A este problema se le denomina selección de atributos y se expone a continuación.

1.2.2. Selección de Atributos

La selección de atributos pretende distinguir una entidad de las que le rodean utilizando para ello una descripción consistente en su tipo y un conjunto de sus características que permitan distinguirla unívocamente de las demás. Imaginemos una situación con tres objetos en una habitación (dos sillas y un sofá) que tienen las siguientes características:

```
object1 = {<type,sofa>,<colour,red>,<size,big>}
object2 = {<type,chair>,<colour,red>,<size,big>}
object3 = {<type,chair>,<colour,red>,<size,small>}
```

Una referencia válida para el objeto 1 podría ser simplemente su tipo, pero no sería suficiente ni para el objeto 2 ni el 3. Para estos dos es necesario realizar una selección de sus atributos que los distinga del resto de objetos en el contexto. En este ejemplo, el tamaño y el tipo serían suficientes (*the big chai*r o *the small chair*), mientras que el color y el tipo no (*the red chair* puede referirse a cualquiera de las dos sillas). Asimismo, una referencia sin tipo como *the small one* sería válida para el objeto 3, pero no para los objetos 1 y 2 (*the big one*).

El problema de la selección de atributos es por tanto un problema complejo, que puede ser estudiado desde distintos puntos de vista y teniendo en cuenta diferentes conjuntos de las características relativas a las expresiones de referencia.

1.2.3. Lexicalización

La Lexicalización es generalmente entendida como la tarea del proceso de generación de texto que decide qué palabras se usarán para presentar un mensaje en el texto. En el caso de la generación de expresiones de referencia se encargará de escoger qué palabra utilizar en el texto final entre todas las que se pueden utilizar para expresar un determinado sustantivo o adjetivo.

Por ejemplo, a la hora de referirnos a un hombre, existen muchas posibilidades (*man*, *guy*, *gentleman*, etc.) que dependerán de cuestiones de contexto y estilo.

La tarea de proporcionar a un algoritmo de generación de expresiones de referencia con la capacidad de realizar elección léxica involucra dos retos diferentes. Para empezar, debemos disponer de suficiente conocimiento para permitir más de una alternativa para la lexicalización de los elementos que se están manejando. Por otro lado, es necesario plantear algún tipo de heurística para decidir qué alternativa léxica a escoger es adecuada para cada aparición de un concepto dentro de un texto dado. Estas heurísticas deben ser capaces de reflejar aspectos como, por ejemplo, las restricciones terminológicas y las prácticas habituales según el estilo o la situación. Por tanto, estas heurísticas serán más adecuadas si están adaptadas a cada dominio particular de aplicación.

1.3. Referencias basadas en Similitudes entre Conceptos

Algo que caracteriza el lenguaje generado por los seres humanos es nuestra capacidad de manejar cantidades enormes de conocimiento, cada una de una fuente distinta, y semánticamente distantes unas de otras. El uso de ciertas figuras retóricas basadas en similitudes entre dominios que complementen la capacidad descriptiva de las expresiones de referencia proporcionaría una mayor naturalidad a los textos generados automáticamente. Algunas de ellas podrían ser las comparaciones, donde se expresa de una manera explícita la semejanza entre dos ideas (*suave como la seda*) o las analogías, que expresan una relación de semejanza entre entidades de distintos dominios (*la comprensión del lenguaje natural es el Santo Grial de la Inteligencia Artificial*), o metáforas, que usan comparaciones de forma implícita (*el aro dorado emerge del mar*, refiriéndose al sol).

El poder establecer este tipo de similitudes entre dominios permitiría enriquecer las expresiones de referencia a generar para que cumplieran funciones descriptivas adicionales. Las comparaciones se podrían utilizar para recalcar propiedades de una entidad si se encuentra una comparación con algún otro elemento que ostenta esa propiedad y en gran medida (*la princesa era tan bonita como una flor*), centrando la atención del lector u oyente en lo importante que resulta que la princesa sea bonita. Las analogías se podrían usar a su vez para introducir entidades desconocidas si se comparan con otras conocidas (*Un soldado de asalto en La Guerra de las Galaxias es como un caballero medieval*), o incluso para recalcar también propiedades de ciertos elementos del discurso (*El carnicero era limpio y preciso, un cirujano entre carniceros*).

1.4. Evaluación de las Expresiones de Referencia Generadas

Como en muchos otros campos dentro de la Inteligencia Artificial, resulta importante saber hasta qué punto las soluciones de generación implementadas resultan adecuadas para resolver los problemas que hayan sido planteados en cada caso. Para poder evaluar los sistemas de esta manera, es necesario definir qué se pretende conseguir y cómo se puede medir.

A la hora de evaluar los resultados obtenidos por un sistema de generación de lenguaje natural, se suele considerar que el standard de referencia a perseguir es la manera en que los seres humanos generarían lenguaje en la misma situación (Gatt, 2007). Para ello es necesario recurrir a corpus específicos de la tarea a evaluar que contengan la manera en que los seres humanos han reaccionado ante diferentes situaciones. Sin embargo, no existe garantía de que estos corpus sean perfectos o tan siquiera suficientemente adecuados.

Existen otras formas de evaluación que pueden ser también útiles en GLN. Por ejemplo, en una *evaluación orientada a tarea*, se comprueba si la salida de un sistema de generación cumple su objetivo de cara al usuario para probar la viabilidad de la tecnología y la efectividad del algoritmo.

En el caso concreto de la generación de expresiones de referencia, una evaluación de este tipo podría consistir en medir hasta que punto las expresiones de referencia generadas automáticamente cumplen la función distintiva de permitir a oyentes o lectores el identificar a la entidad del discurso referida, o la descriptiva de si se ha proporcionado suficiente información para comprender más adelante el discurso.

De esta manera, los métodos de evaluación se pueden dividir en intrínsecos y extrínsecos. Mientras que los MÉTODOS INTRÍNSECOS se ocupan de evaluar los resultados de los algoritmos de manera aislada, bien comparándolos con un corpus o usando métricas de evaluación absolutas, los MÉTODOS EXTRÍNSECOS miden el impacto de un algoritmo sobre algo externo a él o de lo que forma parte.

1.5. Planteamiento del Trabajo y Objetivos

Como se ha esbozado en el resto de la introducción, la generación de expresiones de referencias es un tarea compleja al ser muchas las cuestiones a tener en cuenta para generar referencias apropiadas en distintas situaciones. No sólo es necesario decidir qué información se pretende transmitir en ellas, sino que también se deben tener en cuenta los distintos objetivos comunicativos que pueden estar en juego y las posibles implicaciones erróneas que pueden surgir de una referencia poco adecuada.

Existen numerosos trabajos en la literatura que se encargan de abarcar el problema de la generación de expresiones de referencia desde el punto de vista de su función distintiva, estudiando aspectos como la minimalidad de una expresión, la semejanza de la expresión a las utilizadas por los seres humanos, la no ambigüedad de la referencia generada, etc. Sin embargo, aunque existen trabajos sobre generación de lenguaje natural basada en descripciones (Milosavljevic, 1999), se ha realizado poco trabajo desde el punto de vista del enriquecimiento del discurso con expresiones descriptivas o que realcen cierta información considerada importante, ni sobre su relación con la generación de expresiones de referencia con función distintiva.

Dentro de este marco el objetivo general de este trabajo es estudiar la generación de expresiones de referencia no sólo desde el punto de vista de la distinción de entidades, sino también teniendo en cuenta el uso de técnicas que permitan generar referencias que cumplan con el objetivo de generar textos más descriptivos o de llamar la atención del lector u oyente sobre ciertos aspectos del mismo.

Los objetivos principales de este trabajo se podrían dividir en dos grandes bloques. El primero está relacionado con la generación de expresiones de referencia clásica. Se propondrán soluciones y mejoras a las aproximaciones clásicas de la generación de expresiones de referencia, y se evaluarán teniendo en cuenta las métricas clásicas en este campo. El segundo está relacionado con el enriquecimiento de un discurso dado aportando información descriptiva relacionada con las referencias generadas anteriormente. Para ello, se utilizarán figuras retóricas basadas en similitudes entre dominios como la comparación y la analogía, y se discutirá su repercusión sobre el proceso completo de generación de expresiones de referencia. Como idioma de trabajo utilizaremos el inglés.

Por tanto, se pueden enumerar los siguientes objetivos concretos para este trabajo de tesis doctoral:

1. Dividir la generación de expresiones de referencia básica en forma de frase nominal en subproblemas concretos y estudiar cada uno de ellos por separado. La elección de nivel de abstracción será estudiada como un problema que debe resolverse teniendo en cuenta el contexto concreto, la selección de atributos como un problema de búsqueda, y la lexicalización como una tarea altamente dependiente del estilo. Se propondrán distintas soluciones para cada uno de estos subproblemas, algunas de ellas basadas en paradigmas de resolución de problemas típicos de la Inteligencia Artificial.

2. Evaluar las soluciones implementadas utilizando las métricas típicas que se utilizan para este tipo de soluciones de cara a estudiar su adecuación y semejanza respecto a cómo los seres humanos abordan esta tarea. Para ello será necesario recurrir a un corpus de expresiones de referencia creado utilizando evaluadores humanos.

3. Estudiar la generación de expresiones de referencia desde el punto de vista de su función descriptiva en los casos en que es apropiado enfatizar cierta información de los referentes o describir entidades desconocidas. Se utilizarán para ello figuras retóricas basadas en similitudes entre dominios como son la comparación y la analogía.

4. Implementar una arquitectura que permita introducir las expresiones de referencia enriquecidas con figuras retóricas ya mencionadas en un sistema de generación automática de texto previamente existente. Se buscará algún tipo de arquitectura que permita la interacción de los distintos módulos de manera sencilla y eficiente.

1.6. Estructura de la Memoria

A continuación se muestra la secuencia de capítulos que forman este trabajo, así como una breve descripción de su contenido:

Capítulo 1: Introducción. Este capítulo contiene una revisión general de la problemática de la Generación de Expresiones de Referencia. Se exponen los numerosos problemas a tener en cuenta para la generación de referencias que cumplan su función (tanto distintiva como descriptiva) de manera que además no provoquen que la comunicación falle por motivos como las falsas inferencias que se pueden dar dependiendo de la información proporcionada. Se enumeran además los objetivos concretos de este trabajo y la estructura del mismo.

Capítulo 2: Generación de Lenguaje Natural y Figuras Retóricas. Este capítulo contiene una revisión del estado del arte del campo de la Generación de Expresiones de Referencia. Esta revisión incluye no sólo técnicas y trabajos previos sino también diversos recursos como corpus o bases de datos léxicas que han sido utilizadas más adelante en el trabajo. También se presenta una revisión del trabajo existente en figuras retóricas basadas en similitudes entre dominios, proporcionando asimismo información sobre sistemas y recursos relacionados que también han sido utilizados.

Capítulo 3: Algoritmos Evolutivos, CBR y Agentes. En este capítulo se exponen brevemente tres técnicas de Inteligencia Artificial que han sido utilizadas a lo largo del trabajo: Algoritmos Evolutivos, Razonamiento Basado en Casos y Agentes Inteligentes.

Capítulo 4: Generación de Expresiones de Referencia Básicas. Este capítulo aborda la problemática de la generación de expresiones de referencia que buscan cubrir una función distintiva dividiendo el problema en tres partes: elección del nivel de abstracción, selección de

atributos y lexicalización. Para cada una de ellas se presentan diferentes soluciones. Casi todas ellas son además evaluadas en el marco de una serie de tareas de evaluación competitiva en las que fueron presentadas.

Capítulo 5: Figuras Retóricas basadas en Similitudes entre Dominios para la Descripción de Entidades. Este capítulo explora cómo un discurso puede ser enriquecido usando comparaciones y analogías para resaltar información relevante o para ayudar a la comprensión por parte del usuario de entidades desconocidas.

Capítulo 6: Discusión y Propuestas. Este capítulo contiene la discusión del trabajo, donde se analizan los resultados y limitaciones de las soluciones propuestas, así como las posibles soluciones a los problemas encontrados. Se discute asimismo las repercusiones del trabajo presentado en el proceso completo de generación de lenguaje natural, y se proponen diferentes alternativas de implementación.

Capítulo 7: Conclusiones y Trabajo Futuro. En este capítulo se recogen las conclusiones de este trabajo y las líneas de trabajo futuro.

Capítulo 2

Generación de Lenguaje Natural y Figuras Retóricas

"And here... we... go."
— The Dark Knight (2008)

Este trabajo de tesis doctoral se enmarca dentro del campo de la Generación de Lenguaje Natural, y más concretamente en su subtarea de Generación de Expresiones de Referencia. Será necesario por tanto llevar a cabo una revisión del estado del arte sobre generación de referencias que incluya no sólo su definición y problemática clásica, sino también soluciones ya existentes para su resolución, métricas de evaluación de las mismas, y recursos de los que se pueden beneficiar este tipo de algoritmos.

Dado que en este trabajo se recurrirá a algunas figuras retóricas basadas en similitudes entre dominios será necesario también revisar el estado del arte relacionado, así como las herramientas y recursos ya existentes que podrían ser útiles en el desarrollo posterior de nuestras soluciones.

En primer lugar se realizarán una breve introducción sobre Generación de Lenguaje Natural (sección 2.1), y un estudio exhaustivo sobre la Generación de Expresiones de Referencia, incluyendo definición, problemas a resolver, y soluciones existentes (sección 2.2). En la sección 2.3 se trata el tema de la evaluación en el campo de la Generación de Lenguaje Natural, centrándonos sobre todo en la evaluación de las expresiones de referencia generadas automáticamente. Después se presentarán algunos sistemas y recursos relacionados que serán utilizados más adelante en el trabajo (secciones 2.4 y 2.5). Finalmente se realizará un breve estudio sobre figuras retóricas basadas en similitudes entre dominios, así como de algunos recursos y herramientas relacionados con las mismas que resultarán útiles más adelante (sección 2.6).

2.1. Generación de Lenguaje Natural

La Generación de Lenguaje Natural es el subcampo de la Lingüística Computacional y la Inteligencia Artificial orientada al lenguaje que se encarga del diseño y la construcción de sistemas que producen textos comprensibles en lenguajes humanos a partir de una representación subyacente no lingüística de la información, utilizando conocimiento sobre el lenguaje y el dominio de la aplicación.

El uso más común de la generación de lenguaje natural es crear sistemas que presenten la información al usuario en una representación fácil de comprender. Internamente, estos sistemas usan representaciones que son más directas de manipular, tales como bases de conocimiento de sistemas expertos, simulaciones de sistemas físicos, y similares. En muchos casos, sin embargo, estas representaciones necesitan una gran cantidad de experiencia o conocimientos técnicos para ser interpretadas, y por ello es necesario presentar la información al usuario no experto de una manera más clara. Por ejemplo, se han usado técnicas de GLN para:

- Generar predicciones meteorológicas textuales a partir de representaciones en mapas gráficos, como en (Goldberg et al., 1994; Belz, 2008).
- Explicar información médica de una forma amigable para el paciente, como en (Cawsey et al., 1995; Buchanan et al., 1995; Portet et al., 2007).
- Describir una línea de razonamiento llevada a cabo por un sistema experto, como en (Swartout, 1983; Miyata y Matsumoto, 1998; Alonso-Lavernia et al., 2006).
- Producir automáticamente descripciones de rutas, como en (Dale et al. 2003).
- Generar textos literarios, como en (Callaway y Lester, 2001b; Gervás et al., 2005).

Esta lista de posibles usos es sólo indicativa, y se puede ampliar consultando la bibliografía al respecto.

Existe todavía un gran debate sobre cómo organizar el proceso de la generación automática de texto. Sin embargo, dentro de la comunidad de generación de lenguaje natural, se ha llegado al consenso de que hay seis tipos básicos de actividades que es necesario realizar para transformar unos datos de entrada en un texto final de salida (Reiter y Dale, 2000). Estas seis actividades básicas son:

Determinación de Contenido (*Content Determination*) Es el proceso de decidir qué información debe ser comunicada en el texto. En ocasiones será la aplicación que utiliza el sistema de generación la que proporcione la información exacta que debe ser comunicada, pero en muchos casos será el sistema GLN el que se encargue de seleccionar un subconjunto apropiado dentro la información disponible.

Este proceso se puede describir como la creación de un conjunto de mensajes a partir de las entradas al sistema, siendo éstos los objetos que serán usados por los siguientes procesos de generación. Tanto el proceso de creación de los mensajes, como la forma y el contenido de los mismos, dependen en gran manera de la aplicación.

Planificación del Discurso (*Discourse Planning*) Es el proceso de ordenar y estructurar el conjunto de mensajes que hay que transformar. Un texto no es sólo una colección aleatoria de trozos de información: la información se presenta en un determinado orden y normalmente hay una estructura subyacente para la presentación. Una buena estructuración puede hacer un texto mucho más fácil de leer.

Agregación (*Aggregation*) Aunque no existe un consenso en el campo acerca de su definición exacta (Reape y Mellish, 1999), la agregación se puede ver como la tarea dentro de la generación que se encarga de decidir cómo se compactará la información dada en un texto. Esta tarea opera a diferentes niveles lingüísticos, y debido a ello Reape y Mellish hacen una clasificación de los distintos tipos de agregación: conceptual, de discurso, semántica, sintáctica, léxica y referencial. Sin embargo, la línea entre ellos es muy fina, y en algunos casos un ejemplo específico puede ser clasificado dentro de diferentes tipos de agregación.

Generación de Expresiones de Referencia *(Referring Expression Generation)*

Es el proceso de seleccionar palabras o frases para identificar entidades del dominio o proporcionar información sobre las mismas. La generación de referencias es normalmente considerada como una tarea discriminatoria, donde el sistema necesita comunicar suficiente información para distinguir una entidad de otras dentro del dominio, siempre teniendo en cuenta que la información proporcionada no debe dar lugar a falsas implicaciones o inferencias. Esto generalmente requiere tener en cuenta factores contextuales, incluyendo en particular el contenido de comunicaciones previas con el usuario. Como ya hemos mencionado, nuestro trabajo se centrará en esta tarea.

Lexicalización (*Lexicalization*) Es el proceso de decidir qué palabras y frases específicas deben ser elegidas para expresar los conceptos y relaciones del dominio que aparecen en los mensajes. En muchos casos

la lexicalización puede ser realizada trivialmente si se asocia una palabra o frase específica para cada concepto o relación del dominio. Sin embargo, se puede mejorar la fluidez permitiendo al sistema variar las palabras usadas para expresar cada concepto o relación en distintas apariciones.

Cahill (1998) diferencia entre REALIZACIÓN LÉXICA y ELECCIÓN LÉXICA. El primer término se aplica a la conversión de información conceptual en etiquetas léxicas, mientras que el segundo se emplea para definir la elección entre diferentes alternativas léxicas que representan el mismo contenido conceptual. Tanto la realización como la elección léxica se enfrentan al problema de la adquisición de conocimiento. Para la realización léxica, el lexicón debe contener etiquetas léxicas para todos los conceptos y estructuras que aparezcan en los mensajes. Para la elección léxica, los recursos léxicos necesitan proporcionar suficiente conocimiento para llevar a cabo este tipo de elección.

Realización Superficial (*Surface Realization*) Cada lenguaje está definido, al menos en parte, por un conjunto de reglas que especifican qué es una oración bien formada en dicho lenguaje. Estas reglas de la gramática gobiernan la morfología, que determina cómo se forman las palabras, y la sintaxis, que determina cómo se forman las frases.

La realización superficial es así el proceso de aplicar las leyes de la gramática para producir un texto que es sintáctica, morfológica y ortográficamente correcto.

Reiter (1994) propone una división de todas estas tareas en tres grandes bloques (Figura 2.1). A partir de la información conceptual inicial, y teniendo en cuenta el objetivo final del texto a generar, la PLANIFICACIÓN DEL DISCURSO se encarga de la estructuración del contenido para que tome la forma del texto final. Sobre este plan del discurso, la PLANIFICACIÓN DE ORACIONES se encarga de las referencias y la lexicalización de las oraciones una por una, dando lugar a un plan del discurso enriquecido con toda la información necesaria para la fase final. La REALIZACIÓN LINGÜÍSTICA se corresponde con la tarea de la realización superficial, que se encarga de dar forma al texto final de manera que sea sintáctica, morfológica y ortográficamente correcto.

A este tipo de arquitectura se la denomina secuencial o *pipeline*, y el ejemplo más difundido es el que acabamos de ver. La arquitectura secuencial tiene la ventaja de la simplicidad. Cada módulo está encapsulado al máximo porque recibe la entrada de un origen y envía la salida a un destino. Se asume que la representación intermedia en cada etapa es una representación completa de lo que se sabe en ese punto sobre la información que se va a generar. Sin embargo, su principal desventaja es que las decisiones tomadas

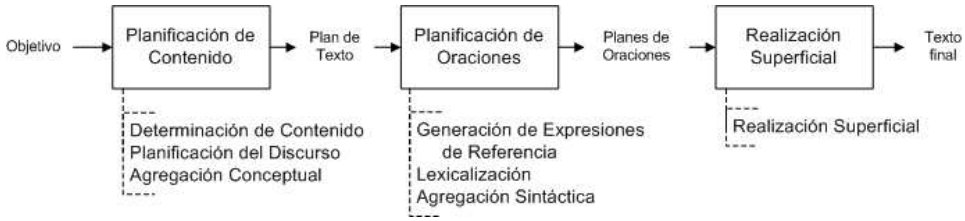

Figura 2.1: Esquema del *pipeline* propuesto por Reiter

en una fase deben mantenerse a lo largo de toda la cadena, sin posibilidad de revisión o mejora. Por tanto, se trata de un modelo poco flexible.

Sistemas como HYLITE+ (Bontcheva y Wilks, 2001) han explorado alternativas a la rigidez de comunicación entre módulos proporcionada por el *pipeline* desde el punto de vista de la Determinación de Contenido y la Realización Superficial. HYLITE+ es un sistema de generación de hipertexto dinámico que produce explicaciones de tipo enciclopédico en un determinado dominio. En este sistema el planificador de contenido produce inicialmente un plan de texto para un hipertexto conciso que sólo contiene hechos sobre el concepto a explicar. Durante la realización superficial, una vez que el formato final ha sido decidido, se escoge la alternativa adaptable más apropiada. En ocasiones esto lleva a la inclusión de un nuevo objetivo comunicativo que resulta en expandir el texto básico inicial utilizando información extra.

2.2. Generación de Expresiones de Referencia

Como ya se ha comentado, la Generación de Expresiones de Referencia es una de las etapas del proceso de generación automática de texto. El lenguaje natural proporciona numerosas maneras de referirse a las entidades. Dependiendo del CONTEXTO DEL DISCURSO operativo en cada momento se pueden usar pronombres, formas reflexivas, demostrativos, frases nominales, nombres propios, descripciones, etc. Sin embargo, el escritor o hablante no es libre de escoger una cualquiera de estas formas, sino que tendrá que tener en cuenta el CONTEXTO SITUACIONAL del discurso en cada momento concreto.

La entrada esperada para la Generación de Expresiones de Referencia son los mensajes que formarán el discurso en el mismo orden en que aparecerán en el texto final. Cada entidad o referente estará representada en estos mensajes utilizando algún tipo de identificador unívoco que representará a esta entidad a lo largo de todo el discurso. De esta manera la tarea podrá trabajar no sólo con la manera específica de referirse a algo, sino también con las decisiones al respecto que ya se han tomado para la misma entidad o para otras relacionadas. Como salida de esta tarea se espera obtener el mismo conjunto de mensajes ordenados donde las entidades o referentes habrán sido sustituidas por algún tipo de representación que indica de qué manera

serán realizadas posteriormente en el texto (pronombre, nombre propio, frase nominal, etc.) y toda la información relacionada que vaya a ser necesaria más adelante.

Desde el punto de vista de la generación de expresiones de referencia, tradicionalmente se han considerado dos cuestiones fundamentales a tener en cuenta. Por un lado, se pretende evitar la ambigüedad, para lo que habrá que proporcionar al oyente o lector suficiente información para que distinga la entidad referida de las demás que se puedan encontrar en ese momento en el contexto. Por otro lado, habrá que tener cuidado con no proporcionar demasiada información que pueda ser irrelevante o redundante y pueda en el extremo, llevar a confusión. En otras palabras, habrá que introducir la mínima información posible para evitar aburrir al lector u oyente, pero la suficiente como para permitir la identificación del referente al que nos estamos refiriendo.

2.2.1. Tipos de expresiones de referencia

El conjunto de posibles formas de presentar en un discurso una referencia es muy amplio. Aquí sólo daremos una breve descripción de los tipos más básicos de expresiones de referencia.

Frase nominal con Determinantes Definidos o Indefinidos

Generalmente las referencias indefinidas se usan para introducir en el contexto del discurso entidades que son nuevas para el lector u oyente. La forma más común de referencia indefinida es la que viene marcada por el determinante *a* (o *an*), pero también puede venir acompañada por un cuantificador como *some*.

El determinante indefinido *a* no indica si la entidad referida es identificable para el hablante o escritor, ya que se puede referir tanto a una entidad específica que es introducida por primera vez en el discurso, como a una entidad no específica. Por ejemplo, en la oración *I am going to the butchers to buy a goose* el hablante puede estar refiriéndose tanto a una oca que ya ha escogido, como a uno que escogerá al llegar a la carnicería.

Las referencias definidas se usan generalmente para referirse a entidades que son identificables para el lector u oyente. Esto incluye tanto entidades que han sido mencionadas previamente en el discurso, y que por tanto han sido introducidas en el modelo del discurso, como entidades que están incluidas en el conjunto de creencias sobre el mundo que tiene el lector u oyente. Otro caso especial es en el que la unicidad del objeto hace que su primera aparición en el discurso sea siempre definida (por ejemplo, *The New York Times*).

Pronombres

Los pronombres son en realidad otra forma de referencia definida. Las restricciones en el uso de una referencia pronominal son mayores que las de uso de frases nominales definidas, ya que requieren que el referente tenga un alto grado de activación o PROMINENCIA en el modelo del discurso. Aunque no siempre, los pronombres normalmente se usan en referencias a entidades que fueron introducidas no más allá de una o dos frases atrás en el discurso, mientras que las frases nominales definidas se pueden referir a entidades que se hayan mencionado mucho más atrás.

Los pronombres también pueden participar en el fenómeno de la *catáfora*, donde son mencionados antes de que lo sean sus referentes. Un ejemplo sería *even before she saw it, Dorothy had been thinking about the Emerald City everyday.*

Demostrativos

Los pronombres demostrativos, como *this* y *that*, se comportan de manera diferente a como lo hacen los pronombres definidos simples como *it*. Pueden aparecer solos o utilizados como determinantes, por ejemplo, *this ingredient*. *This*, el demostrativo de proximidad, indica cercanía literal o metafórica, mientras que *that*, el demostrativo de distancia, indica lejanía literal o metafórica.

Nombres

Los nombres son una forma muy común de expresión de referencia, incluyendo nombres propios, organizaciones y lugares. Los nombres se pueden usar tanto para entidades nuevas o viejas en el discurso.

En general, las frases nominales indefinidas se suelen usar en las referencias iniciales al introducir nuevas entidades en el discurso. En cambio, las frases nominales definidas y los pronombres se usan para las referencias subsiguientes cuando se hace referencia a entidades ya conocidas por el lector. Sin embargo, existen excepciones a esta regla general. Más concretamente, las referencias definidas se pueden usar para referencias iniciales en algunos casos. Por ejemplo, una referencia definida se puede utilizar para introducir una entidad que se asume que es conocida o inferible por el lector, como en *The station where I boarded the train was deserted.*

2.2.2. Normas Conversacionales

Grice (1975) defiende la existencia de una serie de normas conocidas tanto por el hablante como por el oyente que guían la conversación y hacen posible que las inferencias deducidas por el oyente sean exactamente las que

el hablante pretendía transmitir. A este tipo de inferencias que el oyente deduce Grice las denomina implicaciones o inferencias conversacionales, y para identificarlas propone las cuatro *Máximas Conversacionales* que se detallan a continución:

> *Máxima de Calidad*: Trata de que tu contribución sea verdadera. Más específicamente:
>
> 1. No digas lo que crees que es falso.
> 2. No digas algo para lo que te falten evidencias adecuadas.
>
> *Máxima de Cantidad*:
>
> 1. Haz tu contribución tan informativa como sea necesario para los propósitos del intercambio.
> 2. No hagas tu contribución más informativa de lo requerido.
>
> *Máxima de Relevancia*: Sé relevante.
> *Máxima de Manera*: Sé claro. Más específicamente:
>
> 1. Evita expresiones oscuras.
> 2. Evita la ambigüedad.
> 3. Sé breve (evitando prolijidad innecesaria).
> 4. Sé ordenado.

En (Dale y Reiter, 1995) los autores revisaron la literatura psicolingüística existente sobre cómo los humanos generaban expresiones de referencia y se encontraron con que los seres humanos en muchos casos incluyen modificadores innecesarios en las expresiones de referencia que generan. Existen diversas explicaciones para este hecho que parece contradecir la Máxima de Cantidad de Grice. Por ejemplo, en algunos experimentos se podía observar que los sujetos tomaban ciertas decisiones sobre el contenido de la referencia que estaban generando antes de terminar de pensar la expresión de referencia completa que iban a generar. Algunas de estas decisiones les llevaban a tomar atributos innecesarios para la referencia final, y sin embargo ya no volvían atrás para revisar la decisión tomada. Otra posible explicación es que el hablante puede considerar que introduciendo modificadores extra dará lugar a expresiones de referencia que puedan ser entendidas más fácilmente. También es posible que este fenómeno aparezca porque los hablantes usan en algunos casos guiones de referencia precompilados en lugar de comenzar una referencia desde cero. Por ejemplo, si una persona ha referenciado correctamente en varias ocasiones un cierto tipo de entidad de una manera determinada, puede tender siempre a usar la misma forma de referencia aunque no sea la más adecuada a cada caso.

Después de este trabajo las Máximas Conversacionales de Grice fueron estudiadas y reinterpretadas por Dale y Reiter (1996) desde el punto de

vista concreto de la generación de expresiones de referencia para comprobar si eran compatibles con la forma en que los seres humanos se refieren al mundo que les rodea. Veamos las conclusiones que extrajeron para cada una de las máximas.

La *Máxima de Calidad* puede ser interpretada como que una expresión de referencia debe ser una descripción ajustada y verdadera de la entidad a la que nos estamos refiriendo. En general se puede suponer que los sistemas de generación sólo usan la información que conocen y consideran verdadera a la hora de generar un texto, con lo que esta máxima siempre se cumpliría.

La *Máxima de Cantidad* fue resumida por los autores en "No añadas información extra que no es necesaria". En el caso de las referencias con función distintiva se puede considerar que deben contener suficiente información para permitir al usuario identificar el objeto al que se están refiriendo, y no más información que pueda dar lugar a falsas inferencias. Sin embargo, las referencias con función descriptiva tienen como objetivo presentar información al lector u oyente que se considera relevante que conozca pero que no es indispensable para distinguir a la entidad. El uso de este tipo de referencias podría considerarse como una violación de la Máxima de Cantidad de Grice "No hagas tu contribución más informativa de lo requerido". Esto sería cierto si el propósito de la referencia fuera proporcionar suficiente información para que el lector u oyente sepa a qué referente se están refiriendo. Sin embargo, si estamos considerando que el objetivo no es sólo identificar a la entidad, sino alertar al lector de una propiedad de la misma, se puede considerar que la máxima se cumple. Siguiendo el razonamiento sobre implicaciones conversacionales de Grice, si la referencia contiene información que no se requiere para identificar a la entidad se asumirá que dicha información es importante para algo más, y dependiendo del contexto se harán unas inferencias u otras.

La *Máxima de Relevancia* también fue interpretada por Reiter y Dale dependiendo de la función de la referencia en el discurso. En el caso de la función distintiva de las referencias, la información presente en la referencia se considera o no relevante según su valor a la hora de distinguir. Es decir, no se deben mencionar atributos que no tengan poder discriminador y no permitan distinguir al referente de al menos uno de los demás que le rodean. En el caso de la función descriptiva de las referencias, la información presente se considera o no relevante simplemente porque se ha decidido que se quiere aprovechar la referencia para comunicar esa información.

La *Máxima de Manera* fue sobre todo considerada desde el punto de vista de que se debe ser breve. Por tanto, una expresión de referencia debe ser breve siempre que sea posible.

Como conclusión final los autores consideraron que las Máximas de Grice pueden resumirse en "No incluyas elementos que no sirvan para nada". Viendo la generación de expresiones de referencia como un proceso orientado a un objetivo, generalmente no es necesario seguir esta directiva de manera

estricta, ya que este comportamiento de no incluir elementos que no sean útiles en las referencias aparece en el mismo mecanismo de la generación de las mismas.

2.2.3. Evitando Falsas Inferencias

Reiter (1990) tomó como base las Máximas Conversacionales de Grice para tratar de dar unas pautas a seguir si se quieren evitar falsas inferencias a la hora de generar expresiones de referencia en forma de frases nominales. A partir de la definición de COMPONENTE, que puede ser tanto el tipo al que pertenece el referente, como un par atributo-valor que se aplica al referente, formaliza las implicaciones conversacionales usando reglas de preferencia. Una referencia no dará lugar a falsas inferencias si cumple tres reglas:

Brevedad Local (*Local Brevity*)

No debe ser posible generar expresiones de referencias más cortas reemplazando un conjunto de componentes por un único componente diferente. Esta regla no concuerda exactamente con la máxima de cantidad de Grice. Un componente puede ser realizado en el texto usando varias palabras, por lo que un menor número de componentes no siempre implica una realización de la referencia más corta.

Evitar Componentes Innecesarios (*No Unnecesary Components*)

Las expresiones de referencia no deben contener componentes innecesarios. Esta regla surge del hecho de que los seres humanos encuentran anómalos los *pleonasmos* (Cruse, 1986) como *a female mother* o *an unmarried bachelor*, que resultan en información innecesaria por ser redundante.

Preferencia Léxica (*Lexical Preference*)

Si los valores de los atributos y tipos de las entidades usadas en la referencia son miembros de una taxonomía, entonces la expresión de referencia puede realizarse usando diferentes niveles de especificidad. Suponiendo que existe una jerarquía de preferencia sobre la taxonomía, esta regla se cumple siempre que se use el valor válido preferido dentro de la taxonomía.

Sin embargo, el autor realiza este estudio sólo sobre referencias con función exclusivamente distintiva en las que el objetivo perseguido es identificar una entidad entre otras con las que se puede confundir. En el caso de la función distintiva de las referencias estas reglas deberían ser revisadas. Por ejemplo, la regla de *Evitar Componentes Innecesarios* no se cumpliría bajo esta definición exacta.

2.2.4. Otras Consideraciones

Existen muchas más consideraciones a tener en cuenta si se desea modelar la manera en que los seres humanos crean referencias para mencionar las cosas que les rodean. Serán especialmente relevantes para el trabajo aquí presentado las relacionadas con la elección de los modificadores a la hora de generar referencias nominales básicas. En esta línea Dale y Reiter (1995) se hicieron las siguientes preguntas:

1. Si se pueden usar varios atributos, ¿cuál se debería escoger? Por ejemplo, supongamos que *the screw* no es cumple la función distintiva deseada (no permite distinguir el referente de los que le rodean), pero que cada una de las expresiones *the small screw*, *the black screw* y *the screw made in Korea* sí lo son. ¿Cuál es la mejor elección?

2. ¿Es preferible añadir un atributo más a una referencia para garantizar su éxito ante el objetivo de identificación, o usar un tipo más específico para referirnos a la entidad? Por ejemplo, ¿es mejor usar *the small screw* o *the woodscrew*?

3. ¿Es mejor usar adjetivos relativos o absolutos? Por ejemplo, ¿sería mejor decir *the long screw* o *the three-inch screw*?

Basándose en la poca información psicolingüística que pudieron reunir los autores plantearon las siguientes hipótesis como posibles guías a la hora de contestar a las preguntas anteriores:

1. Parece que los seres humanos prefieren usar adjetivos que describen propiedades fácilmente perceptibles como el tamaño, la forma o el color. En general es difícil encontrar situaciones en las que se usen adjetivos que no son visualmente perceptibles.

2. Los seres humanos en ocasiones tienen dificultades para determinar si un objeto pertenece a una clase especializada. Por ejemplo, casi cualquier persona identificaría a una entidad referida por *the dog*, pero tal vez tendría más dificultades con *the fox terrier* si no sabe nada sobre razas caninas. Si hay dudas sobre si el oyente o lector va a saber distinguir la clase especializada, es mejor añadir atributos a la referencia antes que cambiar el tipo de la misma.

3. Las personas parecen preferir el uso de adjetivos relativos (*long*, *big*, etc.) cuando se están refiriendo a algo, y parecen tener menos dificultades también para entenderlos. Sin embargo, existen ciertos tipos de textos escritos, como podrían ser los de una serie de instrucciones, en los que se prefiere el uso de adjetivos absolutos (*two centimeters long*) sobre relativos. Esto puede deberse a que el escritor no puede saber

el contexto en el que será leído el texto, y por tanto quiere evitar que pueda haber cualquier confusión con el uso de los adjetivos relativos.

2.2.5. Algoritmos para la Generación de Expresiones de Referencia

A continuación se exponen algunas de las soluciones de generación de expresiones de referencia más conocidas, todas ellas centradas en la función distintiva de las referencias. La primera de ellas (el algoritmo incremental de Reiter y Dale) ha servido como base a muchas otras y será la que más utilizaremos a lo largo de este trabajo.

Algoritmo Incremental de Reiter y Dale

Reiter y Dale (1992) describen un algoritmo para la generación de expresiones de referencia dentro de un sistema de generación de lenguaje natural. Este algoritmo es una mejora de otro previamente existente, basado en evidencias psicolingüísticas y en el análisis de transcripciones de diálogos entre humanos. Este algoritmo supone una línea de base para comparar el funcionamiento y rendimiento de algoritmos similares.

El algoritmo de Reiter y Dale se basa en las siguientes suposiciones sobre la base de conocimiento subyacente empleada:

1. Toda entidad se caracteriza por una colección de atributos y sus respectivos valores.
2. Toda entidad tiene, entre sus atributos, un tipo.
3. La base de conocimiento puede organizar los valores de algunos atributos en forma de jerarquía de subsunción.

Además, este algoritmo depende de la siguiente información para tomar sus decisiones finales:

- Cada objeto representado en el sistema debería tener un valor básico que se corresponde con el concepto preferido para referirse a dicho objeto. Esto se usa para ofrecer un punto de partida a partir del cual se puedan construir las referencias al objeto: los conceptos más cercanos a este valor básico en la taxonomía serán preferibles a conceptos más lejanos. Cada objeto puede tener asignadas diferentes clases de nivel básico para distintos usuarios. Con esta información es posible decidir si se quieren generar las referencias de manera más específica o más general.
- Para cada objeto debe haber un modo de determinar si el usuario (la persona para que el sistema está generando el texto) es consciente

de que un determinado par atributo-valor es aplicable a dicho objeto. Esto ayuda a determinar si la mención de una característica particular le resultará útil al usuario para identificar el objeto.

- Se debe disponer de una lista de ATRIBUTOS PREFERIDOS o *preferred attributes* que describe el conjunto de atributos que los seres humanos prefieren usar al describir objetos (en su dominio específico, Reiter y Dale observaron que los atributos preferidos eran el *tipo*, el *tamaño*, la *forma* y el *color*).

Para construir una referencia a una entidad en particular, el algoritmo toma como entrada un símbolo correspondiente al referente objetivo (aquel para el que se quiere generar la referencia) y una lista de símbolos que corresponden a otras entidades dentro del foco de atención, basado en el referente objetivo y denominado CONJUNTO DE CONTRASTE. El algoritmo devuelve una lista de pares atributo-valor correspondientes al contenido semántico de la expresión de referencia que se está generando. El algoritmo opera iterando sobre la lista de atributos disponibles, buscando aquél que, siendo conocido por el usuario, elimina el mayor número de elementos restantes del conjunto de contraste. El valor básico considerado para los atributos se usa para dar preferencia a determinados atributos cuando esta decisión no sea clara.

Además, el algoritmo escoge el valor más apropiado para un cierto atributo a partir de un valor inicial. La idea es usar un valor para ese atributo que sea subsumido por el valor inicial, que describa con precisión al referente objetivo, que elimine el mayor número de distractores posibles y que, siguiendo estas restricciones, sea lo más próximo posible al nivel inicial de la taxonomía. Así, se evita usar una referencia como *the chihuahua* en un contexto donde sólo haya un único perro, en cuyo caso la referencia más adecuada sería simplemente *the dog*.

Se puede ver el algoritmo completo en pseudocódigo en la Figura 2.2.

Algoritmo Sensible al Contexto de Khramer y Theune

Krahmer y Theune (2002) presentan una variante del Algoritmo Incremental de Reiter y Dale que tiene como novedad que es sensible al contexto. Cada entidad del dominio tiene asignado un cierto GRADO DE PROMINENCIA o *salience*, y las descripciones son generadas con respecto a un conjunto de contraste definido como el conjunto de entidades que tienen igual o mayor prominencia que la entidad para la que se está generando la referencia. De esta manera el algoritmo permite la generación de referencias más escuetas para las entidades prominentes como podrían ser las entidades que han sido mencionadas recientemente en el discurso.

La principal diferencia entre el Algoritmo Incremental y esta aproximación es la condición de éxito. Mientras que en el Algoritmo Incremental la ejecución finaliza satisfactoriamente cuando el conjunto de contraste queda

```
make-referring-expression ( r, C , P )

L ← { }
D ← C
for each member A_i of list P do
   V = find-best-value( A_i, basic-level-value(r, A_i))
   if V ≠ nil and rules-out(<Ai, V>) ≠ nil then
      L ← L ∪ {<Ai, V>}
      D ← D – rules-out(<Ai, V>)
   endif
   if D = { } then
      if <type, X> ∈ L for some X then
         return L
      else
         return L ∪ {<type, basic-level-value(r, type)>}
      endif
   endif
next
return failure
```

```
find-best-value ( A, initial-value )

if user-knows(r, <A, initial-value>) = true then
   value ← initial-value
else
   value ← nil
endif
for each v_i ∈ taxonomy-children(initial-value) do
   if v_i subsumes value(r, A) and
         (new-value ← find-best-value(A_i,v_i)) ≠ nil and
         (value = nil  or
         |rules-out(<A, new-value>)| > |rules-out(<A, value>)| )
   then
      value ← new-value
   endif
next
return value
```

```
rules-out ( <A, V> )

return { x : x ∈ D and user-knows(x, <A, V>) = false }
```

Figura 2.2: Algoritmo Incremental de Reiter y Dale

vacío, el Algoritmo Sensible al Contexto comprueba si la entidad referida es la más prominente entre aquellas que corresponden al conjunto de propiedades seleccionado para generar la referencia.

Si la referencia corresponde a la entidad más prominente, entonces se considera exitosa, incluso si corresponde también a otras entidades del contexto que son menos prominentes. De esta forma, si se asume que inicialmente todas las entidades tienen el mismo grado mínimo de prominencia, introducir una nueva entidad requerirá distinguirla de todos los elementos en el conjunto de contraste, mientras que volver a referirse a una entidad ya mencionada requiere solamente distinguirla de las entidades que tiene el mismo o mayor grado de prominencia.

2.2.6. Sistemas con Generación de Información Descriptiva sobre Entidades

Según Milosavljevic (2003), la descripción de una entidad es la realización lingüística de un conjunto de una o más proposiciones que tienen como propósito hacer que el oyente se construya un modelo mental de la entidad descrita. En su trabajo de tesis (Milosavljevic, 1999) exploró el uso de comparaciones en la descripción de entidades, postulando que en el proceso de producir una descripción resulta útil el uso de comparaciones con entidades similares y familiares. Otros trabajos han estudiado la generación de descripciones de entidades de diversos tipos.

En (Lavoie et al., 1997) se presenta el sistema MODEX, que genera descripciones en lenguaje natural de modelos software orientados a objetos. MODEX permite al usuario personalizar los planes de texto en ejecución, de manera que cada texto refleja las preferencias individuales del usuario en cuanto al contenido y/o la presentación de la salida generada.

En (Ardissono y Goy, 2000) se presenta SETA, un sistema de generación automática de catálogos web personalizados. Para ello emplea técnicas de GLN basada en plantillas que permiten la generación dinámica de categorías de productos y sus elementos. En estas descripciones el sistema mezcla diferentes tipos de información sobre las características y propiedades de los productos presentados.

El sistema de generación de texto Methodius (Isard, 2007) crea descripciones personalizadas de objetos de museo que pueden ser presentadas de diversas maneras, desde texto o voz en un aparato en mano hasta diálogo con un guía de museo robótico.

2.3. Evaluación para la Generación de Lenguaje Natural

Como ocurre con muchos otros problemas dentro del campo de la Inteligencia Artificial, para conocer el alcance y adecuación de cualquier solución propuesta será necesario evaluarla de alguna manera. En el caso de la generación de lenguaje natural se puede considerar que el resultado ideal sería la generación de textos equivalentes a los generados por los seres humanos, y que por tanto fueran entendidos por ellos de manera eficiente.

Para la evaluación de algoritmos que abordan la tarea de la generación de expresiones de referencia, se pueden encontrar seis cuestiones principales a tener en cuenta (Gatt, 2007; Belz y Gatt, 2007; Gatt et al., 2008):

1. **Unicidad**. Los algoritmos tienen que ser evaluados para comprobar si las referencias generadas que producen distinguen unívocamente a la entidad a referenciar. Normalmente la unicidad es considerada como un requisito indispensable para que un algoritmo funcione correctamente. La unicidad puede medirse automáticamente a partir del contexto, sin necesidad de ninguna fuente de conocimiento adicional.

2. **Minimalidad**. Se considera que una expresión de referencia es mínima si identifica unívocamente a la entidad deseada, y además no hay referencias más cortas que lo hagan también. En general la minimalidad es una característica deseable para un algoritmo de generación de expresiones de referencia, pero no indispensable. Al igual que ocurre con la unicidad, la minimalidad puede medirse automáticamente a partir de la situación inicial sin usar fuentes de conocimiento adicionales.

3. **Semejanza con las Referencias Humanas**. Parece importante comparar las salidas de los algoritmos con las expresiones de referencia generadas por humanos en las mismas situaciones, ya que no existe una definición exacta de cómo las personas se enfrentan al problema de la

generación de referencias. Generalmente será necesario algún tipo de corpus para evaluar esta cuestión.

4. **Precisión de Identificación**. Consiste en comprobar si las expresiones de referencia generadas permiten a los seres humanos identificar de forma apropiada la entidad referenciada. Vendrá identificada por un índice de error que expresa las ocasiones en que los evaluadores humanos no han identificado correctamente a la entidad referenciada. Esta métrica debe ser medida usando evaluadores humanos.

5. **Velocidad de Identificación**. Comprueba si las expresiones de referencia generadas permiten a los seres humanos identificar a la entidad referenciada de manera rápida. Se puede dividir en dos índices diferentes: el tiempo necesario para la lectura de la referencia, y el tiempo requerido desde que finaliza la lectura hasta que se identifica la entidad correspondiente. También en este caso se utilizan evaluadores humanos.

Las dos últimas métricas (precisión y velocidad de identificación) podrían agruparse en una única denominada **Facilidad de Comprensión**, que abarca tanto la claridad de la referencia (dada por la velocidad) como la precisión a la hora de identificar la entidad referenciada.

2.3.1. Comparación con Resultados Humanos

Para evaluar la similitud entre el lenguaje generado por seres humanos con el realizado por un sistema de generación se puede utilizar un corpus. Existe un fuerte tradición de evaluación basada en corpus en la literatura del campo de Procesamiento de Lenguaje Natural. Muchas de las métricas definidas han sido muy utilizadas en campos como la Traducción Automática (Papineni et al., 2002) y la Generación de Lenguaje Natural (Barzilay y Lapata, 2005; Belz y Reiter, 2006). Sin embargo se ha discutido mucho si este tipo de corpus se puede considerar como la referencia para comparar las bondades de las distintas soluciones existentes (Reiter y Sripada, 2002). Dado que estos recursos suelen estar generados por una amplia variedad de autores, se suele encontrar una variación demasiado grande que no permite extraer una forma común en la que se debe realizar la tarea estudiada.

En el caso de la generación de expresiones de referencia, el uso de un corpus es delicado por razones similares. Debido a la naturaleza extremadamente semántica de la generación de referencias, la salida de los algoritmos que abordan esta tarea dependen en gran forma de la manera en que se extrae la semántica de las referencias, incluso aunque sea en un dominio muy restringido.

Corpus de Expresiones de Referencia

Poca investigación se ha centrado en el estudio de la generación de expresiones de referencia en contextos comunes. El corpus GREC (Belz y Gatt, 2008) consta de alrededor de 2000 textos en total, todos escogidos de las secciones introductorias de artículos de la Wikipedia, en cinco dominios distintos (ciudades, países, ríos, montañas y personas). El objetivo es anotar las referencias al tema principal de cada uno de estos textos, llamadas *Main Subject References* (MSR). Tres grandes categorías de expresiones se han anotado en el corpus (frases nominales sujeto, frases nominales objeto, y frases nominales en genitivo y pronombres que realizan la función de sujeto). Estas categorías de expresiones de referencia son relativamente fáciles de identificar, y representan la mayoría de los casos de aparición del MSR en los textos. Se les pidió a los anotadores identificar a sujetos y objetos y decidir si hacen o no referencia al tema principal del texto.

El corpus GRE3D3 (Viethen y Dale, 2008) es una colección de expresiones de referencia de objetos únicos en situaciones 3-D muy simples reunida a través de un experimento on-line utilizando participantes humanos. El estudio estaba orientado a arrojar luz sobre la cuestión de si las relaciones espaciales resultan tan poco utilizadas como se sugiere en la literatura en las situaciones en que existen además descripciones no-relacionales válidas. En el diseño de los datos para el experimento los autores tuvieron mucho cuidado con una serie de factores que se podía esperar que tuvieran influencia en el uso de relaciones espaciales: el protagonismo de otras propiedades como el color y el tamaño (es decir, si la mayoría de los objetos son del mismo o similar tamaño y color, de modo que no son muy distintos desde el punto de vista de sus propiedades directas), lo fácil que resulta distinguir una entidad de los otros objetos a su alrededor, lo fácil resulta identificar al objetivo de la escena sin usar cualquier información de ubicación, y la importancia visual de otros objetos que podrían servir como objetivos de las relaciones en la descripción. Unos ejemplos de las situaciones planteadas se pueden ver en la Figura 2.3.

Uno de los proyectos que más importancia han tenido en el campo de la generación de expresiones de referencia es el proyecto TUNA (van Deemter et al., 2006; Gatt et al., 2007). El proyecto TUNA es un proyecto de investigación financiado por el UK's Engineering and Physical Sciences Research Council (EPSRC), con la colaboración de la Universidad de Aberdeen, la Open University del Reino Unido y la Universidad de Tilburg. El principal objetivo era crear un corpus de expresiones de referencia que permitiera evaluar los algoritmos de generación de las mismas. El proyecto comenzó en Octubre de 2003 y finalizó en Febrero de 2007.

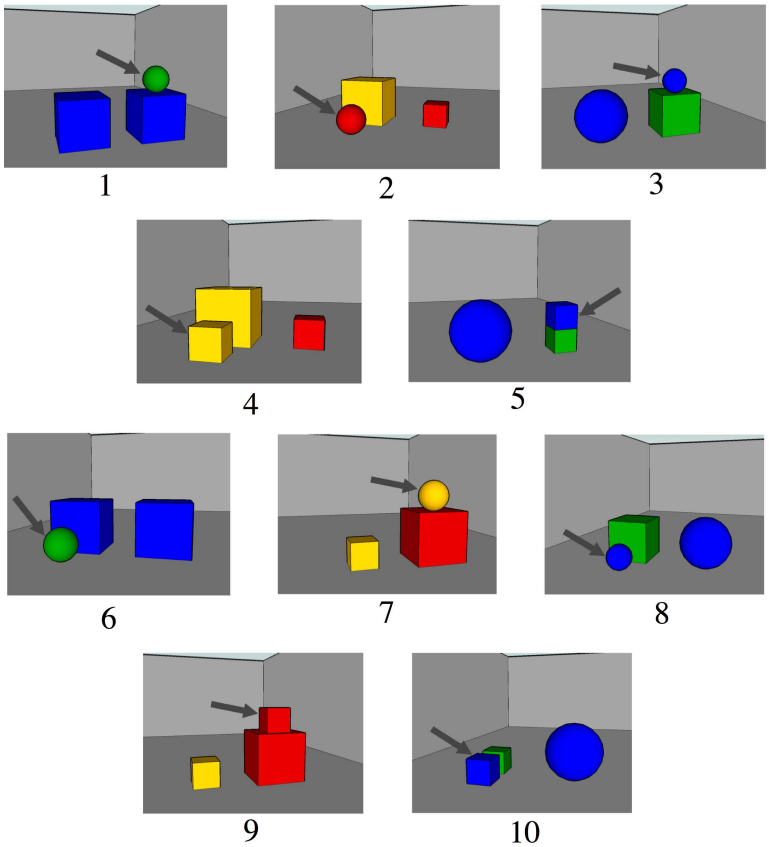

Figura 2.3: Ejemplos de las situaciones planteadas en el experimento del corpus GRE3D3

En el marco del proyecto TUNA se desarrolló un corpus de expresiones de referencia para entidades en dominios visuales de fotografías de personas y muebles[1]. El corpus fue obtenido durante un experimento on-line en el que distintos sujetos escribían descripciones de entidades OBJETIVO en un dominio en el que también se encontraban otras seis entidades denominadas DISTRACTORES. Un ejemplo de las situaciones que se planteaban a los evaluadores se muestra en la Figura 2.4. Cada expresión de referencia del corpus está acompañada por la representación del dominio en el que fue generada, conteniendo el referente objetivo y el conjunto de distractores, cada uno con sus atributos y valores. El corpus final generado ha sido utilizado en las tareas de evaluación competitiva desarrolladas en el campo de la generación de expresiones de referencia que se explicarán más adelante. El corpus TUNA ha sido el corpus utilizado como referencia en este trabajo para la

[1]Consultar http://www.csd.abdn.ac.uk/research/evaluation para más información sobre el corpus

evaluación de las soluciones implementadas. Una definición exhaustiva de su formato y características se puede ver en el apéndice B.

Figura 2.4: Ejemplo de situación presentada a los evaluadores en el proyecto TUNA

Métricas

Se presentan a continuación algunas de las métricas más comunes utilizadas en evaluación basada en corpus. Algunas de ellas se utilizarán más adelante para la evaluación de algunos de los sistemas presentados:

- **Coeficiente Dice.** Es una métrica de comparación de conjuntos. Para dos conjuntos de atributos A y B se calcularía siguiendo la Fórmula 2.1.

$$dice(A, B) = \frac{2 * |A \cap B|}{|A| + |B|} \tag{2.1}$$

- **Coeficiente MASI.** El coeficiente MASI (*Measuring agreement on set-valued items*) (Passonneau, 2006) es una adaptación del coeficiente Jaccard[2] que se decanta ligeramente en favor de la similitud cuando un conjunto es un subconjunto de otro. Se calcula usando la Fórmula 2.2

[2]El coeficiente de Jaccard mide la similitud entre conjuntos, y está definido como el tamaño de la intersección de los conjuntos divido entre el tamaño de la unión de ambos.

$$MASI(A,B) = \delta * \frac{|A \cap B|}{|A \cup B|} \tag{2.2}$$

donde δ es un coeficiente definido como sigue:

$$\delta = \begin{cases} 0 & \text{si } A \cap B = \emptyset \\ 1 & \text{si } A = B \\ \frac{2}{3} & \text{si } A \subset B \text{ o } B \subset A \\ \frac{1}{3} & \text{en otro caso} \end{cases} \tag{2.3}$$

- **Distancia de Edición (*String-edit distance*).** Es la clásica métrica de distancia Levenshtein[3] usada para comparar las diferencias entre dos cadenas de texto como el mínimo número de inserciones, borrados y/o sustituciones de unas palabras por otras para transformar una de las cadenas en la otra. El coste de la inserción o el borrado es de 1 mientras que el de las sustituciones es de 2. Si dos cadenas son idénticas esta métrica devuelve un 0. Si no, el valor devuelto dependerá de la longitud de las cadenas y el número de operaciones requeridas para la transformación.

- **BLEU-x.** Es una medida de comparación de cadenas basada en n-gramas, y fue originalmente propuesta por Papineni et al. (2002) para la evaluación de sistemas de traducción automática. Mide un sistema basándose en la proporción de n-gramas de longitud x o menos que comparte con las opciones de referencia.

- **NIST.** Es una versión de BLEU, pero donde BLEU asigna el mismo peso a todos los n-gramas, NIST da más importancia a los menos frecuentes (y por tanto más informativos). Al igual que BLEU, se trata de una medida agregada.

2.3.2. Evaluación Orientada a Tarea

Recordemos que la evaluación orientada a tarea consiste en comprobar si la salida de un sistema de generación cumple su objetivo. Para ello se puede recurrir tanto a evaluadores humanos que validen los resultados obtenidos por el sistema, como a métricas que pueden ser calculadas automáticamente. Aunque el campo de la Generación de Lenguaje Natural no tiene una gran tradición de evaluación, en los últimos años están surgiendo cada vez más iniciativas que pretenden evaluar de diversas formas aspectos muy específicos del proceso de generación de lenguaje natural.

[3]La distancia Levenshtein entre dos cadenas se calcula como el número mínimo de operaciones necesarias para transformar una cadena en la otra, donde una operación es una inserción, un borrado o una sustitución de un único caracter.

El Desafío GIVE (*First NLG Challenge on Generating Instructions in Virtual Environments*)[4] pretende evaluar la generación de lenguaje natural para una aplicación muy concreta: guiar a un usuario en un entorno 3D para la consecución de un objetivo concreto. El usuario no tiene información de dónde está o qué debe hacer en el entorno, y los sistemas participantes son responsables de guiar mediante órdenes en lenguaje natural al usuario. En el periodo de evaluación los usuarios probarán los diversos sistemas anónimamente creándose un log para cada ejecución a partir de los cuales se realizará la evaluación de los sistemas.

La identificación de referentes a partir de expresiones de referencia generadas automáticamente ha sido otra tarea que también ha sido objeto de evaluación en los últimos años. Desde 2007 se han organizado tres competiciones para la generación de expresiones de referencia sobre los datos del corpus TUNA. El objetivo principal era evaluar la generación de expresiones de referencia que permitieran distinguir referentes objetivo de otros que les rodearan, y estudiar el comportamiento de evaluadores humanos frente a las referencias generadas. Varios de los algoritmos propuestos en este trabajo han sido presentado a estas competiciones, por lo que las describiremos con más detalle en el apartado siguiente.

2.3.3. Evaluación Competitiva para Generación de Expresiones de Referencia

En los últimos años ha surgido dentro de la comunidad GLN la necesidad de obtener resultados que pudieran ser evaluados y comparados. Esto presenta cierto contraste con la investigación realizada hasta el momento, orientada a la generación de lenguaje para distintos dominios, pero pocas veces evaluada exhaustivamente. La comunidad tomó en 2007[5] la decisión de organizar una tarea de evaluación competitiva piloto centrándose en la generación de expresiones de referencia. Ésta fue la tarea elegida dada la amplia investigación que ha sido realizada sobre ella en las últimas décadas con el consiguiente consenso sobre cuáles son los problemas a resolver y su alcance.

Desde entonces se han organizado tres ediciones de este tipo de competiciones sobre la tarea de la generación de expresiones de referencia, pero orientándolas a distintos aspectos de la misma, y realizando distintos tipos de evaluación.

Como base para la definición, desarrollo y evaluación de todas estas tareas se han usado los datos del corpus TUNA (sección 2.3.1). Este corpus contiene ejemplos de situaciones posibles en dos dominios (mobiliario y personas) en las que una entidad ha sido referenciada por evaluadores humanos

[4] http://homepages.inf.ed.ac.uk/v1akolle/proj/give/
[5] Para más información consultar http://www.ling.ohio-state.edu/nlgeval07/

de una manera determinada para que sea posible distinguirla del resto que la rodean.

Primera Tarea Competitiva: Selección de Atributos para Generación de Expresiones de Referencia

La primera de estas iniciativas fue *The First NLG Challenge on Attribute Selection for Generating Referring Expressions (ASGRE'07)*[6], que tuvo lugar en Copenhague en septiembre de 2007. Esta primera tarea de evaluación competitiva fue un éxito tanto por el alto número de equipos participantes como por la variedad y calidad de los trabajos presentados (Belz y Gatt, 2007). La selección de atributos fue la tarea escogida para esta iniciativa pionera ya que, aunque se puede encontrar mucha investigación en el campo de la generación de expresiones de referencia, el subproblema de la selección de atributos quizá sea la subtarea del proceso GLN más estudiada.

Los sistemas participantes debían implementar la tarea de generar un conjunto de atributos que identificara una entidad determinada, partiendo de una cierta representación de entrada que contenía varias entidades con las que ésta se podía confundir. El objetivo en este caso era seleccionar o bien cualquier conjunto de atributos que distinguieran a la entidad de las que le rodeaban, o bien el conjunto mínimo de atributos que la distinguían, o bien el conjunto de atributos que seleccionarían los seres humanos para referirse a ella. Cada uno de estos objetivos estaba representado por una métrica de evaluación diferente. Además se incluyeron también dos tareas más: una abierta para cualquier tipo de problema que los participantes quisieran plantear sobre los datos, y otra de evaluación para métricas de evaluación que los participantes pudieran proponer.

Las entradas definidas para la competición, extraídas del corpus TUNA, son representaciones simbólicas de dominios visuales en los que distintas descripciones fueron escogidas por evaluadores humanos. Cada fichero de entrada representa por tanto un dominio o situación diferente que contiene un conjunto de entidades y para cada una de ellas sus propiedades (pares atributo-valor según la representación de los datos en el corpus). Una de estas entidades es considerada como el referente objetivo, distinguido en los datos del corpus usando la etiqueta *target*. Las demás, también llamadas distractores (marcadas por la etiqueta *distractor*), son las entidades de las que el referente objetivo debe ser diferenciado. Un ejemplo simplificado de los datos que se pueden encontrar en un archivo de entrada podría ser el de la Tabla 2.1. En este ejemplo todas las entidades (`e1`, `e2` y `e3`) tienen el mismo conjunto de atributos (tipo, color, orientación, tamaño), pero con distintos valores según la representación visual original de las mismas.

[6] http://www.csd.abdn.ac.uk/research/evaluation/

		Type	**Colour**	**Orientation**	**Size**
e1	*target*	chair	red	right	small
e2	*distractor*	table	blue	left	large
e3	*distractor*	chair	green	right	small

Tabla 2.1: Ejemplo de dominio de entrada

La salida esperada de los sistemas es un subconjunto de los atributos del referente objetivo que permitan identificarle distinguiéndole del resto de entidades distractoras. En los datos del corpus estos conjuntos de atributos fueron extraídos de expresiones de referencia generadas por evaluadores humanos en forma de lenguaje natural. En el ejemplo de la Tabla 2.1 existen varios conjuntos de atributos que serían considerados válidos, como por ejemplo {`COLOUR:red`} o {`TYPE:chair, COLOUR:red`}. Sin embargo, descripciones como {`ORIENTATION:right, SIZE:small`} pueden referirse a más de un referente y no cumplen su función distintiva.

Los sistemas participantes en la tarea debían implementar algún tipo de algoritmo que les permitiera pasar de una entrada a una salida tal y como se acaban de describir. Los sistemas fueron evaluados según las salidas producidas teniendo en cuenta diversas métricas de evaluación.

El conjunto de datos del corpus TUNA se divide en tres subconjuntos diferentes para llevar a cabo la tarea:

1. Un conjunto de entrenamiento consistente en 468 elementos del corpus (dominio de entrada + descripción por evaluador humano), divididos en los dos dominios de trabajo: mobiliario y personas. Este conjunto de entrenamiento es el que los participantes deben utilizar para entrenar y poner a punto sus sistemas.

2. Un conjunto de desarrollo consistente en 156 elementos del corpus (dominio de entrada + descripción por evaluador humano), divididos también en mobiliario y personas. Estos datos son los que deben usar los participantes para autoevaluar sus sistemas según las métricas de evaluación de cada tarea.

3. Un conjunto de test consistente en 156 elementos del corpus (sólo con el dominio de entrada), divididos en los dominios de mobiliario y personas. Este conjunto de test no formaba parte inicialmente del corpus TUNA, y fue creado replicando el experimento inicial que se utilizó para formar el corpus. Para cada dominio de entrada se tienen como salida dos descripciones en lenguaje natural realizadas por dos individuos diferentes. A la hora de evaluar las descripciones producidas por los sistemas se llevará a cabo una media de las medidas de evaluación sobre ambas descripciones generadas por humanos.

Según la organización de los datos ya expuesta, los participantes se encargan de los valores de evaluación sobre los datos de entrenamiento y desarrollo, y los organizadores llevan a cabo las evaluaciones sobre los datos de test. En esta tarea las salidas de los sistemas participantes fueron evaluadas de acuerdo a cuatro dimensiones:

- **Identificación**.

  Las salidas del sistema son estudiadas para comprobar si el conjunto de atributos devuelto para una determinada situación distingue unívocamente al referente objetivo. En el ejemplo en la Tabla 2.1, el conjunto {`TYPE:chair, COLOUR:red`} distingue unívocamente al objetivo porque no hay ninguna entidad para la que estos pares atributo-valor sean ciertos. Como medida global, se considera la proporción de salidas del sistema que identifican con éxito a los referentes objetivos que les corresponden.

- **Minimalidad**.

  Una descripción mínima se define como el conjunto de atributos más pequeño posible que permite distinguir al referente objetivo. Por ejemplo, una descripción mínima de `e1` en el ejemplo que estamos viendo sería {`COLOUR:red`}, ya que es el único objeto de color rojo. Como medida global, se considera la proporción de salidas mínimas producidas por el sistema.

- **Concordancia sistema-evaluador humano**.

  Para cada entrada proporcionada a los sistemas, existe un conjunto de atributos que corresponden a los utilizados por los evaluadores humanos al generar la expresión de referencia para el referente objetivo. Se mide la similitud entre ambos conjuntos de atributos (humano vs. sistema), esperando obtener resultados diferentes a los proporcionados por las dos métricas anteriores, ya que los humanos suelen sobreespecificar o infraespecificar las referencias que utilizan por diversas razones. Por ejemplo, en el ejemplo mencionado un ser humano podría usar {`TYPE:table, COLOUR:blue`} para referirse a `e2` aunque {`TYPE:table`} es suficiente.

  Las salidas proporcionadas por los sistemas son comparadas con las de los evaluadores usando el coeficiente Dice (sección 2.3.1). Como medida global, se usa la media del coeficiente Dice obtenido por el sistema sobre el conjunto completo de entradas.

- **Evaluación basada en tarea**.

  Consiste en un experimento a pequeña escala con evaluadores humanos en el que se pretende comprobar cómo se comportan los humanos

ante las referencias generadas por los sistemas. Para ello, las salidas generadas son realizadas como descripciones en lenguaje natural reemplazando los atributos y tipos por palabras adecuadas y aplicando algunas reglas sintácticas simples. Estas referencias en lenguaje natural son mostradas a los evaluadores humanos, a los que se les pide que identifiquen la entidad a la que creen que se refieren dada una representación visual del dominio de entrada. Como medida global, se considerará la proporción de ocasiones en las que el lector identifica correctamente la entidad a la que se refiere la expresión de referencia, y cuánto tiempo necesitaba para identificarla.

Segunda Tarea Competitiva para Generación de Expresiones de Referencia

Tras el éxito de la primera tarea competitiva para la Generación de Expresiones de Referencia, se organizó una segunda unos meses después: el *Generation of Referring Expresions Challenge 2008*[7], celebrado dentro del *International Workshop on Natural Language Generation*, parte de ACL 2008, en Ohio. Mientras que la primera tarea se centraba en la selección de atributos para referencias definidas, en esta ocasión se amplió el alcance para incluir tanto selección de atributos como realización, al mismo tiempo que se introducía una nueva tarea sobre referencias en contexto a nombres de entidades o *named entities* utilizando un corpus diferente.

El objetivo de la tarea, basada de nuevo en el corpus TUNA, era generar expresiones de referencia finales para entidades en distintos contextos situacionales. Los resultados obtenidos por los diversos sistemas eran evaluados tanto automáticamente como utilizando evaluadores humanos, intentando valorar qué sistemas se aproximaban más a la manera humana de generar expresiones de referencia. Los resultados de la evaluación y los sistemas fueron presentados en el *International Workshop on Natural Language Generation (INLG08)* en Ohio (Gatt et al., 2008).

En esta edición se abrieron ocho subtareas distintas sobre los dos corpus existentes. Cinco de estas subtareas trabajaban sobre el corpus TUNA, siendo tres de ellas específicas, otra abierta a cualquier trabajo sobre los datos del corpus, y una quinta sobre métodos de evaluación sobre las tres primeras. Las tres subtareas restantes trabajaban sobre el corpus GREC, incluyendo una tarea específica, otra abierta, y otra más sobre evaluación. En este trabajo nos centraremos en las subtareas relativas al corpus TUNA.

- **Subtarea de Selección de Atributos (TUNA-AS)**.

  Esta tarea es equivalente a la que se llevó a cabo en la edición anterior (sección 2.3.3): encontrar para distintas representaciones de dominio

[7] http://www.nltg.brighton.ac.uk/research/reg08/

conjuntos de atributos apropiados, pero utilizando datos nuevos para la fase de test. Se introdujo esta tarea para permitir a los participantes de la competición anterior mejorar sus sistemas, o participar a otros que no pudieron hacerlo en el 2007.

- **Subtarea de Realización (TUNA-R)**.

  En esta subtarea los participantes debían crear sistemas que fueran capaces de trasformar conjuntos de atributos que representan una referencia a una entidad en descripciones en lenguaje natural. Por ejemplo, {`TYPE:fan, COLOUR:red, SIZE:large`} podría dar lugar a *the large red fan* o *big purple windmill*, entre otras posibilidades. De cara a la evaluación se podía optar por intentar maximizar la similitud con las descripciones generadas por los evaluadores humanos para el corpus, o maximizar la optimalidad desde el punto de vista de la comprensión humana y la rápida identificación de los referentes.

- **Subtarea de Generación de Expresiones de Referencia Completa (TUNA-REG)**.

  Esta subtarea combinaba las subtareas TUNA-AS y TUNA-R, de manera que los sistemas participantes tomaban como entradas representaciones de los dominios y generaban como salida descripciones en lenguaje natural que describían al referente objetivo. Como en la tarea anterior, se podía optimizar esta tarea de diferentes maneras según los distintos criterios de evaluación existentes.

- **Subtarea Abierta y Subtarea sobre Métodos de Evaluación**.

  La subtarea abierta pretendía evitar una definición de tareas demasiado estrecha dada la variedad del campo de investigación de la generación de referencias. Por su parte, la subtarea sobre métodos de evaluación permitía a los investigadores desarrollar nuevos métodos de evaluación que pudieran ser también adecuados para cualquiera de las tareas. La idea es que estas subtareas y métodos de evaluación alternativos pudieran ser incluidos en eventos posteriores. Por ejemplo, la métrica MASI incluida en la evaluación de este año fue propuesta por Advaith Siddharthan, de la Universidad de Cambridge, el año anterior (Siddharthan y Copestake, 2007).

Como ya se ha comentado se usaron tres conjuntos de datos para la evaluación. Esta vez los datos de entrenamiento fueron un total de 624, 156 los datos de desarrollo y 112 los de test. Como en la edición anterior, los participantes se encargaban de los valores de evaluación sobre los datos de entrenamiento y desarrollo, y los organizadores llevaban a cabo las evaluaciones sobre los datos de test. Se usaron en esta ocasión distintos métodos de

evaluación, tanto extrínsecos como intrínsecos, calculados automáticamente o evaluados por humanos. Las evaluaciones intrínsecas se encargaban de medir las propiedades de un sistema de manera aislada. Por el contrario, las extrínsecas medían el efecto del sistema evaluado sobre algo externo a él, bien fuera su efecto sobre seres humanos al realizar una tarea como el valor añadido que pudiera proporcionar a una aplicación.

- **Evaluación Extrínseca**

  Para las tareas TUNA-R y TUNA-REG se usaron experimentos basados en tarea como la competición ASGRE. Sin embargo, en esta ocasión las expresiones de referencia y las imágenes de los referentes se mostraban a los evaluadores en dos pasos, de manera que los sujetos leían primero la expresión de referencia de la entidad que debían encontrar, y posteriormente veían las imágenes de la situación correspondiente. Así era posible medir en primer lugar la velocidad de lectura (como una estimación de la facilidad de comprensión), y posteriormente la velocidad de identificación y la precisión de la identificación (ambas como estimación de la claridad de la referencia). Para este proceso se usó software de precisión como DMDX y Time-DX (Forster y Forster, 2003).

- **Evaluación Intrínseca**

  Se pretendía evaluar la similitud de las salidas obtenidas con las referencias generadas por evaluadores humanos usando distintas métricas automáticas. Para la subtarea TUNA-AS se recurrió de nuevo al coeficiente Dice, y a la medida MASI (*Measuring Agreement on Set-valued Items*). Para las tareas TUNA-R y TUNA-REG se usaron medidas de precisión, de distancia de edición de cadenas, BLEU y NIST. Para que fuera posible comparar los resultados obtenidos con el desafío ASGRE, se mantuvieron además las medidas de minimalidad y unicidad de los conjuntos de atributos para la tarea TUNA-AS.

Tercera Tarea Competitiva para Generación de Expresiones de Referencia

La tercera tarea competitiva para la Generación de Expresiones de Referencia forma parte de los Desafíos de Generación 2009, parte del *European Workshop on Natural Language Generation 2009*. Estos desafíos han sido organizados para proporcionar un foro común para una serie de varias tareas competitivas sobre Generación de Lenguaje Natural[8]. La tarea de generación de expresiones de referencia sobre el corpus TUNA en 2009 (*Tuna Referring Expression Generation Task 2009*) es la tercera y última tarea competitiva

[8] http://www.nltg.brighton.ac.uk/research/genchal09/

utilizando como datos las expresiones de referencia del corpus TUNA. Se replica una de las tres tareas del desafío REG 2008, más concretamente la subtarea TUNA-REG que cubría el proceso completo de generación de expresiones de referencia: selección de atributos + realización. Se usan para ello los mismos datos de test, de manera que sea posible comparar directamente con los resultados obtenidos el año anterior.

Esta competición es una réplica de la subtarea TUNA-REG que formó parte del desafío REG en 2008. Como en aquella edición, habrá una amplia variedad de métodos de evaluación para comparar los sistemas, y los participantes pueden por tanto decidir qué criterio de evaluación desean optimizar en sus sistemas.

Siguiendo la definición de problema estándar de este tipo de tareas, la entrada será la representación de un dominio del corpus TUNA, consistente en un conjunto de objetos y sus propiedades, siendo uno de los objetos escogido como el referente objetivo.

Para la salida los sistemas participantes deberán crear una correspondencia entre la representación del dominio del corpus y descripción en lenguaje natural que describe al referente objetivo.

Al igual que en ediciones anteriores los datos se dividen en tres grupos: entrenamiento (593 elementos), desarrollo (150 elementos) y test (112 elementos). Los participantes se encargarán de calcular los resultados de evaluación sobre los datos de entrenamiento y desarrollo, mientras que los organizadores llevarán a cabo las evaluaciones correspondientes sobre los datos de test. Se usarán de nuevo diversos métodos de evaluación, tanto intrínsecos como extrínsecos, calculados automáticamente y evaluados usando evaluadores humanos.

- **Evaluación intrínseca**.

  Al igual que para las subtareas de realización de la edición anterior, se utilizaron como medidas de evaluación intrínseca automática las de precisión, distancia de edición de cadenas, BLEU y NIST. Como evaluación intrínseca humana se incluyeron por primera vez medidas de *adecuación* (corrección del contenido de la referencia) y *fluidez* (naturalidad del lenguaje utilizado). Para ello se realizó un nuevo experimento en el que evaluadores humanos tenían que dar valores de adecuación y fluidez para cada par dominio-referencia.

- **Evaluación extrínseca**.

  Al igual que en otras ediciones, se pretendía realizar una evaluación basada en tarea para comprobar cómo los evaluadores humanos se comportan a la hora de utilizar las referencias generadas para identificar entidades de los dominios. Este año se estaba además barajando la

posibilidad de realizar el experimento con las referencias leídas en voz alta por una voz sintética en lugar de que fueran leídas por los propios evaluadores.

2.4. TAP, un *Pipeline* para Aplicaciones de Generación de Lenguaje Natural

TAP (*Text Arranging Pipeline*) (Gervas, 2007) es una arquitectura software para la generación automática de texto. TAP está formado por una serie de interfaces que definen una funcionalidad genérica para un *pipeline* de tareas orientadas a la generación de lenguaje natural, desde una entrada conceptual inicial hasta la realización superficial final en forma de texto, con etapas intermedias de planificación de contenido y planificación de oraciones. A partir de la entrada conceptual que se va a transformar en texto, los diversos módulos de TAP trabajan sobre representaciones intermedias para almacenar resultados parciales de manera que progresivamente se filtra, agrupa y enriquece la entrada hasta conseguir estructuras cada vez más cercanas al lenguaje natural tanto en estructura como en contenido.

La filosofía seguida en el diseño de TAP es identificar la naturaleza estructural genérica de la información que requiere de procesamiento en las diferentes fases de la generación de lenguaje natural, y proporcionar interfaces para acceder a esta estructura. La arquitectura de TAP ha sido específicamente diseñada para permitir el desarrollo de un conjunto reutilizable de componentes software capaces de resolver tareas básicas de GLN, trabajando con información codificada en formatos estándar, y proporcionando textos adecuados a diferentes tareas en diferentes dominios.

La instancia particular de la arquitectura que hemos usado en este trabajo tiene tres módulos básicos: el Planificador de Contenido, el Planificador de Oraciones, y un Realizador Superficial. Estos módulos están organizados como un pipeline donde la información pasa secuencialmente por los módulos que se encargan de las distintas tareas del proceso. Dentro de ellos nos referiremos en este trabajo especialmente a los módulos de Determinación de Contenido y Generación de Expresiones de Referencia.

Como realizador superficial se ha utilizado SurReal (Gervás, 2006), que proporciona una implementación en Java de los mecanismos de realización de FUF descritos en (Elhadad, 1993). El módulo opera sobre estructuras de datos que corresponden a descripciones funcionales apoyándose en una gramática que es unificada con la entrada. Esta gramática sigue las convenciones de la gramática SURGE (Elhadad y Robin, 1996). Actualmente SurReal tiene menos cobertura que el original, pero la suficiente para tratar el tipo de realizaciones que se abordan en este trabajo.

2.4.1. Determinación de Contenido

La implementación actual del módulo de determinación de contenido de TAP recibe como entrada un conjunto de referentes, un conjunto de propiedades, un conjunto de relaciones y un conjunto de eventos. Las propiedades pueden aplicarse tanto a referentes (en cuyo caso suelen acabar representadas como adjetivos) como a eventos (en cuyo caso acaban representadas como adverbios). Las relaciones pueden establecerse indistintamente entre cualesquiera de los cuatro elementos posibles (referentes, propiedades, relaciones o eventos), de manera que permiten tanto la representación de modificadores de frases nominales, complementos de oraciones, o de relaciones de causalidad.

Dentro del módulo de determinación de contenido, la tarea básica que se lleva a cabo es agrupar los eventos en secuencias de eventos contiguos en el tiempo y que ocurren en el mismo lugar. Estos bloques se consideran como una primera aproximación al concepto de escena, y se utilizan para establecer una planificación inicial del discurso. Las escenas se ordenan según el orden cronológico de los primeros eventos de cada una. Esta primera aproximación del discurso a generar debe enriquecerse con información adicional relativa a las descripciones de los referentes que participan en cada evento. La información que se introduzca en este punto aparecerá en el texto final mencionada explícitamente como parte del discurso en oraciones independientes como *La princesa es rubia*.

2.4.2. Generación de Expresiones de Referencia

La versión de partida del módulo de generación de expresiones de referencia aplicaba una implementación sencilla del algoritmo de Reiter y Dale (sección 2.2.5) que permitía seleccionar el conjunto de atributos (propiedades y relaciones) para un referente que lo identificaban con respecto a posibles menciones previas del mismo en el discurso precedente. Esta tarea se apoyaba en el conjunto de propiedades y relaciones que se habían establecido como válidas para ese referente durante la determinación de contenido. La información que se seleccionaba en este punto aparecería en el texto final mencionada como expresiones de referencia que participaban en oraciones que describían eventos (*The blonde princess went to the castle*).

2.5. Recursos Léxico-Conceptuales para Generación de Lenguaje Natural

Los sistemas basados en conocimiento tienen que resolver dos problemas fundamentales: la representación del conocimiento y su procesamiento para utilizarlo en la resolución de una tarea concreta. Estos sistemas tienen que gestionar su base de conocimiento de una forma coherente, organizando

su contenido según algún criterio razonable. Uno de esos posibles criterios es el uso de ontologías, un modelo de conceptualización que se explica a continuación.

La Generación de Lenguaje Natural es sin duda un campo muy necesitado de distintos tipos de conocimiento a la hora de realizar su función. Por ello, en muchas ocasiones será necesario recurrir a fuentes de conocimiento como ontologías y bases de datos léxicas para cubrir algunas de las necesidades tanto conceptuales como léxicas del proceso de generación.

Una ONTOLOGÍA es una representación formal del conocimiento donde los conceptos, las relaciones y las restricciones conceptuales son definidas mediante algún tipo de formalismo para un determinado dominio. Como mecanismo de inferencia las ontologías generalmente recurren a la herencia, que implica una economía en la codificación de la información: los conceptos superiores transmiten sus características a los conceptos inferiores. Las ontologías no se limitan a clasificar categóricamente un conjunto de conceptos, sino que definen como son las relaciones y los axiomas lógicos que conectan unos con otros, generalmente en términos de los individuos de las bases de conocimiento que se añadirán después para formar el modelo completo y que podrán o no ser aceptados como instancias válidas de dicha conceptualización previa.

El *Ontology Web Language* (OWL) (Bechhofer et al., 2004) es un lenguaje de representación para la semántica declarativa propia de las ontologías y las bases de conocimiento de un sistema basado en conocimiento. El año 2002 este lenguaje se propuso como un estándar internacional para la Web Semántica por parte del Consorcio W3C. Muchas de las ontologías disponibles se encuentran codificadas en este lenguaje.

En el caso de la Generación de Lenguaje Natural, una ontología puede ser uno de los módulos asociados a un sistema de conocimiento donde su función será la de apoyo semántico para distintos puntos del proceso de generación.

Una BASE DE DATOS LÉXICA (Arano, 2003) es un sistema de almacenamiento de información lingüística organizada según un determinado modelo de datos que posibilita el almacenamiento, recuperación y modificación de los mismos. Una base de datos léxica tiene la función de responder a consultas sobre los datos que contiene, ya sea desde prestaciones propias o a partir de aplicaciones externas, permitiendo la reutilización de la información contenida. El comportamiento de una base de datos léxica es pasivo, pues las operaciones sobre sus datos son realizadas por aplicaciones que deben ser iniciadas explícitamente.

Las bases de datos léxicas son utilizadas en la Lingüística como fuentes de información léxica a reutilizar por otros recursos, por ejemplo un lexicón computacional o una base de datos terminológica.

Como herramienta para trabajar automáticamente con ontologías hemos usado DLModel (Peinado, 2008) para acceder a las ontologías en OWL

que usaremos más adelante. DLModel es un conjunto de herramientas para acceder a la información de ontologías escritas en OWL.

A continuación comentaremos algunas ontologías que han sido utilizadas para diversas cuestiones dentro del procesamiento de lenguaje natural. Después estudiaremos con más profundidad WordNet, que es el recurso léxico que ha sido utilizado a lo largo de este trabajo.

2.5.1. Uso de Ontologías para Procesamiento Lingüístico

WordNet (Miller, 1995) es de lejos la base de datos más grande y rica de todos los recursos que se indexan por conceptos. WordNet es un sistema léxico on-line de referencia cuyo diseño está inspirado en las teorías psicolingüísticas actuales sobre la memoria léxica. Su léxico organiza la información en términos de significados de las palabras, en lugar de a partir de las formas de las mismas. Sustantivos, verbos y adjetivos en inglés se organizan en conjuntos de sinónimos, cada uno de ellos en representación de un concepto léxico subyacente. Estos conjuntos de sinónimos o *synsets* están unidos entre sí por relaciones semánticas como la sinonimia o hiponimia. Por su organización utilizando sinónimos utilizaremos WordNet más adelante en este trabajo. Por ello la estudiaremos con detalle en la sección 2.5.2.

El proyecto de FrameNet (Baker et al., 1998) consiste en la creación de un recurso léxico on-line para el inglés basado en la semántica de marcos y apoyado sobre evidencias obtenidas a partir de corpus. El objetivo es codificar el conocimiento semántico de forma que resulte legible por una máquina a partir de la anotación semi-automática de frases de ejemplo. La intuición de los lexicógrafos está limitada y guiada por los resultados de la investigación basada en corpus, utilizando herramientas de software de alto rendimiento. El principal producto de este trabajo, la base de datos léxica FrameNet, contiene actualmente más de 10.000 unidades léxicas en más de 825 marcos semánticos y ejemplificadas en más de 135.000 sentencias anotadas.

En Mikrokosmos (Mahesh y Nirenburg, 1995), la mayor parte de la información en lenguaje natural se almacena en la ontología. Cada concepto está representado por una estructura de marco que permite numerosos vínculos entre los conceptos. Por ejemplo, incluye la habitual relación *isa* y relaciones de pertenencia a conjuntos. La ontología también incluye restricciones sobre roles en los casos. En cambio, el léxico sólo contiene la información necesaria para realizar un concepto en un determinado idioma, información habitual sobre morfología y sintaxis, pero la mayor parte de la semántica se define en términos de un concepto de la ontología. Por lo tanto, la semántica para el léxico de entrada podría ser una asignación directa a los conceptos asociados. La ontología y el léxico de Mikrokosmos son paralelos tanto en las representaciones como en su desarrollo. Los significados de las palabras están representadas en parte en el léxico y en parte en la ontología. En prin-

cipio, la separación entre ontología y léxico es el siguiente: los significados se almacenan en el primero, y la información específica de cada idioma en el segundo. En una situación multilingüe, no es fácil, sin embargo, determinar este límite.

La *Suggested Upper Merged Ontology* (SUMO) (Niles y Pease, 2001) es un *framework* diseñado específicamente para proporcionar una base para ontologías de dominio más específicas. Combina una serie de ontologías de alto nivel para conseguir una amplia cobertura conceptual y una fuerte base de conceptos de semiótica y lingüística. El objetivo era elaborar una ontología estándar de alto nivel que fomente la interoperabilidad de los datos, la búsqueda y recuperación de información, la capacidad automática de inferencia, y el procesamiento del lenguaje natural.

El *Generalized Upper Model* (GUM) (Bateman et al., 1995) es un descendiente del *Penman Upper Model* (Bateman et al., 1990), que a su vez era una ontología lingüísticamente motivada para mediar entre conocimiento del dominio y sistemas de generación de lenguaje natural. Se trata de una ontología general de motivación lingüística e independiente del dominio destinada a la organización de la información para que sea expresada en lenguaje natural. Las categorías de la ontología fuerzan un estilo consistente de modelización sobre cualquier dominio que es también apropiado su traducción a lenguaje natural.

Se han realizado varios esfuerzos para vincular las ontologías lingüísticas con las lógicas descriptivas. Algunos de ellos han consistido en el desarrollo de la versión codificada en OWL de las ontologías (como es el caso de GUM y Mikrokosmos), o proporcionar la manera de conectar una ontología OWL con un recurso existente. En este sentido, Protégé (Crubézy et al., 2005) proporciona varios *plug-ins* para la visualización de ontologías desarrolladas y la importación de bases de datos de diferentes formalismos a una ontología. Dos de estos *plug-ins* están relacionados con el Procesamiento del Lenguaje Natural: la pestaña OntoLing, y la pestaña WordNet. La pestaña OntoLing ofrece funcionalidades para la navegación sobre recursos lingüísticos (tesauros, diccionarios, etc.), el enriquecimiento de ontologías con elementos de estos recursos lingüísticos, y la creación de nuevas ontologías a partir de recursos lingüísticos previamente existentes. La pestaña de WordNet permite a los usuarios buscar on-line en el léxico de WordNet para anotar ontologías con información de la base de datos (tales como términos, identificadores de conceptos, sinónimos, relaciones, etc.), y para crear nuevas clases e instancias en la ontología con contenido de WordNet.

2.5.2. WordNet

WordNet (Miller, 1995) es una base de datos léxica que ha sido diseñada teniendo en cuenta las teorías psicolingüísticas actuales sobre cómo funciona la memoria léxica humana. Implementada inicialmente en inglés, los sustan-

tivos, verbos y adjetivos se organizan en conjuntos de sinónimos, cada uno representando un único concepto léxico. Estos conjuntos de sinónimos se relacionan de diferentes maneras formando la estructura básica de WordNet.

En 1985 un grupo de psicólogos y lingüistas de la Universidad de Princeton decidieron llevar a cabo la implementación de una base de datos léxica según las líneas sugeridas por las investigaciones en el campo (Miller, 1985). La idea inicial era proporcionar una ayuda para realizar búsquedas conceptuales en diccionarios, en lugar de la clásica búsqueda alfabética, en el marco de un diccionario on-line convencional. Según el trabajo avanzó, fue necesaria una reformulación más ambiciosa de sus principios y metas. Como resultado surgió WordNet. Dado que instancia hipótesis basadas en la investigación psicolingüística, se puede decir que WordNet es un diccionario basado en principios psicolingüísticos.

La diferencia más obvia entre WordNet y un diccionario convencional es que WordNet divide el lexicón en cuatro categorías: sustantivos, verbos, adjetivos y adverbios. Al realizar una categorización sintáctica, WordNet se encuentra con un cierto grado de redundancia que otros diccionarios intentan evitar (palabras como *back* pertenecerán a varias categorías sintácticas). Sin embargo, la ventaja principal de este tipo de organización es que las diferencias fundamentales en la organización semántica de estas categorías puede verse claramente y ser explotada sistemáticamente. De esta forma, los nombres en WordNet se organizan como una jerarquía semántica, los verbos como una red vinculada mediante relaciones, y los adjetivos y adverbios como espacios n-dimensionales. Cada una de estas estructuras léxicas refleja una forma diferente de categorización, ya que posibles intentos de utilizar un único principio organizativo para todas las estructuras llevaría probablemente a una representación distorsionada de la complejidad psicológica del conocimiento léxico.

La característica más ambiciosa de WordNet es su intento de organizar la información en términos del significado de las palabras más que según la forma de las mismas. En este aspecto WordNet se asemeja más a un tesauro que a un diccionario, aunque sin el principal de sus problemas: las entradas redundantes. Si una palabra W_x y una palabra W_y son sinónimos, ambas deben ser introducidas en el tesauro dos veces, una alfabéticamente bajo W_x y otra alfabéticamente bajo W_y. En WordNet no nos encontraremos con este problema.

Por ésta y otras razones, WordNet es más que un tesauro on-line, y para poder apreciarlo es necesario estudiar y entender su diseño básico (Miller y Fellbaum, 1991).

La estructura de WordNet

Una palabra es una asociación por convenio entre un concepto lexicalizado y una expresión que juega un papel sintáctico. Dado que la palabra *pala-*

bra se usa comúnmente para referirse tanto a la expresión como a su concepto asociado, se evitarán confusiones si llegamos al convenio de utilizar *forma* para referirnos a la expresión física y *significado* para referirnos al concepto que una forma puede expresar. Por tanto, el punto de partida será buscar las correspondencias entre formas y significados que correspondan (Miller, 1986), partiendo de la premisa inicial de que categorías sintácticas diferentes podrán tener diferentes tipos de correspondencias.

Significados	Formas				
	F_1	F_2	F_3	...	F_n
S_1	$E_{1,1}$	$E_{1,2}$			
S_2		$E_{2,2}$			
S_3			$E_{3,3}$		
...				...	
S_m					$E_{m,n}$

Tabla 2.2: El concepto de matriz léxica en WordNet

En la Tabla 2.2 se puede observar de una forma simple la idea de matriz léxica en la que se basa WordNet. Las formas de las palabras se listan como cabeceras de las columnas, y los significados como cabeceras de las filas. Una entrada en una celda de la matriz implica que la forma de esta columna puede ser usada (dentro de un contexto apropiado) para expresar el significado en esa fila. Por ejemplo, la entrada $E_{1,1}$ implica que la forma F_1 puede ser usada para expresar el significado S_1. Si hay dos entradas en la misma columna, la forma de la palabra es POLISÉMICA (tiene varios significados); si hay dos entradas en la misma fila, las dos formas de la palabra son SINÓNIMAS (tienen el mismo significado) respecto a cierto contexto.

Las correspondencias entre formas y significados son relaciones N a M, donde algunas formas tienen varios significados, y algunos significados pueden ser expresados mediante varias formas. Estos problemas típicos de la lexicografía, la polisemia y la sinonimia, pueden ser vistos como aspectos complementarios de esta correspondencia. Ambos son problemas que surgen a la hora de acceder al conocimiento en el *lexicón mental*: un oyente o lector que reconoce una forma debe tener en cuenta su polisemia, y un hablante o escritor que intenta expresar un significado debe decidir entre sinónimos.

Para simular computacionalmente una matriz léxica es necesario tener alguna forma de representar tanto las formas como los significados de las palabras. Representar la forma de una palabra no supone demasiado problema, pero los significados son otra cuestión. Se puede considerar que la decisión de cómo representar estos conceptos o significados se reduce a utilizar una teoría constructiva o una teoría diferencial para hacerlo. En una teoría constructiva, la representación debe contener suficiente información para permitir una construcción precisa del concepto. Sin embargo, en una

teoría diferencial los conceptos se pueden representar mediante un conjunto de símbolos que permitan distinguirlos de los demás conceptos del dominio.

En el caso de WordNet, la teoría diferencial es suficiente para la construcción de las correspondencias necesarias. De esta forma, el significado S_1 de la Tabla 2.2 se puede representar mediante la lista de formas que pueden ser usadas para expresarlo: $\{F_1, F_2, ...\}$. Estos conjuntos de sinónimos se denominan SYNSETS, y aparecen representados entre llaves ($\{\}$). Los *synsets* no explican qué es un concepto, sino que simplemente quieren decir que un concepto existe, y se asume que cualquier persona que conozca la lengua inglesa ha adquirido anteriormente los conceptos y será capaz de reconocerlos a partir de las palabras expresadas en el *synset*.

Por tanto, una matriz léxica se puede representar a efectos teóricos como una correspondencia entre palabras escritas y conjuntos de sinónimos o *synsets*. Dado que la lengua inglesa es rica en sinónimos, los *synsets* son generalmente suficiente para distinguir entre conceptos. Sin embargo, hay ocasiones en que no existe un sinónimo apropiado y se recurre a una pequeña GLOSA (*gloss*) para resolver la polisemia. La glosa no está pensada para que alguien pueda construir un nuevo concepto si no está familiarizado con él, sino que tiene como propósito de que el usuario de WordNet pueda diferenciar entre este sentido y los demás cuando no hay sinónimos adecuados para realizar esta diferenciación.

Las relaciones semánticas

WordNet está organizada utilizando relaciones semánticas. Teniendo en cuenta que una relación semántica es una relación entre significados, y que un significado se puede representar con *synsets*, resulta natural pensar en las relaciones semánticas como punteros entre *synsets*. Una característica importante de estas relaciones es que son recíprocas: si entre los significados $\{x_1, x_2, ...\}$ e $\{y_1, y_2, ...\}$ existe una relación semántica R, entonces hay también una relación semántica R´ entre $\{y_1, y_2, ...\}$ y $\{x_1, x_2, ...\}$. Para evitar confusiones innecesarias ambas relaciones se denominarán de la misma forma R.

Siguen a continuación algunas de las relaciones semánticas más importantes en WordNet.

Sinonimia. Según lo comentado hasta ahora, parece obvio que la relación más importante de WordNet es la similitud de significado o SINONIMIA, ya que la capacidad de juzgar esta relación entre formas de palabras es un prerrequisito para la representación de significados en la matriz léxica. Se considera que dos expresiones son sinónimas si la sustitución de una por la otra no cambia el significado de la frase en que se haga la sustitución. Según esta definición, los sinónimos verdaderos son raros, si es que existen en absoluto. Una versión más débil de esta definición

hace que los sinónimos lo sean relativos a un contexto: dos expresiones son sinónimos en un contexto lingüístico C si la sustitución de una por la otra en C no cambia el significado de la frase correspondiente.

Es necesario resaltar que la definición de sinonimia en términos de sustituibilidad hace necesaria la partición de WordNet en sustantivos, verbos, adjetivos y adverbios. Es decir, si los conceptos se representan por *synsets*, y si los sinónimos deben ser intercambiables, entonces palabras de diferentes categorías sintácticas no podrán ser sinónimos porque no son intercambiables.

Antonimia. Otra relación familiar es la ANTONIMIA, que resulta ser sorprendentemente difícil de definir. El antónimo de una palabra x es en ocasiones $notx$, pero no siempre. Por ejemplo, *rico* y *pobre* son antónimos, pero no es lo mismo decir que alguien *no es rico* que decir que *es pobre*. La antonimia, que en principio parece ser una simpre relación simétrica, es en realidad bastante compleja.

La antonimia es una relación léxica entre formas de palabras, no una relación semántica entre significados. Por ejemplo, los significados $\{rise, ascend\}$ y $\{fall, descend\}$ pueden ser conceptualmente contrarios, pero no son antónimos. $[rise/fall]$ son antónimos, al igual que $[ascend/descend]$, pero mucha gente duda cuando es preguntada sobre si $rise$ y $descend$, o $ascend$ y $fall$ son antónimos. Este tipo de hechos hace necesario distinguir entre relaciones semánticas entre formas de palabras y relaciones semánticas entre significados de palabras. La antonimia proporciona un principio de organización básico para los adjetivos y adverbios en WordNet.

Hiponimia. Al contrario que la sinonimia y antonimia, que son relaciones léxicas entre formas de palabras, la hiponimia/hiperonimia es una relación semántica entre significados. Esta relación también es denominada en ocasiones subordinación/superordinación, o relación *isa*. Un concepto representado por el *synset* $\{x, x', ...\}$ se dice que es HIPÓNIMO del concepto representado por el *synset* $\{y, y', ...\}$ si se pueden aceptar construcciones como x *is a (kind of)* y. Esta relación se puede representar incluyendo en $\{x, x', ...\}$ un puntero a su hiperónimo, e incluyendo en $\{y, y', ...\}$ punteros a sus hipónimos. La HIPERONIMIA es la relación contraria.

La hiponimia es transitiva y asimétrica, y dado que generalmente hay sólo un hiperónimo para un concepto, se genera una estructura semántica en forma de jerarquía. De esta forma, un hipónimo hereda todas las características del concepto más genérico de la jerarquía, y añade al menos una característica que lo distingue de su hiperónimo y del resto de hipónimos de este hiperónimo. Esta convención constituye el principio central de la organización de los nombres en WordNet.

Ambigüedad en WordNet

Como ya se ha dicho, WordNet no está organizada de acuerdo a palabras, sino de acuerdo a conceptos. Debido a fenómenos lingüísticos como la polisemia o la sinonimia, existe un mapeamiento muchos a muchos que relaciona los conceptos y las palabras. Esto produce el importante problema de la Desambigüación Semántica (WSD) (Ide y Veroni, 1998), que ha sido estudiado por numerosos investigadores. Existen distintas aproximaciones a este problema, desde una perspectiva dependiente del conocimiento, hasta un simple recuento estadístico de palabras en un corpus.

En este punto, WordNet proporciona cierta ayuda: el campo *tag count* para los *synsets*. Este campo permite encontrar, dentro de un *synset*, qué nombres son más comunes en un corpus genérico (en este caso, el Brown Corpus (Nelson y Kucera, 1967)). Sin embargo, esta métrica es muy simple y resulta ser poco adecuada en muchos casos.

2.6. Figuras Retóricas basadas en Similitudes entre Dominios

Desde el punto de vista lingüístico, Carrillo (2005) habla de efecto estilístico a través de estrategias comunicativas, donde el estilo supone dos estrategias retóricas íntimamente relacionadas entre sí: los recursos lingüísticos elegidos por el hablante o escritor, y el efecto estimulado en el oyente o lector. Ambas estrategias tienen como objetivo lograr una efectividad comunicativa.

Kinneavy (1980) señala que el estilo es un asunto de *retórica*, y recuerda que el tratamiento de estilo en la retórica clásica se hace mediante cuatro virtudes de estilo, encontradas en Aristóteles y traducidas como: *claridad*, *dignidad* (o *elegancia*), *propiedad*, y *corrección*. La *claridad* tiene que ver con la realidad o el tema del que se está hablando. La *propiedad* tiene que ver con la adaptación al estilo de un particular auditorio implicado. La *dignidad* (o *elegancia*) tiene que ver con la habilidad del autor para producir una buena impresión por su estilo. La *corrección* es un asunto de seguir las reglas de la lengua implicada. Los principales recursos para lograr esto vinieron a ser llamados las figuras del discurso o figuras retóricas. Estas figuras dieron ornamentación a un discurso, y el componente figurista u ornamental asumió tal importancia que llegó a ser sinónimo de retórica.

Las figuras retóricas basadas en la similitud de dominios (como los símiles, metáforas y analogías) constituyen una de las muchas formas en que los seres humanos enriquecen el lenguaje que usan. Estas figuras han tenido poca presencia en la Generación de Lenguaje Natural en el pasado, debido posiblemente a que tienen poca o ninguna cabida en el tipo de textos técnicos que normalmente se tratan. Se han llevado a cabo algunos esfuerzos

para incluir comparaciones en generadores de texto que operan en escenarios pedagógicos (Milosavljevic, 1997a,b), donde se presenta un sistema de hipertexto dinámico (PEBA-II) para la descripción de animales. También ha habido mucho trabajo sobre metáforas desde un punto de vista cognitivo, pero se ha hecho muy poco en términos de estudiar los usos metafóricos de las palabras en la generación de texto.

2.6.1. Metáfora, Analogía y Alineamiento Estructural

Tanto la metáfora como la analogía son dos mecanismos cognitivos que han sido reconocidos como la base del razonamiento entre diferentes dominios. Para un análisis exhaustivo sobre el tema se puede consultar (Veale, 1995). Está claro que la metáfora y la analogía comparten muchas características, y por ello no se ha llegado a un consenso en la literatura en cuanto a una distinción clara entre ellas.

Muchos de los problemas de interpretación metafórica se pueden resolver utilizando modelos analógicos establecidos como la aproximación de alineación estructural (Gentner, 1983). Como trabajos importantes en este área podemos nombrar SME (Falkenhainer et al., 1989), Sapper (Veale, 1995) o Divago (Pereira, 2005). La idea general tras esta aproximación es que la metáfora y la analogía son fundamentalmente resultado de una interacción entre dos dominios, el de origen y el de destino, denominados *vehicle* y *tenor* en literatura. Por ejemplo, en la analogía *the lion is the king of the jungle* el dominio de origen sería el dominio de la jungla incluyendo a los animales que viven en ella, entre ellos el león. El dominio de destino sería el de los reinos, donde se establece la analogía de que el *king/lion* gobierna el *kingdom/jungle*.

Esta interacción puede ser simplificada y vista como un alineamiento estructural o correspondencia entre los grafos de conceptos que representan los dos dominios. De esta forma, se puede ver un dominio como una red semántica donde los nodos son los conceptos y los arcos las relaciones, y una correspondencia entre dos conceptos de los dos dominios como el resultado de la aplicación de ciertas reglas relativas a la estructura de los grafos. Por ejemplo, si dos nodos comparten la misma conexión al mismo nodo, forman una correspondencia potencial (regla de la triangulación en (Veale, 1995)). Si dos nodos comparten la misma conexión a otros dos nodos que ya están formando una correspondencia, también forman una correspondencia potencial (regla de la cuadrangulación en (Veale, 1995)). Como muchos dominios serán isomórficos (1 a 1), podrá haber una gran cantidad de posibilidades.

Un mapeamiento (por ejemplo de un concepto X a un concepto Y) producido por alineamiento estructural debe hacer hincapié en algunas correspondencias entre dos conceptos, como el papel que tiene un concepto sobre un dominio (por ejemplo, el concepto Y en el dominio T) que se puede proyectar a su homólogo en el otro dominio (por ejemplo, el concepto X en Z).

Esto implica también la proyección implícita del contexto (función, propiedades, etc.) Por ejemplo, cuando alguien dice *my surgeon is like a butcher*, de inmediato algunas de las propiedades de *butcher* se proyectan sobre *surgeon*, como por ejemplo *careless* (en oposición a *careful*) o el usar un *knife* (en lugar de un *scalpel*). Estas correspondencias pueden a su vez ofrecer otro tipo de deducciones o descripciones más elaboradas como *clinical slaughterhouse* por hospital.

Estas propiedades de las metáforas resultan útiles para tareas como la generación de texto y presentan un gran potencial. Algoritmos como los proporcionados por Sapper (Veale, 1995), Mapper (Pereira, 2007) o SME (Gentner, 1983) ayudan a encontrar mapeamientos adecuados para su uso en la generación automática de texto. Esto en sí mismo plantea una serie de nuevos desafíos, algunos de los cuales son considerados en este trabajo.

Mapper

Anteriores intentos de explorar la inserción de metáforas en el proceso de generación automática de texto (Pereira et al., 2006) han utilizado el algoritmo Mapper del sistema Divago descrito en (Pereira, 2007). Este algoritmo de alineamiento es extremadamente dependiente del conocimiento. Además, dada la complejidad de la tarea (búsqueda de grafos isomórficos), los dominios demasiado grandes pueden resultar impracticables para el algoritmo. Para superar este problema, Mapper no está diseñado para lograr la solución óptima, sino una que se considere lo suficientemente aceptable. Para ello utiliza un enfoque probabilístico en ciertos puntos de la ejecución, resultando en diferentes resultados para distintas ejecuciones.

Dados los dominios origen y destino, el algoritmo de búsqueda de correspondencias empieza buscando semillas iniciales por las que comenzar a trabajar. Este proceso se basa en encontrar pares de conceptos que comparten la misma relación a un tercer concepto (regla de triangulación). Los que presenten mayor número de conceptos compartidos comenzarán el proceso de buscar correspondencias 1-a-1 (este proceso se describe en mayor detalle en (Pereira, 2007)). Varias correspondencias potenciales pueden emerger de este proceso, pero el algoritmo elimina la mayoría de ellas durante la generación, de manera que llega al final con el mapeamiento más grande entre todos los posibles que podían encontrarse a partir de esa semilla. Es muy importante, sin embargo, tomar nota de que este algoritmo no es óptimo, ya que la elección de otras semillas puede dar lugar a resultados diferentes.

Para este trabajo se ha explorado el alineamiento estructural proporcionado por el algoritmo Mapper con una realización particular en mente: oraciones *X is the Y of Z* (Fauconnier y Turner, 2002). Como ya hemos visto una correspondencia de un concepto X a un concepto Y debe acentuar alguna similitud entre dos conceptos, es decir, que según alguna perspectiva el papel que un concepto juega en un dominio (por ejemplo el concepto Y

en el dominio T) puede ser proyectado a su equivalente en el otro dominio (por ejemplo el concepto X en el dominio Z). Éste es el razonamiento detrás de la expresión *X is the Y of Z*, donde Z es el dominio en que X está integrada. Por ejemplo, *Freud is the father of Psychoanalysis* es el resultado de la correspondencia *Freud* $\leftrightarrow$ *father* aplicada a los dominios del psicoanálisis y de la estructura familiar, respectivamente. Se puede encontrar este patrón en muchos más ejemplos, como *Brugges is the Venice of Belgium*, *the eyes are the mirror of the soul*, etc.

Una vez que el mapa conceptual está completo, el Mapper busca una correspondencia estructural entre los dominios rellenando los huecos correspondientes de *X is the Y of Z*. Para reducir el espacio de búsqueda, cada par de candidatos es ordenado en términos de su similitud potencial. Esta similitud potencial es directamente dependiente del número de relaciones compartidas que tienen ambos conceptos (por ejemplo, el concepto *perro* es más similar a *gato* que a *coche*, ya que ambos tienen piernas, respiran, son mascotas, etc.). Con esta información se establecen valores umbrales de similitud que evitan la exploración de partes poco prometedoras del espacio de búsqueda.

Sapper

El sistema Sapper (Veale, 1995) sigue el estado del arte general sobre alineamiento estructural. Sapper forma analogías usando activación por difusión (*spreading-activation*) dentro de un modelo de red semántica de memoria a largo plazo, utilizando para ello los puentes conceptuales que se han establecido entre los conceptos de esta red. Estos puentes almacenan los mapeamientos potenciales entre conceptos, y son automáticamente añadidos por Sapper a su red semántica cuando la estructura de dos conceptos comparte algunas regularidades locales de estructura. Estos puentes son tentativos cuando son creados, y por tanto permanecen latentes. Pero estos puentes latentes pueden ser despertados, y posteriormente utilizados en la activación, cuando algunas de las correspondencias analógicas propuestas es realizada por la parte cognitiva del sistema.

En este trabajo no se utilizará Sapper a la hora de establecer las correspondencias entre conceptos, pero sí se recurrirá al rico conocimiento del mundo de que dispone para trabajar sobre él. En Sapper cada dominio se representa como un grafo donde los nodos representan conceptos y las aristas entre nodos representan relaciones entre estos conceptos. Las aristas entre nodos se encuentran cualificadas por etiquetas que indican qué relación existe entre dos nodos (pertenencia a una clase, ser atributo de, control sobre, parte de, etc.) Además de una etiqueta, cada arista/relación está marcada por un número real entre -1 y 1 que indica la intensidad de esa relación entre conceptos. Las intensidades negativas significan que la relación es la opuesta a la representada por la arista, y cuanto mayor es el valor absoluto

de la intensidad más fuerte es la relación. Por ejemplo, una arista de tipo atributo entre los conceptos *surgeon* y *educated* con intensidad 0.85 indica que los cirujanos están bien educados. Sin embargo, una arista de tipo atributo entre los conceptos *butcher* y *precise* con intensidad -0.6 indica que no sólo los carniceros no son precisos, sino que además no lo son con bastante intensidad, es decir, son bastante imprecisos. Un ejemplo del tipo de grafos que forman el conocimiento de Sapper se puede ver en la Figura 2.5.

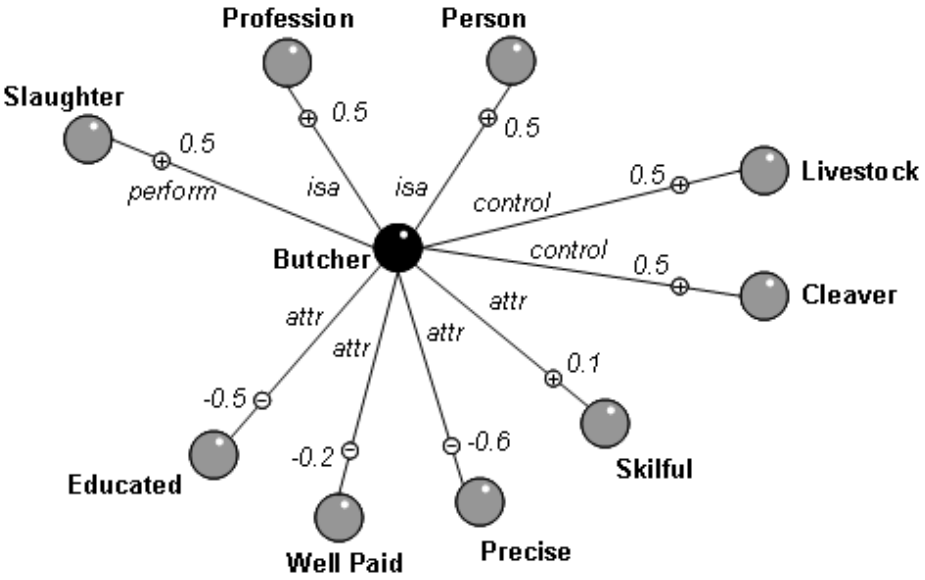

Figura 2.5: Parte del grafo alrededor del concepto *butcher*

2.6.2. Comparaciones

Aunque menos refinada que la analogía o la metáfora, la comparación o símil es otra figura retórica basada en las similitudes entre dominios que pensamos puede ser explorada computacionalmente e introducida en el proceso de generación de texto. Como ocurría con las analogía y las metáforas, aquí también estamos buscando una correspondencia entre dominios. Sin embargo, ahora nos centramos en conceptos que comparten una o más propiedades distintivas en lugar de simplemente buscar el alineamiento estructural.

La comparación es también una herramienta útil del lenguaje. Cuando se describe una entidad a una persona que no está familiarizado con ella, un sistema GLN puede utilizar comparaciones para encaminar la descripción hacia el conocimiento propio del usuario. En particular, es importante distinguir una entidad desconocida de otras entidades que pueden ser fácil-

mente confundidas con ella, y para ello resulta útil comparar la entidad desconocida con conceptos conocidos (Milosavljevic, 1996).

El propósito de la comparación es guiar el aprendizaje del oyente o lector de un concepto no familiar, o concretar su conocimiento previo de un concepto. Por ello, el contenido de una comparación (exactamente qué propiedades del concepto se usan en la comparación) y la realización de la comparación (cómo se expresa la comparación en lenguaje natural), dependerán del conocimiento previo del oyente o lector de los conceptos que aparecen en la misma. En particular, si se tiene un conocimiento muy limitado de los conceptos de la comparación, entonces será necesario ser explícito y detallar las propiedades de estos conceptos.

El trabajo más significativo sobre generación de comparaciones y su relación con el proceso de generación automática de texto es probablemente el de Milosavljevic (1999). En él la autora pretende identificar cuándo y cómo se utiliza la comparación en las descripciones de entidades escritas por seres humanos, y desarrollar métodos para la generación automática de comparaciones semejantes. Las comparaciones se utilizan en las descripciones con dos objetivos en mente: (1) para basar la descripción en los conocimientos previos de los usuarios acerca de otras entidades, facilitando la comprensión de la nueva información, y (2) para evitar malentendidos y falsas suposiciones sobre lo que se está describiendo. Las comparaciones generadas utilizan tanto similitudes como diferencias entre conceptos, y se utilizan en las descripciones generadas por dos sistemas diferentes: Peba-II (Milosavljevic et al., 1996) e ILEX (O'Donnell et al., 2001).

Las comparaciones tratadas en este trabajo corresponden a las COMPARACIONES ILUSTRATIVAS (*illustrative comparisons*) definidas por Milosavljevic (Milosavljevic, 2003). Una comparación ilustrativa es una comparación basada en familiaridad que no contiene diferencias entre las entidades comparadas. En este tipo de comparaciones, la entidad con la que se está comparando es usada como referencia para ayudar al lector u oyente a formarse un modelo conceptual más correcto de la entidad comparada (o al menos de un conjunto particular de atributos de la misma).

Resumen y Conclusiones

En este capítulo hemos realizado revisiones del estado del arte de dos campos principalmente: la generación de expresiones de referencia y las figuras retóricas basadas en similitudes entre dominios.

En el caso de la generación de expresiones de referencia se puede concluir que de los dos posibles objetivos que puede tener una referencia (distinguir vs. describir) sólo el primero ha sido estudiado exhaustivamente en la literatura. Aunque existen algunos trabajos sobre descripción de entidades en

textos generados automáticamente, no se ha estudiado anteriormente esta función desde el punto de vista directo de las referencias a entidades.

En el caso de las figuras retóricas también se ha visto que existe un amplio conjunto de técnicas y sistemas para la generación de figuras retóricas de manera conceptual, pero tampoco se ha estudiado cómo se podrían introducir estas figuras en texto y qué repercusiones podrían tener sobre el mismo en caso de hacerlo.

De cara a posteriores capítulos, la revisión sobre generación de expresiones de referencia nos dará las pautas para la implementación de diversas soluciones en este campo que surgen del replanteamiento de los problemas a resolver, así como para su evaluación. Ambos aspectos se abordarán en el capítulo 4. El trabajo existente sobre figuras retóricas será la base para su utilización de cara a cubrir el objetivo de proporcionar información descriptiva y cuya implementación se muestra en el capítulo 5.

Capítulo 3

Algoritmos Evolutivos, Razonamiento basado en Casos y Agentes

"Ah, people only know what you tell them."
— Catch Me If You Can (2002)

Como ya se comentó en la introducción, la generación de expresiones de referencia con función distintiva puede ser dividida en dos problemas principales: la selección de atributos y la lexicalización. A la hora de resolver estos problemas nos encontramos con que existen paradigmas dentro de la Inteligencia Artificial que pueden resultar útiles. La selección de atributos se puede reformular como un problema de búsqueda en el que, dada una situación determinada, es necesario buscar qué conjunto de atributos sirven para distinguir a un referente objetivo. Por su parte la lexicalización de los conjuntos de atributos generados depende en gran medida de la forma de expresarse de cada persona, por lo que debe ser posible abordar la cuestión desde un punto de vista que permita formalizar distintos estilos a la hora de lexicalizar.

Por otra parte, a la hora de generar información descriptiva para las referencias utilizando figuras retóricas como la comparación o la analogía habrá que conectar diversos mecanismos diferentes que están involucrados en el proceso, desde los módulos que se encargan de generar las figuras hasta los que manejan los recursos de conocimiento necesarios.

Los Algoritmos Evolutivos (sección 3.1) son un paradigma de resolución de problemas de búsqueda donde no es necesario especificar cómo se encuentra el resultado, sino cómo de buena es una posible solución. Por ello, serán adecuados para el tratamiento de la selección de atributos, ya que se trata de un problema en el que no se ha sabido formalizar cómo se debe

buscar una solución, pero sí decidir cuando una solución es buena o no. El Razonamiento Basado en Casos (sección 3.2) permite buscar soluciones a los problemas a partir de una base de soluciones previamente utilizadas. Esto puede resultar muy adecuado en el caso del estilo de lexicalización de las referencias, ya que se pueden formar diferentes bases de casos de manera que cada una represente un estilo diferente. Finalmente se estudian también los Agentes Inteligentes (sección 3.3) que servirán como plataforma de integración de los diversos módulos que entran en escena a la hora de generar figuras retóricas para describir entidades en un texto dado.

3.1. Algoritmos Evolutivos

Los Algoritmos Evolutivos (AEs), conocidos también como Computación Evolutiva (CE), engloban un amplio conjunto de técnicas de resolución de problemas complejos inspiradas en los mecanismos de la evolución y selección natural, las cuales mantienen una población de individuos que evolucionan de acuerdo a reglas de selección y otros operadores genéticos como son el cruce y la mutación.

Los Algoritmos Evolutivos se basan en un modelo de evolución biológica natural que fue propuesto por primera vez por Charles Darwin (Darwin, 1859). La teoría de la evolución de Darwin explica el cambio adaptativo de las especies por el principio de la selección natural, que favorece la supervivencia y evolución de aquellas especies que están mejor adaptadas a las condiciones de su entorno. Como ciencia, la Computación Evolutiva surge a finales de los años 60 cuando John Holland planteó la posibilidad de incorporar los mecanismos naturales de selección y supervivencia a la resolución de problemas de Inteligencia Artificial (Holland, 1992).

La Computación Evolutiva no trata tanto de reproducir ciertos fenómenos que suceden en la Naturaleza como de aprovechar las ideas genéricas que hay detrás de ellos. Efectivamente, en el momento en que se tienen varios candidatos a solución para un problema surge inmediatamente la necesidad de establecer criterios de calidad y de selección, y también la idea de combinar características de buenas soluciones para obtener otras mejores. Dado que fue en el mundo natural donde se han planteado problemas de ese tipo, no tiene nada de extraño el que al aplicar tales ideas en la resolución de problemas científicos y técnicos se obtengan procedimientos bastante parecidos a los ya encontrados por la Naturaleza tras un largo periodo de adaptación.

3.1.1. Estructura de los Algoritmos Evolutivos

Dado un problema específico a resolver, la entrada del AE es una población de soluciones potenciales a ese problema, codificadas de alguna manera, y una métrica llamada función de aptitud que permite evaluar cuantitativamente cada solución candidata. Estas candidatas pueden ser soluciones que

ya se sabe que funcionan, con el objetivo de que el AE las mejore, pero se suelen generar aleatoriamente.

El AE evalúa cada candidata de acuerdo con la función de aptitud. En un conjunto de soluciones candidatas generadas aleatoriamente la mayoría no funcionarán en absoluto, y serán eliminadas. Sin embargo, por puro azar, unas pocas pueden ser prometedoras, pueden mostrar actividad, aunque sólo sea actividad débil e imperfecta, hacia la solución del problema. Estas candidatas prometedoras se conservan y se les permite reproducirse. Se realizan múltiples copias de ellas, pero las copias no son perfectas, ya que se introducen cambios aleatorios durante el proceso de copia. Luego, esta descendencia digital prosigue con la siguiente generación, formando una nueva población de soluciones candidatas, y son sometidas a una ronda de evaluación de aptitud. Las candidatas que han empeorado o no han mejorado con los cambios en su estructura son eliminadas; pero, de nuevo, por puro azar, las variaciones aleatorias introducidas en la población pueden haber mejorado a algunos individuos, convirtiéndolos en mejores soluciones del problema, más completas o más eficientes. De nuevo, se seleccionan y copian estos individuos vencedores hacia la siguiente generación con cambios aleatorios, y el proceso se repite. Las expectativas son que la aptitud media de la población se incrementará en cada generación y, por tanto, repitiendo este proceso cientos o miles de veces, pueden descubrirse soluciones muy buenas del problema.

De esta forma, los AEs combinan la búsqueda aleatoria, dada por las transformaciones de la población, con una búsqueda dirigida, dada por la selección. El esquema general de un algoritmo evolutivo se muestra en la Figura 3.1.

En resumen, los principales componentes de un AE son los siguientes:

- *Población de individuos*, que son una representación (no necesariamente directa) de posibles soluciones.
- *Procedimiento de transformación*, para construir nuevos individuos a partir de los anteriores.
- *Procedimiento de selección*, basado en la aptitud de los individuos para resolver el problema.

3.1.2. Ventajas de los Algoritmos Evolutivos

El primer y más importante punto es que los Algoritmos Evolutivos son intrínsecamente paralelos. La mayoría de los otros algoritmos son en serie y sólo pueden explorar el espacio de soluciones hacia una solución en una dirección al mismo tiempo, y si la solución que descubren resulta subóptima, no se puede hacer otra cosa que abandonar todo el trabajo hecho y empezar de nuevo. Sin embargo, ya que los AEs tienen descendencia múltiple, pueden

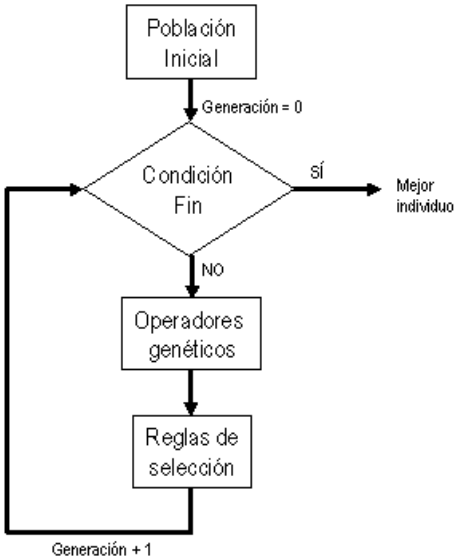

Figura 3.1: Esquema básico de un algoritmo evolutivo

explorar el espacio de soluciones en múltiples direcciones a la vez. Si un camino resulta ser un callejón sin salida, pueden eliminarlo fácilmente y continuar el trabajo en avenidas más prometedoras, dándoles una mayor probabilidad en cada ejecución de encontrar la solución.

Debido al paralelismo que les permite evaluar implícitamente muchos esquemas a la vez, los Algoritmos Evolutivos funcionan particularmente bien resolviendo problemas cuyo espacio de soluciones potenciales es realmente grande, demasiado vasto para hacer una búsqueda exhaustiva en un tiempo razonable.

Otra ventaja notable de los AEs es que se desenvuelven bien en problemas con un paisaje adaptativo complejo, aquellos en los que la función de aptitud es discontinua, ruidosa, cambia con el tiempo, o tiene muchos óptimos locales. La mayoría de los problemas prácticos tienen un espacio de soluciones enorme, imposible de explorar exhaustivamente; el reto se convierte entonces en cómo evitar los óptimos locales -soluciones que son mejores que todas las que son similares a ella, pero que no son mejores que otras soluciones distintas situadas en algún otro lugar del espacio de soluciones. Muchos algoritmos de búsqueda pueden quedar atrapados en los óptimos locales. Los Algoritmos Evolutivos, por otro lado, han demostrado su efectividad para escapar de los óptimos locales y descubrir el óptimo global incluso en paisajes adaptativos muy escabrosos y complejos. Los cuatro componentes principales de los AEs (paralelismo, selección, mutación y cruce) trabajan juntos para conseguir esto.

Sin embargo, el cruzamiento es el elemento clave que distingue a los Algoritmos Evolutivos de los otros métodos como los trepacolinas (*hillclimbing*)

y el recocimiento simulado (*Simulated Annealing*). Sin el cruzamiento, cada solución individual va por su cuenta, explorando el espacio de búsqueda en sus inmediaciones sin referencia de lo que el resto de individuos puedan haber descubierto. Sin embargo, con el cruzamiento en juego, hay una transferencia de información entre los candidatos prósperos: los individuos pueden beneficiarse de lo que otros han aprendido, y los esquemas pueden mezclarse y combinarse, con el potencial de producir una descendencia que tenga las virtudes de sus dos padres y ninguna de sus debilidades.

Otro área en el que destacan los Algoritmos Evolutivos es su habilidad para manipular muchos parámetros simultáneamente. Muchos problemas de la vida real no pueden definirse en términos de un único valor que hay que minimizar o maximizar, sino que deben expresarse en términos de múltiples objetivos, a menudo involucrando contrapartidas: uno sólo puede mejorar a expensas de otro. Los AEs son muy buenos resolviendo estos problemas: en particular, su uso del paralelismo les permite producir múltiples soluciones, igualmente buenas, al mismo problema, donde posiblemente una solución candidata optimiza un parámetro y otra candidata optimiza uno distinto, y luego un supervisor humano puede seleccionar una de esas candidatas para su utilización.

Finalmente, una de las cualidades de los algoritmos evolutivo que, a primera vista, puede parecer poco conveniente, resulta ser una de sus ventajas: los AEs no saben nada de los problemas que deben resolver. En lugar de utilizar información específica conocida a priori para guiar cada paso y realizar cambios pensando en la mejora de las soluciones, como hacen los diseñadores humanos: realizan cambios aleatorios en sus soluciones candidatas y luego utilizan la función de aptitud para determinar si esos cambios producen una mejora. La virtud de esta técnica es que permite a los algoritmos genéticos comenzar con una mente abierta, por así decirlo. Como sus decisiones están basadas en la aleatoriedad, todos los caminos de búsqueda posibles están abiertos teóricamente a un AE. En contraste, cualquier estrategia de resolución de problemas que dependa de un conocimiento previo, debe inevitablemente comenzar descartando muchos caminos a priori, perdiendo así cualquier solución novedosa que pueda existir. Los AEs, al carecer de ideas preconcebidas basadas en creencias establecidas sobre "cómo deben hacerse las cosas" o sobre lo que "de ninguna manera podría funcionar", no tienen este problema.

3.1.3. Limitaciones de los Algoritmos Evolutivos

Aunque los Algoritmos Evolutivos han demostrado su eficiencia y potencia como estrategia de resolución de problemas, no son una panacea. Los AEs tienen ciertas limitaciones; sin embargo, se demostrará que todas ellas pueden superarse y que ninguna de ellas afecta a la validez de la evolución biológica.

La primera y más importante consideración al crear un algoritmo genético es definir una representación del problema. El lenguaje utilizado para especificar soluciones candidatas debe ser robusto; es decir, debe ser capaz de tolerar cambios aleatorios que no produzcan constantemente errores fatales o resultados sin sentido. Hay dos maneras principales para conseguir esto. La primera, utilizada por la mayoría de los Algoritmos Evolutivos, es definir a los individuos como listas de números (binarios, enteros o reales) donde cada número representa algún aspecto de la solución candidata. Si los individuos son cadenas binarias, un 0 o 1 podría significar la ausencia o presencia de una cierta característica. Si son listas de números, estos números podrían representar muchas cosas distintas: los pesos de las conexiones en una red neuronal, el orden de las ciudades visitadas en un recorrido dado, la situación espacial de componentes electrónicos, los valores con los que se alimenta a un controlador, etc. Así, la mutación implica cambiar estos números, cambiar bits o sumar o restar valores aleatorios. En este caso, el propio código del programa no cambia; el código es lo que dirige la simulación y hace un seguimiento de los individuos, evaluando sus aptitudes y quizá asegurando que sólo se producen valores realistas y posibles para el problema dado.

El problema de cómo escribir la función de aptitud debe considerarse cuidadosamente para que se pueda alcanzar una mayor aptitud y verdaderamente signifique una solución mejor para el problema dado. Si se elige mal la función de aptitud o se define de manera inexacta, puede que el algoritmo genético sea incapaz de encontrar una solución al problema, o pueda acabar resolviendo el problema equivocado.

Además de elegir bien la función de aptitud, también deben elegirse cuidadosamente los otros parámetros de un AE: el tamaño de la población, el ritmo de mutación y cruzamiento, el tipo y fuerza de la selección. Si el tamaño de la población es demasiado pequeño, puede que el Algoritmo Evolutivo no explore suficientemente el espacio de soluciones para encontrar buenas soluciones consistentemente. Si el ritmo de cambio genético es demasiado alto o el sistema de selección se escoge inadecuadamente, puede alterarse el desarrollo de esquemas beneficiosos y la población puede entrar en una catástrofe de errores, al cambiar demasiado rápido para que la selección llegue a producir convergencia.

Un problema muy conocido que puede surgir con un AE se conoce como convergencia prematura. Si un individuo que es más apto que la mayoría de sus competidores emerge muy pronto en el curso de la ejecución, se puede reproducir tan abundantemente que merme la diversidad de la población demasiado pronto, provocando que el algoritmo converja hacia el óptimo local que representa ese individuo, en lugar de rastrear el paisaje adaptativo lo bastante a fondo para encontrar el óptimo global. Esto es un problema especialmente común en las poblaciones pequeñas, donde incluso una variación aleatoria en el ritmo de reproducción puede provocar que un genotipo se

haga dominante sobre los otros. Los métodos más comunes implementados por los investigadores en AEs para solucionar este problema implican controlar la fuerza selectiva, para no proporcionar tanta ventaja a los individuos excesivamente aptos.

3.1.4. Aplicación de los Algoritmos Evolutivos a la Generación de Lenguaje Natural

Las técnicas evolutivas han demostrado estar particularmente bien preparadas para la generación de poesía. Los trabajos de (Manurung, 2003) y (Levy, 2001) proponen diferentes modelos computacionales para composición de poesía basada en aproximaciones evolutivas. En ambos casos, la principal dificultad radica en la elección de la función de adaptación para guiar el proceso. Aunque Levy sólo utiliza un modelo simple relacionado con la información silábica, su descripción general de la arquitectura en términos de una población de borradores de poemas que evolucionan, dando prioridad a los borradores que son mejor evaluados, es una idea a tener en cuenta. Levy usa una red neuronal, entrenada con ejemplos de versos válidos, para evaluar los poemas. Por otro lado, el trabajo de Manurung abarca la tarea completa, y presenta un conjunto de evaluadores que clasifican las soluciones candidatas de acuerdo a ciertas heurísticas.

Los algoritmos evolutivos también han sido usados para la planificación de texto. En (Duboue y McKeown, 2002) los autores presentan una técnica para aprender una estructura de árbol para un planificador de contenido a partir de un corpus de entradas semánticas y sus correspondientes salidas escritas por autores humanos. Para ello usan un mecanismo de búsqueda estocástica con dos niveles de funciones de aptitud para crear la estructura del planificador. Los algoritmos evolutivos son usados también en (Mellish et al., 1998), donde los autores plantean cómo, dados un conjunto de hechos y un conjunto de relaciones retóricas que se pueden usar para conectarlos, se puede usar este material para generar el mejor texto posible.

3.2. Razonamiento Basado en Casos

El Razonamiento Basado en Casos o *Case-Based Reasoning* (CBR) (Agnar y Enric, 1994) es un paradigma de resolución de problemas que utiliza el conocimiento específico de experiencias pasadas. Cada problema es considerado como un caso del dominio, y un nuevo problema es resuelto *recuperando* uno o varios casos pasados, *reutilizándolos* de una forma u otra, *revisando* la solución obtenida, y *reteniendo* la nueva experiencia incorporándola en la base de casos. Se emplea conocimiento general durante este ciclo en los cuatro subprocesos del ciclo CBR.

Razonar usando casos pasados es un mecanismo de resolución de problemas utilizado con frecuencia por los seres humanos. Esta afirmación ha

sido corroborada por la investigación en el campo de la psicología cognitiva. Numerosos estudios han proporcionado resultados empíricos de la importancia de las experiencias previamente vividas en la resolución de problemas. Schank (1982) desarrolló una teoría sobre el aprendizaje basado en la retención de experiencias en una estructura dinámica de memoria. Anderson (1983) ha demostrado que la gente utiliza casos pasados como modelos en el aprendizaje de resolución de problemas. Otros resultados, como por ejemplo los de Kolodner (1983), indican que el uso de casos pasados es una técnica de resolución de problemas utilizada incluso por expertos.

En la terminología CBR, un CASO representa generalmente un problema. Una solución a un problema, que ha sido recogida y aprendida de tal manera que puede ser reutilizada para resolver problemas futuros, se denomina CASO PREVIO o CASO PASADO. De la misma forma, un NUEVO CASO o CASO SIN RESOLVER es la descripción de un nuevo problema que habrá que resolver. El razonamiento basado en casos es por tanto el proceso cíclico de resolver un problema, aprender de esta experiencia, resolver un nuevo problema, etc.

3.2.1. El Ciclo CBR

El ciclo CBR general se puede describir según los siguientes cuatro procesos:

1. **RECUPERAR** el caso o casos más similares.
2. **REUTILIZAR** la información y conocimiento de estos casos para resolver el problema.
3. **REVISAR** la solución propuesta.
4. **RECORDAR** la parte de esta experiencia que pueda resultar útil para resolver casos en el futuro.

Un nuevo problema se resuelve recuperando uno o más casos pasados, reutilizándolos de una forma u otra, revisando la solución obtenida, y reteniendo o aprendiendo la nueva experiencia incorporándola en la base de casos ya existente. Estas cuatro tareas contienen a su vez cierto número de pasos intermedios específicos. El ciclo general se muestra en la Figura 3.2. Como se indica en la figura, el conocimiento general sobre el dominio de aplicación también forma parte del ciclo, dando soporte al resto de procesos CBR.

Recuperación

La tarea de recuperación comienza con una descripción parcial de un problema y termina cuando se encuentra el caso pasado que más se ajusta a la consulta. Para ello es necesario identificar el conjunto de características

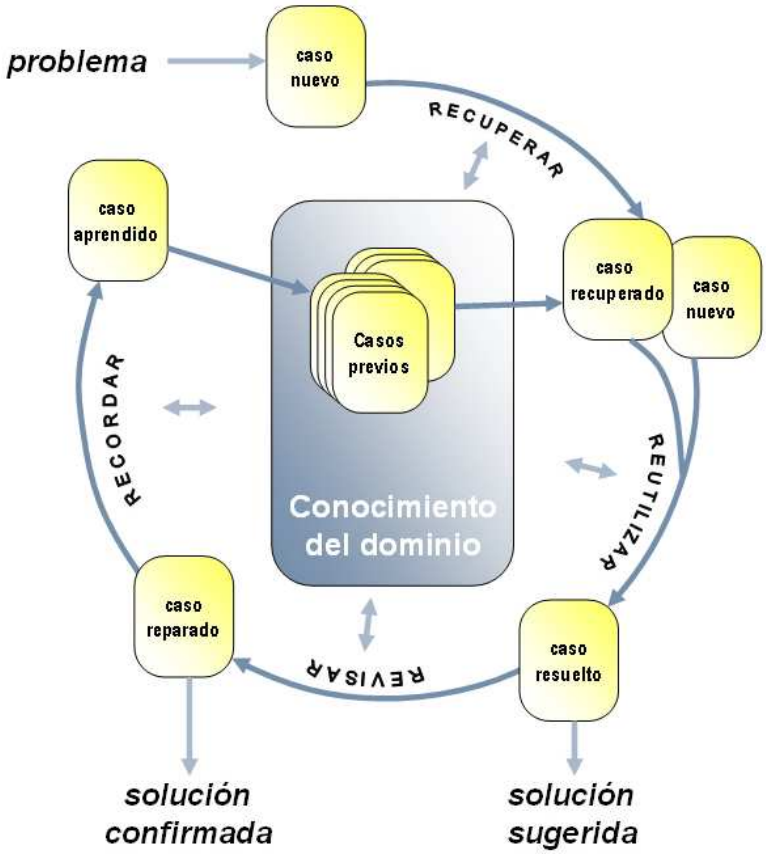

Figura 3.2: Ciclo del Razonamiento Basado en Casos

más relevantes del problema en cuestión, recuperar un conjunto de casos que sean lo suficientemente parecidos al que se busca, y finalmente seleccionar dentro de todos ellos el que más se ajusta a la consulta.

Para permitir una recuperación de casos basada en las similitudes semánticas y la importancia relativa de las diferentes características, será necesario un conocimiento general del dominio. Las aproximaciones pobres en conocimiento (*knowledge-poor*) presentan ventajas en dominios donde el conocimiento general es muy difícil o imposible de conseguir. Por otro lado, las aproximaciones ricas en conocimiento (*knowledge-intensive*) son capaces de utilizar el significado contextual de la descripción de un problema en dominios donde el conocimiento general es accesible.

Reutilización

La reutilización de la solución del caso recuperado en el contexto de un nuevo caso se centra en dos aspectos: (a) las diferencias entre el caso pasado y el actual y (b) qué parte del caso recuperado puede ser transferida al nuevo

caso. Además, dado que la recuperación puede proporcionar varios casos similares a la consulta, también es posible llevar a cabo una reutilización de todos ellos para conseguir la solución buscada.

Existen dos formas de reutilizar un caso pasado. En la reutilización *transformacional* la solución del caso pasado no es directamente la solución del nuevo caso, pero existe cierto conocimiento en el caso pasado en forma de los operadores que deben ser aplicados para transformar la solución pasada en la solución del nuevo caso. La reutilización *derivacional* se basa en cómo el problema fue resuelto en el caso pasado, incluyendo información como justificación de los operadores usados, subobjetivos considerados, alternativas generadas, etc. La reutilización derivacional reinstancia entonces el método recuperado y ejecuta el plan pasado sobre el nuevo contexto de la consulta.

Revisión

Cuando una solución generada por la reutilización no es correcta, surge la oportunidad de aprender de los errores. Esta fase de revisión consiste en evaluar la solución generada en la reutilización. Si la solución es válida, se aprenderá del éxito en la fase de retención. Si no, se repara la solución usando conocimiento específico del dominio.

Retención

Éste es el proceso de incorporar lo que pueda resultar útil para resolver nuevos casos al conocimiento existente. El aprendizaje sobre el éxito o fracaso de la solución propuesta involucra seleccionar qué información del caso hay que retener, en qué forma hacerlo, cómo indexar el caso para recuperaciones posteriores de problemas similares, y cómo integrar el nuevo caso en la estructura de memoria.

3.2.2. Case Retrieval Nets

Las *Case Retrieval Nets* (CRNs) (Lenz y Burkhard, 1996b) son un modelo de memoria asociativa desarrollado para hacer más eficiente la tarea de recuperación dentro del ciclo CBR. Se basan en la idea de que los seres humanos son capaces de resolver problemas sin necesidad de realizar procesos de búsqueda intensivos, sino partiendo de la descripción del problema y ampliando el rango de candidatos potenciales teniendo en cuenta su cercanía a la descripción de partida.

Como su nombre indica, las CRNs organizan la base de casos en forma de red. Los elementos básicos son las ENTIDADES DE INFORMACIÓN (EIs), la unidad de conocimiento más básica en forma de un par atributo-valor. Un caso consiste en un descriptor y un conjunto de EIs, y la base de casos es el conjunto de nodos que representan las entidades de información y los casos. Los nodos de las entidades de información están conectados entre sí por

arcos de similitud, y los nodos de los casos son alcanzables a partir de las EIs que los forman a través de los arcos de relevancia. Se pueden expresar distintos grados de similitud y relevancia variando el peso de los arcos. Dada esta estructura, la recuperación de casos consiste en la activación de las entidades de información que correspondan a la consulta, la propagación de esta activación a través de la red de entidades de acuerdo con los valores de similitud, y la recolección de la activación obtenida en los casos asociados.

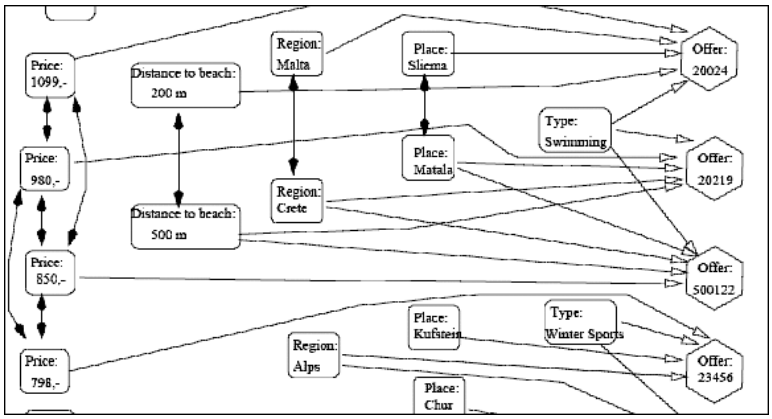

Figura 3.3: Ejemplo de una Case Retrieval Net en el dominio de agencias de viajes

En la Figura 3.3 se puede observar un ejemplo de CRN en el dominio de las agencias de viajes. Los rectángulos representan los nodos de entidad, con su correspondiente par atributo-valor. Los hexágonos son los nodos de casos, con el descriptor que los identifica unívocamente. Los nodos de entidad están relacionados entre sí con los arcos de cabeza de flecha negra, y con los casos a los que pertenecen por los arcos de cabeza de flecha blanca. Los pesos de los arcos no están representados en la figura, y los arcos con peso 0 simplemente no aparecen dibujados.

Algunas características importantes de las CRNs son las siguientes:

- Pueden manejar consultas parciales correctamente y sin pérdidas de eficiencia.

- La recuperación de casos se puede considerar también como un proceso de perfeccionamiento. A la red le basta una parte del caso para completar el resto de su contenido.

- Pueden expresar distintas similitudes y relevancias en tiempo de ejecución, mientras que otros sistemas necesitan ser recompilados para hacerlo.

- No es necesario que todos los casos estén representados por los mismos atributos. Para algunos serán relevantes ciertas características y para otros no.

- La inserción de nuevos casos puede ser realizada de forma incremental insertando nuevos nodos y arcos.

3.2.3. Razonamiento Basado en Casos en el Procesamiento de Lenguaje Natural

Las técnicas de aprendizaje máquina han demostrado reducir significativamente el esfuerzo de ingeniería de conocimiento involucrado en la construcción de sistemas de Procesamiento de Lenguaje Natural (NLP) a gran escala. Estas técnicas ofrecen medios automáticos de adquisición de soluciones robustas para problemas de desambiguación léxica y estructural. Entre los métodos de aprendizaje que han sido utilizados con éxito en el campo del lenguaje natural se encuentran los métodos de aprendizaje basados en casos, donde el sistema de lenguaje natural procesa un texto recuperando ejemplos previamente almacenados que describen cómo se manipulados textos similares en el pasado. Las aproximaciones basadas en casos se han utilizado en tareas tales como adquisición de acentuación (Daelemans et al., 1994), desambiguación de significado (Ng y Lee, 1996) o extracción de conceptos (Cardie, 1993).

Desde un punto de vista centrado en la generación de texto, encontramos el sistema SEGUE (Pan y Shaw, 2004), un generador de lenguaje natural que emplea razonamiento basado en casos pero realiza adaptaciones basadas en reglas. Este sistema utiliza un corpus anotado como fuente de conocimiento y emplea reglas gramaticales para construir nuevas oraciones. Usando recuperación guiada por la adaptación para seleccionar casos que pueden ser adaptados fácilmente para la salida deseada, SEGUE simplifica el proceso y evita la generación de frases gramaticalmente incorrectas.

En (Gervás, 2001) se presenta una aplicación basada en casos que genera versiones poéticas de textos proporcionados por el usuario. Los casos consisten en una oración de prosa (usada como clave en la recuperación) asociada con el correspondiente fragmento poético (usado como punto inicial para la solución). La adaptación se lleva a cabo combinando información fonética, métrica y léxica disponible en diferentes fuentes: el mensaje en prosa, el caso recuperado, y un vocabulario adicional proporcionado por el usuario.

El sistema de generación de texto para cuentos fantásticos PRINCE también incorpora una solución basada en casos en su módulo de lexicalización (Hervás y Gervás, 2005, 2006). En este caso, el razonamiento basado en casos se usa para seleccionar plantillas adecuadas para la realización textual de mensajes que describen acciones. Como modelo de memoria se utiliza una CRN, y para computar las similitudes entre conceptos se usa una taxonomía

de los conceptos que están involucrados en los textos. El proceso de resolver una consulta puede involucrar varios procesos de recuperación para obtener un conjunto de casos que juntos constituyen una buena solución para transcribir los datos de la consulta en mensajes de texto, y un proceso de adaptación rica en conocimiento basada en una base de conocimiento para identificar las sustituciones apropiadas para los conceptos que aparecen en los casos.

3.3. Agentes Inteligentes

Se pueden encontrar en la literatura un gran número de definiciones del concepto de AGENTE sin que ninguna de ellas haya sido plenamente aceptada por la comunidad científica. Quizás la más simple sería la de Russell (1996), que considera un agente como una entidad que percibe y actúa sobre un entorno. Basándose en esta definición, se pueden caracterizar distintos agentes de acuerdo a los atributos que posean y que van a definir su comportamiento (Botti et al., 1999) para resolver un determinado problema.

En la mayoría de las ocasiones, los agentes no se desarrollan de forma independiente, sino como entidades que constituyen un sistema. A estos sistemas se le denomina MULTI-AGENTE (Huhns y Singh, 1998), y en estos casos los agentes deberán interactuar entre ellos. Las interacciones más habituales como son informar o consultar a otros agentes permiten a los agentes "hablar" entre ellos, tener en cuenta las tareas que realiza cada uno y razonar acerca del papel jugado por los diferentes agentes que constituyen el sistema. La comunicación entre agentes se realiza por medio de un lenguaje de comunicación de agentes (*Agent Communication Language* o ACL).

Un agente estará caracterizado por una serie de calificativos, los cuales vienen a denotar ciertas propiedades a cumplir por el agente. Esto nos lleva a plantear otra definición bastante aceptada de agente donde se emplean tres calificativos que según el autor se consideran básicos. Esta definición ve a un agente como un sistema de computación capaz de actuar de forma autónoma y flexible en un entorno (Wooldridge y Jennings, 1995), entendiendo por flexible que sea:

- *Reactivo.* El agente es capaz de responder a cambios en el entorno en que se encuentra situado.

- *Pro-activo.* A su vez el agente debe ser capaz de intentar cumplir sus propios planes u objetivos.

- *Social.* El agente debe poder comunicarse con otros agentes mediante algún tipo de lenguaje de comunicación de agentes.

3.3.1. Sistemas Multiagente

Los sistemas multiagente se diferencian de los de agente único en que varios agentes conviven de forma que influyen en los objetivos y acciones de los demás. Desde el punto de vista de un único agente, los sistemas multiagente difieren de los sistemas de agente único principalmente en que la forma en que cambia el entorno está determinada por otros agentes. Además de la incertidumbre que puede ser inherente al dominio, otros agentes pueden modificar intencionalmente el entorno de forma impredecible. Por tanto, se puede asumir que la mayor parte de los sistemas multiagente tendrán entornos dinámicos.

En general, los sistemas multiagente son sistemas computacionales en que varios agentes semi-autónomos interactúan o trabajan juntos para llevar a cabo cierto conjunto de tareas o para satisfacer ciertos objetivos. Estos sistemas pueden contener agentes que son homogéneos o heterogéneos, que tengan objetivos comunes o no.

La necesidad de interacción en estos sistemas surge porque los agentes están resolviendo problemas que son interdependientes, o por el uso de los recursos o por las relaciones existentes entre los objetivos de los subproblemas. Para que los agentes puedan conseguir soluciones óptimas y no conflictivas deberán tener cierta información del problema global y de los demás agentes del sistema.

De forma esquemática, los sistemas multiagente constarán de:

- *Agentes* de diversos tipos, homogéneos o heterogéneos, que cooperan para proporcionar a los usuarios servicios adaptables y personalizables. Cada agente es un sistema encapsulado y con límites bien definidos, capaz de una acción autónoma y flexible en su entorno para lograr sus objetivos de diseño.

- *Recursos* que pueden estar gestionados por los agentes o por algún otro sistema, y que forman parte del entorno de los agentes. Un recurso puede ser, por ejemplo, un servidor web, una base de datos, un servidor de correo, un sistema de gestión de procesos de negocio, etc., dependiendo del entorno de la aplicación.

- Un *middleware* para facilitar la comunicación entre los agentes y con otros sistemas. Sobre este middleware es posible utilizar herramientas basadas en estándares de comunicación entre agentes como FIPA ACL (2000) o KQML (Finin et al., 1994).

Modelar con agentes tiene ventajas en el sentido de que este paradigma es bastante más cercano a nuestra manera de ver el mundo: muchas organizaciones se definen con un conjunto de roles y relaciones entre ellos, que luego son asumidos por personas (agentes). Aún así, es necesario disponer

de métodos y herramientas para el desarrollo de los sistemas basados en agentes.

3.3.2. Open Agent Architecture, una Arquitectura para Sistemas Multiagente

Open Agent Architecture (OAA) (Martin et al., 1999) es un framework multiagente que se centra en permitir interacciones flexibles entre comunidades dinámicas de agentes software heterogéneos. La idea clave en OAA es la delegación: en lugar de que cada agente defina sus interacciones en profundidad (llamadas a métodos o paso de mensajes), especificando cómo y con quién va a interactuar, los agentes de OAA expresan sus interacciones en forma de necesidades delegadas a un agente llamado FACILITATOR. Éste se encarga de coordinar a la comunidad de agentes para resolver la tareas, suministrando servicios como paralelismo, manejo de errores y detección de conflictos. Los agentes distribuidos de OAA son programas simples, o *wrappers* sobre programas, distribuidos en distintas máquinas, pero compartiendo funcionalidades comunes.

En OAA, el control sobre las interacciones y comunicaciones entre agentes es el producto de la cooperación de cuatro fuentes de conocimiento diferentes:

- El solicitante que especifica un objetivo al *Facilitator* y proporciona sugerencias sobre cómo se puede resolver.
- Proveedores que registran sus capacidades en el *Facilitator*, conocen qué servicios pueden proporcionar y los límites de su capacidad para hacerlo.
- El *Facilitator* que mantiene una lista de los agentes proveedores disponibles y un conjunto de estrategias generales para la resolución de objetivos.
- Meta-agentes que contienen conocimiento específico del dominio o del objetivo y estrategias que se usan como ayuda para el *Facilitator*.

Este conocimiento se emplea para propiciar la coordinación en el conjunto de agentes de OAA. El *Facilitator* hace corresponder una petición con el agente o agentes que proporcionan ese servicio, delega la tarea para que la resuelvan, coordina sus esfuerzos, y devuelve los resultados al solicitante. Esta forma de cooperación entre agentes puede ser aplicada a la resolución tanto de tareas simples como compuestas. Además de la delegación, OAA también proporciona la capacidad de realizar llamadas directas a agentes específicos o a todos los agentes existentes. Un ejemplo de arquitectura OAA se puede ver en la Figura 3.4.

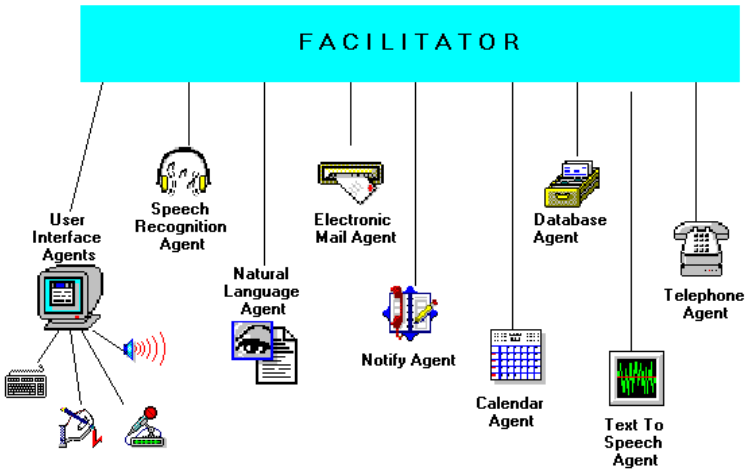

Figura 3.4: Ejemplo de un sistema multi-agente implementado usando OAA

El *Interagent Communication Language* (ICL) de OAA es la interfaz y lenguaje de comunicación usado por todos los agentes, no importa la máquina en la que se ejecutan o en qué lenguaje de programación están escritos. El ICL se ha diseñado como una extensión del lenguaje de programación Prolog, para aprovechar la unificación y otras características útiles. Gran parte de los elementos de los programas se especifican con expresiones en ICL, incluyendo declaraciones, eventos (comunicaciones entre los agentes), solicitudes de servicios, respuestas a estas solicitudes, especificaciones de disparadores, y elementos de datos compartidos.

Resumen y Conclusiones

En este capítulo se ha hecho una breve revisión de tres paradigmas clásicos de la Inteligencia Artificial: Algoritmos Evolutivos, Razonamiento Basado en Casos y Agentes Inteligentes.

Los dos primeros serán utilizados en el capítulo 4 para la implementación de soluciones para selección de atributos y la lexicalización de referencias respectivamente. El tercero será utilizado como base para la implementación de una arquitectura multiagente que integrará los distintos módulos encargados de la generación de figuras retóricas basadas en similitudes con fines descriptivos en el capítulo 5.

Capítulo 4

Generación de Expresiones de Referencia Básicas

"Indy, take the left tunnel! No, Indy! The left tunnel! The left! Indy!"
— Indiana Jones and the Temple of Doom (1984)

Como ya hemos visto en la sección 2.2, la generación de expresiones de referencia es la tarea encargada de referirse a las entidades que aparecen en un discurso hablado o escrito. A lo largo del tiempo ha estado principalmente orientada a la determinación de referencias que permitían distinguir unívocamente algunas de estas entidades de las demás que aparecían a su alrededor.

Seguramente el algoritmo incremental de Reiter y Dale (1992) (sección 2.2.5) puede ser considerado como la más estudiada y reconocida de todas las soluciones existentes en la literatura. Según la visión de Reiter y Dale, la generación de expresiones de referencia se puede dividir en tres procesos separados: la elección de qué nivel de abstracción se considerará al generar la referencia, la selección de atributos que permitirán distinguir a la entidad referenciada del resto que la rodean, y la lexicalización de la referencia escogiendo qué palabras y expresiones se utilizarán para traducir la información escogida a texto.

En este capítulo se cubren los objetivos 1 y 2 planteados en la sección 1.5. En la sección 4.1 se trata el problema de la elección del nivel de abstracción de la referencia a generar, y en las secciones 4.2 y 4.3 se presentan diversas soluciones para la selección de atributos y lexicalización. Algunas de estas soluciones son además evaluadas en el marco de una serie de tareas de evaluación competitivas existentes en el campo (sección 4.4).

4.1. Elección de Nivel de Abstracción de las Referencias

La mayoría de las soluciones para generación de expresiones de referencia en forma de frase nominal se basan en la adicción de modificadores al tipo de la entidad referenciada para que ésta pueda ser distinguida de las que la rodean. En general el tipo de la referencia se considera dado. Pero cuando se dispone de una base de conocimiento en forma de taxonomía que contiene las relaciones entre los tipos de las entidades, es posible mejorar estas soluciones si se escoge el tipo de la entidad en función de la situación en la que se encuentra la misma.

Mientras que la primera aproximación (selección de atributos) ha sido estudiada a menudo en la literatura, la segunda es generalmente pasada por alto asumiendo que los niveles de la taxonomía en los que se enmarcará cada elemento vendrán especificados en la entrada del sistema. Esto permite a la mayoría de las soluciones existentes concentrarse en la tarea de la selección de atributos. Sin embargo, el tener que escoger en qué nivel de la jerarquía se está desarrollando el discurso, y es por tanto el más adecuado para expresar una referencia, aparece un nuevo problema. Si todos los elementos del contexto de trabajo están clasificados en una taxonomía con una raíz única (comúnmente `Thing`), y se establece que la referencia debe ser realizada en un nivel alto de la jerarquía, ocurriría que prácticamente todo elemento del contexto puede ser un distractor para cualquier otro elemento. En este caso la tarea de construir el conjunto de contraste debe ser dinámica y estar probablemente acoplada con la tarea de escoger cuál es el nivel correcto de la taxonomía que debe usarse para establecer cada referencia.

A modo de ejemplo, imaginemos una habitación con los siguientes elementos:

```
Element1 = {<type,sofa>,<size,big>}
Element2 = {<type,doberman>,<size,big>}
Element3 = {<type,chihuahua>,<size,small>}
```

La taxonomía en la Figura 4.1 contiene toda la información conocida sobre el contexto que corresponde a este ejemplo.

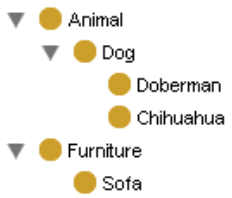

Figura 4.1: Taxonomía que corresponde al ejemplo

Si deseamos referirnos a `Element2` una referencia válida podría ser *the doberman* ya que el tipo del referente es suficiente para distinguir a este elemento de los demás con los que se puede confundir. En este caso el conjunto de contraste podría haber estado formado sólo por `Element2`, por los dos perros, o tal vez por los tres elementos. El resultado habría sido el mismo en todos los casos. Imaginemos ahora que `Element3` desaparece de la habitación. *The doberman* seguiría siendo una referencia distintiva para `Element2` pero contendría información adicional que podría ser irrelevante (la raza del perro). Usando la taxonomía de la Figura 4.1 es posible generar una referencia más adecuada teniendo en cuenta el tipo de los demás elementos que hay en la habitación. En este caso, referencias más apropiadas serían *the dog* o incluso *the animal*. Para poder generar este tipo de referencias el conjunto de contraste no debe estar restringido sólo a los elementos del mismo tipo que la entidad objetivo, sino que también debe incluir elementos de tipos relacionados. En nuestro ejemplo, el conjunto de contraste debe contener todos los elementos de la habitación de manera que la información de la ontología pueda ser usada para decidir a qué nivel se escoge el tipo de un referente dado.

4.1.1. Generación de Referencias Usando Ontologías para Determinar el Nivel de Abstracción

En esta sección se discute el uso de ontologías para apoyar la tarea de generación de expresiones de referencia. Se prestará especial atención a la elección de los distractores que deben ser tenidos en cuenta en el conjunto de contraste, y al uso de la información de la ontología para seleccionar el tipo más apropiado a utilizar para el referente.

Nos hemos centrado para ello en la generación de frases nominales definidas donde el tipo de un elemento y un conjunto de sus propiedades se usan para distinguirlo de otros elementos en el foco de atención. Se supone además que las situaciones en que se están generando las referencias son estáticas, es decir, la percepción del mundo por parte del oyente o lector no cambia durante el proceso de generación de la referencia.

Para probar las ideas aquí presentadas se ha utilizado una ontología sobre vinos. La idea era generar referencias a diferentes instancias de vinos que aparecían juntas en un discurso. El primer paso consiste en seleccionar el conjunto de distractores o conjunto de contraste para el referente específico al que nos estamos refiriendo. Después se aplica un algoritmo para decidir cuál es el tipo más adecuado para la referencia. Se ha considerado que el mejor tipo posible es aquel que distingue al referente del resto de distractores mientras que al mismo tiempo es lo más general posible para no dar lugar a falsas implicaciones. Por ejemplo, si nos estamos refiriendo a una instancia de vino `Chardonnay` (que es un vino blanco) en una situación en la que el resto de vinos son todos tintos, la referencia más adecuada sería *the white wine* y

no *the chardonnay*. Si lo hiciéramos al contrario, la referencia más específica (pero innecesaria) podría hacer que el oyente infiriera que esa información extra es de alguna manera relevante.

Finalmente, si el tipo escogido no es suficiente para distinguir al referente del resto del conjunto de contraste, se aplica selección de atributos para seleccionar un subconjunto de propiedades del elemento que sí permitan distinguirle.

Ontología de Ejemplo

Para probar las ideas aquí presentadas se ha usado una ontología sobre vinos anteriormente desarrollada. Es una ontología de ejemplo usada en los documentos de especificación de OWL, e implementada siguiendo una versión publicada por Brachman et al. (1991) y distribuida con el sistema de representación de conocimiento CLASSIC. Esta ontología no sólo contiene información sobre vinos y conceptos relacionados con ellos. También dispone de información sobre comida ya que estaba inicialmente pensada como base de conocimiento para un agente de la web semántica que recibía una descripción de una comida y recuperaba una selección de vinos de la web que combinaban con dicha comida (Hsu y McGuinness, 2003).

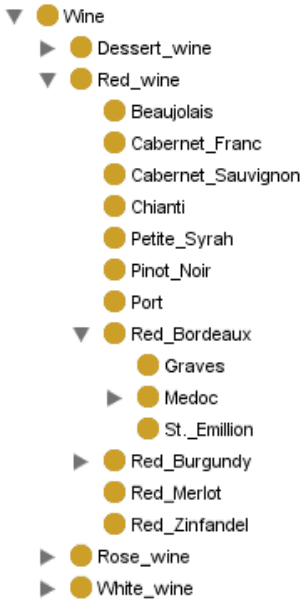

Figura 4.2: Extracto de la ontología sobre vinos

Nos hemos centrado en la taxonomía de vinos proporcionada por la ontología. Los vinos están divididos en cuatro tipos básicos: `Red_Wine`, `White_Wine`, `Rose_Wine` and `Dessert_Wine`, aunque las instancias de estos últimos

también pertenecen a alguno de los otros tipos. A partir de estos tipos principales se tiene una completa taxonomía sobre diferentes tipos de vinos. Además de esta taxonomía, la ontología también contiene conceptos relacionados con vinos como regiones en las que se producen, tipos de uva y una lista de productores. Se muestra en la Figura 4.2 un trozo de la taxonomía de conceptos.

La ontología contiene además numerosas instancias para los diferentes conceptos. Cada una de estas instancias está descrita usando características tales como cuerpo, color, sabor, productor, etc. En la ontología estas características están representadas como relaciones entre instancias, donde los diferentes tipos de colores, cuerpos y demás son también instancias de conceptos en la ontología. En la Figura 4.3 se muestra un fragmento de la información disponible para una de las instancias de vino.

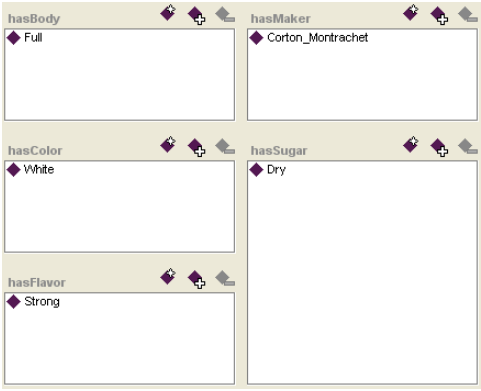

Figura 4.3: Parte de la información disponible para el vino `Corton Montrachet White Burgundy`

Componiendo el Conjunto de Contraste

La información sobre el tipo de una entidad se usa generalmente para determinar qué elementos del mundo pueden ser considerados como conjunto de contraste. En los algoritmos de generación de expresiones de referencia que producen frases nominales formados por el tipo del referente y un conjunto de atributos, el conjunto de contraste suele estar formado por los elementos del mundo que tienen el mismo tipo que el referente objetivo.

En nuestro caso, toda la información sobre el mundo está localizada en la ontología. Cada instancia del mundo contenida en ella tiene un tipo directo (el concepto más específico al que pertenece), y un conjunto de tipos indirectos que son todos los tipos entre el tipo directo y la raíz de la ontología (que es generalmente el concepto `Thing`).

Con el conocimiento representado de esta manera, no resulta trivial decidir qué tipo del referente es apropiado usar para componer el conjunto de

contraste. Si el tipo escogido es demasiado específico, el conjunto de contraste debería ser el referente al que se desea referenciar y tal vez unas cuantas instancias más del mismo tipo directo. Probablemente esto no sería suficiente para una generación de referencias apropiada en un contexto más amplio. Si el tipo escogido es demasiado general, el conjunto de contraste estaría compuesto por todas las instancias de la ontología.

Aunque pueda verse como una aproximación bastante general, en el trabajo desarrollado hemos observado que el uso de la ontología completa como conjunto de contraste es la opción más adecuada en la mayoría de las situaciones. Como veremos más adelante, esta elección evita el uso de referencias más específicas de lo deseado mientras que al mismo tiempo permite al algoritmo escoger el tipo que es más adecuado utilizar para una determinada situación.

Tipo Apropiado para un Referente

Como muchos otros algoritmos de generación de expresiones de referencia, nuestra aproximación toma como atributo distinguible inicial el tipo de los elementos que aparecen en el mundo. Este tipo de solución es suficiente cuando el tipo definido para cada una de las entidades del mundo es fijo y no hay una relación cercana entre los diferentes tipos. Por ejemplo, una solución que toma el tipo estricto definido para un elemento de la base de conocimiento no consideraría que un *doberman* y un *chihuahua* son al mismo tiempo *perros*.

Algunos algoritmos de generación de referencias se enfrentan a este problema proporcionando algún tipo de valor básico o *basic level value* para cada atributo, considerando este valor como el más general conocido por el oyente o lector. Por ejemplo, en el algoritmo incremental de Reiter y Dale (sección 2.2.5) los valores `Doberman` y `Chihuahua` serían subsumidos por `Dog`, siendo este valor básico accesible cuando fuera necesario.

Como ya se ha comentado, en nuestro caso toda la información sobre el mundo está localizada en una ontología. Esta ontología contiene una taxonomía de conceptos que tienen una raíz común llamada `Thing`. La solución más simple podría ser considerar el concepto del que hereda directamente cada instancia como su tipo, pero esto no nos permitiría aprovechar toda la información que nos proporciona la ontología.

El algoritmo que hemos implementado puede verse en la Figura 4.4. El identificador r corresponde al referente objetivo, C es el conjunto de contraste, A es la lista de atributos que se pueden aplicar a las instancias de la ontología, *typeValue* es el tipo que sería asignado al referente por el algoritmo, y L es la lista de pares atributo-valor devueltos si el tipo no es suficiente para eliminar a todos los distractores. La función `rules-out` funciona igual que la del Algoritmo Incremental (sección 2.2.5), y la función

`incremental-algorithm` llama directamente al algoritmo original de Reiter y Dale.

La función `find-best-value-type` es la que devuelve el tipo más apropiado para el referente objetivo r teniendo en cuenta la información disponible en la ontología. Se ha considerado que el nivel del valor básico para el tipo será el tipo más específico de todos los tipos comunes de las instancias de la ontología. A partir de este valor de nivel básico, la rama de conceptos entre él y el tipo directo del referente r es explorada. El tipo que se usará entonces en la referencia es el concepto más general de esta rama que descarte el mayor número de distractores.

```
make-referring-expression ( r, C , A )

L ← {}
C ← instances-ontology()
typeValue ← find-best-value-type( r, C )
C ← C – rules-out( < type , typeValue > )
if C = {} then
   return < typeValue, {} >
else
   L ← incremental-algorithm( r , C , A )
   return < typeValue, L >
endif
```

```
find-best-value-type( r, C )

basic-level-type ← most-specific-common-type( r ∪ C )
value ← basic-level-type
for v_i ∈ ontology-children(basic-level-type)
   if | rules-out( < type,v_i >) |
      > | rules-out( < type,value >) | then
      value ← v_i
   endif
next
return value
```

Figura 4.4: Algoritmo para la elección del nivel de abstracción

Selección de Atributos para Completar las Referencias

En algunos casos el tipo no será suficiente para distinguir dos o más referentes del mundo. Esta situación se produce cuando varios referentes pertenecen exactamente al mismo concepto de la ontología. En estos casos será necesario usar sus propiedades para distinguir entre ellos.

En nuestro caso se ha aplicado a estos casos la selección de atributos según el algoritmo incremental de Reiter y Dale. Las propiedades de las instancias (expresadas en pares atributo-valor) que son del mismo tipo son extraídas de la ontología. La selección de atributos se aplica sobre ellas sin ningún orden predefinido ya que al estar proponiendo una solución genérica que no debe depender del dominio no es posible dar ningún orden a estos atributos.

Algunos Ejemplos

El algoritmo implementado se ha probado sobre diferentes situaciones en las que aparece cierto conjunto de vinos. Para cada una de ellas se proporciona una referencia que les distingue unívocamente usando el tipo más

	Name of the instance	Types	Properties			
			Body	Color	Flavor	Sugar
1	Corban_Dry_White_Riesling	Wine White_wine Riesling Dry_Riesling Corbans_Dry_White_Riesling Chardonnay Bancroft_Chardonnay Rose_wine White_Merlot Forest_Glen_White_Merlot_Rose Red_wine Cabernet_Sauvignon Marietta_Cabernet_Sauvignon Forman_Cabernet_Sauvignon				
2	Bancroft_Chardonnay					
3	Marietta_Cabernet_Sauvignon		Medium	Red	Moderate	Dry
4	Forman_Cabernet_Sauvignon		Medium	Red	Strong	Dry
5	Forest_Glen_White_Merlot_Rose					

Tabla 4.1: Ejemplos de instancias de vinos

apropiado encontrado usando la ontología, y un conjunto de atributos cuando sea necesario.

Las instancias del mundo que han sido consideradas se muestran en la Tabla 4.1 (las propiedades de los vinos que no han sido usadas por el algoritmo no se muestran). Las referencias generadas para cada uno de los referentes son:

1. *The Riesling.* Se puede ver que hay otro vino blanco pero no uno que sea `Dry_Riesling`, así que el tipo más general que descarta más distractores es `Riesling`.

2. *The Chardonnay.* En esta situación hay otro vino blanco, pero el tipo `Chardonnay` es suficiente para eliminar los otros distractores.

3. *The moderate Cabernet_Sauvignon.* En este caso el tipo no es suficiente para distinguir a este referente, así que sus atributos han sido usados también. La propiedad que lo diferencia del otro `Cabernet_Sauvignon` es el sabor moderado.

4. *The strong Cabernet_Sauvignon.* Como en el caso anterior el sabor fuerte se usa para distinguir este vino del otro que también es `Cabernet_Sauvignon`.

5. *The Rose_Wine.* En este caso no hay más vinos rosados, así que este tipo genérico es suficiente para distinguir al referente y no es necesario ser más específico.

Discusión

La principal ventaja de esta solución es que el algoritmo siempre encuentra el valor más adecuado para el tipo teniendo en cuenta el resto de

entidades en el contexto. Como se puede ver en los ejemplos, si sólo hay un vino rosado el sistema no dará información redundante sobre a qué tipo de vino rosado pertenece. Dado que esta solución es completamente genérica e independiente del dominio, el algoritmo funcionará de la misma manera con ontologías más generales. Por ejemplo, si la ontología utilizada contuviera información no sólo sobre vinos sino también sobre otros tipos de bebidas, los valores que serían usados para los tipos de los referentes se elegirían de la misma manera. En este tipo de situación el referente podría ser el único vino entre otras bebidas, y el tipo seleccionado para él sería la más apropiada: *the wine*.

En la definición del algoritmo incremental Reiter y Dale no se ocupan de la cuestión de cómo se construye el conjunto de contraste o como su contenido puede ser actualizado durante el proceso de generación. Por ello consideran que el conjunto de contraste es una de las entradas del algoritmo. En nuestro trabajo se ha escogido como conjunto de contraste todas las instancias que pueden encontrarse en la ontología. Esta solución permite al algoritmo trabajar con suficiente información para elegir exactamente a qué nivel de la ontología se debe desarrollar el discurso (más general o más específico). Con esta información las referencias generadas están adaptadas al nivel de especificidad requerido en cada caso.

4.2. Selección de Atributos como Problema de Búsqueda

En el caso en el que se está realizando una referencia a una entidad en forma de frase nominal, la tarea de selección de atributos consiste en decidir qué modificadores de los que se pueden aplicar a dicha entidad se pueden añadir a la referencia. Existen diversas razones por las que los seres humanos incluyen o no ciertos atributos de las entidades a las que se están refiriendo. En general se tiende a que las referencias no sean excesivamente largas y detalladas, sino a que contengan el mínimo de información que es necesario para la correcta identificación del referente, y sin embargo hay muchas situaciones en las que se tiende a utilizar información redundante si eso puede facilitar la comprensión. También existen diversas cuestiones de estilo o situación que pueden hacer que las referencias generadas en una misma situación sean muy distintas dependiendo de la persona que las está realizando. Éstas, y otras muchas cuestiones, hacen que el proceso de selección de atributos tal y cómo lo llevan a cabo los seres humanos esté lejos de estar definido de una manera determinista que permita implementar algoritmos que lo simulen.

Pese a estas limitaciones, el problema de la selección de atributos se puede considerar como un problema de búsqueda, ya que consiste en encontrar qué conjunto de todos los subconjuntos posibles que se pueden formar con

los atributos que describen a una determinada entidad es el más apropiado para referirse a ella. En (Bohnet y Dale, 2004) los autores realizan una definición formal de la generación de expresiones de referencia nominales como un problema de búsqueda, y utilizando esta definición formalizan como algoritmos de búsqueda los algoritmos más importantes de la literatura en este campo. Sin embargo, no abordan la cuestión de qué paradigmas de resolución de problemas basados en búsqueda pueden beneficiar a la tarea de selección de atributos.

En esta sección se presentan tres algoritmos diferentes para la selección de atributos como parte de la tarea completa de generación de expresiones de referencia. Los dos primeros se basan en buscar distintos órdenes de evaluación de los atributos a la hora de ser escogidos para aparecer en una referencia. Dado que los atributos para crear la referencia se van comprobando en orden, parece lógico pensar que el orden en que se coloquen los mismos tendrá repercusiones sobre la solución final. Para estudiar qué orden de los atributos es más adecuado se usará un corpus de expresiones de referencia generadas por humanos. El último de estos algoritmos se basa en Algoritmos Evolutivos para realizar dicha búsqueda. Dicho paradigma parece apropiado para este problema ya que en lugar de codificar cómo se encuentra la solución al problema, se basa en cómo se puede describir que una solución es apropiada.

Cada uno de estos algoritmos se ha evaluado utilizando las métricas clásicas en este campo, y tomando como corpus de referencia el corpus TUNA (sección 2.3.1). Para la implementación de las soluciones específicas para la selección de atributos que se exponen en esta sección, el fragmento del módulo de resolución de referencias de TAP que se encargaba de la selección de atributos fue separado de las demás decisiones a tomar. Por tanto, no se tienen en cuenta ni los pronombres ni las referencias definidas e indefinidas en este submódulo.

4.2.1. Selección de Atributos basada en Agrupamientos Relativos de los Mismos

Dada la importancia que tiene el orden de los atributos del referente objetivo a la hora de realizar la selección de atributos según el algoritmo de Reiter y Dale, se decidió estudiar qué tipos de órdenes utilizan los seres humanos cuando se refieren a elementos de su entorno. Para ello se tomó como referencia el corpus TUNA (sección 2.3.1), considerándolo un ejemplo de cómo los seres humanos seleccionan los atributos a la hora de crear una referencia. Para comparar los resultados obtenidos con los del corpus se utilizó el coeficiente Dice (sección 2.3.1).

Análisis del Corpus

Para determinar de qué manera el orden de los atributos influye en la generación de referencias, los datos de entrenamiento del corpus TUNA fueron estudiados por separado dependiendo del dominio (mobiliario vs. personas). La idea era que no sólo el conjunto de atributos a tener en cuenta en ambos dominios era muy distintos, sino que probablemente las consideraciones psicológicas que tiene en cuenta una persona a la hora de referirse a un mueble o a una persona son significativamente diferentes.

Siguiendo esta idea, se consideró el uso del atributo tipo (*type*) como un atributo adicional con poder de distinción cuando se estaban generando descripciones en el dominio de los muebles, pero no en el dominio de las personas (donde todos los elementos que aparecen en las entradas son de tipo *person*).

En el dominio de los muebles había que trabajar con un conjunto de seis atributos. Todas las posibles combinaciones de los mismos en diferentes órdenes nos daban $6! = 720$ posibilidades a explorar. Esto pertenece a un orden de magnitud abordable computacionalmente, así que generamos automáticamente todos los posibles órdenes de atributos y para cada uno de ellos ejecutamos el proceso completo de realizar la selección de atributos correspondiente con todos los ejemplos del corpus de entrenamiento. La media de los valores del coeficiente Dice obtenidos fue calculada para cada caso.

El estudio de estos resultados reveló qué combinaciones de atributos daban los mejores resultados. Pero además también mostraron una peculiaridad en la forma en que la calidad de los resultados dependía del orden de consideración de los atributos, ya que parecía que los resultados dependían del orden relativo de ciertos grupos de atributos más que del orden de todos los atributos en general. En otras palabras, los resultados eran casi los mismos para ciertos órdenes de grupos dentro de los atributos, independientemente del orden interno dentro de dichos grupos.

En el dominio del mobiliario los grupos identificados fueron:

- [*colour*, *type*, *size*]
- [*orientation*, *x-dimension*, *y-dimension*]

Esta distinción tiene cierta coherencia si consideramos que uno de los grupos está más relacionado con la situación espacial del objeto, y los otros con sus propias características. Parece posible que cada persona se sienta más cómoda usando uno u otro, dependiendo de su visión general del mundo que le rodea.

Como ya se mencionó anteriormente, en el dominio de las personas el tipo no fue considerado como un atributo distintivo ya que siempre es el

mismo (*person*) para todos los referentes considerados. Esto nos dejaba con 11 atributos a tener en cuenta, y 11! = 39.916.800 órdenes posibles para ellos, un orden de magnitud inabordable computacionalmente.

Siguiendo la intuición sobre los agrupamientos de los atributos extraída de los resultados sobre el dominio de los muebles, llevamos a cabo diferentes experimentos creando distintas combinaciones de los atributos dados. Nuestra primera aproximación fue dividir los atributos en tres conjuntos agrupados según su relación con las personas:

- [*hasShirt*, *hasGlasses*, *hasSuit*, *hasTie*] (relacionados con la ropa)
- [*hasBeard*, *hairColour*, *hasHair*, *age*] (relacionados con la apariencia)
- [*x-dimension*, *y-dimension*, *orientation*] (relacionados con la situación espacial)

cada uno de ellos conteniendo atributos relacionados semánticamente. Muchas combinaciones diferentes del orden de los grupos y de los elementos dentro de ellos fueron probados, pero los resultados obtenidos con estas divisiones no fueron muy buenos.

Así que se intentó otra aproximación agrupando los atributos dependiendo de la relevancia que su presencia o ausencia tienen para distinguir a una persona de otra. Por ejemplo, tener barba o llevar gafas son generalmente más fáciles de percibir que llevar corbata (especialmente si la persona lleva además un traje). Cuatro nuevos grupos se usaron para los experimentos:

- [*hasSuit*, *hasTie*, *hasShirt*]
- [*hasBeard*, *hasGlasses*, *hasHair*, *hairColour*]
- [*age*]
- [*x-dimension*, *y-dimension*, *orientation*]

obteniendo resultados más altos que con los grupos anteriores.

Modificaciones sobre el Algoritmo

Como ya se ha comentado, el módulo de resolución de referencias fue aislado del resto del flujo de control de TAP para poder utilizar su funcionalidad de selección de atributos por separado. Se creó una interfaz especial de manera que los datos de entrada del módulo pudieran ser directamente las representaciones del corpus que contienen un objetivo y un conjunto de distractores, todos ellos representados por sus atributos.

Se modificó además el módulo para aceptar no sólo un referente objetivo y un conjunto de distractores, sino también un orden particular en el que

considerar los atributos que son usados en el proceso de generar la expresión de referencia. Este orden de los atributos se corresponde con la lista de *preferred attributes* que mencionaban Reiter y Dale. La selección de atributos es así completamente dependiente del orden en el que los atributos disponibles son considerados.

Evaluación del Algoritmo

Para evaluar el algoritmo consideramos tres dimensiones de evaluación: identificación, minimalidad y concordancia sistema-evaluador humano calculada usando el coeficiente Dice.

El módulo de resolución de referencias inicial estaba previamente implementado para generar siempre descripciones de las entidades del discurso que identificaran unívocamente a la entidad objetivo entre el resto con las que se pudiera confundir. De esta forma, no fue necesaria ninguna modificación del módulo para ajustarse a la evaluación de la *identificación*.

Para comprobar la adecuación de las *referencias mínimas* respecto a las correspondientes al corpus, se probó a definir el orden de los atributos usando el algoritmo de Máxima Brevedad o *Full Brevity* (Dale, 1989). De acuerdo con este algoritmo, la lista de atributos a ser considerados para crear una referencia que permita distinguir unívocamente a una entidad objetivo pueden ser ordenados por su poder discriminatorio. Una vez conseguido este orden (de más discriminatorio a menos) se usan en orden como en otros algoritmos hasta que la entidad es distinguida unívocamente por la referencia generada.

Los primeros experimentos realizados demostraron que no era posible obtener buenos resultados simultáneamente para las evaluaciones de minimalidad y coincidencia con el corpus. En realidad este resultado no es sorprendente ya que los seres humanos no siempre usan la expresión óptima cuando se están refiriendo a algo, sino que tienden a usar información redundante si eso hace que la referencia sea más fácil de formular o entender.

En la Tabla 4.2 se pueden ver los resultados obtenidos para el coeficiente Dice (sobre los datos de entrenamiento) usando el módulo de selección de atributos ajustado para producir los conjuntos mínimos de atributos para las referencias. Se puede ver que los porcentajes obtenidos son bastante pequeños.

	Referencias Mínimas	**Dice**
Mobiliario	100,00 %	24,33 %
Personas	100,00 %	31,33 %

Tabla 4.2: Resultados obtenidos usando el algoritmo para la mínima referencia sobre los datos de entrenamiento

A la vista de estos resultados, decidimos concentrar nuestros esfuerzos en mejorar las medidas del coeficiente Dice (semejanza de las referencias generadas con las del corpus) sin tener en cuenta si el orden considerado para los atributos podía producir referencias mínimas. Tanto para el dominio de los muebles como para el de las personas, se utilizó como orden para la lista de atributos de entrada el que había dado los mejores resultados al analizar el corpus, y se evaluaron las proporciones de referencias mínimas y el valor del coeficiente Dice como se requería en la competición.

Los mejores resultados obtenidos (sobre los datos de entrenamiento) en ambos dominios se muestran en la Tabla 4.3. El mejor orden encontrado para los atributos en el dominio del mobiliario fue:

$$[\textit{type}, \textit{colour}, \textit{size}] + [\textit{orientation}, \textit{x-dimension}, \textit{y-dimension}]$$

El orden de los atributos usado en el dominio de las personas fue:

$$[\textit{hasGlasses}, \textit{hasBeard}, \textit{hairColour}, \textit{hasHair}]+[\textit{hasSuit}, \textit{hasTie}, \textit{hasShirt}]+ [\textit{age}] + [\textit{x-dimension}, \textit{y-dimension}, \textit{orientation}]$$

	Referencias Mínimas	**Dice**
Mobiliario	0,00 %	82,45 %
Personas	42,72 %	43,57 %

Tabla 4.3: Mejores resultados obtenidos sobre los datos de entrenamiento

Los resultados obtenidos sobre los datos de entrenamiento para la evaluación de expresiones mínimas merecen ser revisados. En el caso del dominio de los muebles, en general los valores del coeficiente Dice fueron altos, mientras que los que correspondían a expresiones mínimas fueron muy bajos. Sin embargo, ambos resultados fueron muy similares en el dominio de las personas.

Nuestra impresión es que estas diferencias se deben al número de atributos a considerar en cada uno de los dominios. En el caso de los muebles, con sólo seis atributos es difícil para una persona encontrar inmediatamente la referencia mínima entre seis distractores. Sin embargo, en el caso de las personas, donde se tienen once atributos y sólo seis distractores, es más fácil encontrarse con que la entidad objetivo tiene uno o dos atributos que la distinguen unívocamente del resto de elementos, y que además éstos sean fáciles de encontrar por la persona que está produciendo la referencia.

4.2.2. Selección de Atributos dependiendo del Valor más Frecuente

Después de los resultados obtenidos para el algoritmo anterior (sección 4.2.1), que demostraban que el orden de los atributos a considerar para

generar una referencia es importante, se realizó una nueva aproximación. En contraste con el algoritmos de Reiter y Dale, buscamos una solución en la que el orden en el que considerar los atributos no fuera siempre el mismo, sino que dependiera de la entidad para la que se fuera a generar la referencia. Más concretamente se tienen en cuenta los valores de los atributos a la hora de decidir su orden.

La idea subyacente es que los valores más o menos sobresalientes para un atributo pueden influir también en su elección a la hora de generar una descripción. Por ejemplo, el que una persona sea calva llama mucho más la atención que el que no lo sea, de manera que un atributo como el pelo se tenderá a usar sólo cuando tenga como valor algo que llame la atención, en este ejemplo, no tenerlo.

El algoritmo de selección de atributos funcionaría entonces como el de Reiter y Dale, iterando sobre la lista de atributos preferidos hasta que se consigue una descripción para el referente que lo identifica unívocamente respecto a los demás con los que se puede confundir. Sin embargo, para cada entidad se calcula un orden diferente de los atributos a considerar. Para ello se requiere algún tipo de base de conocimiento que contenga información sobre la relevancia de ciertos valores de los atributos respecto a otros. La lista ordenada de atributos preferidos se calcularía añadiendo al principio los atributos que tienen valores para la entidad a describir más relevantes.

Análisis del Corpus

Para configurar el módulo de manera que cumpliera los requerimientos de la solución propuesta, se estudiaron los datos de entrenamiento para obtener la probabilidad de aparición de cada atributo cuando el referente objetivo tenía un valor específico para el mismo. Esta probabilidad se calculó usando la Fórmula 4.1:

$$prob_{val_i} = \frac{\sum appsValueInAttSet}{\sum appsValueInTarget} \tag{4.1}$$

Para cada posible valor de cada uno de los atributos de los dominios del corpus TUNA, se calculan la suma de las apariciones del valor en los elementos `ATTRIBUTE-SET` de la respuesta (*appsValueInAttSet*), y la suma de las apariciones de este valor en los atributos de los referentes objetivos (*appsValueInTarget*). La división de estos dos valores es la probabilidad de mencionar un atributo del referente objetivo cuando tiene un valor específico.

Las probabilidades obtenidas sobre los datos de entrenamiento se pueden ver en la Tabla 4.4. Por ejemplo, se puede ver que el atributo `hasGlasses` en el dominio de las personas se menciona en el 60 % de las situaciones cuando su valor es `1`, y en el 0 % de las situaciones cuando su valor es `0`. Sin embargo, el atributo `hasShirt` casi nunca se menciona (0.8 % de las veces cuando su

valor es `1`, y en un 0 % con valor `0`). En el caso de los muebles parece que los valores de los atributos no influyen tanto en la decisión, aunque se puede ver que atributos como el tipo o el color se utilizan muy a menudo sea cual sea su valor.

Mobiliario			**Personas**		
X-DIMENSION	*1*	16.45 %	X-DIMENSION	*1*	35.71 %
	2	18.18 %		*2*	25.92 %
	3	24.59 %		*3*	37.50 %
	4	30.76 %		*4*	15.68 %
	5	18.64 %		*5*	26.00 %
Y-DIMENSION	*1*	33.65 %	Y-DIMENSION	*1*	39.24 %
	2	16.32 %		*2*	23.71 %
	3	29.56 %		*3*	40.25 %
ORIENTATION	*back*	49.01 %	ORIENTATION	*right*	2.43 %
	right	29.47 %		*front*	2.25 %
	front	35.29 %		*left*	1.26 %
	left	25.58 %	TYPE	*person*	83.79 %
SIZE	*small*	43.07 %	HASHAIR	*0*	15.78 %
	large	32.62 %		*1*	14.41 %
TYPE	*chair*	94.40 %	HASBEARD	*0*	0.00 %
	desk	85.88 %		*1*	65.68 %
	fan	97.61 %	HASGLASSES	*0*	0.00 %
	sofa	91.48 %		*1*	60.35 %
COLOUR	*blue*	87.20 %	HAIRCOLOUR	*light*	27.43 %
	green	87.50 %		*dark*	26.96 %
	grey	83.33 %	AGE	*young*	0.00 %
	red	86.31 %		*old*	0.00 %
			HASSHIRT	*0*	0.00 %
				1	0.80 %
			HASSUIT	*0*	0.00 %
				1	4.59 %
			HASTIE	*0*	0.00 %
				1	2.29 %

Tabla 4.4: Probabilidades de aparición para los valores de los atributos

Modificaciones sobre el Algoritmo

El algoritmo funciona igual el selector de atributos original, pero antes de comenzar la ejecución se determina dinámicamente el orden de la lista de los atributos a considerar dependiendo del referente objetivo. En la Figura 4.5 se puede ver un esquema del funcionamiento del algoritmo.

La única excepción considerada en el algoritmo es el atributo `type` en el dominio de las personas. Como todas las entidades en este dominio son de tipo *person*, el selector de atributos no escogería este atributo porque ningún distractor es descartado por él. Sin embargo, en los experimentos se ha observado que en el corpus hay muchas descripciones que incluyen el tipo *person* incluso siendo redundante. Siguiendo esta idea, nuestro algoritmo siempre incluye el atributo `type` en la lista de atributos escogidos en el dominio de las personas.

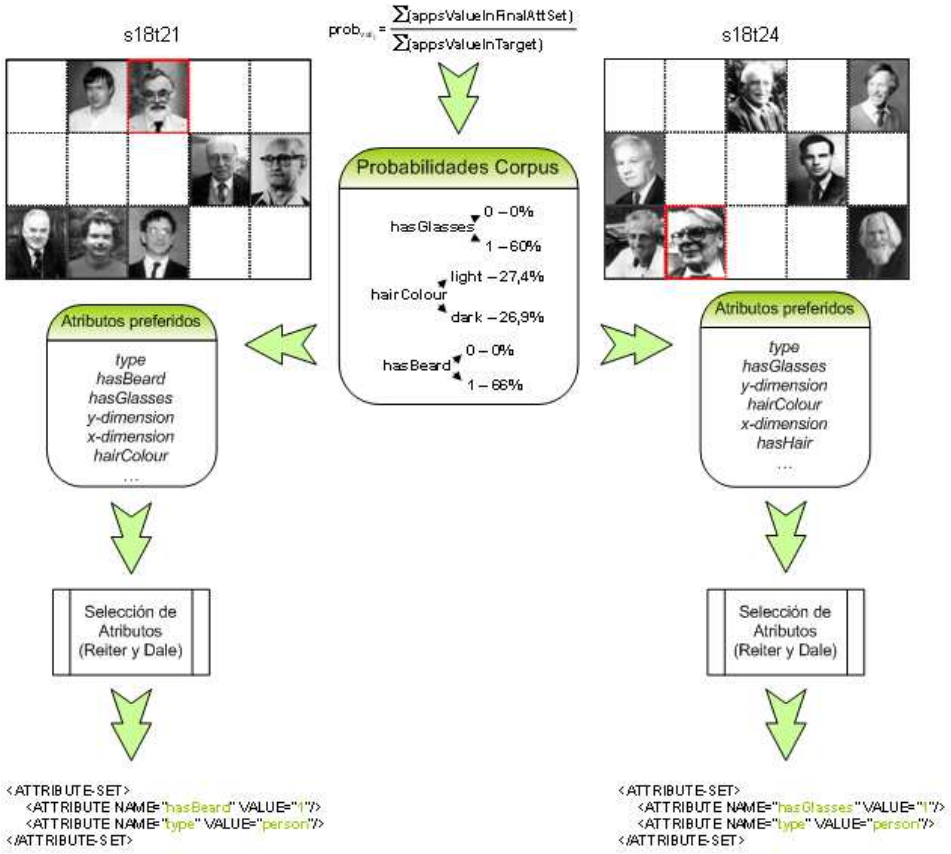

Figura 4.5: Esquema de funcionamiento del algoritmo basado en el valor más frecuente

Evaluación del Algoritmo

Se consideraron cinco métricas para evaluar el algoritmo: identificación, minimalidad, precisión, Dice y MASI. El módulo de selección de atributos de TAP ya cumplía el criterio de evaluación de la *identificación*, ya que todas las referencias generadas garantizan que la entidad referenciada no puede ser confundida con las que la rodean. Teniendo en cuenta además que en la

tarea competitiva anterior se pudo comprobar que era muy difícil conseguir buenos resultados en los valores Dice y de *minimalidad* al mismo tiempo como vimos en la sección 4.2.1, esta vez se concentraron los esfuerzos en obtener buenos resultados para el coeficiente Dice (y para el MASI, que está muy relacionado), en lugar de buscar referencias mínimas. También optamos por no usar la evaluación de *precisión* (que consistía en conseguir referencias exactamente iguales a las del corpus), ya que el corpus contiene un amplio conjunto de estilos diferentes a la hora de generar las referencias y estábamos más interesados en mejorar el sistema para todos los estilos en lugar de acoplarlo sólo con ciertas formas de generar referencias.

Se muestran en la Tabla 4.5 los resultados obtenidos con este algoritmo sobre los datos de entrenamiento. Se pueden ver por separado los resultados por dominios y en conjunto, y comprobar que los resultados obtenidos son más que aceptables.

	Dice	**MASI**	**Precisión**	**Identif.**	**Minimalidad**
Muebles	79,18 %	0,57	41,69 %	100 %	0 %
Personas	69,71 %	0,42	22,99 %	100 %	0 %
Ambos	74,80 %	0,50	34,81 %	100 %	0 %

Tabla 4.5: Resultados de evaluación para los datos de entrenamiento

4.2.3. Búsqueda de los Atributos usando Algoritmos Evolutivos

Uno de los paradigmas de resolución de problemas de búsqueda que resulta muy adecuado cuando no se puede definir algorítmicamente el proceso a seguir para resolver un problema son los Algoritmos Evolutivos (AEs) (sección 3.1). El objetivo es evolucionar posibles soluciones para el problema, evaluar cuáles son las mejores, y al final obtener la solución aparecida que mejor lo resuelve. El proceso de búsqueda se guía escogiendo un conjunto de operadores para transformar las soluciones en otras similares pero más aptas, y elegir un conjunto de funciones de aptitud que especifican el mínimo conjunto de restricciones para las soluciones válidas.

En el caso que nos ocupa proponemos el uso de Algoritmos Evolutivos para resolver la tarea de selección de atributos, ya que esta tarea se puede considerar un proceso de búsqueda en el que no están claras las heurísticas a utilizar para obtener las soluciones. La búsqueda en este caso estará guiada por la función de aptitud, en la que se debe codificar lo que se considera una expresión de referencia correcta y adecuada.

En la implementación de cualquier algoritmo evolutivo existen tres elementos básicos a tener en cuenta:

- *Población de individuos*, que son una representación (no necesariamente directa) de posibles soluciones.
- *Procedimiento de transformación*, para construir nuevos individuos a partir de los anteriores. Para ello se utilizan los operadores genéticos, que actuarán sobre los individuos de la población de forma aleatoria.
- *Procedimiento de selección*, basado en la aptitud de los individuos para resolver el problema. Según esta aptitud los individuos más cercanos a la solución, que serán los de mayor valor de aptitud, tendrán más posibilidades de pasar a las siguientes generaciones.

En el caso de nuestro algoritmo, se han escogido como individuos de la población los conjuntos de atributos que optan a formar la referencia final, y como genes la aparición o no de cada uno de estos atributos en la referencia a la entidad objetivo. El sistema trabajará sobre esta población durante un número de generaciones determinado por el usuario. Para transformar la población en cada generación se usarán dos operadores genéticos distintos: cruce y mutación. Una vez transformada la población, todos los individuos (que corresponden a referencias) de la misma serán evaluados, y sólo los más adaptados (con mayor función de aptitud, en este caso referencias más adecuadas) tendrán más posibilidades de pasar a la siguiente generación.

Representación de los Individuos

Los individuos de la población corresponderán a los conjuntos de atributos que se pueden utilizar para realizar una referencia sobre una entidad determinada. Cada uno de estos atributos corresponderá a un gen, que contendrá un valor de 0 o 1 dependiendo de si aparecerá en la descripción final (1) o no (0).

Operadores Genéticos

El sistema trabaja sobre la población de posibles referencias durante un número de generaciones definido por el usuario. En cada generación se aplican sobre la población dos operadores genéticos: cruce y mutación. Ambos aseguran una variación razonable entre las distintas opciones igual que se pueden encontrar en las referencias generadas por seres humanos.

- *Operador de cruce.* Para la operación de cruce se seleccionan aleatoriamente dos referencias y se cruzan por un punto aleatorio de su estructura. De esta forma, cada uno de los dos hijos tendrá una parte de cada uno de los padres.
- *Operador de mutación.* En el caso de la operación de mutación, algunos de los genes de los textos son escogidos aleatoriamente para ser

mutados. La mutación de los genes, al tener valores simples, es fácil: si el gen no aparece en la referencia (tiene valor 0) muta para aparecer (valor 1), y viceversa.

Función de Evaluación

La clave del algoritmo evolutivo se encuentra en la elección de la función de evaluación, que debe codificar las restricciones que debe cumplir una referencia para ser aceptable. En el caso que nos ocupa se debe encontrar un equilibrio entre que la referencia representada por el individuo identifique unívocamente a la entidad referenciada, y un uso natural de los atributos. La fórmula que se ha usado como función de aptitud se puede ver en la Fórmula 4.2:

$$fit_{ind_i} = f_{att_i} * weight_{att} + ident * weight_{id} \tag{4.2}$$

donde $weight_{att} + weight_{id} = 1$.

Los parámetros $weight_{att}$ y $weight_{id}$ representan el peso relativo que se le da respectivamente a la elección de atributos y a la función de identificación de la referencia. El valor de *ident* representa si la referencia sirve para identificar unívocamente al referente entre los posibles distractores, y f_{att_i} será una función cualquiera que defina el rol de los atributos en la referencia según el caso concreto en que se encuentre el algoritmo. Por ejemplo, si se considera que a la hora de juzgar la bondad de una referencia es tan importante que cumpla su función distintiva como que los atributos seleccionados sean adecuados, tanto $weight_{att}$ como $weight_{id}$ podrían valer 0'5. Si en cambio sólo nos importa la función de distinción y no los atributos a escoger, se usarían como valores $weight_{att} = 0$ y $weight_{id} = 1$.

De estos cuatro parámetros, el único que debe ser ajustado dependiendo de la orientación que se le quiera dar al algoritmo de selección de atributos es f_{att_i}. Cada solución diferente tendrá una forma distinta de considerar la adecuación de los atributos para la referencia.

Modificaciones sobre el algoritmo

Para poder evaluar el funcionamiento del algoritmo se realizó una adaptación del mismo para que pudiera trabajar con los datos del corpus TUNA. Los individuos están representados por un conjunto de genes que son la lista de atributos disponibles para la entidad dependiendo del dominio. Cada gen tiene un valor binario asociado, con valor 0 si el atributo no se incluirá en la referencia, o 1 si sí será incluido. La población inicial debe tener un número bajo de genes con valor 1, ya que las referencias del corpus tienden a ser cortas (igual que las utilizadas por personas generalmente lo son) y el uso de todos los posibles atributos en la referencia debe evitarse.

La implementación de los operadores genéticos es la más común. Para el operador de cruce se escogen individuos aleatoriamente y se les cruza por un punto también aleatorio de su estructura. Para el operador de mutación, algunos genes se escogen aleatoriamente para cambiar su valor de 1 a 0 o viceversa.

Para la función de aptitud se ha utilizado la Fórmula 4.2 ya comentada. Los pesos $weight_{att}$ y $weight_{id}$ se han ajustado para optimizar los resultados obtenidos, y el parámetro f_{att_i} se ha definido teniendo en cuenta las especificaciones de los datos del corpus TUNA que se usan en la tarea. Concretamente f_{att_i} calcula el rol de los atributos en la referencia como la suma del peso (dependiendo de su frecuencia absoluta en los elementos `ATTRIBUTE-SET` de las respuestas en el corpus) de todos los atributos que estarán presentes ($gene_{att_i} = 1$), dividido en entre el número total de atributos del referente ($\#attsRef$). En la Fórmula 4.3 se puede ver la definición formal:

$$f_{att_i} = \frac{\sum(gene_{att_i} * weight_{att_i})}{\#attsRef} \tag{4.3}$$

Evaluación del Algoritmo

Se midieron los resultados obtenidos por el algoritmo según las métricas de evaluación de los módulos de selección de atributos como en los algoritmos anteriores: Dice, MASI, precisión, minimalidad e identificación. Los resultados sobre los datos entrenamiento se pueden ver en la Tabla 4.6. Los resultados para el dominio del mobiliario son bastante aceptables, mientras que para las personas parece que la función de adaptación escogida no es la más apropiada.

	Dice	**MASI**	**Precisión**	**Identif.**	**Minimalidad**
Muebles	77,64 %	0,55	38,99 %	100 %	0 %
Personas	56,97 %	0,28	10,22 %	100 %	0 %

Tabla 4.6: Resultados de evaluación del algoritmo evolutivo para los datos de entrenamiento

4.3. Sinonimia y Preferencias Personales para la Lexicalización de las Referencias

A la hora de lexicalizar una expresión de referencia una vez que se ha definido el contenido conceptual que ésta expresará, se deben escoger qué palabras o expresiones utilizar para cada uno de los conceptos que aparecen en la misma. En este trabajo, orientado a las referencias expresadas en forma de frases nominales, habrá que decidir cómo lexicalizar tanto el nombre del

tipo con el que nos vamos a referir a la entidad como los modificadores que le acompañarán. En este marco existen dos cuestiones relacionadas a tener en cuenta.

La primera es la elección léxica que se debe realizar entre las distintas palabras o expresiones que se pueden utilizar para realizar un concepto en forma de texto. Aquí juega un papel importante la sinonimia, ya que será necesario recopilar el conjunto de todas las posibles palabras o expresiones que sirven para expresar el mismo significado que el concepto o modificador que se desea lexicalizar.

Para poder elegir entre este conjunto de sinónimos sería necesario emplear algún tipo de heurística, que es donde entra en juego la segunda cuestión a tener en cuenta: las preferencias personales. En general los seres humanos tienden a usar una parte relativamente pequeña del amplio vocabulario que conocen. Esto se debe tanto a una cuestión del contexto en el que se están comunicando, como a las propias preferencias o costumbres personales que les hacen utilizar ciertas palabras o expresiones con más frecuencia que otras. Sin embargo, dado un mismo contexto de comunicación, es fácil observar diferentes usos del lenguaje por parte de diferentes individuos.

En esta sección se presenta una arquitectura general para añadir capacidades de elección léxica a módulos de generación previamente existentes apoyándose en un recurso léxico que proporciona diferentes alternativas a la hora de lexicalizar. En este caso se pone como ejemplo de recurso de estas características WordNet. Además se presentan dos algoritmos diferentes de lexicalización de referencias nominales. El primero de ellos presenta una solución muy simple de elección léxica según las opciones más frecuentes en los datos disponibles, y el segundo trata de modelar los estilos o preferencias personales usando una solución basada en casos.

4.3.1. Elección Léxica como Parte de la Lexicalización Usando WordNet

En esta sección se propone un módulo independiente para la expansión de las alternativas léxicas proporcionadas por un método de lexicalización ya existente en cualquier sistema de generación de lenguaje natural. El módulo propuesto se apoya en WordNet como base de material léxico indexado conceptualmente, y está específicamente diseñado como una solución reconfigurable que permite una adaptación sencilla de las heurísticas a cada dominio particular.

La decisión de desarrollar un módulo independiente para añadir elección léxica a una solución existente de realización léxica se puede justificar en términos de sus ventajas y desventajas.

Por un lado, la expansión de módulos de lexicalización previamente implementados será una posible aplicación. Sin embargo, dado que el módulo es independiente de la representación conceptual y se basa únicamente en la

entrada en forma de palabras, podría ser también utilizado para el enriquecimiento del léxico de textos ya terminados e incluso no generados de forma automática. Esta segunda aplicación no ha sido considerada en este trabajo, pero se mantiene como una posibilidad. También proporciona una relación directa entre el léxico existente en una aplicación determinada y WordNet, sin necesidad de que el resto del sistema deba ser ampliado para manejar información en la representación utilizada por WordNet.

Por otro lado, esta solución particular conlleva la introducción de un módulo extra al sistema existente, además de su solución original para la lexicalización. Esto puede penalizar la complejidad de la estructura pero reduce las dificultades que podrían surgir en términos de incompatibilidades entre las diferentes representaciones internas involucradas. De esta forma, la asignación de etiquetas léxicas a las distintas apariciones de un concepto ocurre dos veces. Primero se lleva a cabo en el módulo original. Después, la elección es revisada por el módulo de elección léxica (con el correspondiente coste en eficiencia).

La opción de posponer la elección léxica ya se contemplaba en (Shieber et al., 1989) para evitar indeterminismos innecesarios cuando las decisiones léxicas para alguna parte de una oración pueden ser dependientes de la forma particular que tomen otras partes vecinas.

La solución presentada aquí se puede ver desde dos puntos de vista: como un algoritmo genérico para realizar elección léxica gracias a la consulta un recurso léxico indexado conceptualmente, y como una arquitectura para implementar este algoritmo reutilizando soluciones software previamente existentes.

Los objetivos que deben ser satisfechos para la solución propuesta son:

- Debe proporcionar acceso a un recurso léxico indexado conceptualmente, de gran tamaño y previamente existente.
- Debe resultar fácilmente adaptable a módulos de lexicalización ya implementados.
- Debe permitir una sencilla reconfiguración de las heurísticas para adaptarlas a los distintos dominios de aplicación.

El Algoritmo

La operación del módulo de elección léxica se puede describir como un algoritmo básico que opera sobre una elección léxica inicial para una palabra determinada en un contexto dado (recibida como entrada), y que accede para ello a un recurso léxico que proporciona las alternativas.

Para una palabra dada, que es el resultado en un proceso previo de realización léxica aplicada al concepto que debe ser lexicalizado, el módulo de elección léxica debe:

1. Asegurarse de que la palabra corresponde a una forma canónica, llevando a cabo un análisis morfológico si es necesario.

2. Recolectar los significados posibles y desambigüar.

3. Recuperar las alternativas proporcionadas por el recurso léxico para el significado escogido.

4. Aplicar determinadas heurísticas para seleccionar la alternativa más apropiada para la aparición del concepto en cuestión, teniendo para ello en cuenta su contexto o historia del discurso.

5. Reemplazar la palabra original por la alternativa escogida, asegurándose de que está apropiadamente derivada si la palabra original lo estaba.

Veamos cada uno de estos puntos con más detalle:

1. **Análisis Morfológico**

   Los pasos inicial y final serán innecesarios en los casos en que el módulo de elección léxica esté trabajando dentro de un sistema de generación de texto que posponga la realización morfológica a una etapa posterior de realización superficial. Sin embargo, para que sea posible aplicar el módulo de elección léxica en otros contextos, estos pasos se han mantenido. Un posible caso sería el enriquecimiento léxico de textos completos ya mencionado. Cuando el módulo se encuentre operando dentro de sistemas de generación, los pasos inicial y final se pueden reducir a una simple comprobación, o incluso ser completamente deshabilitados durante la configuración del módulo.

2. **Desambigüación Semántica**

   Como ya se comentó en el apartado 2.5.2, WordNet no está organizado respecto a palabras individuales, sino respecto a conceptos. Debido a fenómenos lingüísticos como la polisemia o la sinonimia, existe una correspondencia muchos a muchos que relaciona los conceptos y las palabras. Esto produce el importante problema de la Desambigüación Semántica ya comentada.

   En el presente trabajo el algoritmo de desambigüación para una palabra dada es simple: encontrar los *synsets* que contengan la palabra, ordenarlos por su valor de *tag count*, y finalmente seleccionar el *synset* con el *tag count* mayor. La idea intuitiva detrás de este proceso es que si de los *synsets* existentes para una palabra el S_1 es más usado que el resto, lo más probable es que el concepto buscado corresponda a este *synset* y no a otro. Existen otras opciones más complejas para tratar la desambiguación, pero su estudio cae fuera del ámbito de este trabajo.

3. **Alternativas a Considerar**

   El conjunto de alternativas a considerar está estrechamente ligado con el recurso léxico particular que se esté empleando. El número y la naturaleza de las relaciones semánticas representadas en el recurso léxico actuarán como restricciones básicas sobre el conjunto de alternativas.

4. **Heurísticas de Elección**

   Se pueden usar numerosas heurísticas diferentes pueden ser usadas para decidir en la elección léxica, desde una simple asignación aleatoria hasta técnicas complejas de planificación, con opciones intermedias hechas a mano en términos de redes discriminantes. Sea cual sea la solución empleada, tener en cuenta el contexto en el que aparece un concepto (si el concepto ha aparecido anteriormente, a qué distancia, cómo fue lexicalizado entonces, etc.) generalmente tiene un efecto positivo en los resultados.

La Arquitectura

La lexicalización aquí propuesta se implementa en tres pasos:

1. *Realización léxica*, donde una etiqueta del vocabulario del sistema es escogida para cada concepto o relación que aparece en los mensajes de entrada. El resultado de este paso contiene suficiente información para generar el texto final.

2. *Comunicación con WordNet*, que proporciona información para la elección léxica. Para cada concepto que aparece en los mensajes, se obtiene de WordNet información sobre sus sinónimos e hiperónimos. Hay que tener en cuenta que las consultas a WordNet deben hacerse usando una palabra que corresponda al concepto buscado. En nuestro caso, las palabras obtenidas del vocabulario en el paso anterior son las utilizadas.

3. *Elección léxica*, donde diferentes heurísticas se aplican para decidir entre las opciones proporcionadas por WordNet para los conceptos que aparecen en los mensajes.

En términos arquitectónicos, esta división en pasos está justificada como sigue. La realización léxica inicial es llevada a cabo por el módulo de lexicalización inicial del sistema GLN. La comunicación con el recurso léxico elegido, en este caso WordNet, es implementada como un módulo independiente para facilitar sustituciones posteriores por otros recursos léxicos. El módulo de elección léxica es independiente para asegurar que se puedan considerar implementaciones basadas en heurísticas alternativas con un mínimo esfuerzo en términos de adaptación del resto de los componentes.

De los cuatro posibles usos de WordNet citados por Jing (Jing, 1998), el módulo aquí presentado está relacionado con dos de ellos directamente: uso de WordNet para lexicalizar en general, y uso de WordNet para proporcionar paráfrasis léxicas para textos existentes. La última propuesta de Jing (indexación del léxico de una aplicación directamente en términos de conceptos de WordNet) ha sido considerada y rechazada a favor de una compatibilidad más sencilla con módulos de lexicalización previamente existentes.

4.3.2. Lexicalización Escogiendo la Opción Más Frecuente

Una vez que los mensajes han sido generados de forma conceptual por el módulo de Planificación de Discurso de TAP, cada uno de ellos es tratado de manera individual por el módulo de Planificación de Oraciones. Aquí se realizan las tareas de Generación de Expresiones de Referencia (que incluye entre otros la selección de atributos ya comentada anteriormente), Elección Sintáctica (asignación de una forma sintáctica concreta al contenido del mensaje), y Lexicalización (asignación de una forma léxica a los conceptos que aparecen en el mensaje). Para realizar estas funciones el módulo de Planificación de Oraciones necesita que estén disponibles diccionarios de asignaciones entre conceptos y formas léxicas, y acciones o verbos y formas sintácticas.

El diccionario que se utiliza a la hora de lexicalizar en TAP añade a la funcionalidad de un diccionario típico, que asocia a cada concepto un conjunto de etiquetas léxicas, la posibilidad de asociar a cada elemento léxico una combinación dada por un tipo semántico más una lista de propiedades y relaciones adicionales. Esta información permite asociar elementos léxicos a conceptos semánticamente complejos que se expresan con una definición más que con el nombre de su tipo. Por ejemplo, a la combinación de conceptos *man* y *single* se le asigna la etiqueta léxica *bachelor* para indicar que estamos hablando de un hombre soltero. Esto enriquece la lexicalización que se produciría por defecto: *single man*.

Aunque el diccionario de TAP permite la inserción de numerosas posibilidades léxicas para un mismo concepto, por el momento no se han implementado heurísticas para elegir entre unas y otras y se devuelve siempre la primera opción.

A partir de la lexicalización realizada por TAP, se implementó un algoritmo que se basaba en los datos del corpus TUNA para escoger la opción de lexicalización más frecuente en cada caso. Este algoritmo implementa más una solución de realización léxica que de elección léxica en el sentido que les da Cahill (1998). Para implementar realmente una solución de elección léxica sería necesario considerar más de una alternativa para un concepto específico y seleccionar una de ellas siguiendo algún tipo de criterio. Esto no se ha llevado a cabo en el presente trabajo. En su lugar, una única al-

ternativa ha sido tenida en cuenta para cada concepto usándola para todos los ejemplos del corpus. La selección de esta alternativa particular ha sido llevada a cabo empíricamente de cara a obtener buenos resultados de evaluación sobre los datos de entrenamiento, y teniendo como idea que la mejor alternativa es la que más ha sido usada en el corpus.

Aunque los datos de entrada del corpus contenían únicamente el conjunto de propiedades seleccionadas para identificar al referente objetivo, esta información es sólo una parte de la que se requiere para generar expresiones de referencia adecuadas. Por tanto se deben tomar decisiones adicionales para llevar a cabo el proceso completo. Con respecto a la variación lingüística, se han diferenciado las decisiones que dan lugar a diferentes estructuras sintácticas (que se consideran en la fase de elección sintáctica) y las decisiones que dan lugar a las mismas estructuras sintácticas pero con distintas etiquetas léxicas (que se consideran en la fase de elección léxica). Para cada caso se han tomado las decisiones que eran más frecuentes en los datos del corpus.

Generación de Expresiones de Referencia

Una decisión a tomar que resultó ser particularmente relevante tiene que ver con el uso de determinantes. Los ejemplos en el corpus incluyen tres posibles alternativas: usar artículos indefinidos, usar artículos definidos, u omitir el uso de determinantes completamente. Analizando los datos del corpus se tomó como decisión no utilizar ningún tipo de determinante al inicio de las referencias, ya que era la opción más frecuente en el mismo.

Otra decisión que debía ser tenida en cuenta era la generación de expresiones para describir la localización espacial de los referentes, transformando la información disponible sobre su posición en la cuadrícula a lenguaje natural. El corpus presenta un amplio rango de opciones, algunas usando diferentes sistemas de referencia (norte-sur vs. arriba-abajo). En nuestro caso se usó el sistema de referencia arriba-abajo y derecha-izquierda a la hora de referirse a la situación espacial de los referentes.

Otra situación encontrada al analizar el corpus involucraba el uso de características particulares de los objetos en la descripción, como en *the desk with the drawers facing the viewer* o *the chair with seat facing away*. Estos casos no han sido considerados para ser incluidos en el sistema ya que requieren el uso de más información de la que se dispone en el conjunto de atributos proporcionado como entrada al módulo.

Además, muchas descripciones en el corpus se apoyaban en expresiones relativas basadas en la comparación con todos o algunos de los distractores. Estas expresiones pueden tomar la forma de complementos describiendo la posición relativa de los referentes objetivo respecto a los demás elementos, como en *the blue fan next to the green fan* y *the red chair above the blue one*, o adjetivos comparativos usados para ciertos atributos, como en *the*

largest red couch y *the larger grey chair* (incluso combinaciones de las dos como en *the smaller of the two blue fans*). Tampoco estos casos han sido contemplados porque para obtener este tipo de información sería necesario utilizar no sólo la información dada en el conjunto de atributos, sino también la situación del dominio.

Finalmente, hay ejemplos de referencias en el corpus donde se usa elipsis o expresiones gramaticalmente incorrectas como *glasses white beard*. Tampoco estos casos han sido contemplados. Sin embargo, es importante conocer que existen este tipo de expresiones en el corpus, ya que influirán negativamente en los resultados de evaluación, no teniendo nada que ver con la corrección de las expresiones generadas por los sistemas.

Elección Sintáctica

Con respecto a la elección sintáctica, algunos atributos muestran más de una posible opción a la hora de escoger la estructura sintáctica a utilizar. Algunos ejemplos son el atributo color (*grey chair - chair that is gray*), el atributo tener barba (*with beard - with the beard - with whiskers - the bearded man - with a beard - with facial hair*), y hasta el atributo orientación que llega a tener hasta 12 alternativas sintácticas diferentes para expresar la orientación *back*.

Para ajustar el módulo de Elección Sintáctica de TAP se escogieron en cada caso las opciones que más veces se encontraban presentes en el corpus.

Elección Léxica

Existen ligeras variaciones de elección léxica a lo largo del corpus, como en *sofa/couch/settee/loveseat*, *ventilator/fan/windmill* o *man/guy/bloke* (para sustantivos), y *large/big* o *small/little* (para adjetivos). Además es importante considerar que existe un buen número de errores ortográficos en el corpus, lo que también tiene un impacto significativo en los valores de evaluación.

Otro problema importante es la existencia en algunos casos de correspondencias conceptuales erróneas entre la anotación para el conjunto de atributos y la realización proporcionada (*purple/blue*, *black and white/grey*,etc.)

También en esta subtarea se tomaron siempre como decisión las opciones más frecuentes en el corpus.

Evaluación del Algoritmo

Se muestran en la Tabla 4.7 los resultados obtenidos con este algoritmo sobre los datos de entrenamiento utilizando las métricas de distancia de edición y precisión. Se pueden ver por separado los resultados por dominios, y en conjunto.

	Distancia de Edición (*String-edit distance*)	**Precisión**
Mobiliario	4,26	14,15 %
Personas	5,43	9,12 %
Ambos	4,8	11,82 %

Tabla 4.7: Resultados de evaluación para los datos de entrenamiento

Una cuestión importante a considerar con respecto a la adaptación aquí presentada es si una solución que implementara realmente elección léxica podría haber obtenido mejores resultados. Una solución así podría haberse beneficiado de la información que puede ser extraída del elemento `ANNOTATED-WORD-STRING` para entrenar un proceso de decisión sobre las distintas características. En la siguiente sección se presenta un algoritmo de este tipo.

4.3.3. Razonamiento Basado en Casos para Reproducir el Estilo de las Referencias

Para proporcionar una solución que fuera posible adaptar al estilo de referencia de un determinado conjunto de datos hemos utilizado el paradigma del razonamiento basado en casos descrito en la sección 3.2. Cuando los seres humanos se expresan oralmente o por escrito, no inventan palabras o expresiones nuevas cada vez que quieren comunicar una idea concreta que no habían oído o transmitido nunca, sino que automáticamente establecen relaciones entre la nueva idea a expresar y otras ya usadas con anterioridad, tomando así el mismo vocabulario y adaptándolo según sea necesario. Utilizan la experiencia pasada para resolver un nuevo caso. En esta idea se encuentra implícito también el hecho de que cada persona tiene un estilo diferente a la hora de leer o escribir que depende del vocabulario que utiliza normalmente para expresarse.

El algoritmo aquí presentado se basa en una solución de generación a base de plantillas, reutilizando fragmentos de texto extraídos de textos típicos en un dominio, y aplicando un proceso de abstracción que identifica qué parte de ellos es común para cualquier uso y qué huecos deben ser rellenados con los detalles que corresponden a un nuevo uso. La idea en el caso que nos ocupa es obtener la solución que se necesita para lexicalizar una referencia de ejemplos de uso apropiado que se habrán transformado en plantillas.

Para implementar este tipo de solución es imprescindible un corpus de referencias que ya se han considerado como adecuadas para poder extraer la información requerida para los casos. En nuestro sistema hemos utilizado de nuevo el corpus TUNA.

La Base de Casos

Para el almacenamiento de los casos se ha implementado una Case Retrieval Net (sección 3.2.2). El modelo es adecuado en el caso que nos ocupa tanto porque nuestros casos están formados por pares atributo-valor que se relacionan entre sí (elementos `ATTRIBUTE` dentro del `ATTRIBUTE-SET`), como porque las consultas realizadas al módulo no siempre tendrán los mismos elementos. Al realizar una consulta para buscar la etiqueta léxica apropiada para una nueva referencia el sistema busca referencias relacionadas según el conjunto de atributos que la definen.

En nuestra aproximación un caso consiste en la descripción de un problema (`ATTRIBUTE-SET`) y una solución para ese problema (`ANNOTATED-WORD-STRING`) interpretada como una plantilla. Para cada par atributo-valor del `ATTRIBUTE-SET` se crea una entidad de información (EI), y por cada caso un nodo que contiene enlaces a las entidades de información que lo forman. Si cuando se introduce una EI ésta ya ha aparecido en otro caso no se crea un duplicado, sino que simplemente se crea otra asociación al nuevo caso al que también pertenece. Como solución de los casos se utilizan las plantillas de lexicalización que se pueden extraer a partir del nodo `ANNOTATED-WORD-STRING` de los elementos del corpus. En la Figura 4.6 se muestra un ejemplo.

Según las entidades son insertadas formando la red, es necesario establecer la similitud entre las mismas. Teniendo en cuenta que los valores que pueden tomar los distintos atributos del corpus son muy dispares entre sí (nada tiene que ver el color con llevar corbata, por ejemplo), se han establecido similitudes diferentes dependiendo del tipo de atributo concreto. Además, la similitud de valores iguales para el mismo atributo será siempre uno, la máxima posible, y la similitud entre valores de diferentes atributos siempre será cero. El resto de similitudes definidas son:

- `type` y `colour`. Todos los valores que sean distintos tienen una similitud de 0.2, ya que son todos del mismo tipo (dependiendo del caso muebles o colores), y eso siempre tiene más similitud que simplemente ser cosas distintas.

- `orientation`. Se establecen las similitudes según la distancia relativa entre los cuatro valores (*left*, *right*, *front* y *back*). Los valores adyacentes tienen similitud 0.5, y los que no lo son 0. Se puede ver la idea en la Figura 4.7.

- `x-dimension` e `y-dimension`. Se calcula la distancia entre dos valores usando la Fórmula 4.4. Así la similitud entre valores iguales para el mismo atributo será 1, pero por ejemplo para x=2 y x=4 será 0.5.

$$sim(v_1, v_2) = 1 - \frac{|v_1 - v_2|}{4} \tag{4.4}$$

```
<TRIAL CONDITION="-LOC" ID="s60t22">
  <DOMAIN>
    [. . .]
  </DOMAIN>

  <WORD-STRING>middle aged dark haired dark bearded man</WORD-STRING>

  <ATTRIBUTE-SET>
     <ATTRIBUTE ID="a6" NAME="type" VALUE="person"></ATTRIBUTE>
     <ATTRIBUTE ID="a5" NAME="hasBeard" VALUE="1"></ATTRIBUTE>
     <ATTRIBUTE ID="a4" NAME="hairColour" VALUE="dark"></ATTRIBUTE>
     <ATTRIBUTE ID="a3" NAME="hasHair" VALUE="1"></ATTRIBUTE>
     <ATTRIBUTE ID="a2" NAME="hairColour" VALUE="dark"></ATTRIBUTE>
     <ATTRIBUTE ID="a1" NAME="age" VALUE="young"></ATTRIBUTE>
  </ATTRIBUTE-SET>

  <ANNOTATED-WORD-STRING>
    <ATTRIBUTE ID="a1" NAME="age" VALUE="young">
        middle aged
    </ATTRIBUTE>
    <ATTRIBUTE ID="a3" NAME="hasHair" VALUE="1">
      <ATTRIBUTE ID="a2" NAME="hairColour" VALUE="dark">
          dark
      </ATTRIBUTE>
      haired
    </ATTRIBUTE>
    <ATTRIBUTE ID="a5" NAME="hasBeard" VALUE="1">
      <ATTRIBUTE ID="a4" NAME="hairColour" VALUE="dark">
          dark
      </ATTRIBUTE>
      bearded
    </ATTRIBUTE>
    <ATTRIBUTE ID="a6" NAME="type" VALUE="person">man</ATTRIBUTE>
  </ANNOTATED-WORD-STRING>

</TRIAL>
```

Texto de la referencia

Representación del caso

Plantilla de la solución

Figura 4.6: Ejemplo de caso del corpus

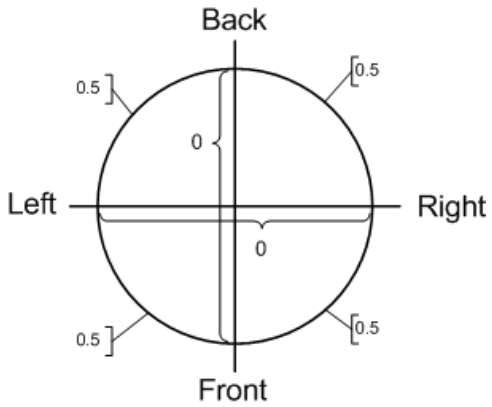

Figura 4.7: Similitudes entre las distintas orientaciones

- El resto de atributos (`size`, `age`, `hairColour`, `hasShirt`, `hasBeard`, `hasHair`, `hasGlasses`, `hasTie` y `hasSuit`) tienen valores que siempre son contrarios entre sí (*young* vs. *old*, 0 vs. 1, etc.), por lo que la similitud entre valores diferentes de estos atributos siempre será 0.

El Ciclo CBR

El módulo realizado en este trabajo lleva a cabo cada uno de los procesos del ciclo CBR de la forma que se expone a continuación:

1. **Recuperación de casos**

   Los pares atributo-valor del `ATTRIBUTE-SET` que hay que lexicalizar en una cadena final se utilizan como consulta. En el módulo que estamos describiendo la recuperación de casos se realiza directamente con la Case Retrieval Net y su método de propagación de similitud. A partir de los pares atributo-valor que hay que lexicalizar la recuperación de los casos más similares se lleva a cabo calculando un valor de activación por cada caso de la base de casos. Los que tengan mayor activación son los más similares a la consulta realizada. Este cálculo se realiza en tres pasos:

   - Los nodos de entidad que corresponden a la consulta son activados. Si no se encuentran en la red porque no pertenecían a ningún caso de la base de casos, se insertan los nodos correspondientes en el momento de la consulta, calculando sus pesos de similitud y relevancia usando la base de conocimiento. Se asigna un valor de activación de 1 a los nodos que corresponden a la consulta, y al resto un valor de 0.
   - La activación se propaga de acuerdo con los valores de los arcos de similitud. Esto se realiza recorriendo todos los nodos de entidad de la red, y calculando para cada uno de ellos su valor de activación a partir de su propia activación inicial y de su similitud con el resto de nodos de entidad. En este cálculo se utiliza la siguiente fórmula:

     $$activation(e) = \sum_{i=1}^{N} (sim(e_i, e) * activation(e_i))$$

     donde N es el número total de nodos de entidad.
   - Las activaciones obtenidas en el paso anterior se recogen en los nodos de casos asociados, calculando la activación final de los casos teniendo en cuenta los pesos de relevancia de los arcos que conectan los casos con las entidades que los forman. Este valor final de activación de los casos se calcula con la fórmula:

     $$activation(c) = \sum_{i=1}^{N} (rel(e_i, c) * activation(e_i))$$

     donde N es el número total de nodos de entidad.

Una vez obtenida la activación final de los casos, se toman los casos que tengan el valor de activación más alto y se ordenan por orden de preferencia teniendo en cuenta los atributos que contienen. Para ello se clasifican en cuatros grupos, que en orden de mayor a menor preferencia serían:

- *Preferidos.* Casos que contienen exactamente los mismos atributos que la consulta.
- *Más Extra.* Casos que contienen todos los atributos que tiene la consulta, y además algunos extra.
- *Menos Extra.* Casos a los que les falta alguno de los atributos de la consulta, pero que tienen también algunos atributos extra que no estaban en ella.
- *Menos Sin Extra.* Casos a los que les falta alguno de los atributos de la consulta, y que además no tienen atributos extra.

De esta manera los casos con la activación máxima se clasifican en estos grupos, y se escoge el que esté en el grupo con mayor preferencia.

2. **Reutilización de casos**

   La adaptación o reutilización del caso obtenido depende de su tipo según los mencionados anteriormente. La idea es quedarse con todas las partes de la plantilla que corresponden a atributos comunes entre la consulta y el caso recuperado. Los atributos extra que aparecen en el caso pero no en la consulta son descartados, y si hay atributos de la consulta que no aparecen en el caso recuperado se perderán.

   Para adaptar cada uno de los atributos dentro de la plantilla, se consulta si los valores de cada atributo son los mismos para la consulta y para el caso recuperado. En caso de ser así se usa para el atributo la etiqueta léxica que aparece en el caso recuperado. Si no, se utilizan etiquetas por defecto para cada posible valor del atributo.

3. **Revisión y Aprendizaje de casos**

   Cuando una solución generada en la fase de reutilización es correcta o incorrecta, se abre la posibilidad de aprender del éxito o fracaso. Por el momento, las tareas de revisión y aprendizaje no están implementadas en el módulo CBR. Sería muy útil incorporar nuevo conocimiento a la base de casos, pero tratándose de lenguaje natural puede ser complicado. Debido a las restricciones asociadas al uso del lenguaje, sería necesaria la presencia de un experto en el dominio para revisar los resultados obtenidos por el módulo, refinándolos y reteniéndolos si fuera posible.

Ejemplos de Funcionamiento

Para mostrar el funcionamiento del módulo presentamos algunos ejemplos. Supongamos que se realiza al módulo una consulta como:

```
TYPE: chair, COLOUR: grey, Y-DIMENSION: 2
```

La referencia que le corresponde a este ejemplo tomado de los datos del corpus sería *the grey chair in the middle row.*

El proceso de recuperación recupera un caso de los considerados preferidos, ya que contiene exactamente los mismos atributos que la consulta:

```
TYPE:    COLOUR:  Y-DIMENSION:
chair    grey     3
```

con referencia asociada *the gray chair in the bottom row* que corresponde a la siguiente plantilla:

```
"the"
<ATTRIBUTE name=colour value=grey string="gray"/>
<ATTRIBUTE name=type value=chair string="chair"/>
<META-ATTRIBUTE name=location string="in the bottom row">
   <ATTRIBUTE name=y-dimension value=3/>
</META-ATTRIBUTE>
```

Para adaptar la plantilla a la consulta se comprueban si los valores del caso recuperado coinciden con los valores de la consulta. En caso afirmativo se mantiene el texto asociado a ese valor, y en caso contrario se usa el texto por defecto que se ha asignado a cada valor. En este ejemplo los valores *grey* y *chair* coinciden, pero no ocurre lo mismo con la dimensión y. En lugar de usar *in the bottom row* se utiliza el valor por defecto para el valor 2: *in row two.* La referencia final obtenida sería *the gray chair in row two.*

Imaginemos ahora que realizamos la siguiente consulta:

```
TYPE: person, HASGLASSES: 1
```

La referencia que le corresponde a este ejemplo tomado de los datos del corpus sería *the man with glasses.*

El proceso de recuperación recupera un caso de los que contienen todos los atributos de la consulta, y además alguno extra (en este caso `Y-DIMENSION`):

```
TYPE:    HASGLASSES:  Y-DIMENSION:
person   1            3
```

con referencia asociada *the man with glasses on the bottom* que corresponde a la siguiente plantilla:

```
"the"
<ATTRIBUTE name=type value=person string="man"/>
<ATTRIBUTE name=hasGlasses value=1 string="with glasses"/>
<META-ATTRIBUTE name=location string="on the bottom">
   <ATTRIBUTE name=y-dimension value=3/>
</META-ATTRIBUTE>
```

A la hora de adaptar el caso se descarta el atributo que no está en la consulta y obtenemos la referencia *the man with glasses*.

En ocasiones es posible que surjan realizaciones gramaticalmente incorrectas para las referencias. Imaginemos que con esta consulta

```
TYPE: person, HAIRCOLOUR: dark, HASBEARD: 1
```

se recuperara un caso con la siguiente plantilla:

```
"a"
<ATTRIBUTE name=type value=person string="man"/>
"with"
<ATTRIBUTE-NESTED name=hasBeard value=1>
   "a"
   <ATTRIBUTE name=hairColour value=dark string="dark"/>
   "beard"
</ATTRIBUTE-NESTED>
"and"
<ATTRIBUTE-NESTED name=hasHair value=1>
   <ATTRIBUTE name=hairColour value=dark string="dark"/>
   "hair"
</ATTRIBUTE-NESTED>
```

Al adaptar el caso se descarta el atributo que no está en la consulta (*hasHair*) y obtendríamos la referencia *a man with a dark beard and*, que resulta ser gramaticalmente incorrecta ya que ha conservado la conjunción al reutilizar la plantilla.

Resultados y Discusión

Se midieron los resultados obtenidos por el algoritmo según las métricas clásicas de evaluación de los módulos de lexicalización: distancia de edición (*string-edit distance*) y precisión.

La base de casos se formó en todas las pruebas usando los datos de entrenamiento, y se probó a llevar a cabo la realización de las referencias tanto para el conjunto de datos de entrenamiento (Tabla 4.8) como para los de desarrollo (Tabla 4.9).

	Distancia de Edición (*String-edit distance*)	**Precisión**
Mobiliario	1,75	44,09 %
Personas	2,75	35,63 %
Ambos	2,25	39,86 %

Tabla 4.8: Resultados de evaluación del módulo basado en casos sobre los datos de entrenamiento

	Distancia de Edición (*String-edit distance*)	**Precisión**
Mobiliario	4,66	8,75 %
Personas	5,25	5,88 %
Ambos	4,95	7,43 %

Tabla 4.9: Resultados de evaluación del módulo basado en casos sobre los datos de desarrollo

Se calcularon los valores obtenidos por el módulo usando los datos de entrenamiento tanto para el entrenamiento como para la ejecución para comprobar cuáles eran los valores mínimos que se podían conseguir si los datos que se consultaban estaban ya en la base de casos. Los resultados obtenidos son bastante bajos, pero aún así no lo suficiente según sería de esperar. Idealmente el módulo CBR debería devolver siempre el mismo caso si existe en la base de casos, pero en realidad existen muchos casos diferentes que tienen el mismo conjunto de atributos, que es sobre el que realiza la consulta, pero con distintas lexicalizaciones de la referencia. Esto provoca que los resultados no sean perfectos.

Los resultados obtenidos sobre los datos de desarrollo muestran que nuestro sistema no es especialmente bueno cuando se trata de coincidir con las expresiones específicas que se usan en el corpus. Esto se debe probablemente a la naturaleza del corpus que presenta una amplia variación en el tipo de expresiones usadas, y que por tanto no tiene un estilo homogéneo que se pueda reproducir. Nuestro método basado en casos podría mejorarse si fuera posible modelar estilos particulares de generación de expresiones de referencia en lugar de intentar trabajar en una manera genérica que abarque razonablemente todas las variaciones posibles. Esto se podría llevar a cabo separando las referencias del corpus que han sido producidas por una misma persona y usando un modelo de la forma en que esta persona genera expresiones de referencia. Para una solución de este tipo la aproximación

del razonamiento basado en casos sería muy conveniente ya que la base de casos almacenaría ejemplos de como una persona concreta genera referencias, y a la hora de lexicalizar se usarían distintas referencias pero siempre conservando el mismo estilo.

4.4. Evaluación de los Algoritmos en las Tareas Competitivas para la Generación de Expresiones de Referencia

Se presentan a continuación los resultados de evaluación obtenidos por las adaptaciones de algunos de los algoritmos ya presentados en cada una de las ediciones de las tareas de evaluación competitiva sobre generación de expresiones de referencia. Los resultados muestran tanto los resultados obtenidos por cada sistema en solitario como en comparación con el resto de algoritmos participantes.

4.4.1. Resultados de Evaluación en la Primera Tarea Competitiva para la Generación de Expresiones de Referencia

A la primera tarea competitiva para la generación de expresiones de referencia (sección 2.3.3) se presentó el algoritmo de selección de atributos basado en agrupamiento de atributos de la sección 4.2.1. Recordemos que en esta tarea se debía generar un conjunto de atributos correspondientes a la definición conceptual del referente objetivo de un dominio o situación dada. Se utilizaron cuatro métricas de evaluación: identificación, minimalidad, concordancia sistema-evaluador humano usando el coeficiente Dice y evaluación basada en tarea. Las tres primeras debían ser calculadas por los participantes sobre los datos de desarrollo, mientras que con los datos de test los organizadores aplicarían las cuatro métricas para todos los sistemas participantes.

Resultados sobre los Datos de Desarrollo

Para los experimentos sobre los datos de desarrollo se usaron las combinaciones de atributos que habían mostrado mejores resultados sobre los datos de entrenamiento. Los resultados finales obtenidos usando el algoritmo final sobre los datos de desarrollo se muestran en la Tabla 4.10. Los resultados obtenidos con el módulo modificado para obtener las expresiones de referencia mínimas se muestran en la Tabla 4.11. Los valores obtenidos con los datos de desarrollo son consistentes con los obtenidos a partir de los datos de entrenamiento (ver sección 4.2.1).

	Identificación	Referencias Mínimas	Dice
Mobiliario	100,00 %	0,00 %	75,21 %
Personas	100,00 %	33,82 %	44,78 %

Tabla 4.10: Resultados obtenidos sobre los datos de desarrollo del algoritmo basado en agrupamientos de atributos

	Referencias Mínimas	Dice
Mobiliario	100,00 %	20,95 %
Personas	100,00 %	30,93 %

Tabla 4.11: Resultados sobre los datos de desarrollo usando el algoritmo adaptado para generar referencias mínimas

Resultados sobre los Datos de Test y Comparativa con otros Participantes

La competición fue un éxito de participación con 22 sistemas repartidos entre seis equipos que englobaban un total de 13 investigadores de diferentes universidades internacionales. Todos los sistemas participantes fueron evaluados según las medidas de identificación, minimalidad y concordancia, mientras que sólo 15 se probaron en la evaluación orientada a tarea usando evaluadores humanos. Para más detalles consultar (Belz y Gatt, 2007).

Desde el punto de vista de la evaluación de la identificación, todos los sistemas excepto uno describían los referentes objetivo con un conjunto de atributos que los identificaban unívocamente.

En la Tabla 4.12 se muestra la proporción de expresiones mínimas generadas por cada uno de los sistemas. Se puede ver que nuestro sistema no se comporta especialmente bien desde el punto de vista de las referencias mínimas, pero ya se comentó que la implementación no se orientaba a esta medida de evaluación ya que parecía hacer empeorar la evaluación de concordancia.

En la Tabla 4.13 se muestran los resultados obtenidos por los sistemas usando el coeficiente Dice, tanto por dominios como en media, y se muestran los resultados de comparación de los sistemas usando el Test de Diferencia Honestamente Significativa de Tukey. Éste es un tipo de test ANOVA que determina si las medias entre grupo son significativamente diferentes. Así, los sistemas que no comparten ninguna letra se consideran significativamente diferentes. Aunque nuestro sistema obtiene uno de los valores más altos para el dominio de los muebles, la media se resiente debido a los malos resultados obtenidos para el dominio de las personas.

Los resultados de la evaluación orientada a tarea se pueden dividir en dos partes a discutir por separado. En el caso de la precisión de referentes

identificados correctamente por los evaluadores humanos no existe ninguna diferencia significativa entre sistemas. Los resultados de la medición del

Sistema	**Referencias Mínimas (%)**
TITCH-AW-DYN	93.92
TITCH-BS-STAT	93.92
TITCH-BS-DYN	92.56
CAM-BU	83.10
CAM-B	81.08
TITCH-AW-STAT	81.08
GRAPH-SC	77.03
TITCH-RW-DYN	74.32
TITCH-RW-STAT	74.32
TITCH-RW-DYN-PLUS	66.18
TITCH-RW-STAT-PLUS	66.18
TITCH-AW-DYN-PLUS	63.24
TITCH-AW-STAT-PLUS	63.23
IS-FBS	60.08
GRAPH-FP	59.46
CAM-T	40.54
CAM-TU	41.21
NIL	37.16
DIT-DS	31.76
IS-IAC	31.75
IS-FBN	16.22
DIT-DI	14.86

Tabla 4.12: Proporción de referencias mínimas por sistema

Sistema	**Muebles**	**Personas**	**Media**						
IS-FBN	0.80	0.74	0.7709	A					
DIT-DS	0.80	0.69	0.7501	A	B				
IS-IAC	0.80	0.68	0.7461	A	B	C			
CAM-T	0.79	0.65	0.7249	A	B	C	D		
CAM-TU	0.78	0.65	0.7214	A	B	C	D		
GRAPH-FP	0.71	0.67	0.6898	A	B	C	D	E	
GRAPH-SC	0.71	0.63	0.6715	A	B	C	D	E	
TITCH-RW-STAT	0.69	0.61	0.6551	A	B	C	D	E	
TITCH-RW-DYN	0.69	0.61	0.6551	A	B	C	D	E	
TITCH-AW-STAT-PLUS	–	0.65	0.6532	A	B	C	D	E	
TITCH-AW-DYN-PLUS	–	0.65	0.6532	A	B	C	D	E	
TITCH-AW-STAT	0.67	0.62	0.6455		B	C	D	E	F
TITCH-AW-DYN	0.66	0.62	0.6411		B	C	D	E	F
TITCH-RW-STAT-PLUS	–	0.64	0.6400		B	C	D	E	F
TITCH-RW-DYN-PLUS	–	0.64	0.6400		B	C	D	E	F
CAM-BU	0.61	0.65	0.6300			C	D	E	F
NIL	0.80	0.42	0.6251				D	E	F
DIT-DI	0.71	0.53	0.6243				D	E	F
CAM-B	0.59	0.65	0.6203				D	E	F
TITCH-BS-DYN	0.60	0.58	0.5934					E	F
TITCH-BS-STAT	0.60	0.58	0.5928					E	F
IS-FBS	0.62	0.42	0.5276						F

Tabla 4.13: Resultados del coeficiente Dice de los sistemas participantes sobre los datos de test. Los sistemas que no comparten una letra son significativamente diferentes

tiempo de identificación que utilizaban los evaluadores para cada sistema se muestran en la Tabla 4.14, donde el Test de Diferencia Honestamente Significativa de Tukey ha sido utilizado nuevamente para comparar los sistemas. Parece en este caso que nuestro algoritmo no se comporta significativamente diferente a como lo hacen los que han obtenido menores tiempo de identificación.

Sistema	**Tiempo Medio de Identificación (ms)**			
TITCH-RW-STAT	2514.367	A		
CAM-TU	2572.821	A		
CAM-T	2626.022	A		
TITCH-AW-STAT-PLUS	2652.845	A		
CAM-BU	2659.369	A		
GRAPH-FP	2724.559	A		
TITCH-RW-STAT-PLUS	2759.758	A		
CAM-B	2784.804	A		
DIT-DS	2785.396	A		
GRAPH-SC	2811.091	A		
IS-IAC	2844.172	A	B	
TITCH-AW-STAT	2864.933	A	B	
NIL	2894.770	A	B	
IS-FBN	3570.904		B	C
IS-FBS	4008.985			C

Tabla 4.14: Tiempos medios de identificación para los sistemas evaluados. Los sistemas que no comparten una letra son significativamente diferentes

Discusión

Los resultados obtenidos sobre los datos de desarrollo indican que el algoritmo está actuando adecuadamente de cara a la identificación unívoca de los referentes. Las referencias mínimas también podrían haber sido obtenidas sin ningún problema (como se muestra en la Tabla 4.11) si la minimalidad fuera considerada una prioridad. Otra opción que no se ha explorado es intentar mejorar los resultados para el coeficiente Dice mientras se mantienen las referencias mínimas. En cuanto a los resultados del coeficiente Dice sobre nuestro sistema, se consigue el valor más alto en el dominio de los muebles, mientras que los resultados son muy pobres para el dominio de las personas. Esto se debe a que se ha conseguido encontrar un orden adecuado de los atributos para el primer dominio, pero no para el segundo.

4.4.2. Resultados de Evaluación en la Segunda Tarea Competitiva para la Generación de Expresiones de Referencia

En la segunda tarea competitiva para la generación de expresiones de referencia (sección 2.3.3) se presentaron tres algoritmos, uno a cada una de las subtareas diferentes realizadas sobre el corpus TUNA: selección de atributos (TUNA-AS), realización (TUNA-R), y generación completa de referencias (TUNA-REG). Para más detalles consultar (Gatt et al., 2008).

Resultados para la Subtarea de Selección de Atributos (TUNA-AS)

En la subtarea de selección de atributos se presentó el algoritmo basado en el valor más frecuente ya comentado en la sección 4.2.2, al que nombraremos a partir de aquí como NIL-UCM-MFVF (*Most Frequent Value First*). Recordemos que en esta tarea se debía generar un conjunto de atributos correspondientes a la definición conceptual del referente objetivo de un dominio o situación dada. Se emplearon cinco métricas de evaluación: Dice, MASI, Precisión, Identificación y Minimalidad. Los participantes debían calcularlas sobre los datos de desarrollo, y los organizadores sobre los datos de test para todos los sistemas participantes.

Los resultados obtenidos sobre los datos de desarrollo se muestran en la Tabla 4.15. Si se comparan con los obtenidos sobre los datos de entrenamiento (ver sección 4.2.2) se puede ver que no hay sorpresas en los resultados finales y que el sistema obtiene resultados similares para los dos conjuntos de datos y los dos dominios de trabajo. Estos resultados confirman que la frecuencia de aparición de cada valor para los atributos de los referentes objetivo se mantiene razonablemente constante en todo el corpus.

	Dice	**MASI**	**Precisión**	**Identif.**	**Minimalidad**
Mobiliario	77,55 %	0,54	41,25 %	100 %	0 %
Personas	70,86 %	0,42	22,06 %	100 %	0 %
Ambos	74,48 %	0,49	32,43 %	100 %	0 %

Tabla 4.15: Resultados de evaluación para los datos de desarrollo del algoritmo NIL-UCM-MFVF

En las evaluaciones realizadas sobre los datos de test se comprobó que todos los sistemas identificaban unívocamente al referente objetivo con los conjuntos de atributos que generaban. Los resultados para el resto de métricas se pueden ver en la Tabla 4.16. Las comparaciones entre sistemas utilizando el Test de Diferencia Honestamente Significativa de Tukey se muestran en la Tabla 4.17. Se puede ver también que el algoritmo presentado no presenta diferencias significativas con los que han obtenido los resultados más altos.

	Dice			MASI			Precisión		
	mob.	*pers.*	*amb.*	*mob.*	*pers.*	*amb.*	*mob.*	*pers.*	*amb.*
GRAPH	0.858	0.729	0.794	0.705	0.465	0.585	0.53	0.56	0.40
JU-PTBSGRE	0.858	0.762	0.810	0.705	0.501	0.603	0.55	0.58	0.41
ATT-DR-b	0.852	0.722	0.787	0.663	0.441	0.552	0.52	0.54	0.36
ATT-DR-sf	0.852	0.722	0.787	0.663	0.441	0.552	0.50	0.52	0.36
DIT-FBI	0.850	0.731	0.791	0.661	0.451	0.556	0.50	0.53	0.36
IS-FP	0.828	0.723	0.776	0.641	0.475	0.558	0.52	0.54	0.37
NIL-UCM-MFVF	0.821	0.684	0.753	0.601	0.383	0.492	0.44	0.46	0.31
USP-EACH-FREQ	0.820	0.663	0.742	0.616	0.404	0.510	0.46	0.48	0.33
DIT-TVAS	0.814	0.684	0.749	0.580	0.383	0.482	0.43	0.46	0.29
OSU-GP	0.640	0.443	0.541	0.352	0.114	0.233	0.17	0.20	0.06
ATT-FB-m	0.357	0.263	0.310	0.164	0.119	0.141	0.13	0.14	0.00
ATT-FB-f	0.231	0.307	0.269	0.093	0.138	0.116	0.13	0.12	0.00
ATT-FB-sf	0.231	0.307	0.269	0.093	0.138	0.116	0.13	0.12	0.00
ATT-FB-sr	0.231	0.307	0.269	0.093	0.138	0.116	0.13	0.12	0.00

Tabla 4.16: Resultados de evaluación de los sistemas participantes sobre los datos de test

Dice				MASI					
ATT-FB-f	A			ATT-FB-f	A				
ATT-FB-sf	A			ATT-FB-sf	A				
ATT-FB-sr	A			ATT-FB-sr	A				
ATT-FB-m	A			ATT-FB-m	A	B			
OSU-GP		B		OSU-GP		B			
USP-EACH-FREQ			C	DIT-TVAS			C		
DIT-TVAS			C	NIL-UCM-MFVF			C	D	
NIL-UCM-MFVF			C	USP-EACH-FREQ			C	D	E
IS-FP			C	ATT-DR-b			C	D	E
ATT-DR-b			C	ATT-DR-sf			C	D	E
ATT-DR-sf			C	DIT-FBI			C	D	E
DIT-FBI			C	IS-FP			C	D	E
GRAPH			C	GRAPH				D	E
JU-PTBSGRE			C	JU-PTBSGRE					E

Tabla 4.17: Comparación de los sistemas participantes para los coeficientes Dice y MASI

A la vista de los resultados parece que cuando se está generando una expresión de referencia en cualquier contexto el valor del atributo está relacionado con su probabilidad de ser escogido. En nuestro algoritmo hemos estudiado esta relación usando solamente las características correspondientes a la entidad objetivo. Sin embargo, cuando una persona está generando una expresión de referencia no sólo tiene en cuenta si un atributo específico del referente es sobresaliente en general, sino también si sobresale en la situación en la que el referente está inmerso. Por ejemplo, tener barba es una característica destacable que se suele usar para referirse a las personas, pero si en una situación específica todo el mundo tiene barba, su inclusión en una descripción es inútil. Siguiendo esta idea, el presente algoritmo podría mejorarse si se tiene en cuenta no sólo los valores de los atributos del referente objetivo, sino también los de los distractores.

Resultados para la Subtarea de Realización (TUNA-R)

En la subtarea de realización se presentó el algoritmo de lexicalización según la opción más frecuente ya comentado en la sección 4.3.2, al que nombraremos a partir de aquí NIL-UCM-BSC (*Best Score Choice*). Recordemos que esta tarea consistía en lexicalizar conjuntos de atributos que representaban la descripción conceptual de una referencia usando lenguaje natural. Se utilizaron cuatro métricas de evaluación intrínseca: distancia de edición, precisión, BLEU y NIST. Sólo las dos primeras eran evaluadas por los participantes sobre los datos de desarrollo, mientras que los organizadores usaban todas ellas para medir los sistemas participantes.

Los resultados obtenidos sobre los datos de desarrollo se muestran en la Tabla 4.18. Si se comparan con los obtenidos sobre los datos de entrenamiento (Tabla 4.7), se observa una vez más que los resultados son consistentes con ambos conjuntos de datos y dominios.

	Distancia de Edición (*String-edit distance*)	**Precisión**
Mobiliario	4,21	15 %
Personas	4,94	7,35 %
Ambos	4,54	11,48 %

Tabla 4.18: Resultados de evaluación para los datos de desarrollo del algoritmo NIL-UCM-BSC

En las evaluaciones realizadas sobre los datos de test se comprobó que todos los sistemas se comportaban de manera consistente sin diferencias significativas. Los resultados se pueden ver en la Tabla 4.19. No se realizó la comparación entre sistemas utilizando el Test de Diferencia Honestamente Significativa de Tukey porque todos eran significativamente iguales.

	Distancia de edición			**Precisón**			**NIST**	**BLEU**
	mobiliario	*personas*	*ambos*	*mobiliario*	*personas*	*ambos*	*ambos*	*ambos*
IS-GT	7.750	9.768	8.759	0.02	0.00	0.01	0.4526	0.0415
NIL-UCM-BSC	7.411	9.143	8.277	0.05	0.04	0.04	17.034	0.0784
ATT-1-R	7.143	9.268	8.205	0.02	0.00	0.01	0.1249	0
DIT-CBSR	7.054	10.286	8.670	0.09	0.02	0.05	11.623	0.0686
DIT-RBR	6.929	9.857	8.393	0.04	0.00	0.02	0.9151	0.0694

Tabla 4.19: Resultados de evaluación de los sistemas participantes sobre los datos de test

Tal y como se ve en los resultados, nuestro algoritmo obtiene los segundos mejores valores de distancia de edición. Cabe destacar que los valores de NIST y BLEU son significativamente superiores a los de los demás sistemas.

Resultados para la Subtarea de Generación Completa de Expresiones de Referencia (TUNA-REG)

En la subtarea de generación completa de expresiones de referencia se presentó un algoritmo compuesto por los dos que acabamos de comentar. Para la parte de la selección de atributos se utilizó el algoritmo NIL-UCM-MFVF, y para la lexicalización de las salidas generadas por éste se usó el algoritmo NIL-UCM-BSC. Nos referiremos a este algoritmo combinado como NIL-UCM-FVBS (*Frequent-Value Selection with Best-Scoring Realization*). Recordemos que esta tarea pretendía abordar la tarea completa de generación de expresiones de referencia generando frases en lenguaje natural a partir de representaciones conceptuales de situaciones de entidades.

En este caso se realizó tanto evaluación intrínseca como extrínseca. Como medidas de evaluación intrínseca se utilizaron las mismas que en la subtarea TUNA-R (distancia de edición, precisión, BLEU y NIST). Para la evaluación extrínseca se midió el tiempo que tardaban los evaluadores humanos en leer la referencia, el tiempo que tardaban después en identificar al referente, y la tasa de error. Sólo las dos primeras medidas intrínsecas eran evaluadas por los participantes sobre los datos de desarrollo, mientras que los organizadores utilizaban todas ellas (tanto intrínsecas como extrínsecas) para medir los sistemas participantes.

Los resultados obtenidos sobre los datos de desarrollo se muestran en la Tabla 4.20. Los resultados obtenidos son un poco peores que los obtenidos por los módulos de selección de atributos y realización por separado. En realidad esto no es un resultado inesperado ya que las malas decisiones que se han podido tomar en la selección de atributos se propagan hasta la realización, dando lugar a errores acumulados en la evaluación final.

	Distancia de Edición (*String-edit distance*)	**Precisión**
Mobiliario	5,06	3,75 %
Personas	6,24	1,47 %
Ambos	5,60	2,70 %

Tabla 4.20: Resultados de evaluación para los datos de desarrollo del algoritmo NIL-UCM-FVBS

Las evaluaciones realizadas sobre los datos de test se estudiaron por separado desde los puntos de vista intrínseco y extrínseco. En la Tabla 4.21 se pueden ver los resultados de los diferentes sistemas para las medidas de evaluación intrínsecas. Además se compararon los sistemas para la medida de distancia de edición utilizando el Test de Diferencia Honestamente Significativa de Tukey, y se comprobó que todos los sistemas eran significativamente iguales exceptuando el OSU-GP que obtenía valores muy altos para la distancia de edición. Nuestro algoritmo presenta una de las medias más bajas

de distancia de edición, y volvemos a obtener valores significativamente altos para el coeficiente NIST.

	Distancia de Edición			Precisión			NIST	BLEU
	mob.	*pers.*	*amb.*	*mob.*	*pers.*	*amb.*	*amb.*	*amb.*
ATT-PermuteRank-ws	8.339	8.304	8.321	0.00	0	0	0.007	0.0288
ATT-Template-ws	8.304	8.161	8.232	0.00	0	0	0	0.0059
ATT-Dependency-ws	8.232	8.000	8.116	0.00	0	0	0.0001	0.0139
ATT-TemplateS-ws	8.214	8.161	8.188	0.00	0	0	0	0.0057
OSU-GP	7.964	13.232	10.598	0.00	0	0	1.976	0.0236
ATT-PermuteRank-drws	7.464	8.411	7.938	0.02	0.04	0.03	0.603	0.0571
DIT-TVAS-RBR	6.893	8.161	7.527	0.05	0	0.03	10.233	0.0659
ATT-TemplateS-drws	6.786	7.679	7.232	0.07	0.02	0.04	0.6786	0.0958
ATT-Template-drws	6.768	7.696	7.232	0.07	0.02	0.04	0.6083	0.0929
NIL-UCM-FVBS	6.643	8.411	7.527	0.07	0.04	0.05	18.277	0.0684
IS-FP-GT	6.607	7.304	6.955	0.05	0.02	0.04	0.8708	0.1086
DIT-FBI-CBSR	6.536	7.643	7.089	0.16	0.05	0.11	0.8804	0.1259
ATT-Dependency-drws	6.482	7.446	6.964	0.07	0	0.04	0.3427	0.0477
GRAPH	5.946	9.018	7.482	0.18	0	0.09	1.141	0.0696

Tabla 4.21: Resultados de evaluación intrínseca de los sistemas participantes sobre los datos de test

Los valores obtenidos durante la evaluación extrínseca para los tiempos de lectura e identificación y la tasa de error se pueden ver en la Tabla 4.22. La comparación entre sistemas para estas medidas se muestra en la Tabla 4.23.

	Tiempo de Lectura (ms)			Tiempo de identificación (ms)			Tasa de error (%)		
	mob.	*pers.*	*ambos*	*mob.*	*pers.*	*ambos*	*mob.*	*pers.*	*ambos*
HUMAN-1	2155	2187	2171	1973	1911	1942	11.86	6.78	9.32
OSU-GP	2080	3204	2637	2063	2274	2167	6.67	18.97	12.71
HUMAN-2	1823	2298	2061	1873	1945	1909	16.66	5	10.83
ATT-PremuteRank-drws	1664	1420	1543	1765	1719	1742	10	8.47	9.24
DIT-FBI-CBSR	1581	1521	1551	1528	1932	1732	10.17	10	10.08
NIL-UCM-FVBS	1561	1933	1747	1531	1723	1627	6.67	3.33	5
GRAPH	1499	1516	1508	1706	2026	1866	5	5	5
DIT-TVAS-RBR	1485	1442	1463	1559	1734	1647	8.33	13.33	10.83
ATT-Dependency-drws	1460	1583	1522	1505	2078	1791	1.67	18.33	10
ATT-TemplateS-drws	1341	1641	1490	1656	1720	1687	3.33	10.34	6.78
IS-FT-GT	1292	1614	1453	1616	1884	1750	6.67	1.67	4.17
ATT-PermuteRank-ws	1218	1450	1334	1876	1831	1854	31.67	13.33	22.5

Tabla 4.22: Resultados de evaluación extrínseca de los sistemas participantes sobre los datos de test

Tiempo de Identificación				Tiempo de Lectura				
NIL-UCM-FVBS	A			ATT-PermuteRank-ws	A			
DIT-TVAS-RBR	A			IS-FT-GT	A			
ATT-TemplateS-drws	A	B		DIT-TVAS-RBR	A			
DIT-FBI-CBSR	A	B		ATT-TemplateS-drws	A			
ATT-PremuteRank-drws	A	B		GRAPH-4+B	A	B		
IS-FT-GT	A	B		ATT-Dependency-drws	A	B		
ATT-Dependency-drws	A	B		ATT-PremuteRank-drws	A	B		
ATT-PermuteRank-ws	A	B		DIT-FBI-CBSR	A	B		
GRAPH-4+B	A	B		NIL-UCM-FVBS	A	B	C	
HUMAN-2	A	B	C	HUMAN-2		B	C	
HUMAN-1		B	C	HUMAN-1			C	D
OSU-GP			C	OSU-GP				D

Tabla 4.23: Comparación de los sistemas participantes para las medidas de evaluación extrínsecas

Podemos ver en las tablas que nuestro algoritmo se comporta especialmente bien en cuanto al tiempo de identificación y la tasa de error, y que se mantiene en la media respecto al tiempo de lectura.

Es conveniente discutir que existen deficiencias adicionales que surgen de considerar la tarea TUNA-REG como una composición de las tareas TUNA-AS y TUNA-R. La reducción de los tipos de expresión producidas por seres humanos a un conjunto de atributos provoca en algunos casos pérdidas de información. Esto es por ejemplo lo que ocurre cuando las expresiones humanas del corpus contienen atributos para los que se da información extra, como se puede comprobar comparando el `ANNOTATED-WORD-STRING` con el conjunto de atributos real que consta en el corpus. Por ejemplo, el corpus contiene ejemplos en los que el atributo `hasBeard` tiene un atributo anidado para indicar el color de la barba. Otros ejemplos proporcionan información sobre el color de la ropa que llevan los sujetos. Esta información no puede ser utilizada por la tarea de realización si antes se ha perdido en la tarea de selección de atributos, lo que reduce el formato disponible a un conjunto de atributos individuales sin estructurar.

Se podría conseguir un rango más amplio de realizaciones al utilizar una versión para la tarea TUNA-REG que abarcara la generación de referencias completa desde los datos iniciales de entrada sin representaciones intermedias de la solución generada hasta el momento. Probablemente una solución de este tipo mejoraría los valores de evaluación con respecto a las expresiones generadas manualmente en el corpus.

En términos más generales, parece que el corpus no contiene datos adecuados para ciertos niveles de planificación de oraciones como son las subtareas de elección léxica y elección sintáctica. Además, algunas de las variaciones encontradas en el corpus, como por ejemplo el uso libre de determinantes o la flexibilidad que exhiben los sujetos en la manera en que se refieren a las imágenes del dominio introduce cierto ruido. Cuestiones como éstas ocurren cuando las descripciones generadas por los evaluadores humanos involucran formas de elipsis o de estructuras gramaticalmente incorrectas para los atributos. Algunas de estas cuestiones se podrían reducir si se hiciera una versión refinada del corpus con más control sobre el experimento, asegurándose que los evaluadores o bien describen los elementos como imágenes o bien como las cosas reflejadas en las imágenes, por ejemplo.

4.4.3. Resultados de Evaluación en la Tercera Tarea Competitiva para la Generación de Expresiones de Referencia

La tercera tarea competitiva para la generación de expresiones de referencia (sección 2.3.3) estaba definida como la tarea TUNA-REG de la edición anterior, es decir, cubriendo la generación completa de expresiones de referencia a partir de la situación de los referentes. Para más detalles consultar (Gatt et al., 2009).

Al igual que en la competición anterior, se realizó tanto evaluación intrínseca como extrínseca. Como medidas de evaluación intrínseca se volvieron a utilizar la distancia de edición (DE), la precisión, y los coeficientes BLEU y NIST. Para la evaluación extrínseca se leía en voz alta las referencias a los evaluadores humanos y se medía el tiempo que tardaban en identificar el referente objetivo y la tasa de acierto. Las medidas intrínsecas eran evaluadas por los participantes sobre los datos de desarrollo y por los organizadores sobre los datos de test. Las medidas extrínsecas eran sólo utilizadas por los segundos.

A esta competición se presentaron dos algoritmos, el evolutivo de selección de atributos expuesto en la sección 4.2.3, y el basado en casos para lexicalización comentado en la sección 4.3.3. Como las medidas de evaluación sólo evaluaban las salidas en lenguaje natural, se probaron las dos soluciones juntas, y por separado junto con otras soluciones ya evaluadas otros años. Así se presentaron un total de tres sistemas a la competición:

- **NIL-UCM-EvoTAP**. Realiza la selección de atributos usando el algoritmo evolutivo, y la realización usando el algoritmo NIL-UCM-BSC evaluado el año anterior.

- **NIL-UCM-ValuesCBR**. Selecciona los atributos usando la solución NIL-UCM-MFVF y lleva a cabo la realización usando la aproximación basada en casos.

- **NIL-UCM-EvoCBR**. La selección de atributos es llevada a cabo por la solución evolutiva, y la realización por el algoritmo basado en casos.

Los resultados obtenidos por los tres sistemas sobre los datos de desarrollo se muestran en la Tabla 4.24.

		Precisión	Dist. Edición	BLEU 1	BLEU 2	BLEU 3	BLEU 4
EvoTAP	**Mobiliario**	8 %	4,87	0,44	0,33	0,24	0,18
	Personas	3 %	6,04	0,39	0,25	0,15	0,00
	Ambos	6 %	5,41	0,41	0,29	0,20	0,13
ValuesCBR	**Mobiliario**	1 %	5,91	0,44	0,31	0,20	0,13
	Personas	1 %	5,80	0,43	0,28	0,17	0,08
	Ambos	1 %	5,86	0,44	0,30	0,19	0,11
EvoCBR	**Mobiliario**	4 %	5,77	0,39	0,26	0,18	0,13
	Personas	1 %	6,94	0,41	0,25	0,16	0,08
	Ambos	3 %	6,31	0,41	0,26	0,17	0,11

Tabla 4.24: Resultados sobre los datos de desarrollo para los tres sistemas

Las evaluaciones sobre los datos de test se estudiaron por separado desde los puntos de vista intrínseco y extrínseco. En la Tabla 4.25 se pueden ver los resultados de los sistemas participantes para las medidas de evaluación intrínsecas. En esta tarea se calcularon además todas las métricas comparando los resultados de los sistemas participantes con dos conjuntos de soluciones distintas creadas por evaluadores humanos. La tabla de resultados

también incluye los valores de comparar los dos conjuntos de referencias humanas (HUMAN-1 y HUMAN-2) entre sí usando las mismas métricas que se utilizan sobre el resto de sistemas. De nuestros tres algoritmos el que obtiene mejores resultados es EvoTAP, seguido por ValuesCBR y EvoCBR. Aunque ninguno de los ellos es significativamente distinto al resto de sistemas, los tres presentan resultados bastante mejores que los conjuntos de referencias generadas por humanos.

	Ambos dominios				**Personas**				**Mobiliario**			
	Prec.	*DE*	*BLEU*	*NIST*	*Prec.*	*DE*	*BLEU*	*NIST*	*Prec.*	*DE*	*BLEU*	*NIST*
GRAPH	12.50	6.41	0.47	2.57	8.93	7.04	0.43	2.16	16.07	5.79	0.51	2.26
IS-FP-GT	3.57	6.74	0.28	0.75	3.57	7.04	0.37	0.94	3.57	6.45	0.13	0.36
EvoTAP	6.25	7.28	0.26	0.90	3.57	8.07	0.20	0.45	8.93	6.48	0.34	1.22
USP-EACH	7.14	7.59	0.27	1.33	0.00	9.04	0.11	0.46	14.29	6.14	0.41	2.28
ValuesCBR	2.68	7.71	0.27	1.69	3.57	8.07	0.23	0.94	1.79	7.34	0.28	1.99
EvoCBR	2.68	8.02	0.26	1.97	0.00	9.07	0.19	1.65	5.36	6.96	0.35	1.69
HUMAN-2	2.68	9.68	0.12	1.78	3.57	10.64	0.12	1.50	1.79	8.71	0.13	1.57
HUMAN-1	2.68	9.68	0.12	1.68	3.57	10.64	0.12	1.41	1.79	8.71	0.12	1.49

Tabla 4.25: Resultados de evaluación intrínseca de los sistemas participantes sobre los datos de test

Los valores obtenidos durante la evaluación extrínseca para el tiempo de identificación y el porcentaje de precisión se muestran en la Tabla 4.26. En este caso el algoritmo EvoTAP vuelve a ser el que mejor se comporta, seguido otra vez por ValuesCBR y EvoCBR. Todos ellos obtienen resultados razonablemente similares al resto de sistemas.

	Ambos dominios		**Personas**		**Mobiliario**	
	% Prec.	*Vel. de ident.*	*% Prec.*	*Vel. de ident.*	*% Prec.*	*Vel. de ident.*
GRAPH	0.96	3069.16	0.95	3081.01	0.96	3057.31
HUMAN-1	0.91	3517.58	0.95	3323.76	0.88	3711.41
USP-EACH	0.90	3067.16	0.86	3262.79	0.95	2871.53
EvoTAP	0.88	3159.41	0.88	3375.17	0.89	2943.65
ValuesCBR	0.87	3262.53	0.80	3447.50	0.93	3077.56
HUMAN-2	0.83	3463.88	0.89	3647.41	0.77	3280.35
EvoCBR	0.81	3362.22	0.75	3779.64	0.88	2944.80
IS-FP-GT	0.68	3167.11	0.89	2980.30	0.46	3353.91

Tabla 4.26: Resultados de evaluación extrínseca de los sistemas participantes sobre los datos de test

La solución evolutiva presenta resultados ligeramente inferiores a los de otras soluciones de selección de atributos en competiciones anteriores. El punto clave en los algoritmos evolutivos es la función de aptitud, y parece que en este caso la escogida no resulta apropiada para el problema que nos ocupa. Por ejemplo, los pesos de los atributos se calculan usando su frecuencia absoluta en los datos de entrenamiento, pero se podría haber utilizado un algoritmo más refinado para este fin. Una posibilidad sería la utilizada en la solución basada en el valor más frecuente (sección 4.2.2) que tenía en cuenta la frecuencia de un atributo dependiendo de su valor.

Para la solución basada en casos presentada fue necesario descartar algunos de los casos de los datos de entrenamiento que introducían excesivo

ruido en los resultados. Frases como *the guy on the northeast corner is in red box* o *the third man* son algunos ejemplos. Los resultados mejoraron razonablemente usando los datos de entrenamiento refinados para crear la base de casos. Se puede por tanto considerar que si se implementa un solución basada en casos para realizar cualquier tipo de tarea, la base de casos debe ser construida cuidadosamente para que sea representativa de todas las situaciones posibles pero sin casos extremadamente extraños que puedan afectar negativamente a los resultados.

Finalmente, en la comparación entre los dos conjuntos de referencias humanas HUMAN-1 y HUMAN-2, se puede ver que aunque son más fáciles de entender e identificar en la evaluación extrínseca, se comportan bastante peor que los sistemas participantes en la intrínseca. Esto refuerza la idea que ya hemos ido comentando de que la forma en que los seres humanos generan referencias es muy personal y no se puede generalizar de manera sencilla.

Resumen y Conclusiones

En este capítulo se han presentado distintas soluciones a los problemas que nos encontramos al generar expresiones de referencia básicas en forma de frase nominal.

Para empezar se ha abordado el problema de la elección del nivel de abstracción para la generación de las referencias cuando se encuentra disponible una fuente de conocimiento organizada como una taxonomía sobre los conceptos del dominio.

En el caso de la selección de atributos se han presentado dos soluciones simples basadas en el algoritmo incremental de Reiter y Dale (1992) y otra utilizando algoritmos evolutivos con la idea de que el problema se puede formalizar como uno de búsqueda en el que se conoce cómo de buena se considera una solución pero no cuál es el proceso para conseguirla.

Para la lexicalización se ha propuesto una solución general de elección léxica que puede ser agregada a cualquier método general de lexicalización y que usa WordNet como recurso lingüístico. También se ha presentado una lexicalización sencilla basada en las opciones más usadas, y finalmente una solución basada en casos para intentar tener en cuenta el estilo personal a la hora de realizar la lexicalización de las referencias.

En el caso de las soluciones de selección de atributos y lexicalización, se ha llevado a cabo además la evaluación de algunas de ellas según los parámetros de las tareas de evaluación competitiva para la generación de expresiones de referencia. En general los resultados obtenidos son buenos y los sistemas se comportan adecuadamente en comparación con el resto de participantes.

Capítulo 5

Figuras Retóricas basadas en Similitudes entre Dominios para la Descripción de Entidades

" – I think it was "Blessed are the cheesemakers!"
– What's so special about the cheesemakers?
– Well, obviously it's not meant to be taken literally.
It refers to any manufacturers of dairy products."
— Life of Brian (1979)

Figuras retóricas como la analogía, la metáfora o la comparación, basadas en similitudes entre dominios, constituyen algunas de las maneras con las que los seres humanos enriquecen el lenguaje que usan.

Este tipo de lenguaje figurativo se usa comúnmente para enfatizar ciertos aspectos de las entidades involucradas en un discurso. Tanto si la intención es facilitar la comprensión de estas entidades por parte del lector u oyente, o simplemente subrayar la importancia de un trozo de información del texto, figuras retóricas como la analogía o la comparaciones son generalmente usadas por los seres humanos para este propósito. Por ejemplo, la analogía *the butcher was clean and precise, a surgeon between butchers* subraya lo limpio y preciso que era el carnicero, algo poco común entre los de su clase, más que la oración *the butcher was clean and precise*. De la misma manera, una comparación como *she was pretty as a rosebud* centra la atención en cómo de importante es saber que ella era guapa, o cómo de importante es esta propiedad para ella, mucho más que el simple uso de la frase *she was pretty*.

En este capítulo se explora cómo un discurso dado puede ser enriquecido usando comparaciones y analogías para resaltar información relevante o para ayudar a la comprensión por parte del usuario de entidades desconocidas. Se pretende utilizar este tipo de figuras como un primer paso hacia la

comprensión de la función distintiva de las referencias, según los objetivos 3 y 4 de la sección 1.5. Para ello se realizará en primer lugar una propuesta de alto nivel de cómo y en qué ocasiones puede ser apropiado introducir una comparación o analogía en un discurso (sección 5.1). Posteriormente se presentará como caso de estudio una implementación realizada en forma de arquitectura multiagente que resuelve las diferentes cuestiones que surgen al implementar la propuesta anterior (sección 5.2).

5.1. Análisis Teórico de Analogías y Comparaciones en Texto

Tanto la analogía como la comparación son dos mecanismos cognitivos que han sido reconocidos como la base del razonamiento entre diferentes dominios teniendo en cuenta las similitudes entre éstos, y se usan frecuentemente en la comunicación humana. Asumiendo que el oyente o lector tiene suficiente conocimiento de un dominio destino, usar una analogía o comparación adecuada puede ser una forma más económica de comunicar información que explicar un conjunto de hechos sobre un concepto dado. Si los ordenadores fueran capaces de usar este tipo de figuras, esto constituiría una gran ventaja en contextos como aplicaciones pedagógicas, donde se necesita transmitir conocimiento novedoso y complejo, o simplemente como una herramienta adicional para comunicar información compleja en cualquier tipo de entorno interactivo. La tarea de identificar analogías o comparaciones adecuadas es difícil incluso para los seres humanos, y en muchas ocasiones se considera que tiene un alto grado de creatividad. Sin embargo, el proceso de introducir una analogía o comparación como parte de un discurso sí está dentro de las posibilidades de la tecnología de Generación de Lenguaje Natural actual.

5.1.1. Propuesta para la Introducción de Analogías y Comparaciones en Texto

La tarea de generar analogías o comparaciones como parte de cualquier tipo de discurso involucra un conjunto de retos que necesitan ser resueltos en módulos separados. El trabajo aquí presentado pretende estudiar la generación de este tipo de figuras basadas en similitudes entre dominios desde el punto de vista de enriquecer un texto dado representado en algún tipo de forma conceptual que un generador es capaz de transformar en texto. El proceso completo constará de tres fases diferenciadas. Para empezar, una tarea básica es decidir en qué situaciones resulta adecuado utilizar una de estas figuras en el texto. Para tomar esta decisión se debe considerar qué función se espera que realicen de cara a la información a transmitir, y en qué puntos del discurso se pueden encontrar las estructuras que nos indican que una

figura de este tipo puede ser útil. Después es necesario identificar el dominio destino en el que se buscarán las analogías o comparaciones y realizar la correspondencia entre los dos dominios en base a conceptos de los mismos. Finalmente está la tarea de insertar la estructura lingüística apropiada para la analogía o comparación en el discurso final, incluyendo tanto su construcción como la elección de su colocación dentro del texto original si es necesario.

Para poder llevar a cabo el proceso completo ya comentado es necesario recurrir a alguna fuente de conocimiento que sea lo suficientemente rica como para permitir la creación de analogías y comparaciones a partir de ella. Esta riqueza debe incluir no sólo gran cantidad de datos, sino también cierta profundidad en los mismos. Si un concepto de este conocimiento no está abundantemente relacionado con otros conceptos del mismo será difícil encontrar una base conceptual sobre la que construir las correspondencias que son la base de las analogías y comparaciones. Más adelante veremos también que cierta información sobre la intensidad de las relaciones puede resultar útil para decidir cómo de importantes o sobresalientes son ciertas relaciones.

La elección de en qué partes del discurso resulta apropiado utilizar una comparación o analogía depende mucho de cuál de ellas estemos tratando. Tanto los objetivos a cubrir como la información del discurso a tener en cuenta depende mucho de la figura concreta que estemos considerando. Realizaremos por tanto esta discusión más adelante. Los otros dos aspectos a tratar (identificación del dominio objetivo y la realización lingüística de la figura) sí tienen partes en común que veremos a continuación.

Identificar el Dominio Destino y las Correspondencias

La tarea de identificar un dominio destino que resulte apropiado como dominio objetivo es bastante compleja. Dado que la analogía o comparación debe contribuir a un acto de comunicación, parece razonable asumir que el dominio a buscar debe ser suficientemente conocido por los supuestos lectores u oyentes del discurso de manera que no requiera ninguna explicación adicional. Esto provoca una reducción del conjunto de posibilidades. También hace que la solución del problema dependa del tipo particular de usuario al que el discurso generado esté dirigido.

Una vez establecido un dominio destino particular, el próximo paso es identificar las correspondencias relevantes. Esto supone una operación elemental de comparación de los dos dominios dados para identificar posibles correspondencias válidas. Es necesario definir alguna heurística para seleccionar los candidatos aceptables para la analogía o comparación dentro del conjunto completo de correspondencias parciales. Una posible forma de hacerlo sería comprobar cuantas relaciones determinan una asociación, relaciones existentes en los datos que son las que sirven de base para la

analogía estructural. Se podría considerar que las correspondencias entre conceptos basadas en mayor número de relaciones son mejores candidatas para la analogía. También se podrían utilizar otras heurísticas dependiendo del objetivo que se pretende conseguir con la analogía o comparación.

Realización de la Analogía o Comparación en el Texto

Las comparaciones o analogías no pueden ser estudiadas como un fenómeno aislado. En muchos casos, el contexto que acompaña a un texto es necesario para guiar al lector a través de las inferencias que debe realizar para captar el significado de la analogía o comparación. Imaginemos la analogía *his uncle was a well-turned out shark*, que será difícilmente entendida por el lector si no sabe que estamos hablando de abogados. Sin embargo, en la frase *it was the well-turned out shark who won the trial*, la palabra *trial* hace que la frase sea identificada como dentro del dominio legal, y la analogía fácilmente inferida.

La inserción de una analogía o comparación en un texto dado se puede llevar a cabo de dos formas diferentes. Una manera consiste en respetar el texto original en su forma dada, y simplemente construir una oración adicional para expresar la analogía o comparación e insertarla en un lugar determinado. Una solución más compleja y rica sería añadir el correspondiente mensaje, representado en la misma notación conceptual usada por el contenido original del texto, a la entrada del sistema, y permitir al generador que se encargue de convertir el conjunto de mensajes como un todo en un texto coherente. Esta solución tiene la ventaja de permitir al generador reformular el discurso alrededor de la analogía para hacer que el texto final sea lingüística y estilísticamente más coherente.

Por ejemplo, imaginemos una analogía como *the princess was the Aphrodite of the royalty*. Tomemos un discurso que podría dar lugar a un texto simple como:

> *A princess lived in a castle. The princess loved a knight. She was the daughter of a king.*

La primera opción descrita correspondería a insertar una oración que describa la analogía en algún lugar del texto, en este caso después de la primera aparición del concepto al que corresponde:

> *A princess lived in a castle. The princess was the Aphrodite of royalty. The princess loved a knight. The princess was the daughter of a king.*

Sin embargo, si la analogía se añade a nivel conceptual, estará proporcionando al generador información útil que puede ser utilizada más adelante. Por ejemplo, sería posible que el sistema generara un texto como:

> *A princess lived in a castle. The princess was the Aphrodite of royalty. She loved a knight. The Aphrodite was the daughter of a king.*

donde la analogía ha ampliado el rango de elección a la hora de generar las referencias y permite mencionar a la princesa usando la analogía realizada.

En otros casos también podrían mejorar cuestiones como el uso uniforme de pronombres, por ejemplo.

5.1.2. Comparaciones

Las comparaciones tratadas en este trabajo corresponden a las *comparaciones ilustrativas* definidas por Milosavljevic y comentadas en la sección 2.6. Recordemos que una comparación ilustrativa es una comparación basada en familiaridad que no contiene diferencias entre las entidades comparadas.

Aquí hemos considerado la introducción de comparaciones en un texto desde el punto de vista de su utilidad para resaltar propiedades de ciertas entidades del discurso. Así, expresiones como *the boy is tall as a pine* refuerzan la idea de que el niño es alto mucho más que frases como *the boy is tall*, proporcionando al mismo tiempo más naturalidad al texto como haría un ser humano.

En el trabajo aquí presentado se han considerado los siguientes casos como contextos apropiados para emplear comparaciones que resaltan propiedades:

Propiedad importante. Cuando la entidad tiene una propiedad que es muy importante o destacable de su clase. La inserción de una comparación en estos casos, además de recalcar la propiedad desde el punto de vista de la descripción de la entidad, refuerza su pertenencia a esa clase. Por ejemplo, un caso en que podría ser apropiado usar una comparación de este tipo sería para mencionar lo guapa que es una princesa o lo valiente que es un príncipe, ya que en el dominio de los cuentos fantásticos se supone generalmente que las princesas son guapas y los príncipes valientes.

Propiedad atípica. Cuando la entidad tiene una propiedad que es atípica de su clase. Las propiedades atípicas de una clase no son aquellas que no están especificadas en la clase, sino las que son opuestas a las propiedades propias de una clase. Así, que un cirujano sea alto no es atípico de su clase, ya que el ser cirujano no implica ninguna altura específica. Sin embargo, el que un cirujano sea sucio o descuidado sí se puede considerar atípico.

Para que sea posible distinguir estas situaciones será necesario recurrir a algún tipo de conocimiento general que nos permita consultar las propie-

dades que pertenecen a diversos conceptos, así como si estas propiedades se pueden considerar importantes o atípicas para los mismos.

Elementos del Discurso a Considerar para Introducir las Comparaciones

Una vez decidida qué función tendrán las comparaciones, es necesario estudiar en qué puntos del texto se puede encontrar la información que lleva a crear una comparación, y dónde se insertará.

Dado que las comparaciones son formas de resaltar propiedades, los puntos del texto a considerar para la creación de comparaciones son aquellos en los que exponen cualidades o características de las entidades. Estas situaciones se dan en dos casos:

1. *Mensaje copulativo.* Cuando se utiliza un mensaje copulativo para dar información sobre las propiedades que ostenta una entidad. Por ejemplo, éste sería el caso de oraciones como *the butcher was precise* o *the princess was pretty.*

2. *Referencia.* Cuando las propiedades acompañan a la entidad en una referencia. Ejempos serían *the precise butcher* o *a pretty princess.*

Sin embargo, no todas estas situaciones son adecuadas para insertar una comparación. Si estamos considerando que una comparación tiene como finalidad resaltar una propiedad de una entidad, debería ser usada solamente en situaciones donde se está describiendo esa entidad. Un texto en el que se inserte la comparación en cualquier parte del discurso sólo porque se ha mencionado una propiedad no tendría sentido.

Las oraciones del tipo 1 expuesto anteriormente serán siempre consideradas como una descripción, y por tanto válidas a la hora de considerar la inserción de una comparación. Las oraciones del tipo 2 sólo se pueden considerar descripciones cuando estamos ante la primera mención de la entidad. En ese caso, una primera mención de una entidad acompañada por una propiedad es también una descripción de ésta.

Por tanto, de los casos anteriores nos quedaremos con el caso 1 y el caso 2 cuando la referencia sea la primera mención de la entidad. Cuando alguna de estas situaciones aparezcan en un discurso se considerará la introducción de una comparación.

Identificando el Destino de la Comparación

Una vez decidido según todo lo anterior que se va a realizar una comparación para una entidad del discurso y una de sus propiedades, será necesario aplicar algún tipo de algoritmo o solución para encontrar qué otro concepto

es adecuado para ser comparado. Esta tarea dependerá del tipo de implementación que se lleve a cabo para el proceso completo de enriquecimiento del texto, y será estudiado en secciones posteriores.

Realización de las Comparaciones

Dependiendo de la situación en la que nos encontremos, el mensaje correspondiente a la comparación se introducirá en el texto de una manera u otra.

- *Mensaje copulativo.* Si nos encontramos en el caso de que la propiedad a destacar aparece en un mensaje copulativo que describe a la entidad, simplemente se sustituye un mensaje por otro. Por ejemplo, si tenemos una oración *the girl is pretty* se transformaría en *the girl is as pretty as a rose.*

- *Referencia.* Si estamos en la situación en la que la propiedad aparece modificando la primera mención a una entidad, se añadiría el mensaje de la comparación justo después de la oración en la que aparece esta primera mención. Por ejemplo, si tuviéramos una referencia a un concepto X de la forma *a pretty girl is coming* obtendríamos algo como *A girl is coming. She is as pretty as a rose.*

5.1.3. Analogías

Otra figura retórica basada en similitudes entre conceptos que también puede ser útil para enfatizar información en un discurso es la analogía. Una analogía puede ser usada en lugar de un conjunto de propiedades si el destino de la analogía es algo para el que esas propiedades son representativas, además de ser conocido por el oyente o lector. En este trabajo se ha considerado que una analogía puede ser útil para enriquecer un texto dado en uno de los siguientes casos:

Entidad de clase desconocida. Cuando la entidad pertenece a una clase que se sabe es desconocida para el lector u oyente. En este caso, en lugar de dar una lista detallada de las propiedades de la entidad o la clase para que el lector sepa a qué nos estamos refiriendo, se usa una analogía. Así el lector u oyente se formará una idea de qué es la entidad y qué cosas puede inferir sobre ella a partir de la información ya conocida. Imaginemos por ejemplo que en un discurso se está hablando de la película La Guerra de las Galaxias, y que ésta es desconocida para el lector. En este caso, una analogía como *a storm trooper is like a knight in Star Wars* da información sobre que los *storm trooper* son como los caballeros de la Edad Media.

En este caso la utilidad que se persigue con la analogía es hacer entender algo desconocido utilizando cosas semejantes que pertenecen a dominios previamente conocidos.

Entidad con propiedades atípicas. Cuando una entidad tiene algunas propiedades que son atípicas en su clase. Al igual que con las comparaciones, se consideran propiedades atípicas de la clase las que son opuestas a las propiedades propias de una clase, y no sólo aquellas que simplemente no están especificadas. Un ejemplo podría ser un carnicero que es limpio y preciso, cualidades las dos que son atípicas para los carniceros, pero muy comunes entre los cirujanos. Una posible analogía sería *he was a surgeon between butchers.*

A diferencia del caso de las comparaciones, en las que se pretendía enfatizar una propiedad de una entidad, con las analogías lo que se pretende resaltar es que la entidad correspondiente se sale de lo común dentro de su tipo, y con la analogía se proporciona implícitamente por qué.

Para saber si una clase o concepto es desconocida para el lector u oyente será necesario tener algún tipo de modelo de usuario que contenga esta información. En este modelo debería constar todo el conocimiento previo que se puede considerar como conocido, de manera que a la hora de introducir una entidad en un discurso sea posible saber si su clase es conocida o no por el usuario.

Elementos del Discurso a Considerar para Introducir las Analogías

Una vez decidida la función que se espera de las analogías a la hora de enriquecer un discurso, es necesario estudiar en qué puntos del mismo se puede encontrar la información necesaria para decidir si una analogía es apropiada, y para crearla en caso de que se considere útil. Esto dependerá de si nos encontramos en el caso en el que la analogía se utiliza para presentar un concepto en función de otro, o para resaltar lo atípico de una entidad con respecto a su clase.

- *Entidad de clase desconocida.* Si la analogía tiene como función el dar información sobre un concepto que no es conocido utilizando otro que sí lo es, ésta debe situarse justo después de la presentación del concepto desconocido. Esta presentación podría no sólo referirse a la primera aparición del concepto, sino que dependiendo de la situación podría incluir toda la descripción que se haga de él. Por ejemplo, podría ocurrir que la descripción del concepto contuviera al menos parte de las propiedades que dan lugar a la analogía, y por tanto pudiera ser mejor incluirla antes de la analogía para facilitar su comprensión.

- *Entidad con propiedades atípicas.* Si la analogía tiene como función destacar lo atípica que resulta una entidad para su clase, será necesario incluir la analogía en un punto del texto donde no sólo se haya descrito a la entidad correspondiente, sino donde además se hayan mencionado las propiedades que la hacen tan atípica. Así, una analogía como *the butcher was like a surgeon between butchers* puede no tener mucho sentido si antes no se ha comentado que este carnicero en particular era cuidadoso y muy preciso en su trabajo.

Aunque en el primer caso parece que la analogía podría ser comprendida sin dificultad tanto antes como después de la descripción completa de un concepto, para simplificar el mecanismo completo de enriquecimiento a través de analogías hemos considerado que las analogías se insertarán siempre una vez que la entidad ha sido completamente descrita en el discurso.

Así, habrá que identificar en qué punto del discurso se puede considerar completa una descripción. Al igual que en el caso de las comparaciones, hemos supuesto que las descripciones se realizan a través de mensajes copulativos como *the man was brave*, por lo que la descripción de una entidad del discurso se considerará terminada tras el último mensaje copulativo que mencione alguna de sus propiedades.

Identificando el Destino de la Analogía

Una vez decidido según todo lo anterior que se va a realizar una analogía para una determinada entidad del discurso, habrá que utilizar algún tipo de algoritmo que sea capaz de encontrar los conceptos necesarios para ello. Esta tarea dependerá, al igual que en el caso de las comparaciones, de la implementación que se realice del proceso completo de enriquecimiento del texto. También será estudiado más adelante en este trabajo.

Lo que sí será común a cualquier tipo de algoritmo encargado de buscar el destino de la analogía es la información que se considerará como dominio completo de la entidad. También serán comunes los requisitos que se considerarán indispensables para escoger un dominio determinado como destino de la analogía. Veamos estos dos puntos con un poco más de detalle.

Generalmente el proceso de construcción del dominio origen a partir de un discurso ya existente requiere un preprocesamiento adicional. La información conceptual disponible para generar un texto no suele ser suficientemente rica para constituir un dominio con una estructura debidamente completa. Mientras que la representación de un dominio completo que participa en un texto debe corresponder a un complejo árbol taxonómico, el tipo de entrada que un generador recibe suele ser más un conjunto de hechos que podrían haber sido las hojas del árbol taxonómico correspondiente.

A partir de este tipo de representación conceptual, el contexto en el que la analogía se va a utilizar puede ser extraído, asumiendo que este contexto

es un subconjunto del conjunto de hechos recibido como entrada por el generador. Antes de que pueda ser utilizado para encontrar una correspondencia con el dominio destino completo, debe ser enriquecido para asegurar que al menos una parte significativa de su estructura taxonómica es representada. El dominio completo de la entidad para la que se está buscando la analogía, y que se considerará dominio origen de la misma, debe estar formado no sólo por las propiedades de la entidad sino también por toda la información relacionada con ella y que pueda ser útil a la hora de buscar otro concepto con el que establecer una analogía.

Esto se consigue expandiendo el contexto con el uso de la información relacionada que se puede encontrar en la entrada del generador, además de con una base de conocimiento general. Una vez que este enriquecimiento ha sido llevado a cabo, el dominio inicial ya enriquecido puede ser comparado con el dominio destino en busca de analogías estructurales. El proceso proporcionará una correspondencia parcial entre el contexto enriquecido y el objetivo, donde ciertos conceptos en el dominio origen serán asociados con ciertos conceptos del dominio destino. Cada una de las asociaciones en estas correspondencias parciales es un candidato para establecer una analogía.

A la hora de buscar el dominio destino no bastará con encontrar una posible correspondencia entre dominios para considerar que ésta es válida para dar lugar a una analogía. Dependiendo de la situación inicial en la que nos encontremos, una analogía sólo será válida si cumple ciertos criterios.

En el caso en que el propósito de la analogía sea resaltar una entidad que dadas sus propiedades es considerada atípica para su clase, las propiedades que la hacen atípicas deben estar presentes en la clase destino de la analogía. De hecho, para que la relación entre las propiedades atípicas y la analogía sea comprendida correctamente, dichas propiedades tienen que ser consideradas importantes o relevantes para la clase destino.

Si el propósito de la analogía es facilitar la comprensión de un concepto desconocido a partir de un concepto conocido, será necesario recurrir al modelo de usuario ya mencionado para comprobar que la clase destino es apropiadamente conocida por el usuario.

Si se cumplen estas condiciones, cada una en el caso que le corresponde, se considerará que se puede insertar la analogía correspondiente en el texto.

Realización de las Analogías

Como ya hemos comentado, el punto del discurso donde se debe estudiar la inserción de una analogía es cuando se da por finalizada la descripción de una entidad. Si consideramos que las descripciones están formadas por mensajes copulativos, el punto del discurso donde debe insertarse la realización lingüística que corresponde a la analogía es después del último mensaje copulativo que mencione alguna de las propiedades de la entidad.

5.2. Caso de Estudio: Sistema Multiagente para la Generación de Analogías y Comparaciones

Como caso de estudio de las ideas teóricas aquí presentadas, se ha realizado una implementación que apoyándose en fuentes de conocimiento previamente existentes genera automáticamente analogías y comparaciones siguiendo las directrices ya comentadas. Esta arquitectura toma la forma de un sistema multiagente que permite conectar entre sí módulos ya implementados que realizan parte de la funcionalidad necesaria. Veamos en esta sección qué decisiones se han tomado para cada una de las cuestiones previamente planteadas.

5.2.1. Conocimiento General

Cuando los seres humanos utilizan cualquier tipo de figura retórica se pone en marcha un complejo proceso de inferencia sobre todo el conocimiento aprendido previamente. Gracias a este conocimiento sobre el mundo que le rodea, y a las inferencias que puede realizar sobre él, una persona es capaz de entender frases como *he is a finance shark* o *she is as pretty as a flower*.

El proceso llevado a cabo para generar o comprender este tipo de figuras es complejo. Requiere por un lado conocimiento sobre las características de los distintos conceptos involucrados, destacando especialmente las que son más sobresalientes en ellos. Además es necesaria cierta capacidad para encontrar similitudes y diferencias entre las características de los conceptos de manera que sea posible realizar deducciones correctas y lógicas. Por ejemplo, en el caso de *su tío era un tiburón de las finanzas* se entiende que se está hablando de una persona que es despiadada en su trabajo en el mundo de las finanzas, pero no se asume que se parezca a un tiburón ni en su anatomía ni en que tenga que vivir bajo el agua. En el caso de *la niña era tan bonita como una flor* se está considerando que las flores son generalmente bonitas, y se está resaltando esa propiedad sobre la niña.

Como ya se comentó en la sección 5.1.1, si se desea realizar una generación automática de este tipo de figuras es necesario disponer de algún tipo de conocimiento general que permita realizar los razonamientos e inferencias expuestos, al menos parcialmente. Para este trabajo nos hemos decidido por dos fuentes de conocimiento diferentes. Utilizaremos WordNet (sección 2.5.2) para la generación de comparaciones basándonos en su estructura taxonómica y las glosas que acompañan a los conceptos. Además se usarán los dominios definidos por Tony Veale para su sistema Sapper (sección 2.6.1) como conocimiento general tanto para la generación de analogías como para la de comparaciones.

Recordemos que en Sapper los dominios se representan como grafos donde los nodos son conceptos y las aristas entre nodos son relaciones entre estos conceptos. Cada arista está marcada con una etiqueta que indica a

Valor de la Intensidad	**Significado Considerado**	**Fuerza de la Relación**
[-1,-0.8]	Relación inversa	Alta
(-0.8,-0.3)	Relación inversa	Normal
[-0.3,0)	Relación inversa	Débil
[0,0.3]	Misma relación	Débil
(0.3,0.8)	Misma relación	Normal
[0.8,1]	Misma relación	Alta

Tabla 5.1: Significado considerado dependiendo del valor de la intensidad

qué relación corresponde, y por un número real entre -1 y 1 que denota a la intensidad de la relación. En la Tabla 5.1 se muestran los significados relativos que han sido considerados dependiendo del valor real de las intensidades de las relaciones en Sapper.

Como se puede ver las intensidades negativas significan que la relación correspondiente a la arista es la contraria a la indicada por la etiqueta, y cuanto mayor es el valor absoluto de la intensidad, mayor es la fuerza de la relación entre los conceptos.

Una parte del conocimiento de Sapper se puede ver en el apéndice C de esta memoria. Se muestran principalmente los conceptos que serán usados más adelante en los ejemplos.

5.2.2. Arquitectura Multiagente para la Creación e Inserción de Figuras Retóricas en Texto

A la hora de escoger una arquitectura sobre la que montar todas las piezas consideradas en este trabajo fue necesario valorar las necesidades inmediatas y futuras para una implementación de estas características. De esta manera, era necesario proporcionar una arquitectura que incluyera distintos métodos de enriquecimiento de un texto, así como la posibilidad de que fueran usados por separado o utilizando distintas fuentes de conocimiento. También era necesaria la integración de distintas soluciones previamente implementadas en distintos lenguajes y paradigmas de programación.

Tanto para permitir una perspectiva amplia a la hora de implementar el enriquecimiento de los textos con diversas figuras retóricas, como para permitir la integración de distintos módulos pertenecientes a diversas contribuciones, una plataforma multiagente puede cubrir nuestras necesidades apropiadamente. De esta forma, se pueden distribuir diferentes papeles entre distintos agentes, siendo cada uno responsable de una tarea de propósito específico, actuando autónomamente e interactuando sólo a través de canales de comunicación claramente definidos. Esta descripción coincide con la arquitectura OAA expuesta en la sección 3.3.2.

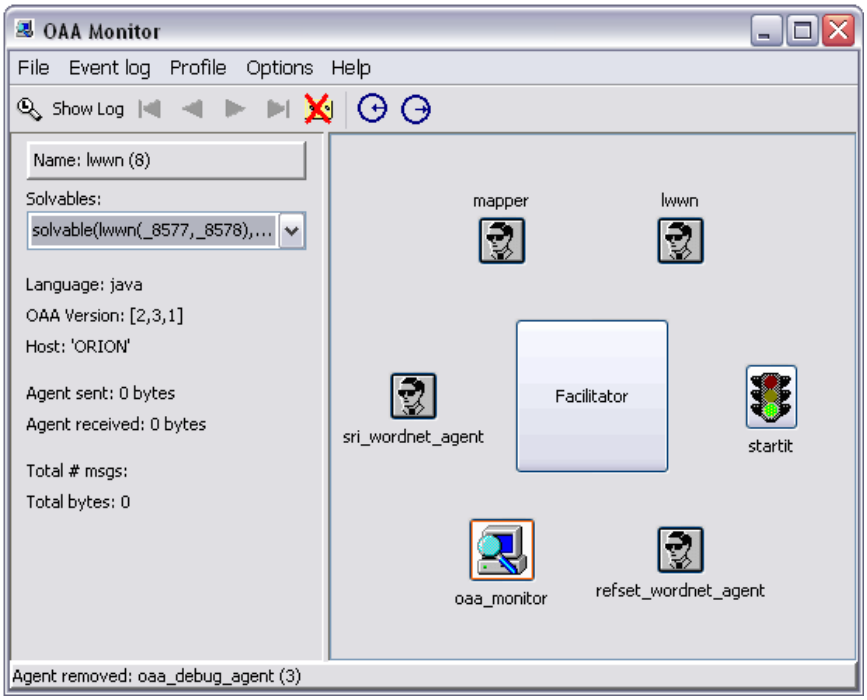

Figura 5.1: La sociedad de agentes vista en el monitor de OAA

La arquitectura OAA es suficientemente abierta y modular para la implementación y prueba del trabajo aquí presentado, además de permitir la conexión de futuras funcionalidades fácilmente. En este caso, la arquitectura multiagente se compone de un agente de comunicación con WordNet que se encarga de manejar las consultas a la base de datos léxica, un agente que obtiene las posibles lexicalizaciones a utilizar para un concepto, un agente proxy (el *Facilitator* de OAA que se encarga de administrar las consultas y comunicaciones entre los agentes), el agente que se encarga de crear las analogías, y otro que se usa para obtener comparaciones usando WordNet. El agente encargado del proceso de generación de texto se conectaría a este sistema multiagente. Excepto el agente generador de texto, el resto son agentes implementados previamente a la realización de este trabajo. El hecho de que a pesar de ello puedan ser utilizados conjuntamente sin problemas es una de las ventajas de este tipo de arquitectura.

Esta sociedad de agentes se muestra en la Figura 5.1. A continuación describiremos en detalle cada uno de estos agentes.

El AGENTE WORDNET es esencialmente un servidor para todas las consultas dirigidas a WordNet, y fue implementado en Prolog por Chris Culy (Culy, 2002). Para las consultas sobre cada uno de los elementos y categorías de WordNet existe un predicado OAA determinado. Por ejemplo, para pre-

guntar por la glosa (la descripción informal de un concepto) de un *synset*, se envía el siguiente mensaje a este agente:

$$wn_g(SynsetID, Gloss).$$

donde $SynsetID$ es el número que identifica el *synset*, mientras $Gloss$ contendrá la respuesta que este agente proporcionará.

El AGENTE JMAPPER se encarga de producir las analogías propiamente dichas, realizando los mapeamientos necesarios entre dos dominios dados como origen y destino. Se trata de una implementación del algoritmo descrito en la sección 2.6.1 desarrollado en Java por Rui P. Costa, Hugo Costa y Francisco Pereira. En esta implementación el algoritmo original ha sido ligeramente modificado para mejorar su eficiencia y escalabilidad, pero siempre manteniendo sus principios generales.

El AGENTE REFSET se encarga de proporcionar un conjunto de lexicalizaciones alternativas para una palabra dada, y fue desarrollado en Prolog por Francisco Pereira (Pereira et al., 2006). Recibe una etiqueta léxica que llamaremos VBWord para un concepto, junto a su categoría gramatical (sustantivo, adjetivo, etc.). La VBWord será la semilla para la búsqueda del correspondiente *synset* en WordNet. Después de seleccionar el *synset* a explorar, el agente RefSet recolecta su conjunto de sinónimos ordenados por el valor de *tag count*, y el conjunto de sus hiperónimos ordenados por su distancia en la jerarquía (primero el *padre*, luego el *abuelo*, etc.). En esta arquitectura, el agente RefSet usa servicios de WordNet y del Facilitator.

El AGENTE LIGHTWEIGHT WORDNET o LWWN es un *wrapper* escrito en Java para un motor desarrollado por Nuno Seco (Seco, 2005). Este motor usa el servidor de base de datos Lucene (Gospodnetic y Hatcher, 2004) para proporcionar un mecanismo de consulta eficiente sobre Extended WordNet (Mihalcea y Moldovan, 2001). El agente transmite las consultas a Lucene y busca la información que rellenará los huecos correspondientes de la estructura para las comparaciones *X is as Y as Z*.

5.2.3. Comparaciones

En la sección 5.1.2 se mostraba la base teórica del uso de comparaciones para enriquecer un discurso dado con información que permita resaltar ciertas propiedades de las entidades del discurso. En esta sección se tratarán las cuestiones específicas de implementación dentro de la arquitectura multiagente presentada.

Recordemos que se había tomado la decisión de utilizar comparaciones para resaltar ciertas propiedades de las entidades del discurso, y que se desea resaltar estas propiedades cuando son o bien importantes para la clase de la entidad, o bien atípicas para la misma. Para determinar si una propiedad destaca o es atípica de una clase se utiliza el conocimiento general de Sapper comentado en la sección 5.2.1, considerando propiedades de una clase

los conceptos con los que está relacionado siguiendo la relación *attr*. Se considerará por tanto que una propiedad destaca respecto a una clase cuando su intensidad es mayor que 0.8 o menor que -0,8. Sería éste el caso de un neurocirujano, que en el conocimiento general que manejamos tiene como atributo *precise* con una intensidad de 0.9. Lo mismo ocurre con hacker y su atributo *social* con intensidad -0.8, que refleja que los hackers por lo general no son sociables, y además no lo son en gran medida.

Otra cuestión a tratar es la elección del concepto con el que se comparará la entidad que aparece en el discurso. Como ya se comentaba en la sección 5.1.1, es necesario gran cantidad de conocimiento para realizar las inferencias necesarias para generar comparaciones. En este trabajo presentamos dos soluciones, una basada en las glosas de WordNet, y otra basada en las similitudes entre los dominios utilizados como conocimiento general.

Pares Referente-Propiedad como Fuente de la Comparación

Cuando se da una de las dos situaciones que hemos considerado como válidas para dar lugar a una comparación, tendremos como información para su generación la entidad que será comparada y la propiedad que se desea resaltar. El procedimiento será diferente dependiendo de si la propiedad pertenece a la clase o no. Para saberlo se realiza la comprobación pertinente en el conocimiento general utilizado.

Si la propiedad pertenece a la clase, el significado de la propiedad y el proceso a seguir es diferente dependiendo del signo de la intensidad. Se consulta la intensidad que une al concepto con su propiedad:

- *Intensidad positiva.* Si el valor de la intensidad es positivo, significa que la propiedad pertenece a la clase y nos encontramos ante el caso de que se quiere destacar una propiedad importante para la clase. Como ya hemos mencionado, en este caso sólo se generará una comparación si el valor de la intensidad es mayor que 0.8.

- *Intensidad negativa.* Si el valor de la intensidad es negativo, significa que la propiedad que pertenece a la clase es la contraria a la que se ha mencionado en el texto. Por tanto, nos encontramos ante el caso en el que se usa la comparación para resaltar una propiedad atípica de la clase. También en este caso sólo se genera la comparación si el valor de la intensidad es menor que -0.8.

Si la propiedad no pertenece a la clase todavía quedan opciones por considerar. Dada la organización del conocimiento general utilizado en este trabajo, es posible que una propiedad sí pertenezca a una clase si por ejemplo su antónimo pertenece con intensidad negativa.

Para comprobarlo se buscan todos los antónimos de la propiedad realizando las llamadas necesarias al agente WordNet del sistema multiagente presentado. Se pueden dar dos casos:

- *Alguno de los antónimos de la propiedad pertenece a la clase.* La situación dependerá de la intensidad de la relación entre la clase y el antónimo:

  - *Intensidad positiva.* Si la intensidad es positiva, significa que la propiedad inicial es atípica, ya que su antónimo es lo corriente para la clase. La comparación para resaltar esta propiedad atípica se realiza, como en otros casos, sólo si la intensidad es mayor que 0.8.
  - *Intensidad negativa.* Si la intensidad es negativa, significa que estamos ante una propiedad común de la clase, sólo que en lugar de estar expresada en sentido positivo, está expresada en sentido negativo usando intensidades negativas. Se realizará la comparación sólo si la intensidad es menor que -0.8.

- *Ninguno de los antónimos de la propiedad pertenece a la clase.* Eso significa que la propiedad no está relacionada con la clase ni de forma positiva ni de forma negativa. No se realiza por tanto ninguna comparación al no encontrarnos en ninguno de los supuestos considerados.

Destino de la Comparación utilizando WordNet

Cuando la fuente de conocimiento a utilizar para la generación de comparaciones es WordNet, se realizan las llamadas necesarias al agente LWWN del sistema multiagente para encontrar las intersecciones de propiedades que estamos buscando, y que realizaremos en oraciones *X is as Y as Z*. Estas consultas buscan tanto en la taxonomía de conceptos como en los contenidos de las glosas. El proceso consiste en recorrer la lista de hiperónimos de la entidad X hasta encontrar uno para el que el agente LWWN pueda encontrar otra entidad Z con la que comparar a X, siempre dada una determinada propiedad Y. Por ejemplo, si preguntamos por algo que es un *male* y tiene la palabra *handsome* en su glosa, obtendremos el nombre *Adonis*. De esta forma obtenemos salidas muy interesantes, pero con frecuencia la respuesta de la consulta será el conjunto vacío.

En el marco de la Guerra de las Galaxias, uno de los dominios de Sapper, algunas comparaciones obtenidas son las siguientes:

- El concepto *Chewbacca* tiene como uno de sus atributos *hairy*. Después de buscar en la lista de sus hiperónimos (*wookie*, *alien*, *creature*), el agente LWWN encuentra que para el hiperónimo *creature* la palabra *mammoth* contiene *hairy* en su glosa. De esta forma, la comparación obtenida sería *Chewbacca is as hairy as a mammoth.*

- El concepto *Darth Vader* tiene como uno de sus atributos *evil*. Siguiendo el mismo proceso, el agente LWWN encuentra que para el hiperóni-

mo *person* la palabra *hellhound* contiene *evil* en su glosa. Así obtenemos la comparación *Darth Vader is as evil as a hellhound.*

- El concepto *Princess Leia* tiene como uno de sus atributos *beautiful.* El agente LWWN recupera que para el hiperónimo *woman* la palabra *nymph* tiene *beautiful* en su glosa. La comparación obtenida sería *Princess Leia is as beatiful as a nymph.*

- El concepto *Emperor* tiene como uno de sus atributos *powerful.* El agente LWWN recupera que para el hiperónimo *man* la palabra *big businessman* tiene *powerful* en su glosa. La comparación obtenida sería *The Emperor is as powerful as a big businessman.*

Las tres primeras comparaciones obtenidas son más que aceptables. Sin embargo, la última resulta poco adecuada porque no hay similitudes entre un hombre de negocios y el Emperador aparte de que ambos son hombres. Más adelante exploraremos otra solución a la hora de generar referencias que tenga más en cuenta las similitudes entre conceptos.

En el marco de los cuentos fantásticos se han obtenido también las siguientes comparaciones:

- El concepto *princess* tiene como uno de sus atributos *pretty.* Después de recorrer la lista de hiperónimos del concepto *princess*, el resultado dado por el agente LWWN es que para el hiperónimo *person* ha encontrado los conceptos *rosebud* y *pin up* que contienen *pretty* en su glosa. De esta forma, obtenemos las comparaciones *the princess was as pretty as a rosebud* o *the princess was as pretty as a pin up.*

- El concepto *knight* tiene como uno de sus atributos *handsome.* Siguiendo el mismo proceso, el agente LWWN encuentra que para el hiperónimo *person* el concepto *Adonis* tiene *handsome* en su glosa. Así obtenemos la comparación *the knight was as handsome as Adonis.*

- El concepto *king* tiene como atributo *stern.* El agente LWWN recupera que para el hiperónimo *person* el concepto *Dutch uncle* tiene *stern* en su glosa. Así se obtiene la comparación *the king was as stern as a Dutch uncle.*

Uno de los momentos más gratificantes durante el desarrollo del sistema ocurrió cuando el sistema empezó a producir frases que indicaban un conocimiento del inglés más amplio que el de alguno de los desarrolladores involucrados. Esto se debe directamente a la amplia cobertura proporcionada por WordNet. A pesar de todo, es llamativo cuando un programa que has desarrollado, lo que implica que uno sabe muy bien cómo trabaja, consigue producir resultados sorprendentes. Incluso más espectacular es el hecho de que el sistema parece saber inglés mejor que uno mismo.

Destino de la Comparación utilizando Técnicas de Alineamiento Estructural

Una correspondencia entre un concepto X y un concepto Y producidos por alineamiento estructural debería enfatizar alguna correspondencia particular entre los dos conceptos, de manera que respecto a cierto punto de vista, el papel de uno de los conceptos en su dominio puede ser proyectado al otro concepto en su propio dominio. Si dos conceptos pueden considerarse análogos o muy similares, y uno de ellos tiene como propiedad remarcable la propiedad que se va a destacar del otro concepto, parece apropiado llevar a cabo una comparación entre ellos.

Para la creación de comparaciones usando alineamiento estructural sobre los dominios de Sapper se ha utilizado el agente jMapper. Una vez que un concepto y una de sus propiedades han sido escogidos para dar lugar a una comparación, se utiliza el agente jMapper para buscar posibles alineamientos entre este concepto y todos los conceptos de los dominios de Sapper. Cada una de las correspondencias obtenidas es comprobada, y si el concepto encontrado tiene como propiedad importante o remarcable la que se está buscando, se selecciona el concepto para la comparación. Por ejemplo, si estamos buscando una comparación para un carnicero que es atípicamente limpio, la correspondencia carnicero-cirujano aparece destacando para la clase cirujano el ser limpio (intensidad 0.8). Se obtendría la comparación *the butcher is clean as a surgeon.* Sin embargo, aunque hay otras entidades que también destacan por ser limpias en los dominios (como un quirófano o un escalpelo), no son consideradas para la comparación porque no se ha establecido ninguna correspondencia entre ellos y la clase carnicero.

Se han intentado encontrar comparaciones sobre los conceptos referidos a las películas de Star Wars de los ejemplos anteriores, pero utilizando esta vez técnicas de alineamiento estructural con elementos de la saga del Rey Arturo. Se han encontrado correspondencias como *Princess Leia-Guinnevere* o *Obi Wan Kenobi-Merlin*, pero ninguno de los elementos destino contenían el atributo buscado con suficiente intensidad como para dar lugar a una comparación.

Ejemplos de Comparaciones Generadas

A continuación se muestran pequeños textos en los que se menciona una entidad y alguna de sus propiedades. Se incluyen ejemplos de propiedades tanto importantes para la clase como atípicas, en positivo y en negativo, para mostrar ejemplos de todos los casos posibles. Veamos qué comparaciones se introducen según lo expuesto anteriormente.

Partimos del siguiente texto:

> *There was a surgeon. He was educated. He was influential. The surgeon was careless. He was handsome.*

El texto resultante después de pasar por el proceso de enriquecimiento con comparaciones usando WordNet es el siguiente:

> *There was a surgeon. He was influential. He was educated as a hakham. The surgeon was careless as a scrawler. He was handsome.*

Y usando el alineamiento estructuras sobre los dominios de Sapper:

> *There was a surgeon. He was influential. He was educated as a military general. The surgeon was careless. He was handsome.*

En este ejemplo tenemos cuatro propiedades candidatas a formar una comparación: *educated*, *influential*, *careless* y *handsome*.

Ser educado es una propiedad importante para un cirujano (intensidad 0.85), mientras que ser influyente es una propiedad también típica de la clase, pero no tan importante (intensidad 0.6). Por eso la oración *he was educated* se transforma en una comparación. Usando WordNet se obtiene *he was educated as a hakham*, donde según WordNet *hakham* es *a Hebrew title of respect for a wise and highly educated man*. Usando los dominios de Sapper se obtiene *he was educated as a military general*, ya que la clase general militar tiene como atributo *educated* con intensidad 0.8. En este caso el referente de tipo cirujano tiene una de las propiedades consideradas muy importantes para su clase.

Al contrario, se supone que los cirujanos son cuidadosos (*careful* con intensidad 0.8), pero este cirujano en concreto no lo es. Por eso se resalta esta propiedad y en lugar de *the surgeon was careless* tenemos *the surgeon was careless as a scrawler*, donde *scrawler* es *a writer whose handwriting is careless and hard to read* en WordNet. Sin embargo, usando el alineamiento estructural no se ha creado ninguna comparación porque ninguna correspondencia con cirujano tiene como propiedad importante el ser cuidadoso.

Finalmente, la propiedad *handsome* no aparece en la clase cirujano, ni positiva ni negativamente. Por tanto no se resalta y no es considerada para realizar una comparación.

En el segundo ejemplo el texto inicial es el siguiente:

> *There was a clean and precise butcher.*

El texto resultante después del enriquecimiento con comparaciones usando WordNet es el siguiente:

> *There was a precise butcher. He was clean as a window cleaner.*

Y usando el alineamiento estructural sobre los dominios de Sapper:

There was a precise butcher. He was clean as a brain surgeon.

Tendremos dos propiedades candidatas para realizar una comparación: *clean* y *precise*.

En este caso, se supone que los carniceros no son limpios ni precisos. La propiedad de no ser limpios (*clean* con intensidad -0.8) tiene valores de intensidad altos, por lo que a partir de *he was clean* se obtiene *he was clean as a window_cleaner* usando WordNet y *he was clean as a brain surgeon* con el alineamiento estructural. El referente de tipo carnicero tiene una propiedad que se considera altamente incompatible con su clase, y por eso hay que destacarla.

En el tercer ejemplo el texto inicial es el siguiente:

There was a corpse. It was unhealthy. It was unpleasant.

El texto resultante después de su enriquicimiento usando comparaciones con WordNet es el siguiente:

There was a corpse. It was unhealthy. It was unpleasant as a pill.

Se supone que los cadáveres son sobre todo desagradables (*pleasant* con intensidad -0.9), y como el que tenemos en el discurso también lo es, tenemos *it was unpleasant as a pill.* Sin embargo, aunque los cadáveres también son insanos (*healthy* con intensidad -0.9), el sistema multiagente no consigue encontrar ninguna comparación adecuada usando WordNet.

No se han encontrado comparaciones posibles usando los dominios de Sapper, ni para *unhealthy* ni para *unpleasant.*

5.2.4. Analogías

En la sección 5.1.3 se mostró la base teórica del uso de analogías para enriquecer un discurso dado con dos propósitos diferentes dependiendo de la situación: o bien para explicar un concepto desconocido utilizando otro conocido por el usuario, o bien para resaltar entidades que presentan propiedades atípicas de su clase. En esta sección se tratarán las cuestiones específicas de implementación dentro de la arquitectura multiagente presentada.

Una de ellas es la necesidad de tener algún tipo de modelo de usuario para saber si un concepto que aparece en el discurso puede ser confuso para el lector porque no conoce a qué nos estamos refiriendo. La definición de modelos de usuario es una línea de investigación en sí misma, y no ha sido considerada en esta tesis nada más que como una posibilidad de trabajo futuro. Para los ejemplos de analogías que se presentan más adelante se han considerado dominios enteros como desconocidos para el usuario sin ningún tipo de formalización de este conocimiento.

Al igual que con las comparaciones, para saber si una propiedad destaca o es atípica de una clase se recurre al conocimiento general de los dominios de Sapper comentado en la sección 5.2.1, considerando propiedades de una clase los conceptos con los que está relacionado siguiendo la relación *attr*. Se considerará que una propiedad destaca cuando su intensidad es mayor que 0.8 o menor que -0,8.

Finalmente, la elección del concepto destino de la analogía a generar se realiza utilizando las capacidades metafóricas de los agentes de la arquitectura multiagente propuesta.

Conjunto de Propiedades como Fuente de la Analogía

Cuando se identifique un punto del discurso en el que se considera terminada la descripción a un concepto, será necesario estudiar si nos encontramos en el caso en el que el concepto es desconocido y se puede usar una analogía para clarificar su significado, en el caso en el que el concepto es atípico respecto a su clase, o en ninguno de los dos.

Para comprobarlo habrá que estudiar las propiedades que forman la descripción de una entidad en el discurso. Sin embargo, las propiedades que forman la descripción de la entidad no son sólo las que se mencionan explícitamente en el discurso. Si el concepto en cuestión es previamente conocido, muchas de sus propiedades conocidas son asumidas por el lector u oyente y puede ser tenidas en cuenta para posteriores razonamientos.

Consideraremos por tanto que las propiedades a tener en cuenta para una entidad que aparece mencionada en el discurso son tanto las mencionadas explícitamente, como las mencionadas implícitamente al dar el nombre de la clase independientemente de que ésta sea conocida o no por el lector, ya que lo que importa en este caso es que la clase sea conocida por el sistema para realizar los razonamientos que sean necesarios.

Al igual que ocurría en el caso de las comparaciones, se consideran PROPIEDADES EXPLÍCITAMENTE MENCIONADAS en el texto las que aparecen en una de las siguientes estructuras:

1. En un *mensaje copulativo* que da información sobre una propiedad de la entidad.

2. En una *referencia* a la entidad que contiene la propiedad en sí.

Como ya se discutió en la sección 5.1.2, en el caso 2 sólo se pueden considerar como propiedades que forman parte de una descripción los casos en los que la referencia es la primera que se realiza a la entidad.

Las PROPIEDADES IMPLÍCITAMENTE MENCIONADAS son las propiedades que pueden deducirse de la clase a la que pertenece a la entidad, y pueden por tanto considerarse como implícitamente mencionadas al mencionar la clase de la entidad. Cuando un elemento del discurso aparece mencionado

por primera vez, se realiza una consulta al conocimiento general para saber qué características del mismo se pueden dar por conocidas. Por ejemplo, cuando se menciona un gato en cualquier tipo de discurso, el lector u oyente realiza una serie de inferencias sobre lo que implica ser gato: que es un animal, que tiene cuatro patas, que seguramente es arisco, etc.

Sin embargo, en ocasiones puede ocurrir que las propiedades inferidas se contradigan con las mencionadas, y por tanto no se puedan aplicar a la instancia concreta de la clase que aparece en el discurso. Tal sería el caso de un gato que sólo tiene tres patas porque perdió una en un accidente, por ejemplo. Para obtener el conjunto de propiedades implícitas que no contradicen las explícitamente mencionadas será necesario recurrir al conocimiento general de una manera parecida a como se hacía en las comparaciones.

Antes de considerar una propiedad inferida del conocimiento general como implícitamente mencionada, es necesario comprobar si es incompatible con las explícitamente mencionadas en el discurso. Nos podemos encontrar con distintos casos:

- *La propiedad inferida coincide con una de las mencionadas de manera explícita.* Dependiendo de la intensidad de la propiedad inferida tendremos dos casos diferentes.
    - *Intensidad positiva.* La intensidad de la propiedad inferida es positiva. El significado considerado para la propiedad es el original, y por tanto ambas propiedades son compatibles. La propiedad inferida puede considerarse como implícitamente mencionada.
    - *Intensidad negativa.* La intensidad de la propiedad inferida es negativa. Esto significa que la propiedad que realmente corresponde a la clase es la contraria, y al coincidir con la mencionada, estamos en el caso en el que las dos propiedades son contrarias. Esto significa que la propiedad mencionada en el discurso es atípica para la clase, quedando marcada la entidad como una muestra atípica de la clase a la que pertenece. La propiedad inferida, al no corresponderse con la instancia concreta de la clase con la que se está trabajando, es desechada.
- *La propiedad inferida no coincide con ninguna de las mencionadas de manera explícita.* En este caso habrá que comprobar si además es incompatible con alguna de ellas. Para ello se buscan sus antónimos recurriendo de nuevo al agente WordNet para obtener todos los anónimos del adjetivo correspondiente a la propiedad.
    - *Alguno de los antónimos de la propiedad inferida coincide con una de las propiedades mencionadas explícitamente.* La propiedad mencionada en el discurso es atípica para la clase, y por tanto la entidad también lo es. La propiedad inferida es desechada.

- *Ninguno de los antónimos de la propiedad inferida coincide con las propiedades explícitamente mencionadas.* En este caso la propiedad inferida no está relacionada con nada de lo que se dice en el discurso, y por tanto se puede considerar que la entidad del discurso ostenta esta propiedad al no decirse lo contrario.

Destino de la Analogía utilizando Técnicas de Alineamiento Estructural

Como ya se comentó en la sección 5.1.3, para establecer una analogía es necesario construir un dominio inicial a partir del texto con el que se está trabajando, y buscar un dominio destino que proporcione analogías válidas según las consideraciones ya comentadas.

Para encontrar este dominio inicial, se enriquece la información completa que corresponde a la entidad para la que se está buscando la analogía, tanto con información propia del contexto como con información extraída del conocimiento general del sistema. En el trabajo aquí presentado, el dominio inicial para una entidad del discurso estará formado por:

- Las propiedades de la entidad explícitamente mencionadas en el discurso.
- Las propiedades de la entidad implícitamente mencionadas, es decir, inferidas de su clase y compatibles con las mencionadas en el discurso.
- Las relaciones que no son propiedades extraídas del conocimiento general sobre la clase. Éstas serán todas las relaciones de la clase distintas de *attr*.

Sin embargo, aunque un dominio formado según estas directrices es completo desde el punto de vista de la entidad, resultará bastante plano a la hora de buscar una analogía. Será por ello necesario enriquecerlo con la información disponible para cada una de las entidades con las que está relacionada la entidad inicial. Para cada una de ellas se repetirá el proceso mencionado, pero siempre teniendo cuidado de evitar ciclos, que pueden surgir si por ejemplo dos entidades ya examinadas están relacionadas entre sí en algún punto.

La prueba de las capacidades analógicas de la arquitectura ha sido realizada buscando los dominios destino entre los dominios de Sapper que fueron generados anteriormente para otras investigaciones en el campo de las metáforas y analogías (sección 2.6.1). Estos datos presentan dos importantes ventajas. Por un lado constituyen un conjunto de datos coherentes ya utilizados para la generación de analogías estructurales. Por otro lado, están generados independientemente del esfuerzo de investigación actual, luego son menos propensos a estar orientados hacia la obtención de resultados interesantes usando nuestra aproximación.

Para buscar una analogía entre una entidad del texto y los posibles dominios destino, se intentan buscar correspondencias entre el dominio inicial y cada uno de los posibles dominios finales utilizando el agente jMapper. Para muchos de los pares de dominios no se encontrará ninguna correspondencia entre conceptos, pero es posible que sí se encuentren dichas correspondencias para más de uno. De todos ellos, se escogerá el que cumpla los requisitos requeridos dependiendo del propósito de la analogía que se está generando, y entre ellos el de mayor intensidad en la correspondencia con el concepto inicial.

Ejemplos de Analogías

Se presentan a continuación ejemplos de analogías generadas a partir de discursos que presentaban las dos situaciones que hemos considerado como apropiadas para la inserción de una analogía.

Como ejemplos de analogías para conceptos desconocidos, se han usado del conjunto completo de datos empleados en el sistema Sapper dos dominios: la Guerra de las Galaxias y la saga del Rey Arturo. El primero ha sido escogido como el dominio desconocido sobre el que se pretende escribir un pequeño texto. Para facilitar su legibilidad, ya que se asume que el lector no ha oído hablar nunca de la Guerra de las Galaxias, se ha utilizado el segundo dominio para extraer las analogías para referirse a los conceptos del primero que faciliten su comprensión.

Fue necesaria una conversión de los datos de los dominios a la representación utilizada por los agentes, y en este proceso parte de la información inicial tuvo que ser desechada. Algunos ejemplos son los pesos de las relaciones, que en este tipo de analogías no son utilizados, y algunos tipos de conceptos narrativos. Por el momento nos centraremos sólo en las propiedades de los personajes y objetos, y sus relaciones dentro de la historia (por ejemplo, `have`, `friend_of`, `teach`, `loves`, etc.).

El primer paso es enviar el dominio inicial extraído del texto sobre la Guerra de las Galaxias completo al agente jMapper para que se realicen las correspondencias posibles entre los dos dominios. A partir de ahí, se podrá preguntar al agente qué analogías se han obtenido para los distintos conceptos. El grafo enriquecido de relaciones obtenido del contexto inicial es enfrentado con las relaciones de la saga del Rey Arturo. Un fragmento de ambos dominios se puede ver en la Tabla 5.2.

Algunas de las asociaciones obtenidas como parte de la correspondencia están únicamente basadas en relaciones muy generales como el género o la relación *isa*. Estas analogías se consideran poco interesantes y son directamente descartadas por el generador. La correspondencia obtenida para este ejemplo se muestra en la Tabla 5.3. Para cada asociación se puede ver la lista de relaciones que han producido la correspondencia.

Star Wars	**Algunas Relaciones**
storm_trooper	[warrior,man,person,evil]
light_saber	[hand_held,narrow,long,weapon]
princess_leia	[beautiful,young,royal_personage,brunette,...]

Rey Arturo	**Algunas Relaciones**
knight	[warrior,man,person,medieval]
excalibur	[hand_held,narrow,long,weapon,magical,steely,...]
guinnevere	[beautiful,young,royal_personage,queen,blonde,...]

Tabla 5.2: Ejemplos de relaciones para algunos conceptos de los dos dominios

Correspondencias	**Relaciones**		**Intensidad**
obi_wan_kenobi ↔ *merlin*	good wise person man	powerful old magician	0.52
storm_trooper ↔ *knight*	warrior person	man	0.66
light_saber ↔ *excalibur*	hand_held long	narrow weapon	0.73
han_solo ↔ *lancelot*	skilful handsome man	brave young person	0.43
princess_leia ↔ *guinnevere*	beautiful royal_personage woman	young person	0.63

Tabla 5.3: Correspondencia resultante entre La Guerra de las Galaxias y la saga del Rey Arturo

Según el algoritmo ya explicado, y usando las propiedades y correspondencias de las Tablas 5.2 y 5.3, se obtiene el siguiente texto en el dominio de la Guerra de las Galaxias, que se considera desconocido, usando información sobre el dominio de la saga del Rey Arturo, que se considera conocido:

> *Luke Skywalker was a young jedi knight. He had a light saber. The light saber was powerful. The light saber was the Excalibur of Luke Skywalker.*
>
> *Han Solo loved Princess Leia. He was the Lancelot of Luke Skywalker. She was the Guinnevere of Han Solo.*
>
> *Obi Wan Kenobi taught Luke Skywalker. Obi Wan Kenobi was the Merlin of the Jedi Knights.*

A continuación se muestran ejemplos de analogías utilizadas para enfatizar propiedades atípicas de un elemento del discurso respecto a su clase. Para cada texto inicial se muestra el texto final resultante después de haber establecido las analogías.

Imaginemos el siguiente texto inicial:

> *He proposed her a contract. It was illegal. It was dangerous.*

Después del enriquecimiento con analogías, el texto generado es el siguiente:

> *He proposed her a contract. It was illegal. It was dangerous. <u>The contract was like a burglary.</u>*

En los dominios con los que estamos trabajando un contrato se considera como algo legal (intensidad 0.5), y por tanto para este texto el contrato que se menciona se considera atípico de su clase. Recurriendo al agente jMapper, se obtiene como analogía que el contrato en cuestión es como un robo: ilegal y peligroso (ambos con intensidad 0.6).

Otro posible texto de entrada sería:

> *The policeman was unethical and dishonest.*

Después del enriquecimiento con analogías se obtiene el siguiente texto:

> *The policeman was unethical and dishonest. <u>He was like a criminal.</u>*

En este caso los policías se consideran en el conocimiento general como éticos y honestos (ambos con intensidad 0.5) entre otras cosas. A la hora de buscar una analogía para estas características atípicas del policía del discurso, surge el concepto criminal, que en el conocimiento general tiene como atributos *unethical* y *dishonest* con intensidades de 0.5 para ambos.

Resumen y Conclusiones

En este capítulo se ha abordado el uso de figuras retóricas basadas en similitudes entre dominios como son la comparación y la analogía a la hora de cubrir el objetivo de describir entidades de la generación de expresiones de referencia. Para ello se ha estudiado en qué casos puede resultar útil resaltar una entidad o describirla si es desconocida, y cómo se pueden conseguir las figuras necesarias recurriendo a algoritmos y conocimiento previamente existentes.

Como marco para la implementación de estas ideas se ha presentado una arquitectura multiagente que permite conectar todos los módulos que entran en funcionamiento de manera eficiente y sin necesidad de que cada uno conozca cómo funcionan los demás.

Se han mostrado además ejemplos de comparaciones y analogías generadas para cada uno de los casos considerados.

Capítulo 6

Discusión y Propuestas

"There is a difference between knowing the path and walking the path."
— The Matrix (1999)

El trabajo realizado en esta tesis va más allá de la tarea de generación de expresiones de referencia. En el proceso de generar referencias con función descriptiva o distintiva intervienen factores que dependen de tareas como la lexicalización (a la hora de lexicalizar las referencias), la planificación de oraciones (cuando se modifican las oraciones para insertar una comparación o analogía), la determinación de contenido (donde se deciden qué dominios se usarán para buscar las analogías), etc. Aunque gran parte de estas cuestiones no se encontraban dentro del alcance de este trabajo, resulta adecuado mencionarlas y discutirlas de cara a trabajo futuro.

Se discuten a continuación las diferentes soluciones proporcionadas para la generación y evaluación de expresiones de referencia básicas (sección 6.1) y las apoyadas en el uso de figuras retóricas (sección 6.2). Se realizará además una breve discusión de las repercusiones que puede tener la inclusión de este tipo de figuras en el proceso completo de generación automática de texto (sección 6.3). Finalmente, se estudia el problema de la relación entre las referencias tratadas en este trabajo y el proceso completo de generación de lenguaje natural (sección 6.4).

6.1. Generación y Evaluación de Expresiones de Referencia Básicas

En el capítulo 4 se aplicaron y evaluaron diversos algoritmos a la resolución de referencias básicas separando el problema completo en tres tareas diferentes: elección del nivel de abstracción, selección de atributos y lexicali-

zación. En esta sección se discuten cuestiones relacionadas con estos trabajos y posibles cuestiones pendientes que se podrían considerar en el futuro.

6.1.1. Elección del Nivel de Abstracción de las Referencias

El trabajo presentado sobre el uso de ontologías para determinar el nivel de abstracción al que desarrollar un discurso se centra en situaciones estáticas donde no hay disponible un contexto que defina qué entidades han sido mencionadas previamente. No obstante, la generación de expresiones de referencia suele ser parte de un proceso más complejo de generación de lenguaje natural como serían la generación de diálogos u otro tipo de discursos. En estos casos el conjunto de contraste podría restringirse empleando distintos tipos de heurísticas.

Una de estas posibles heurísticas sería el uso de grados de prominencia o *salience* empleada por Krahmer y Theune (sección 2.2.5) donde se asigna una determinada prominencia a cada entidad del dominio. Esto les permite crear un conjunto dinámico de contraste denominado *conjunto de contexto* que se compone de aquellas entidades en el mundo que tienen una prominencia mayor o igual que el referente objetivo.

Otro posible enfoque sería restringir el algoritmo para que sólo trabaje con una parte de la ontología. Por ejemplo, podríamos considerar un sistema de generación de narrativa que emplea una ontología general acerca del mundo. Si las acciones narradas suceden en una ubicación específica, el conjunto de contraste podría restringirse a aquellas instancias de la ontología que pertenecen a dicha ubicación. Alternativamente, si el texto que estamos generando consistiese en una descripción de los vinos que hay en una mesa, el conjunto de contraste solo consideraría la sub-ontología sobre vinos en lugar de la ontología completa sobre el mundo. Estos posibles enfoques serán explorados como trabajo futuro.

Por otro lado, los atributos considerados en la tarea de selección de atributos deben ser tratados con cautela. En la ontología las propiedades y las relaciones se representan de la misma manera. En este trabajo hemos considerado sólo propiedades (`body`, `color`, etc.), pero también sería posible usar las relaciones (`maker`, `madeWithGrape`, etc.) disponibles. Para la selección de atributos las propiedades y relaciones pueden usarse del mismo modo, pero en las siguientes etapas de la generación de texto (concretamente la tarea de elección sintáctica) las relaciones se deberían tratar por separado.

Aplicando el mismo mecanismo que para el tipo de los referentes, el algoritmo presentado también podría emplearse para encontrar el valor de nivel básico y el valor más apropiado para los atributos de los elementos del mundo. Aún así, para afrontar esta tarea se necesitaría más información acerca de los distintos valores de los atributos y su organización en forma de taxonomía.

Ya que se ha usado WordNet para otras tareas relacionadas con la generación de expresiones de referencia, se podría probar si las ideas propuestas usando ontologías se podrían replicar usando la jerarquía semántica de WordNet. En la Figura 6.1 se muestra una parte de la taxonomía que WordNet contiene sobre vinos. Aunque es semejante a la ontología con la que hemos trabajado en algunos aspectos, el hecho de que los niveles de jerarquía no estén bien definidos podría dar problemas a la hora de generar las referencias. Un ejemplo de esto son los vinos concretos que están a la misma altura que los vinos tintos en general.

```
wine
|
|=> vintage
|=> red wine
|          |=> Medoc
|          |=> Pinot noir
|          |=> claret, red Bordeaux
|          |     |=> Saint Emilion
|          |=> Rioja
|=> white wine
|          |=> Canary wine
|          |=> Rhine wine
|          |     |=> Riesling
|          |=> Soave
|=> altar wine, sacramental wine
|=> sparkling wine
|          |=> champagne
|=> Burgundy
|          |=> Chablis
|          |=> Montrachet
|=> Bordeaux
|          |=> Medoc
|          |=> claret, red Bordeaux
|          |     |=> Saint Emilion
|
```

Figura 6.1: Parte de la jerarquía sobre vinos de WordNet

6.1.2. Soluciones Evolutivas para la Búsqueda de Conjuntos de Atributos

De cara al funcionamiento del módulo evolutivo para selección de atributos sería interesante comparar los resultados obtenidos por nuestra solución con otras aproximaciones que se pueden encontrar en la literatura, como por ejemplo las de Reiter y Dale o Krahmer y Theune (sección 2.2.5). De estas comparaciones podrían surgir nuevas características a tener en cuenta para la implementación de los operadores genéticos y las funciones de evaluación, por ejemplo.

Una evaluación como la propuesta por Callaway y Lester (2001a) podría resultar muy útil para estimar la bondad de las referencias generadas. Los autores describen la evaluación del sistema STORYBOOK, un sistema para la generación de prosa narrativa que produce textos originales para cuentos en el dominio de Caperucita Roja. Ellos pretenden evaluar múltiples versiones de una misma historia asegurándose que el contenido es el mismo en todas ellas. Se producen cinco versiones de dos historias separadas y un conjunto de veinte estudiantes de inglés los comparan. Finalmente, son analizados con el test estadístico ANOVA. En nuestro caso se podrían generar diferentes referencias finales con distintas funciones de aptitud o configuraciones de los parámetros del algoritmo. Usando evaluadores humanos se podrían comparar para ver cuál de todas las posibilidades es considerada más apropiada.

Otra posibilidad a la hora de realizar la evaluación de la población al final de cada generación es el uso de redes neuronales para reproducir la evaluación humana de la calidad de los textos resultado. Las redes neuronales han sido utilizados a menudo como evaluadores en algoritmos evolutivos. Algunos ejemplos en el campo de la generación automática de poesía pueden encontrarse en Levy (2001) y Manurung (2003). Sería posible utilizar el corpus para entrenar una red neuronal que midiera cómo de acertada es la referencia generada respecto al resto de referencias que se encuentran en el corpus.

6.1.3. Elección léxica con WordNet

El problema de la ambigüedad en el uso de WordNet es importante. Los resultados muestran que el problema es solucionado parcialmente con la elección de la palabra con el mayor *tag count*, pero en ocasiones esto no es suficiente. No siempre el significado más común en un corpus genérico será el más común en un dominio concreto. Un ejemplo claro es de la palabra *ball* que según WordNet usando el mayor *tag count* es *un objeto redondo que se usa en diversos deportes*, pero que en un caso como el cuento de la Cenicienta es *un baile formal*. Una opción a explorar sería tener en cuenta el dominio en el que se están generando las referencias. Se usaría un corpus de textos en el dominio anotado con la aparición de sustantivos. Cuando fuera necesario decidir entre diferentes *synsets*, el que tuviera mayor número de apariciones en el corpus sería el escogido.

En términos generales, los resultados se pueden mejorar usando WordNet para realizar la elección léxica no sólo para los sustantivos en singular, sino también para los plurales. Para ello sería necesaria la implementación de un nuevo módulo para llevar a cabo las derivaciones morfológicas requeridas. WordNet contiene un procesador morfológico llamado Morphy (Beckwith et al., 1993) que es aplicado a la cadena buscada para generar una forma de la misma que esté presente en WordNet, usando durante el proceso un

conjunto de funciones morfológicas previamente definidas. El uso de Morphy podría ser una solución a nuestro problema.

En la implementación actual sólo se ha considerado WordNet para realizar la elección léxica a nivel de sustantivos, pero podría ser utilizado también a nivel de adjetivos. Sin embargo, relaciones semánticas diferentes se usan para organizar los adjetivos en WordNet (Fellbaum et al., 1990). Estudiar sus ventajas potenciales para la tarea de lexicalización será necesario.

6.1.4. Lexicalización de Referencias basada en Razonamiento Basado en Casos

Como hemos visto, entrenar el módulo CBR con un corpus heterogéneo como el TUNA no resulta ser una buena idea para reproducir el estilo de las referencias. Se podría ser separar el corpus en distintos conjuntos homogéneos de referencias según un modelo de la percepción del usuario. Por ejemplo, un profesional del mundo de la moda describiría a una persona en términos de las ropas que lleva puestas, mientras que un doctor podría fijarse más en su complexión física. Para ello se entrenaría el módulo usando un conjunto de referencias u otro dependiendo del estilo que se desee utilizar.

Otra mejora del funcionamiento del módulo sería la implementación del mecanismo de *lazy spreading activation* (Lenz y Burkhard, 1996a) en la Case Retrieval Net. En lugar de propagar la activación a todos los nodos de entidad, y de ahí a todos los nodos de caso, la propagación tiene lugar progresivamente desde los nodos más similares a los menos. Una vez que suficientes nodos de caso han sido activados en respuesta a la petición, la propagación termina. En ocasiones en los que la base de casos tiene gran tamaño este mecanismo puede suponer una mejora de eficiencia significativa.

6.1.5. Evaluación de las Referencias Generadas

A la vista de los resultados de evaluación obtenidos por los sistemas participantes en las tareas de evaluación competitiva para generación de expresiones de referencia, se podría deducir que no son los sistemas que tienen mejor evaluación intrínseca sobre el corpus lo que son más apropiados desde el punto de vista de la evaluación extrínseca con evaluadores humanos.

Esto se debe probablemente a que es difícil crear un corpus que resulte representativo de cómo los seres humanos generan referencias en general. A pesar de haber sido realizado usando un alto número de evaluadores, no es posible que se cubran todas las posibilidades. Además, el hecho de que las referencias del corpus provengan de tantas soluciones diferentes puede provocar que las métricas de evaluación sobre el corpus no sean uniformes dado que el mismo corpus no lo es.

6.2. Generación y Evaluación de Expresiones de Referencia Usando Figuras Retóricas

En el capítulo 5 de este trabajo se estudia cómo abordar la generación de expresiones de referencia utilizando información de otros dominios para generar comparaciones y analogías. En esta sección se discuten cuestiones relacionadas con lo expuesto en ese capítulo y cuestiones pendientes que se considerarán de cara a trabajo futuro.

6.2.1. Flujo de Control para la Generación de Figuras Retóricas

La arquitectura OAA implementada para el tratamiento de las figuras retóricas permite que cada agente se ejecute en una máquina diferente, manejando las comunicaciones entre agentes sin ningún esfuerzo adicional. Esto puede conducir a una configuración donde varios agentes mapeadores (uno por cada dominio candidato) pueden ser ejecutados en máquinas cliente para computar las correspondencias. La arquitectura multiagente proporciona también de esta manera distintas estrategias para procesar los resultados. Las opciones van desde seleccionar el dominio procesado por el agente mapeador que contesta primero, hasta establecer negociaciones entre los diferentes agentes para seleccionar la mejor correspondencia disponible.

6.2.2. Conocimiento General Necesario para la Generación de Figuras Retóricas

El conocimiento general necesario para la generación de las figuras retóricas ha sido obtenido de los dominios generados por Tony Veale para su sistema Sapper (sección 2.6.1). Sin embargo, se podrían usar otras fuentes de conocimiento para realizar esta función.

Una herramienta que parece apropiada para esta tarea son las ontologías, que almacenan información conceptual sobre conceptos e instancias y las relaciones que hay entre ellos. Al igual que usábamos los dominios de Sapper, sería posible extraer la información relacionada con un determinado concepto usando una ontología y explorando las relaciones de ese concepto con los demás.

Ontologías de conocimiento sobre sentido común como Cyc (Lenat, 1995), Open Mind Common Sense (OMCS) (Singh, 2002) y Thought Treasure (Mueller, 1998) podrían ser usadas para ello. Sin embargo, en una ontología seguramente no sería posible encontrar intensidades para las relaciones como en Sapper. Algún tipo de mecanismo será por tanto necesario para decidir si una propiedad es importante para la definición de un concepto. Sin esta información, las comparaciones y analogías sólo podrían generarse sin tener en cuenta si las propiedades de una entidades son remarcables o no.

6.2.3. Generación de Comparaciones

La generación de comparaciones usando las glosas de los conceptos de WordNet presenta algunos problemas. El agente LWWN se encarga de buscar conceptos que contengan en su glosa el adjetivo para el que queremos hacer la comparación, siempre comprobando que esos conceptos tienen como hiperónimos alguno de los hiperónimos del concepto que queremos comparar. Esto nos garantiza por un lado que los dos conceptos a comparar sean de tipos parecidos, o al menos relacionados. Por otro lado, se parte de la suposición de que si un concepto tiene en su glosa un determinado adjetivo, este adjetivo debe ser importante en la definición del concepto.

Sin embargo, en ocasiones se obtienen comparaciones extrañas como *it was unpleasant as a pill*. En WordNet la glosa que corresponde a *pill* es *something unpleasant or offensive that must be tolerated or endured; "his competitor's success was a bitter pill to take"*, pero al realizar la comparación es difícil entender que nos estamos refiriendo a esta definición de *pill*.

Una solución para este tipo de problemas sería realizar un tratamiento un poco más cuidadoso de las glosas. Donde ahora simplemente se comprueba si en la glosa se encuentra una determinada palabra, se podría aplicar un análisis sintáctico que nos permitiera saber qué función cumple el adjetivo buscado en la oración. Un analizador de dependencias como MINIPAR (Lin, 1998) podría servirnos para este propósito.

Cuando se está utilizando mapeamiento estructural entre dominios para generar las comparaciones, se evitan generalmente estas confusiones de significado. Los conceptos que se usan como destinos de las comparaciones son siempre similares o análogos a la entidad con la que están siendo comparados, por lo que es más difícil encontrar este tipo de comparaciones confusas. Sin embargo, esto también provoca que no sea posible la aparición de comparaciones más creativas como serían *the butcher is clean as a operating theater* o *the surgeon is precise as a scalpel*.

Desde el punto vista de la relación de nuestro trabajo con el de Milosavljevic (sección 2.6.2), mientras que ella estudia cómo las comparaciones son necesarias cuando una entidad ha de ser descrita, utilizando los conocimientos previos del usuario para construir la nueva información sobre similitudes y diferencias con otros conceptos, nuestro trabajo se centra en el uso de la comparación fuera de proceso de descripción. Nuestro objetivo no es proporcionar nueva información al oyente, sino recalcar algunas partes del discurso que pueden ser consideradas importantes.

6.2.4. Generación de Analogías

En la descripción de la arquitectura multiagente presentada en la sección 5.2 estamos considerando una aproximación de alto nivel para la identificación e introducción de una analogía entre un concepto simple de un dominio

dado y un concepto en el dominio destino. En los ejemplos presentados en la sección 5.2.4 se puede ver que dados dos dominios análogos estructuralmente, el proceso descrito devuelve una serie de asociaciones entre conceptos de los dos dominios. Desde un punto de vista estilístico está claro que un texto en que todas las posibles asociaciones entre los dominios son incluidas estará sobrecargado de analogías. Esto sugiere que las heurísticas a considerar para la selección y localización de las analogías deben ser revisadas en busca de resultados más naturales. Varias cuestiones deberán tenerse en cuenta.

Para empezar será necesario proporcionar un criterio estricto para filtrar el conjunto de correspondencias devueltas por el módulo que busca las analogías. Cuando se genera una correspondencia entre dos conceptos de distintos dominios, un registro de qué relaciones han llevado a esta correspondencia es almacenado. Las heurísticas actuales están diseñadas para seleccionar sólo una analogía para su inserción en el texto, y están orientadas a asegurar que ninguna analogía es introducida a menos que esté justificada por un número mínimo de relaciones. Cuando son varias asociaciones que han llevado a cada analogía, esta restricción no es suficiente para reducir significativamente el número de candidatos.

Otro aspecto importante es que el conjunto de todas las analogías encontradas como resultado de la correspondencia de dos dominios estructuralmente análogos puede ser a su vez tan rico que constituya casi un texto paralelo en sí mismo. En estos casos, en lugar de insertar oraciones individuales describiendo cada una de las posibles asociaciones, puede merecer la pena introducir un sub-texto completo describiendo la vista del dominio inicial que corresponde al dominio destino. En el ejemplo presentado en la sección 5.2.4 podríamos acabar obteniendo un texto como el siguiente:

> *King Arthur had his Excalibur. He was in love with Guinnevere. Merlin taught him.*
>
> *Luke Skywalker was the King Arthur of the Jedi Knights. He had a light saber. The light saber was powerful. ...*

O bien insertar los mensajes relacionados con la analogía como grupos de asociaciones más que como mensajes individuales. Por ejemplo, está claro que los mensajes que describen al Rey Arturo y Excalibur como análogos a Luke Skywalker y su sable láser pueden ser presentados de una manera más intuitiva si se agrupan:

> *Luke Skywalker had a light saber. The light saber was powerful. He was the King Arthur of the Jedi Knights and the light saber was his Excalibur. ...*

Esto correspondería a considerar el conjunto de asociaciones devueltas por la correspondencia como una adición al contenido conceptual que se

va a pasar a texto, susceptible de las consiguientes etapas de Determinación de Contenido, donde las asociaciones menos interesantes pueden ser descartadas, y Planificación del Discurso, donde las asociaciones restantes son reagrupadas, posiblemente teniendo en cuenta las relaciones del dominio destino que son las responsables de unir los conceptos que aparecen en las distintas asociaciones.

Sin embargo, este tipo de arreglo podría no ser tan adecuado si resulta en dos textos paralelos completamente diferentes. Algún tipo de clustering intermedio deberá tener lugar durante la Planificación del Discurso, donde los elementos son agrupados por alguna relación (como por ejemplo Luke y su sable láser y/o Arturo y Excalibur) antes de de establecer la analogía. Esto ha ocurrido por coincidencia en el texto de ejemplo para Han Solo/Lancelot y Princesa Leia/Guinnevere, y también para Merlín/Obi Wan Kenobi, dado que comparten una relación recíproca. La heurística deberá ser redefinida para asegurar que estos efectos son resultado de decisiones explícitas y no fruto de la casualidad.

6.2.5. Evaluación de las Referencias Generadas

En el caso de las referencias generadas usando figuras retóricas basadas en similitudes entre dominios no se han encontrado en la literatura métricas de evaluación apropiadas. Como trabajo futuro se podría abordar el problema de definir métricas tanto intrínsecas como extrínsecas para medir este tipo de construcciones.

Una posibilidad sería crear un corpus de comparaciones o analogías aplicadas a la generación de expresiones de referencia sobre el que fuera posible comparar las generadas automáticamente. Sin embargo, sin contar lo dificultoso que podría resultar, se ha visto en el caso de las referencias básicas que un corpus no siempre es el fiel reflejo de cómo los seres humanos realizan este tipo de tareas.

Otra posibilidad más apropiada podría ser realizar algún tipo de evaluación extrínseca sobre humanos en la que se midieran cómo de apropiadas consideran las distintas referencias generadas dependiendo del conocimiento que tienen sobre los dominios.

6.3. Interacción entre las Comparaciones y Analogías y la Generación Automática de Texto

En el capítulo 5 hemos estudiado la generación de comparaciones y analogías como un fenómeno aislado. A continuación se realizará una breve discusión de cómo la inserción de comparaciones y analogías tendrá repercusión en el resto del proceso de generación, y se propone una implementación para el sistema de generación de lenguaje natural TAP.

6.3.1. Relación de las Comparaciones con el Resto del Proceso de Generación

La inserción de comparaciones interacciona con el resto del proceso de generación en varios puntos.

Desde el punto de vista de la Planificación del Discurso, la inserción de comparaciones enriquece la información inicial seleccionada para ser expresada en el discurso. En el caso en el que un mensaje copulativo que describe a una entidad es sustituido por una comparación, se está añadiendo información extra con la que no se contaba en la fase de Planificación del Discurso anterior. En el caso en que la primera mención de una entidad va acompañada por una propiedad y da lugar a una comparación, el nuevo mensaje insertado con la comparación modifica de alguna manera el discurso planificado previamente.

Las interacciones con la Generación de Expresiones de Referencia son todavía más complejas. Como se ha mencionado anteriormente, uno de los casos en los que se considera que puede resultar útil una comparación es cuando la primera mención a una entidad aparece acompañada por alguna de las propiedades que la describen. Aquí es importante que la mención a la entidad sea la primera, y no una de las siguientes. La razón para esto es que, en la mayor parte de las ocasiones, las siguientes menciones a un concepto aparecerán acompañadas de propiedades cuando éstas sean necesarias para distinguir a la entidad de otras con las que se pudiera confundir. En estos casos no tendría sentido introducir una comparación relacionada con estas propiedades.

Otra interacción interesante que surge en el proceso de generación de comparaciones es la decisión de en qué punto se decide cómo se va a describir una entidad, ya sea utilizando mensajes copulativos o acompañando las primeras menciones de una entidad con propiedades (ya hemos visto que las posteriores menciones con propiedades no se consideran descripciones). Si las descripciones de entidades se deciden en la fase de Planificación del Discurso, en la Planificación de Oraciones se podrá encontrar toda la información necesaria para la generación de comparaciones según lo que hemos expuesto anteriormente. Sin embargo, si es la etapa de Generación de Expresiones de Referencia la que encarga de añadir propiedades a las entidades para describirlas, y no sólo para distinguirlas, sería necesaria una nueva pasada por la Planificación de Oraciones completa para comprobar si alguna de las propiedades añadidas podría dar lugar a una comparación.

6.3.2. Relación de las Analogías con el Resto del Proceso de Generación

La tarea de generar textos donde las analogías se usan adecuadamente requiere resolver diferentes problemas que deben ser tratados en módulos se-

parados. Por un lado está la tarea inicial de identificar el dominio adicional en el que se buscarán las analogías y el concepto correspondiente del dominio inicial. Por otro lado está la tarea de insertar las estructuras lingüísticas apropiadas para la analogía en el texto original, incluyendo tanto la construcción de las mismas como la elección del lugar en que serán insertadas en el texto.

Con respecto a la estructura clásica de *pipeline* de un generador de lenguaje natural simple, la introducción de una analogía conceptual debería tener lugar en la fase de Determinación de Contenido. El proceso de identificación del dominio objetivo y la correspondencia tendrán lugar aquí. Dado que la analogía debe contribuir a un acto comunicativo, parece razonable pensar que para que un dominio destino sea apropiado en una analogía, este dominio debe ser suficientemente bien conocido para los lectores del texto de forma que no requiera ninguna explicación adicional. Esto obliga a trabajar en contextos de interacción particulares, donde cierta información debe estar disponible sobre el lector. Una solución simple a esta cuestión sería considerar como dominios candidatos sólo aquellos que han sido descritos por el sistema en el pasado. Esto se basa en la consideración de que las salidas del sistema han sido leídas siempre por el mismo lector, y requiere el almacenamiento de un registro de las salidas del sistema y de los conceptos involucrados en cada ocasión.

Sería posible que el sistema computara correspondencias entre la entrada actual y los dominios más recientes para los que ha proporcionado salidas, y entre ellos seleccionar el dominio con la mejor correspondencia obtenida. Sin embargo, esta solución podría ser computacionalmente ineficiente.

El problema de decidir dónde tendrá lugar la realización lingüística de la analogía con respecto al resto del texto debería ser resuelto en la fase de Planificación del Discurso. Sin embargo, existen ciertas peculiaridades en las analogías que podrían requerir un cierto grado de interacción entre esta fase y la anterior de Determinación de Contenido. Una cuestión importante a tener en cuenta es que si cualquiera de las relaciones de un concepto ha sido considera necesaria para entender la analogía, la realización lingüística de la analogía no debe tener lugar en el texto antes de que estas relaciones hayan sido mencionadas explícitamente. Una aproximación alternativa sería presentar primero la analogía y luego proporcionar una explicación de las relaciones que han dado lugar a la misma.

6.3.3. Propuesta de Implementación sobre un Generador de Lenguaje Natural

Considerando la arquitectura multiagente presentada para la generación de figuras retóricas basadas en similitudes y su inserción en texto, cualquier sistema que quiera utilizar la funcionalidad implementada deberá tener presencia en la sociedad de agentes implementada. En el caso que nos ocupa se

podría implementar un agente TextGenerator, que se encarga del proceso de generación de texto, y podría ser simplemente un *wrapper* para el módulo original de generación de lenguaje natural escogido. Éste es el agente que pone en marcha el flujo de control del proceso completo. El TextGenerator sigue el flujo de control del módulo GLN, interactuando con los agentes de la arquitectura cuando necesita información sobre los diferentes conceptos del discurso.

El proceso completo de generación de texto para las entidades de un discurso comienza después de la fase de Planificación del Discurso. En este punto ya se han tomado las decisiones sobre qué información de entre toda la disponible se desea comunicar en el texto final, y cómo se organizará esta información en forma de mensajes. Aquí ya se habrán decidido por tanto qué entidades aparecerán mencionadas en el texto y dónde, así como las propiedades que se utilizarán para sus descripciones.

A partir de este discurso inicial organizado como un conjunto de mensajes conceptualmente representados, se pueden estudiar las entidades mencionadas y sus descripciones o propiedades mencionadas para decidir qué tipo de enriquecimiento puede ser llevado a cabo.

Para las comparaciones, cada aparición de una propiedad de un concepto en un mensaje copulativo o en la primera aparición de éste en el texto es estudiada para comprobar si puede ser útil enfatizarla como ya se ha visto. En caso de que así sea, se generará el mensaje con la comparación correspondiente.

Para las analogías, el conjunto de propiedades mencionadas que forman la descripción de un concepto es localizado, de manera que sea posible tomar una decisión para la inserción de las mismas. Dependiendo de si el conjunto de propiedades que ostenta una entidad la hace atípica para su clase, o si a través de un modelo de usuario se sabe que el tipo de esta entidad es desconocido, se recurrirá a las propiedades que forman la descripción para generar la analogía correspondiente.

Una vez enriquecido el discurso con analogías y comparaciones, el tratamiento de las entidades del discurso pasa por la tarea de la generación de expresiones de referencia. Esta tarea se lleva a cabo apoyándose en la información conceptual existente sobre las entidades, así como sobre el contexto de lo que se ha dicho en el discurso. Sin embargo, tras el enriquecimiento han aparecido nuevas entidades en el discurso que no aparecen en la información conceptual inicial de la que surge el texto. Esto puede ser un problema, ya que faltará información básica para llevar a cabo la Generación de Expresiones de Referencia, como por ejemplo pueden ser las propiedades de dichas entidades o información sobre su tipo para encontrar las entidades con las que se pueden confundir.

En realidad, estas nuevas entidades del discurso, que se corresponderán con los conceptos destino de las comparaciones y analogías, no deberían entrar en el proceso de Generación de Expresiones de Referencia. La razón es que no pertenecen al contexto completo del texto, y por tanto no deben ser consideradas para ser realizadas como un pronombre, una frase nominal o usando un nombre propio. Más bien deberán aparecer siempre mencionadas por su nombre, ya sea propio o no, directamente, y sin ser consideradas en el proceso de Generación de Expresiones de Referencia. Para que esto sea posible, en el momento de su inserción en el texto deberán ser insertadas directamente como referencias ya generadas, no como referentes para los que se debe encontrar una referencia posteriormente.

6.4. Referencias Distintivas y Descriptivas y su Relación con el Resto del Proceso GLN

Aunque en este trabajo se han tratado las funciones distintiva y descriptiva de las referencias por separado, es importante discutir cómo ambas se entrelazan dentro del proceso de generación de texto. La función descriptiva de las expresiones de referencia está estrechamente ligada con otro de los problemas de la Generación de Lenguaje Natural: la descripción de entidades y su posterior influencia en la generación de referencias a las mismas. En esta sección se estudia la relación entre la Determinación de Contenido y la Generación de Expresiones de Referencia al proporcionar información descriptiva sobre entidades y referirse posteriormente a ellas buscando distinguirlas. Nos centraremos en discursos generados automáticamente, donde la única información disponible para el lector u oyente es la contenida en el propio texto. Para ello será necesario revisar la arquitectura clásica de *pipeline*, estudiando las interacciones que pueden ser necesarias entre los distintos módulos involucrados.

6.4.1. Interacciones en el Flujo de Control del Proceso de Generación

Dos tareas dentro de la Generación de Lenguaje Natural están involucradas en el proceso de generación de información descriptiva sobre entidades en un discurso y su posterior relación con el resto del texto: la Determinación de Contenido y la Generación de Expresiones de Referencia. La relación entre ambas a la hora de referirse a las entidades en un discurso es importante para transmitir la información deseada y evitar ambigüedades. Esta relación se basa en que la introducción de información con intención descriptiva ocurre en función de criterios específicos del dominio (que se suelen tratar en la Determinación de Contenido), pero la introducción de información con

intención distintiva ocurre en función del contexto (que se suele tratar en la Generación de Expresiones de Referencia).

La Determinación de Contenido (DC) se encarga generalmente de seleccionar la información descriptiva que se pretende comunicar sobre la entidad. Dependiendo del lugar que ocupa dicha descripción dentro del discurso, y de la importancia de la entidad en el mismo, la información descriptiva a añadir podrá ser más o menos extensa. Según el tipo de discurso habrá entidades que sobresalen sobre el resto o son más importantes que las demás. En un cuento, por ejemplo, los personajes protagonistas o el lugar donde se desarrolla la acción serán presentados con descripciones mucho más detalladas que los personajes secundarios o los objetos menos importantes. La DC deberá determinar qué elementos del discurso sobresalen sobre los demás y deberá seleccionar información acorde a este hecho a la hora de describirlos.

La Generación de Expresiones de Referencia (GER) se encarga de generar referencias a entidades que permitan distinguirlas del resto de elementos del discurso, al mismo tiempo que proporciona cualquier información extra que el lector pueda encontrar útil más adelante. Para ello depende no sólo de la información proporcionada por la DC, sino también del contexto del texto en el momento de realizar la referencia. Este contexto está determinado, entre otras cosas, por cómo ha sido descrita previamente cada entidad, es decir, por la información que se puede considerar conocida por el lector.

Cuando se pretende cubrir la función distintiva de las referencias, la GER debe tener en cuenta a la hora de generar la referencia no la información completa de que dispone, sino la información que ha sido transmitida previamente en el texto. A la hora de cubrir la función descriptiva, es posible que la información extra que se pretende proporcionar ya haya sido expuesta anteriormente. Por ejemplo, en el cuento de La Cenicienta, casi todas las menciones a los zapatos de la protagonista incluyen la información extra de que éstos son de cristal. En estos casos la reiteración de dicha información pretende destacar este hecho al lector. Por otro lado, si la información a presentar no ha sido dada previamente, es tarea de la GER encargarse de que sea entendida correctamente y no produzca ningún tipo de confusión. Esto podría ocurrir, por ejemplo, si no se relaciona correctamente el concepto referido anteriormente con la nueva característica. Si por ejemplo tenemos una historia en la que ha aparecido una princesa de la que sólo se ha dicho que es guapa, y más tarde se usa una referencia como *la princesa morena*, el lector podría pensar que la princesa morena es una nueva princesa y no tiene nada que ver con la anterior.

Generación de Información Descriptiva Guiada por la Determinación de Contenido

La Determinación de Contenido podría ser la encargada de decidir qué información comunicar para la descripción de entidades teniendo en cuenta

todos los elementos que aparecerán en el texto y sus características. En el momento de decidir qué información incluir para describir a una entidad, la DC debe ser consciente de todos elementos que aparecerán en el discurso e incluir la información necesaria para que todos ellos sean distinguibles entre sí.

Una consecuencia de esta solución es que la DC tendrá que prever parte de las decisiones que tomará más adelante la GER para la distinción de entidades. Para ello deberá asegurarse de añadir al discurso suficiente información para que algoritmos como la selección de atributos, que se encarga de elegir qué atributos de un elemento deben ser mencionados para distinguirlo del resto de elementos del mismo tipo, no se encuentren con menos información de la que necesitan para hacer las referencias.

Generación de Información Descriptiva Guiada por la Generación de Expresiones de Referencia

Otra posible solución para el problema planteado sería proporcionar una comunicación bidireccional entre los módulos de Determinación de Contenido y Generación de Expresiones de Referencia. Ante la falta de información para crear una referencia adecuada, como en el ejemplo de las princesas mencionado anteriormente, el módulo de GER haría una solicitud al módulo de DC para obtener más información sobre las dos princesas involucradas. Supongamos que se sabe que la segunda princesa es morena, aunque inicialmente este hecho no parecía relevante para el discurso y no había sido incluido para describirla. Con esta nueva información sí será posible generar referencias que distingan a las dos princesas.

Por supuesto, podría darse la situación en que el módulo de DC no pudiese proporcionar información adicional para la descripción de una entidad. En estos casos, la GER deberá recurrir a otro tipo de técnicas, como por ejemplo el uso de demostrativos (*this princess* o *that princess*) o de otros identificadores (*the first princess*). Pero estos aspectos de la GER quedan fuera del estudio realizado en este trabajo.

6.4.2. Propuesta de Implementación en una Arquitectura GLN

En esta sección se estudian las implicaciones de las dos propuestas expuestas anteriormente desde el punto de vista de una posible implementación de las mismas dentro del sistema de generación previamente existente TAP (sección 2.4). Con respecto a las tareas que se discuten aquí, son relevantes el módulo de determinación de contenido y el módulo de generación de expresiones de referencia.

Adaptaciones Necesarias

De cara a implementar las soluciones descritas de manera abstracta más arriba, resulta importante dotar al módulo de generación de expresiones de referencia de la capacidad de identificar este tipo de situaciones anómalas, y, en cada caso, almacenar los datos que permitan identificar los referentes entre los que surge el conflicto. Esto es tan sencillo como, en cada ocasión en que el conjunto de contraste no esté vacío al final del proceso, almacenar ese remanente del conjunto de contraste junto con el referente original: estos referentes son los que quedan indistinguibles con la información proporcionada por el módulo de determinación de contenido.

Con respecto al módulo de determinación de contenido, sería necesario añadir una funcionalidad que permitiera solicitar una revisión de una asignación de contenido previa, de modo que esta solicitud pueda venir acompañada de un conjunto de conjuntos de referentes indistinguibles según la asignación previa.

Opciones Básicas de Combinación

Todo lo expuesto anteriormente daría lugar a tres opciones de utilización.

En la primera, la determinación de contenido no añadiría ningún tipo de información sobre los referentes en la primera pasada. Durante la generación de expresiones de referencia se detectan los conjuntos de referentes indistinguibles, generando una lista de los mismos para los que no ha podido generar una referencia correcta por falta de información. Se solicitaría entonces una segunda pasada de determinación de contenido para añadir información que los distinga. Este bucle podría continuar hasta que el módulo de generación de expresiones de referencia no encuentre problemas a la hora de generar las referencias, en cuyo caso el flujo del pipeline continuaría hasta la generación del texto final.

En la segunda opción, la determinación de contenido puede añadir información descriptiva en la primera pasada, pero no necesariamente tiene que ocuparse de que esa información sea distintiva, sino que serán más bien descripciones iniciales apropiadas para cada entidad según su papel en el discurso completo. Por ejemplo, los personajes y lugares principales siempre se describen más exhaustivamente que los secundarios. La generación de expresiones de referencia se lleva a cabo como en la primera opción y se solicita al módulo de determinación de contenido una segunda pasada para completar las descripciones con la información necesaria para distinguir. En esta segunda opción, pueden combinarse las funcionalidades de las descripciones para facilitar la identificación y para añadir información que no distingue de otros referentes pero que se considera importante.

La tercera opción incluiría en el módulo de determinación de contenido llamadas a los fragmentos de la generación de expresiones de referencia que

son independientes del discurso. Para cada referente se haría una exploración genérica del conjunto de contraste máximo posible aplicando los criterios mencionados anteriormente, y se utilizaría la realimentación obtenida para añadir la información que pudiera ser necesaria ya en la primera pasada de la Determinación del Contenido.

6.4.3. Discusión

La generación de información distintiva guiada por la Determinación de Contenido puede resultar un proceso muy costoso dependiendo de la forma en que sea abordado. A la hora de seleccionar la información correspondiente para la descripción de una entidad, el resto de entidades con que se compara pueden haber sido ya descritas, es decir, la información que se usará para su descripción puede haber sido seleccionada anteriormente. Habrá por tanto que decidir si la DC considera esta información seleccionada a la hora de realizar la descripción de las otras entidades o no. No es lo mismo tener en cuenta toda la información disponible sobre una entidad a la hora de distinguirla de otra, que tener sólo en cuenta la información que va a ser mencionada sobre esa entidad en el discurso. En el segundo caso la descripción estará teniendo en cuenta el contexto del discurso y no sólo la información disponible inicialmente.

La generación de descripciones guiada por la Generación de Expresiones de Referencia provocaría que en las descripciones del texto sólo se incluyera la información necesaria para la distinción de los elementos del discurso. Esto podría provocar textos demasiado escuetos, con pocas descripciones, si el número de entidades que pueden ser confundidas con otras es bajo. Además, esta aproximación no se encarga del uso de información adicional no necesaria para distinguir pero que se considera interesante para la comprensión completa del texto. Una posible solución sería que en la DC se decidiera qué información descriptiva no necesaria para distinguir se quiere expresar en el texto, y que la fase posterior de REG se encargue de solicitar información adicional a ésta cuando lo necesite.

Como trabajo futuro quedaría la implementación de algún algoritmo que se encargara de la generación de información descriptiva guiada por la Determinación de Contenido teniendo en cuenta el contexto de lo que se mencionará en el discurso. Un algoritmo de este tipo no es trivial, ya que al mismo tiempo que se va seleccionando la información que será mencionada para describir a las entidades, el contexto sobre el que se tomaron anteriores decisiones irá cambiando. Las decisiones que se han tomado para las descripciones ya generadas deberían ser revisadas según esto ocurre.

Una posible implementación para un algoritmo de este tipo podría ser la utilización de un algoritmo evolutivo. Los genes representarían los atributos que se mostrarán como información descriptiva de cada una de las entidades que aparecerán en el discurso. La función de aptitud evaluaría si las des-

cripciones generadas para cada elemento contendrían suficiente información para que se pueda distinguir de todos los demás. La ventaja proporcionada en este caso por los algoritmos evolutivos es que se están considerando al mismo tiempo todas las descripciones de los elementos del discurso, sin necesidad de realizar una revisión cada vez que se genera una nueva.

Resumen y Conclusiones

En este capítulo se han discutido las diferentes partes que forman este trabajo de tesis doctoral, y se han propuesto algunas líneas de trabajo futuro tanto para perfeccionar las soluciones ya implementadas, como para explorar nuevas líneas en las que aplicar los resultados obtenidos.

Desde el punto de vista de las referencias básicas, se podrían buscar mejoras a los módulos existentes y nuevas formas de evaluación que pudieran aproximarse todavía más a como los seres humanos resuelven el problema de la generación de referencias. Desde el punto de vista de las referencias apoyadas en figuras retóricas, sería adecuado encontrar alguna forma de evaluar las referencias generadas, ya sea usando evaluadores humanos o algún tipo de métrica que pueda ser calculada automáticamente.

Se discuten también las repercusiones que tienen cada una de las líneas estudiadas en el proceso completo de generación automática de texto.

Capítulo 7

Conclusiones y Trabajo Futuro

"All right, Mr. DeMille, I'm ready for my close-up."
— Sunset Blvd. (1950)

En este trabajo hemos tratado la generación de expresiones de referencia desde dos perspectivas diferentes. En primer lugar, se ha estudiado la generación de referencias básicas con el objetivo de distinguir referentes a través de la propuesta y la evaluación de soluciones basadas en diferentes técnicas de Inteligencia Artificial. En segundo lugar, se ha abordado el uso de figuras retóricas como la analogía y la comparación como recursos para la generación de referencias con función descriptiva.

En este capítulo se resumen las principales conclusiones derivadas de los resultados y las discusiones presentadas en esta tesis. En primer lugar se resumen las principales aportaciones y conclusiones (sección 7.1). También analizamos por separado las principales conclusiones en relación con la generación básica de referencias (sección 7.2) y la generación de referencias usando figuras retóricas (sección 7.3). La evaluación y los resultados obtenidos por las soluciones propuestas se analizan también en la sección 7.4. Las interacciones entre las dos funciones de la generación de referencias y el proceso completo de generación de texto se describen en la sección 7.5. Por último, se presentan algunas líneas de trabajo futuro en la sección 7.6.

7.1. Distinción y Descripción de Entidades

Uno de los principales aspectos analizados en este trabajo es que las referencias en un texto puede tener dos funciones u objetivos diferentes. La función distintiva de las referencias se ha definido como la intención de identificar unívocamente un referente en contraste con otros con los que se puede confundir. La función descriptiva de las referencias se puede definir como la inserción de información que no es necesariamente distintiva en una

referencia, pero que proporcionar información descriptiva o adicional sobre la entidad que se considera importante para el lector u oyente.

La función distintiva de las referencias ha sido ampliamente estudiada en la literatura, al contrario que ocurre con la descriptiva. Esto se debe al hecho de que el objetivo de distinguir referentes es un elemento básico en la generación de lenguaje natural, y más concretamente en la generación de expresiones de referencia. La definición de esta tarea es sencilla, sólo depende del contexto del discurso, y puede ser calculada automáticamente con algoritmos deterministas que producen como resultado si una referencia es distintiva o no. Por el contrario, la función descriptiva de las referencias es más difícil de definir, ya que es una función vaga que depende del contexto en el que se presenta el discurso. La decisión de añadir información descriptiva no sólo depende de la información presentada en el texto, sino que también está relacionada con la intención del hablante o escritor cuando crea el mismo.

Toda referencia que se haga en un texto tiene tanto una parte distintiva como una parte descriptiva, cada una de ellas correspondiente a uno de los objetivos que se pueden abordar. La relación entre estas dos partes en la misma referencia es un problema que tampoco ha sido estudiado en la literatura. Para hacer frente al mismo, sería necesario comenzar estudiando por separado los dos objetivos para encontrar definiciones precisas, soluciones para resolverlos, y métricas para evaluar su exactitud. Este estudio es precisamente la principal contribución de esta tesis.

En este trabajo hemos estudiado la función distintiva de las referencias en la generación de referencias básicas en forma de frases nominales, y la función descriptiva usando figuras retóricas basadas en similitudes entre dominios. Este trabajo pretende ser un primer paso hacia el estudio del proceso completo de generación de expresiones de referencia que cubran tanto la función distintiva como la descriptiva de las referencias. En las siguientes secciones se examinan con más detalle las principales conclusiones para cada una de estas dos funciones.

7.2. Valoración de la Generación de Referencias con Función Distintiva

En el caso de la función distintiva de las referencias básicas se puede encontrar mucho trabajo acerca de su definición, posibles soluciones y métricas de evaluación. En esta tesis hemos estudiado las soluciones más comunes encontradas en la literatura, y hemos propuesto mejoras que han dado lugar a buenos resultados de evaluación. También hemos encontrado que este proceso da lugar a interacciones más allá de la tarea de generación de expresiones de referencia. Por ejemplo, al decidir el nivel de abstracción que se utilizará en una referencia, estamos tratando cuestiones más relacionadas con la

información conceptual disponible durante la Determinación de Contenido que con la generación de referencias. Además, a la hora de elegir las palabras para expresar la referencia resultante, ciertas decisiones acerca de su forma sintáctica o qué palabra que se puede utilizar están más relacionadas con el proceso completo de Lexicalización.

A la hora de generar expresiones de referencia nominales que tienen como objetivo distinguir entidades en un discurso, se ha considerado la división del problema en tres partes diferenciadas: elección del nivel de abstracción, selección de atributos y lexicalización. Algunas de ellas están más allá de una definición estricta de la etapa de Generación de Expresiones de Referencia, ya que interfieren en las elecciones conceptuales y sintácticas, por ejemplo.

En la sección 4.1 se discutió la importancia de utilizar diferentes niveles de abstracción para las referencias en función del contexto. En situaciones donde la información necesaria para la Generación de Expresiones de Referencia se almacena en una taxonomía o jerarquía, es posible utilizar esta información para determinar si es preferible referirse a un referente utilizando un tipo muy concreto (por ejemplo, *doberman*), o utilizar un valor más genérico (por ejemplo, *dog*). Esta decisión depende directamente de la situación en la que se desarrolla el discurso. Si hay otros perros de diferentes razas en el contexto del discurso, la referencia específica sería más apropiada. Si sólo hay otros tipos de animales pero no perros, la referencia genérica sería suficiente y no se daría más información que la necesaria.

En este trabajo hemos estudiado el uso de ontologías para hacer frente a este problema como un primer paso en el proceso completo de generación de referencias. Como se indica en el algoritmo incremental de Reiter y Dale (1992), el nivel de abstracción para el tipo o *basic level value* se obtiene a partir de la base de conocimiento o del modelo de usuario. En nuestro caso hemos implementado un método dinámico para obtener este valor que sólo depende de los conocimientos disponibles sobre el mundo que aparece en el discurso.

Como vimos en la sección 4.2, la selección de qué atributos mencionar a la hora de generar una referencia en forma de frase nominal se puede considerar como un problema de búsqueda. En este trabajo se propusieron tres posibles soluciones: el uso de agrupamientos relativos de los atributos a la hora de elegir cuáles mencionar (sección 4.2.1), tener en cuenta el valor de un atributo a la hora de decidir si usarlo o no (sección 4.2.2), y el uso de algoritmos evolutivos a la hora de decidir el conjunto de atributos más apropiada (sección 4.2.3). Las dos primeras están basadas en el algoritmo incremental de Reiter y Dale (1992). De los resultados obtenidos en las evaluaciones de estas soluciones se pueden extraer diversas conclusiones.

Para las soluciones basadas en el algoritmo incremental, es evidente que la manera en que se consideran los posibles atributos tiene una fuerte influencia en los resultados obtenidos. En el caso de la solución basada en agrupamientos relativos se identificaron distintos grupos de atributos. Los resultados fueron muy diferentes en función del orden de consideración elegido para estos grupos, mientras que el orden de los atributos dentro de los grupos tenía poca repercusión en los resultados. Cuando se consideró también la influencia de los valores de los atributos, los resultados mostraron que en función de su valor algunos atributos son más susceptibles de ser utilizados que otros.

Al considerar el uso de algoritmos evolutivos para resolver la selección de atributos, la principal cuestión con que nos encontramos fue la elección de la función de aptitud. Este tipo de algoritmo resultó ser un buen enfoque para un problema en el que la bondad de una solución puede ser calculado, pero el proceso de obtención de la misma no puede ser formalizado. El enfoque presentado en este trabajo trata adecuadamente con la tarea y obtiene resultados razonables.

En la sección 4.3 se discutió acerca de diferentes aspectos de la lexicalización de referencias. Una arquitectura general para la ampliación de la elección léxica usando alternativas extraídas de WordNet se abordó en la sección 4.3.1. En la sección 4.3.2 se presentó una solución de lexicalización basada en las elecciones más frecuentes utilizadas, mostrando que la lexicalización va más allá de la simple elección de palabras. Por último, la importancia del estilo de la hora de decidir qué palabras son apropiadas para expresar una referencia se introdujo en la sección 4.3.3.

El uso de sinónimos durante la generación de texto entra en conflicto con otras tareas de la GLN. Por ejemplo, si una ocurrencia de un concepto ha sido sustituida por un pronombre durante la Generación de Expresiones de Referencia, la disponibilidad de sinónimos se convierte en irrelevante. Este tipo de dependencias deben ser tenidas en cuenta. Además, el enriquecimiento léxico utilizando un recurso léxico debe ser manejado cuidadosamente para evitar un uso excesivo. Si una palabra específica tiene muchos sinónimos en WordNet, por ejemplo, es necesario realizar un uso cuidadoso de los mismos para evitar la producción de un texto que emplee demasiadas palabras para referirse a la misma cosa. Un texto como éste sería muy difícil de entender.

La lexicalización de referencias usando la opción más común muestra algunas dependencias que hacen casi imposible separarla de otras tareas de la generación tales como la Generación de Expresiones de Referencia o la Elección Sintáctica. Al generar la lexicalización para una referencia, es necesario elegir qué determinantes deben ser utilizados (Generación de Expresiones de Referencia), qué palabras específicas escoger (Elección Léxica) y qué estructura sintáctica es la más adecuada (Elección Sintáctica). A pesar

de que estas decisiones son bastante independientes, todas ellas deben ser tenidas en cuenta a la hora de lexicalizar la representación conceptual de la referencia.

Otra decisión que se ha abordado durante la lexicalización de expresiones de referencia es el estilo de referencia que se utilizará en cada situación concreta. Cada persona tiene una forma diferente de expresarse, y ello puede dar lugar a varias posibles lexicalizaciones que son igualmente adecuadas. La única diferencia entre ellas sería el estilo de discurso en el que están incluidas. Alguna manera de generar las referencias usando diferentes estilos debe ser por tanto contemplado. En este trabajo se ha propuesto la utilización del paradigma del razonamiento basado en casos para abordar esta cuestión. La idea es entrenar el sistema usando referencias generadas de acuerdo a un estilo específico, de manera que la generación pueda basarse en un estilo que el sistema entienda. Existen tantas maneras de crear una referencia, que no es posible dar una solución general que se ocupe de todas ellas en la misma forma.

7.3. Valoración de la Generación de Referencias usando Figuras Retóricas basadas en Similitudes entre Dominios

En el capítulo 5 abordamos el uso de figuras retóricas basadas en similitudes entre dominios a fin de enriquecer las referencias en un texto con información descriptiva. Más concretamente, hemos considerado el uso de comparaciones y analogías para hacer frente a esta tarea.

El propósito de la comparación y la analogía tal y como se ha estudiado en este trabajo es completar los conocimientos previos del oyente sobre un concepto a base de similitudes con otros conceptos. Las comparaciones y analogías son aparentemente un buen enfoque para este fin, ya que remarcan una propiedad mediante la búsqueda de una entidad que también la ostenta. El alineamiento estructural entre entidades también ha sido estudiado para evitar inferencias incorrectas que pueden surgir cuando se considera de manera superficial el tipo o las similitudes.

El proceso de generar una figura retórica para referirse a un concepto específico implica varias tareas que se tratan por separado. Teniendo en cuenta que el concepto inicial pertenece a un dominio dado, el primer paso es encontrar otro dominio donde buscar las comparaciones o analogías. Una vez que se ha encontrado, y que la correspondencia entre los dos dominios queda establecida, es necesario generar un conjunto de posibles referencias para cada uno de los conceptos susceptibles de ser mencionados usando una figura retórica. Este conjunto de referencias retóricas deben ser estudiadas y evaluadas en términos de claridad y adecuación, de manera que las no apropiadas puedan ser filtradas. Por último, para cada ocurrencia del concepto

en un contexto determinado dentro del texto, es necesario decidir si utilizar una de las comparaciones o analogías para referirse a este concepto en ese momento o no, siempre evitando la pérdida de significado o ambigüedades innecesarias.

En este trabajo hemos elegido implementar nuestras ideas en forma de un sistema multiagente. Este tipo de arquitectura nos ha permitido utilizar fácilmente fuentes de conocimiento y sistemas previamente existentes que se requerían para el proceso de generación de analogías y comparaciones. Como bases de conocimiento hemos utilizado los dominios de Sapper y WordNet para obtener la información necesaria para la generación de las comparaciones y analogías.

7.4. Evaluación de las Referencias Generadas

En el capítulo 4 se han evaluado y discutido las soluciones propuestas para la generación referencias básicas usando las métricas habituales del campo. La mayoría de estas métricas se basan en la comparación de las referencias generadas con un corpus de referencias creado por evaluadores humanos para distintos dominios. Además, también se llevaron a cabo pruebas para comprobar cómo reaccionaban distintas personas ante las referencias generadas en determinadas situaciones. Los resultados han mostrado que los sistemas que presentaban un buen rendimiento en las comparaciones con el corpus no eran siempre las más adecuadas ante personas reales. Además, el corpus no era suficientemente homogéneo, dándose el caso de que determinadas soluciones que tenían un buen resultado en un cierto conjunto de referencias en el corpus, obtenían malos resultados para otras referencias del mismo corpus. La conclusión obtenida de estos resultados es que un corpus creado a partir de las referencias planteadas por personas diferentes no puede ser visto como un ejemplo de cómo se deberían hacer las cosas. Este tipo de corpus podría ser utilizado de forma más fructífera si se tienen en cuenta factores como los estilos personales.

En el caso de la adición de información descriptiva a las referencias mediante el uso de figuras retóricas, no hemos encontrado ningún recurso disponible (como un corpus) que pudiese actuar como ejemplo de cómo la gente usa este tipo de figuras para enriquecer la información o para describir entidades desconocidas. Como resultado, no ha sido posible llevar a cabo una evaluación formal similar a la realizada para las referencias básicas. Cabe destacar que, en caso de existir, un corpus de estas características podría no ser útil para evaluar formalmente las estructuras retóricas generadas. En el caso de las referencias básicas, donde la tarea es más específica y está más mejor delimitada, la evaluación basada en corpus ha resultado no ser tan buena como cabría esperar. En el caso de las figuras retóricas, que son mucho más subjetivas, la comparación con un corpus difícilmente sería una métrica

adecuada, por lo que en el futuro será necesario explorar nuevos enfoques para la evaluación.

7.5. Interacciones con el Proceso Completo de Generación

En la generación y posterior uso de las expresiones de referencia tanto para distinguir como para describir ya no se encuentra involucrada simplemente la tarea de Generación de Expresiones de Referencia (GER), sino que también habrá que tener en cuenta la Determinación de Contenido (DC).

La Determinación de Contenido se encarga de decidir qué información se desea transmitir en un discurso a partir de toda la información disponible para el mismo. Desde el punto de vista de la función descriptiva de las referencias, puede ser al final de la Determinación de Contenido, cuando ya se ha decidido qué entidades van a aparecer en el texto y cómo se va a hablar de ellas, donde se decide cuando es apropiado o necesario realizar una descripción y qué información se va a proporcionar en ella. Así, a la hora de realizar una referencia a un concepto, la Generación de Expresiones de Referencia deberá tener en cuenta qué se ha descrito anteriormente sobre ese concepto y por tanto qué información conocida por el lector puede ser utilizada para referirse a él. En la mayoría de los sistemas GLN la relación entre estas dos tareas es unidireccional.

La arquitectura más extendida para sistemas de GLN es la arquitectura secuencial o de *pipeline*, donde la elección del contenido del discurso y su organización en mensajes es realizada al principio del proceso (sección 2.1). Los subsiguientes pasos de la generación usan esta información, entre ellos la GER, sin posibilidad de que sea revisada por etapas anteriores del pipeline. Desde el punto de vista de la relación de la DC y la GER en la generación de referencias a entidades, el flujo de información proporcionado por esta arquitectura puede no resultar adecuado en ciertas ocasiones. Imaginemos que en la fase de GER el sistema se da cuenta de que no dispone de suficiente información sobre una entidad para distinguirla del resto que están en el mismo contexto. Con una arquitectura secuencial el problema no tendría remedio y la expresión de referencia generada sería ambigua.

Un problema como éste puede solucionarse de varias maneras. Una de ellas sería que la DC se encargara, además de decidir qué información es importante para el discurso como un todo, de comprobar si la información seleccionada es suficiente para que no se produzcan ambigüedades a la hora de generar las expresiones de referencia distintivas. Otra solución sería que la DC seleccionara poca o nada de información para la parte descriptiva de las referencias, y que fuera la GER la que se encargara de solicitar la inclusión de nueva información cuando resultara necesaria para generar las referencias.

7.6. Trabajo futuro

La línea más importante de trabajo futuro sería la combinación de las soluciones aportadas para la generación de referencias básicas con función distintiva y para la generación de figuras retóricas que aportan información descriptiva. Esto requiere completar el estudio de las referencias básicas, analizando su posible uso con funciones descriptivas, así como analizar las posibles interacciones sobre la función distintiva que pueden surgir al introducir figuras retóricas. Una vez analizadas las posibles repercusiones que puede tener cada una de las funciones sobre la otra, será el momento de aportar soluciones concretas para su combinación.

También se han identificado otras posibles líneas de trabajo futuro, centradas en las referencias básicas. En primer lugar, se ha identificado la posibilidad de usar modelos de usuario que representen el nivel de experiencia del lector u oyente dentro de un dominio específico de cara a determinar el nivel de abstracción. Por otro lado, también se ha identificado la necesidad de definir funciones de aptitud más precisas de cara a la aplicación de algoritmos evolutivos. También es importante resaltar que en este trabajo sólo hemos considerado el uso de referencias nominales para hacer referencia a las entidades del discurso. Existen otros tipos de referencias que también se podrían emplear, dado que añaden fluidez y naturalidad al discurso: referencias pronominales, relaciones como modificadores de frases nominales, nombres propios, etc.

Por otro lado, dentro de nuestro enfoque para la generación de referencias que emplean analogías para añadir información descriptiva, sólo hemos considerado las propiedades de los conceptos y de las instancias a la hora de construir las referencias. Aún así, los elementos en dominios generales también se relacionan con otros elementos a través de otros tipos de relación. Por ejemplo, en los dominios que hemos empleado sería posible encontrar relaciones como *Han Solo is in love with Princess Leia* o *Excalibur is stuck in a stone*. Estas relaciones entre elementos podrían usarse no sólo al generar la correspondencia entre dominios, sino también al crear la analogía.

Finalmente, otra de las líneas que se han dejado abiertas sería la creación de un corpus de figuras retóricas, aunque dicho trabajo caería fuera del alcance de esta línea de investigación. Además, cabe la posibilidad de que dicho corpus no llegase a resultar una herramienta útil para evaluar comparaciones y analogías generadas automáticamente. Para este tipo de evaluación podría resultar interesante llevar a cabo experimentos con evaluadores humanos que juzgasen las referencias generadas y valorasen en qué medida cumplen los distintos objetivos planteados.

El trabajo presentado en esta memoria de tesis no ha estado asociado a ningún dominio concreto. Hemos intentado proponer soluciones genéricas que puedan ser empleadas para generar textos en distintos dominios. Aún

así, es cierto que determinados dominios presentan sus propias particularidades y requisitos que deben ser tenidos en cuenta.

En esta línea, un tipo especial de textos que puede merecer un análisis más detallado en el futuro es de los textos narrativos. Casi todas las soluciones existentes para la generación de expresiones de referencia basadas en selección de atributos asumen que las características de los objetos tratados no cambian a lo largo del tiempo. Por tanto, estos métodos son válidos principalmente en la generación de textos descriptivos o diálogos en los que el contexto no varía.

Sin embargo, la premisa de los atributos que se mantienen constantes en el tiempo no se cumple en los textos narrativos. Por su propia definición, la narrativa se basa en secuencias de sucesos que ocurren cambiando el estado del mundo constantemente: los personajes y objetos de las historias nacen, crecen, mueren, cambian de forma, aparecen, desaparecen,...

Así, a la generación de expresiones de referencia en textos narrativos no se puede aplicar una solución simple basada en la selección de atributos dado que los atributos pueden cambiar durante el discurso, ya sea explícitamente (por ejemplo, después de *the princess heard the scream and she got scared*, la princesa cambia su estado a *scared*), o implícitamente (por ejemplo, después de *the prince killed the dragon* el estado del dragón cambia a *dead* incluso cuando no se ha mencionado directamente). Aspectos tales como líneas temporales (tiempo de la historia vs. tiempo del discurso), o características existentes y mencionadas, son sólo algunos ejemplos de cuestiones que deben abordarse en el trabajo futuro sobre referencias en textos narrativos.

Otra figura retórica basada en similitudes entre dominios que puede ser estudiada a raíz del trabajo realizado en materia de comparaciones y analogías es la metáfora. Las metáforas son muy similares a las analogías, pero se suelen utilizar de una manera diferente. En lugar de tener mensajes que comparan los dos conceptos como en las analogías, las metáforas se utilizan en lugar del concepto original.

La generación de metáforas es un proceso delicado ya que una metáfora sólo se entenderá si es posible para el lector encontrar un número suficiente de elementos comunes entre el concepto metafórico utilizado y el concepto real. Por ejemplo, en los ejemplos presentados en este trabajo hemos encontrado una correspondencia entre Lancelot y Han Solo: ambos son hombres hábiles, valientes y jóvenes. Sin embargo, si generamos la frase *Han Solo arrived to Camelot* la metáfora no se entenderá. Se debe por tanto estudiar cómo este tipo de malentendidos puede evitarse. Por ejemplo, la metáfora *the steely light saber* en lugar de *Excalibur* podría ser más comprensible porque la diferencia entre los conceptos (ser de acero) se menciona explícitamente.

Bibliografía

(2000). FIPA ACL message representation in bit-efficient specification. `http://www.fipa.org/specs/fipa00069/` [24/2/2009].

Agnar, A., y Enric, P. (1994). Case-based Reasoning : Foundational issues, methodological variations, and system approaches. *AI Communications*, *7*(1).

Alonso-Lavernia, M., la Cruz-Rivera, A. D., y Sidorov, G. (2006). Generation of natural language explanations of rules in an expert system. En *Computational Linguistics and Intelligent Text Processing, proc. of CICLing 2006*, (pp. 311–314). Springer-Verlag.

Anderson, J. R. (1983). *The architecture of cognition*. Harvard University Press.

Arano, S. (2003). La ontología: una zona de interacción entre la lingüística y la documentación. *Hipertext.net [on-line] [14/03/2009]*, (2).

Ardissono, L., y Goy, A. (2000). Dynamic generation of adaptive web catalogs. En *Proc. Conference on Adaptive Hypermedia and Adaptive Web-based Systems*. Italy.

Baker, C. F., Fillmore, C. J., y Lowe, J. B. (1998). The berkeley framenet project. En *Proceedings of the COLING-ACL*. Montreal, Canada.

Barzilay, R., y Lapata, M. (2005). Collective content selection for concept-to-text generation. En *HLT '05: Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, (pp. 331–338). Morristown, NJ, USA: Association for Computational Linguistics.

Bateman, J. A., Henschel, R., y Rinaldi, F. (1995). Generalized Upper Model 2.0: documentation. Tech. rep., GMD/Institut für Integrierte Publikations- und Informationssysteme.

Bateman, J. A., Kasper, R. T., Moore, J. D., y Whitney, R. A. (1990). A general organization of knowledge for natural language processing: the PENMAN upper model. En *Proceedings of the Fifth International Workshop on Natural Language Generation (INLG-90)*.

Bechhofer, S., van Harmelen, F., Hendler, J., Horrocks, I., McGuinness, D. L., Patel-Schneider, P. F., y Stein, L. A. (2004). OWL web ontology language reference. `http://www.w3.org/TR/2004/REC-owl-ref-20040210/` [24/2/2009].

Beckwith, R., Miller, G. A., y Tengi, R. (1993). Design and implementation of the WordNet lexical database and searching software.

Belz, A. (2008). Automatic generation of weather forecast texts using comprehensive probabilistic generation-space models. *Natural Language Engineering*, *14*, 431–455.

Belz, A., y Gatt, A. (2007). The attribute selection for GRE challenge: Overview and evaluation results. En *Proc. 2nd UCNLG Workshop: Language Generation and Machine Translation (UCNLG+MT)*, (pp. 75–83). Copenhaguen, Denmark.

Belz, A., y Gatt, A. (2008). REG Challenge 2008: Participants pack. `http://www.nltg.brighton.ac.uk/research/reg08/` [24/2/2009].

Belz, A., y Reiter, E. (2006). Comparing automatic and human evaluation of NLG systems. En *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics, EACL-06*.

Bohnet, B., y Dale, R. (2004). Referring expression generation as a search problem. En *Proceedings of the Australasian Language Technology Workshop 2004*.

Bontcheva, K., y Wilks, Y. (2001). Dealing with dependencies between content planning and surface realisation in a pipeline generation architecture. En *Proc. of the International Joint Conference on Artificial Intelligence (IJCAI'01)*.

Botti, V., Carrascosa, C., Julian, V., y Soler, J. (1999). The ARTIS agent architecture: Modelling agents in hard real-time environments. En *Proceedings of the MAAMAW'99*, (pp. 63–76). Springer-Verlag.

Brachman, R. J., Mcguinness, D. L., Patel-schneider, P. F., Resnick, L. A., y Borgida, A. (1991). Living with classic: When and how to use a KL-ONE-like language. *Principles of Semantic Networks*, (pp. 401–456).

Buchanan, B., Moore, J., Forsythe, D., Carenini, G., Banks, G., y Ohlson, S. (1995). An intelligent interactive system for delivering individualized information to patients. *Artificial Intelligence in Medicine*, *7*, 117–154.

Cahill, L. (1998). Lexicalisation in applied NLG systems. Tech. Rep. ITRI-99-04.

Callaway, C., y Lester, J. (2001a). Evaluating the effects of natural language generation techniques on reader satisfaction. En *Proceedings of the Twenty-Third Annual Conference of the Cognitive Science Society*. Edinburgh, UK.

Callaway, C., y Lester, J. (2001b). Narrative prose generation. En *Proceedings of the 17th IJCAI*, (pp. 1241–1248). Seattle, WA.

Callaway, C. B., y Lester, J. C. (2001c). Pronominalization in generated discourse and dialogue. En *ACL '02: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, (pp. 88–95). Association for Computational Linguistics.

Cardie, C. (1993). A case-based approach to knowledge acquisition for domain-specific sentence analysis. En *National Conference on Artificial Intelligence*, (pp. 798–803).

Carrillo, L. (2005). Marco comunicativo del estilo en el uso de la lengua. *Ámbitos*, (13-14), 135–153.

Cawsey, A., Binsted, K., y Jones, R. (1995). Personalised explanations for patient education. En *Proceedings of the 5th European Workshop on Natural Language Generation*, (pp. 59–74).

Cheng, H. (1998). Embedding new information into referring expressions. En *ACL-36: Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics*, (pp. 1478–1480). Morristown, NJ, USA: Association for Computational Linguistics.

Crubézy, M., Dameron, O., Fergerson, R. W., Knublauch, H., Musen, M. A., Noy, N. F., Rubin, D., Tu, S. W., y Vendetti, J. (2005). Protégé project. Tech. Rep. 24/3.

Cruse, D. (1986). *Lexical Semantics*. New York: Cambridge University Press.

Culy, C. (2002). WordNet Agent. `http://www.ai.sri.com/oaa/contributions/wordnet` [24/2/2009].

Daelemans, W., Gillis, S., y Durieux, G. (1994). The acquisition of stress: a data-oriented approach. *Comput. Linguist.*, *20*(3), 421–451.

Dale, R. (1989). Cooking up referring expressions. En *Proceedings of the 27th Annual Meeting of the Association for Computational Linguistics*.

Dale, R., y Reiter, E. (1995). Computational interpretation of the Gricean maxims in the generation of referring expressions. *Cognitive Science*, *19*(8), 233–263.

Dale, R., y Reiter, E. (1996). The role of the Gricean maxims in the generation of referring expressions. En *Working Notes for the AAAI Spring Symposium on Computational Implicature*, (pp. 16–20). AAAI Press.

Darwin, C. (1859). *On the Origin of Species by Means of Natural Selection*. London: J. Murry.

Duboue, P., y McKeown, K. (2002). Content planner construction via evolutionary algorithms and a corpus-based fitness function. En *Proceedings of the Second International Natural Language Generation Conference (INLG 2002)*. Ramapo Mountains, NY.

Elhadad, M. (1993). FUF: the universal unifier user manual. Tech. Rep. Technical Report CUCS-038-91, Columbia University.

Elhadad, M., y Robin, J. (1996). SURGE: a comprehensive plug-in syntactic realization component for text generation. Tech. rep., Technical Report 96-03, Department of Computer Science, Ben Gurion University.

Falkenhainer, B., Forbus, K. D., y Gentner, D. (1989). The structure mapping engine: Algorithm and examples. *Artificial Intelligence*, *41*, 1–63.

Fauconnier, G., y Turner, M. (2002). *The Way We Think*. Basic Books.

Fellbaum, C., Gross, D., y Miller, K. (1990). Adjectives in WordNet. *International Journal of Lexicography*, *3*(4), 265–277.

Finin, T., Fritzson, R., McKay, D., y McEntire, R. (1994). KQML as an Agent Communication Language. En N. Adam, B. Bhargava, y Y. Yesha (Eds.) *Proceedings of the 3rd International Conference on Information and Knowledge Management (CIKM'94)*, (pp. 456–463). Gaithersburg, MD, USA: ACM Press.

Forster, K. I., y Forster, J. C. (2003). DMDX: a windows display program with millisecond accuracy. *Behavior research methods, instruments and computers : a journal of the Psychonomic Society, Inc.*, *35*, 116–124.

Gatt, A. (2007). *Generating Coherent References to Multiple Entities*. Ph.D. thesis.

Gatt, A., Belz, A., y Kow, E. (2008). The TUNA challenge 2008: Overview and evaluation results. En *Proceedings of the 5th International Conference on Natural Language Generation, INLG-08*, (pp. 198–206). Ohio, USA.

Gatt, A., Belz, A., y Kow, E. (2009). The TUNA-REG Challenge 2009: Overview and evaluation results. En *Proceedings of the 12th European Workshop on Natural Language Generation, ENLG-09*. Athens, Greece.

Gatt, A., van der Sluis, I., y van Deemter, K. (2007). Evaluating algorithms for the generation of referring expressions using a balanced corpus. En *Proc. of the 11th European Workshop on Natural Language Generation (ENLG 07)*.

Gentner, D. (1983). Structure-mapping: A theoretical framework for analogy. *Cognitive Science*, *7*(2).

Gervás, P. (2001). An expert system for the composition of formal Spanish poetry. *Journal of Knowledge-Based Systems*, *14*, 200–1.

Gervás, P. (2006). SurReal: a surface realization module. Tech. rep., Natural Interaction based on Language Group,Universidad Complutense de Madrid, Spain.

Gervas, P. (2007). TAP: a text arranging pipeline. Tech. rep., Natural Interaction based on Language Group, Universidad Complutense de Madrid, Spain.

Gervás, P., Díaz-Agudo, B., Peinado, F., y Hervás, R. (2005). Story plot generation based on CBR. *18*(4-5), 235–242.

Goldberg, E., Driedgar, N., y Kittredge, R. (1994). Using natural-language processing to produce weather forecasts. *IEEE Expert*, *9*, 45–53.

Gospodnetic, O., y Hatcher, E. (2004). *Lucene in Action*. Manning Publications. GOS o 05:1 1.Ex.

Grice, H. P. (1975). Logic and conversation. En P. Cole, y J. L. Morgan (Eds.) *Syntax and semantics*, vol. 3. New York: Academic Press.

Hervás, R., y Gervás, P. (2005). Case Retrieval Nets for heuristic lexicalization in natural language generation. En *Proceedings of the 12th Portuguese Conference on Artificial Intelligence (EPIA 05)*, (pp. 55–66). Springer-Verlag.

Hervás, R., y Gervás, P. (2006). Case-Based Reasoning for Knowledge-Intensive template selection during text generation. En *Proceedings of the European Conference on Case-Based Reasoning (ECCBR 06)*, (pp. 151–165). Springer-Verlag.

Holland, J. (1992). *Adaptation in Natural and Artificial Systems. An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence*. Cambridge, Massachusetts, Second Edition: MIT Press.

Hsu, E. I., y McGuinness, D. L. (2003). Wine agent: Semantic web testbed application. En *Proceedings of the Workshop on Description Logics*.

Huhns, M., y Singh, M. P. (1998). *Readings in Agents*.

Ide, N., y Veroni, J. (1998). Word Sense Disambiguation: The state of the art. *Computational Linguistics*, (24), 1–40.

Isard, A. (2007). Choosing the best comparison under the circumstances. En *Proceedings of the International Workshop on Personalization Enhanced Accessto Cultural Heritage (PATCH'07)*. Corfu, Greece.

Jing, H. (1998). Usage of WordNet in Natural Language Generation. En *COLING-ACL-98 Workshop on Usage of WordNet in Natural Language Processing Systems*.

Jurafsky, D., y Martin, J. (2000). *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Prentice Hall.

Kinneavy, J. (1980). *A Theory of Discourse: The Aims of Discourse*. Norton & Company.

Kolodner, J. (1983). Reconstructive memory, a computer model. *Cognitive Science*, *7*, 281–328.

Krahmer, E., y Theune, M. (2002). Efficient context-sensitive generation of referring expressions. En *Information Sharing: Givenness and Newness in Language Processing*, (pp. 223–264).

Lavoie, B., Rambow, O., y Reiter, E. (1997). Customizable descriptions of object-oriented models. En *Proceedings of ANLP'97*.

Lenat, D. (1995). CYC: A large-scale investment in knowledge infrastructure. *Communications of the ACM*, *38(11)*, 33–38.

Lenz, M., y Burkhard, H. (1996a). Case Retrieval Nets: Foundations, properties, implementation, and results. Tech. rep., Humboldt University, Berlin.

Lenz, M., y Burkhard, H.-D. (1996b). Case Retrieval Nets: Basic ideas and extensions. En *KI - Kunstliche Intelligenz*, (pp. 227–239).

Levy, R. (2001). A computational model of poetic creativity with neural network as measure of adaptive fitness. En *Proceedings of the ICCBR-01 Workshop on Creative Systems*.

Lin, D. (1998). Dependency-based evaluation of MINIPAR. En *Proc. of Workshop on the Evaluation of Parsing Systems*. Granada, Spain.

Mahesh, K., y Nirenburg, S. (1995). A situated ontology for practical NLP. En *Proceedings of the Workshop on Basic Ontological Issues in Knowledge Sharing, International Joint Conference on Artificial Intelligence (IJCAI-95)*. Montreal, Canada.

Manurung, H. (2003). *An evolutionary algorithm approach to poetry generation*. Ph.D. thesis, School of Informatics, University of Edinburgh.

Martin, D. L., Cheyer, A. J., y Moran, D. B. (1999). The Open Agent Architecture: A framework for building distributed software systems. *Applied Artificial Intelligence*, *13*(1-2), 91–128. OAA.

McCoy, K. F., y Strube, M. (1999). Generating anaphoric expressions: Pronoun or definite description? En *ACL workshop on Discourse and Reference Structure*, (pp. 63–71).

Mellish, C., Knott, A., Oberlander, J., y O'Donnell, M. (1998). Experiments using stochastic search for text planning. En E. Hovy (Ed.) *Proceedings of the Ninth International Workshop on Natural Language Generation*, (pp. 98–107). New Brunswick, New Jersey: Association for Computational Linguistics.

Mihalcea, R., y Moldovan, D. (2001). Extended Wordnet: progress report. En *Proceedings of NAACL Workshop on WordNet and Other Lexical Resources*.

Miller, G. (1985). WordNet: A dictionary browser. En *Information in Data, Proceedings of the First Conference of the UW Centre for the New Oxford Dictionary*. University of Waterloo, Canada.

Miller, G. (1986). Dictionaries in the mind. *Language and Cognitive Processes*, *1*, 171–185.

Miller, G., y Fellbaum, C. (1991). Semantic networks of English. *Cognition*, (pp. 197–229).

Miller, G. A. (1995). WordNet: a lexical database for English. *Commun. ACM*, *38*(11), 39–41.

Milosavljevic, M. (1996). Introducing new concepts via comparison: A new look at user modelling in text generation. En *Proceedings of the Fifth International Conference on User Modeling, Doctoral Consortium*, (pp. 228–230).

Milosavljevic, M. (1997a). Augmenting the user's knowledge via comparison. En *Proceedings of the 6th International Conference on User Modelling*. Sardinia, Italy.

Milosavljevic, M. (1997b). Content selection in comparison generation. En W. Hoeppner (Ed.) *6th European Workshop on Natural Language Generation (6th EWNLG)*, (pp. 72–81).

Milosavljevic, M. (1999). *The Automatic Generation of Comparisons in Descriptions of Entities*. Ph.D. thesis, Department of Computing, Macquarie University, Sydney, Australia.

Milosavljevic, M. (2003). Defining comparison. En P. Slezak (Ed.) *Proceedings of the Joint International Conference on Cognitive Science with the Australasian Society for Cognitive Science*. University of New South Wales.

Milosavljevic, M., Tulloch, A., y Dale, R. (1996). Text generation in a dynamic hypertext environment. En *Proceedings of the 19th Australian Computer Science Conference*.

Miyata, T., y Matsumoto, Y. (1998). Natural language generation for legal expert system and visualization of generation process. *Journal of Advanced Computational Intelligence*, *2*.

Mueller, E. (1998). ThoughtTreasure: A natural language/commonsense platform.

Nelson, W., y Kucera, H. (1967). *Computing Analysis of Present-day American English*. Brown University Press, Providence, RI.

Ng, H. T., y Lee, H. B. (1996). Integrating multiple knowledge sources to disambiguate word sense: an exemplar-based approach. En *Proceedings of the 34th annual meeting on Association for Computational Linguistics*, (pp. 40–47). Morristown, NJ, USA: Association for Computational Linguistics.

Niles, I., y Pease, A. (2001). Toward a standard upper ontology. En C. Welty, y B. Smith (Eds.) *Proceedings of the 2nd International Conference on Formal Ontology in Information Systems (FOIS-2001)*. Ogunquit, Maine.

O'Donnell, M., Mellish, C., Oberlander, J., y Knott, A. (2001). ILEX: An architecture for a dynamic hypertext generation system. *Natural Language Engineering*, (pp. 225–250).

Pan, S., y Shaw, J. (2004). SEGUE: A hybrid case-based surface natural language generator. En *Proceedings of the International Workshop on Natural Language Generation (INLG 2004)*, (pp. 130–140).

Papineni, K., Roukos, S., Ward, T., y Zhu, W.-J. (2002). BLEU: a method for automatic evaluation of machine translation. En *ACL '02: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*,

(pp. 311–318). Morristown, NJ, USA: Association for Computational Linguistics.

Passonneau, R. (2006). Measuring agreement on setvalued items (MASI) for semantic and pragmatic annotation. En *Proceedings of the 5th Language Resources and Evaluation Converence (LREC'06)*.

Peinado, F. (2008). DLModel, a tool for dealing with description logics. `http://federicopeinado.com/projects/dlmodel/` [24/2/2009].

Pereira, F., Hervás, R., Gervás, P., y Cardoso, A. (2006). A multiagent text generator with simple rhetorical habilities. En *Proceedings of the AAAI-06 Workshop on Computational Aesthetics: AI Approaches to Beauty and Happiness*. AAAI Press.

Pereira, F. C. (2005). *A Computational Model of Creativity*. Ph.D. thesis, University of Coimbra.

Pereira, F. C. (2007). *Creativity and AI: A Conceptual Blending Approach*. Mouton de Gruyter.

Portet, F., Reiter, E., Hunter, J., y Sripada, S. (2007). Automatic generation of textual summaries from neonatal intensive care data. En *AIME '07: Proceedings of the 11th conference on Artificial Intelligence in Medicine*, (pp. 227–236). Berlin, Heidelberg: Springer-Verlag.

Reape, M., y Mellish, C. (1999). Just what is aggregation anyway? En *Proceedings of the 7th EWNLG*. Toulouse, France.

Reiter, E. (1990). The computational complexity of avoiding conversational implicatures. En *Proceedings of the 28th annual meeting on Association for Computational Linguistics*, (pp. 97–104). Morristown, NJ, USA: Association for Computational Linguistics.

Reiter, E. (1994). Has a consensus NL generation architecture appeared, and is it psycholinguistically plausible? En *Proceedings of the Seventh International Workshop on Natural Language Generation*, (pp. 163–170).

Reiter, E., y Dale, R. (1992). A fast algorithm for the generation of referring expressions. En *Proceedings of the 14th conference on Computational linguistics*. Nantes, France.

Reiter, E., y Dale, R. (2000). *Building Natural Language Generation Systems*. Cambridge University Press.

Reiter, E., y Sripada, S. (2002). Should corpora texts be gold standards for NLG. En *In Proceedings of the Second International Conference on Natural Language Generation*, (pp. 97–104).

Russell, S. (1996). *Inteligencia Artificial: un enfoque moderno*. Prentice - Hall.

Schank, R. (1982). *Dynamic memory; a theory of reminding and learning in computers and people*. Cambridge University Press.

Seco, N. (2005). LightWeight WordNet. `http://eden.dei.uc.pt/~nseco/lwwn.tar.gz` [24/2/2009].

Shieber, S., van Noord, G., Moore, R., y Pereira, F. (1989). A semantic-head-driven generation algorithm for unification-based formalisms. En *Meeting of the Association for Computational Linguistics*, (pp. 7–17).

Siddharthan, A., y Copestake, A. (2007). Evaluating an open domain GRE algorithm on closed domains. En *First NLG Challenge on Attribute Selection for Generating Referring Expressions (ASGRE'07), Proc. of the MT Summit XI Workshop: Using Corpora for Natural Language Generation*. Copenhagen,Denmark.

Singh, P. (2002). The public acquisition of commonsense knowledge. En *Proceedings of AAAI Spring Symposium*. Palo Alto, CA: AAAI.

Swartout, W. (1983). XPLAIN: a system for creating and explaining expert consulting systems. *Artificial Intelligence*, *21*, 285–325.

van Deemter, K., van der Sluis, I., y Gatt, A. (2006). Building a semantically transparent corpus for the generation of referring expressions. En *Proceedings of the 4th International Conference on Natural Language Generation (Special Session on Data Sharing and Evaluation), INLG-06*.

Veale, T. (1995). *Metaphor, Memory and Meaning: Symbolic and Connectionist Issues in Metaphor Interpretation*. PhD Thesis, Dublin City University.

Viethen, J., y Dale, R. (2008). The use spatial relations in referring expressions. En *Proceedings of the 5th International Conference on Natural Language Generation*.

Wooldridge, M., y Jennings, N. (1995). Intelligent agents: Theory and practice. *The Knowledge Engineering Review*, *10*(2), 115–152.

Apéndice A

Publicaciones

En este apéndice se muestran las publicaciones sobre las que se apoya este trabajo.

A.1. Generación de Lenguaje Natural

1. García, C., Hervás, R., Gervás, P.: **"Una arquitectura software para el desarrollo de aplicaciones de generación de lenguaje natural"**. Sociedad Española para el Procesamiento del Lenguaje Natural, Procesamiento de Lenguaje Natural, nº 33, Septiembre de 2004, pp. 111-118. ISSN: 1135-5948.

2. Gervás, P., Díaz-Agudo, B., Peinado, F., Hervás, R.: **"Story Plot Generation based on CBR"**. Journal of Knowledge-Based Systems 18, 4-5: Special issue AI-2004, pp. 235-242. Elsevier Science, 2005. ISSN: 0950-7051.

3. Hassan, S., León, C., Gervás, P., Hervás, R.: **"A Computer Model that Generates Biography-Like Narratives"**. International Joint Workshop on Computational Creativity (IJWCC'07), London, 2007.

4. Hervás, R., Gervás, P.: **"Descripción de Entidades y Generación de Expresiones de Referencia en la Generación Automática de Discurso"**. Sociedad Española para el Procesamiento del Lenguaje Natural, Procesamiento de Lenguaje Natural, nº 41, Septiembre de 2008, pp. 217-224. ISSN: 1135-5948.

5. Dionne, D., de la Puente, S., León, C., Hervás, R., Gervás, P.: **"A Model for Human Readable Instruction Generation Using Level-Based Discourse Planning and Dynamic Inference of Attributes Disambiguation"**. 12th European Workshop on Natural Language Generation (EWNL'09), Marzo de 2009.

A.2. Generación de Expresiones de Referencia

1. Hervás, R., Gervás, P.: **"Uso flexible de soluciones evolutivas para tareas de Generación de Lenguaje Natural"**. Sociedad Española para el Procesamiento del Lenguaje Natural, Procesamiento de Lenguaje Natural, nº 35, Septiembre de 2005, pp. 187-194. ISSN: 1135-5948.

2. Hervás, R., Gervás, P.: **"Agent-based Solutions for Natural Language Generation Tasks"**. XI Conferencia de la Asociación Española para la Inteligencia Artificial (CAEPIA 05), Santiago de Compostela, España, Springer LNAI Series, 2005.

3. Hervás, R., Gervás, P.: **"Case Retrieval Nets for Heuristic Lexicalization in Natural Language Generation"**. En Bento, C., Cardoso, A., Dias, G., eds: Progress in Artificial Intelligence (Proc. of the 12th Portuguese Conference on Artificial Intelligence (EPIA 05)), Covilha, Portugal, Springer-Verlag, LNAI 3808, pp. 55-66, 2005.

4. Hervás, R., Gervás, P.: **"An Evolutionary Approach to Referring Expression Generation and Aggregation"**. (Póster) En Wilcock, G., Jokinen, K., Mellish, C., Reiter, E., eds.: Proceedings of the 10th European Workshop on Natural Language Generation, pp. 168-173, Aberdeen, Scotland, 8-10 August, 2005.

5. Hervás, R., Gervás, P.: **"Case-Based Reasoning for Knowledge-Intensive Template Selection During Text Generation"**. 8th European Conference on Case-Based Reasoning (ECCBR 06), Fethiye, Turkey, Springer-Verlag, LNAI 4106, September 2006.

6. Francisco, V., Gervás, P., Hervás, R.: **"Automatic Knowledge Acquisition in Case-Based Text Generation"**. Proceedings of the ECCBR 06 Workshop on Textual Case-Based Reasoning, Turquía, pp. 68-77, Septiembre de 2006.

7. Francisco, V., Gervás, P., Hervás, R.: **"Dependency Analysis and CBR to Bridge the Generation Gap in Template-Based NLG"**. Computational Linguistics and Intelligent Text Processing (CICLing 2007), Méjico, Springer-Verlag, LNCS 4394, pp. 432-443, 2007.

8. Hervás, R., Gervás, P.: **"Degree of Abstraction in Referring Expression Generation and its Relation with the Construction of the Contrast Set"**. (Póster) En Proc. of the Fifth International Natural Language Generation Conference (INLG'08), pp. 161-164, Ohio, USA, Junio 2008.

A.3. Figuras Retóricas

1. Pereira, F.C., Hervás, R., Gervás, P., Cardoso, A.: **"A Multiagent Text Generator with Simple Rhetorical Habilities"**. In the AAAI-06 Workshop on Computational Aesthetics: AI Approaches to Beauty & Happiness, July 2006.

2. Hervás, R., Pereira, F.C., Gervás, P., Cardoso, A.: **"Cross-Domain Analogy in Automated Text Generation"**. Proceedings of the 3rd Joint Workshop on Computational Creativity, Italia, Agosto 2006.

3. Hervás, R., Pereira, F.C., Gervás, P., Cardoso, A.: **"A Text Generation System that Uses Simple Rhetorical Figures"**. Sociedad Española para el Procesamiento del Lenguaje Natural, Procesamiento de Lenguaje Natural, nº 37, Septiembre de 2006, pp. 199-206. ISSN: 1135-5948.

4. Hervás, R., Robinson, J., Gervás, P.: **"Evolutionary Assistance in Alliteration and Allelic Drivel"**. En Giacobini, M. et al., eds: Applications of Evolutionary Computing (Proc. of EvoMusArt 2007, EvoWorkshops 2007, dentro de Evo* 2007 (conferencia múltiple en el campo de la Programación Evolutiva)), Valencia, España, Springer-Verlag, LNCS 4448, pp. 537-546, 2007. ISSN: 0302-9743.

5. Hervás, R., Costa, R., Costa, H., Gervás, P., Pereira, F.C.: **"Enrichment of Automatically Generated Texts using Metaphor"**. En A. Gelbukh, A.F. Kuri Morales, eds: MICAI-07: Advances in Artifical Intelligence (Proc. of 6th Mexican International Conference on Artificial Intelligence (MICAI-07)), Méjico, Springer-Verlag, LNAI 4827, pp. 944-954, 2007. ISSN: 0302-9743.

A.4. Tareas Competitivas para la Generación de Expresiones de Referencia

1. Hervás, R., Gervás, P.: **"NIL: Attribute Selection for Matching the Task Corpus Using Relative Attribute Groupings Obtained from the Test Data"**. First NLG Challenge on Attribute Selection for Generating Referring Expressions (ASGRE), UCNLG+MT Workshop, Machine Translation Summit XI, Copenhague, 2007.

2. Gervás, P., Hervás, R., León, C.: **"NIL-UCM: Most-Frequent-Value-First Attribute Selection and Best-Scoring-Choice Realization"**. Referring Expression Generation Challenge 2008, Proc. of the 5th International Natural Language Generation Conference (INLG '08), Ohio, USA, Junio de 2008.

3. Hervás R, Gervás P.: **"Evolutionary and Case-Based Approaches to REG: NIL-UCM-EvoTAP, NIL-UCM-ValuesCBR and NIL-UCM-EvoCBR"**. Generation Challenges 2009, Proc. of the 12th European Workshop on Natural Language Generation (ENLG '09), Atenas, Grecia, Abril de 2009.

Apéndice B

Características del Corpus TUNA

En este apéndice se describe el formato XML utilizado en los datos del corpus TUNA. Cada archivo de los datos consiste en un único ejemplo del corpus, es decir, un único par formado por un dominio (la representación de las entidades y sus propiedades) y una descripción que describe una entidad en el dominio (el referente objetivo). Estos datos son el resultado de un experimento en el que se pidió los participantes que describieran entidades dominios visuales. Los dominios se mostraban en pantalla y consistían en las imágenes de siete entidades en una tabla de tamaño 3 (fila) x 5 (columna). Para cada ejemplo los participantes tenían que escribir una descripción de la entidad objetivo usando las características de la misma según lo observado en la imagen.

Este corpus ha sido utilizado para evaluar algunas de las soluciones presentadas para la generación de expresiones de referencia básica en el marco de las tareas de evaluación competitiva comentadas en la sección 2.3.3.

B.1. Descripción del Formato de las Instancias del Corpus

El formato básico de las instancias del corpus se muestra en la Figura B.1.

B.1.1. El Nodo `TRIAL`

El nodo TRIAL empareja dominios y descripciones. Este nodo contiene los siguientes atributos XML:

1. `ID:` Identificador único de la instancia dentro del corpus.

2. `CONDITION:` Este atributo toma uno de dos valores: ±LOC. El valor de `CONDITION` representa la condición experimental en la que la

```
<TRIAL CONDITION=''+/-LOC'' ID=''sNtM''>

    <DOMAIN>
        representation of entities and their properties
    </DOMAIN>

    <WORD-STRING>
        the string describing the target referent in the domain
    </WORD-STRING>

    <ANNOTATED-WORD-STRING>
        the string in WORD-STRING, where the relevant substrings are annotated
        with attributes in ATTRIBUTE-SET
    </ANNOTATED-WORD-STRING>

    <ATTRIBUTE-SET>
        the set of domain attributes included in the description
    </ATTRIBUTE-SET>

</TRIAL>
```

Figura B.1: Formato de las instancias del corpus

instancia fue creada. Si la condición es +LOC, es que se dijo a los participantes que podían referirse a las entidades usando cualquiera de sus propiedades, incluyendo su localización. Si la condición es -LOC, se les sugirió que no usaran la localización, aunque no se les prohibió. Esto significa que mientras que es más posible encontrar expresiones de lugar en descripciones con la condición +LOC, no siempre será así.

B.1.2. El Nodo `DOMAIN`

En este nodo se representan las entidades de un dominio. Cada dominio consiste exactamente en siete entidades únicas (nodos `ENTITY`), cada uno representado en términos de sus atributos (nodos `ATTRIBUTE`). Las entidades se dividen en un objetivo y seis distractores. El objetivo es el referente para el que se está buscando la descripción, siempre teniendo en cuenta los distractores del dominio que le acompañan.

El nodo `ENTITY` contiene los siguientes atributos XML:

1. `ID:` Un identificador entero único.
2. `IMAGE:` El nombre del fichero que contiene la foto o dibujo de esta entidad.
3. `TYPE:` Especifica si la entidad es objetivo o distractor en esta instancia concreta del corpus.

Los nodos `ATTRIBUTE` representan las propiedades de una entidad usando pares atributo-valor. En la Tabla B.2 se pueden ver todos los atributos y valores considerados en el corpus para los dos dominios. Cada nodo `ATTRIBUTE` contiene los siguientes atributos XML:

1. `NAME:` El nombre de los atributos, como por ejemplo *colour* o *size*.

Attributes in the furniture/household domain		
Attribute	**Type**	**Possible values**
TYPE	literal	*chair, sofa, desk, fan*
COLOUR	literal	*blue, red, green, grey*
SIZE	literal	*large, small*
ORIENTATION	literal	*left, right, front, back*
X-DIMENSION (column number)	gradable	1, 2, 3, 4, 5
Y-DIMENSION (row number)	gradable	1, 2, 3
OTHER		
Attributes in the people domain		
Attribute	**Type**	**Possible values**
TYPE	literal	*person*
ORIENTATION	literal	*front, left, right*
AGE	literal	*young, old*
HAIRCOLOUR	literal	*dark, light, other*
HASSHIRT	boolean	0 (false), 1 (true)
HASBEARD	boolean	0 (false), 1 (true)
HASHAIR	boolean	0 (false), 1 (true)
HASGLASSES	boolean	0 (false), 1 (true)
HASTIE	boolean	0 (false), 1 (true)
HASSUIT	boolean	0 (false), 1 (true)
X-DIMENSION (column number)	gradable	1, 2, 3, 4, 5
Y-DIMENSION (row number)	gradable	1, 2, 3
OTHER		

Figura B.2: Atributos y valores usados en el corpus TUNA

2. `VALUE:` El valor de los atributos, como por ejemplo *red* o *large*.

3. `TYPE:` El tipo del valor. Este atributo puede ser literal (cuando el valor es un elemento del conjunto de posibles valores), booleano (cuando el valor es 1 o 0) o gradual, significando que el atributo toma un valor numérico. De este último tipo sólo existen dos atributos: *x-dimension* e *y-dimension*, que corresponden a la columna (X) y fila (Y) de la entidad en la cuadrícula en la que se divide el dominio.

B.1.3. El Nodo `WORD-STRING`

Ésta es la descripción de la entidad objetivo tal y como fue escrita por participantes humanos en el experimento TUNA, sin ninguna anotación adicional.

B.1.4. El Nodo `ATTRIBUTE-SET`

El `ATTRIBUTE-SET` es el conjunto de `ATTRIBUTEs` para las que las descripciones humanas contienen palabras o frases equivalentes. Las descripciones humanas fueron anotadas para identificar subcadenas que correspondían a la

realización de atributos del dominio. Por ejemplo, la descripción *a bearded man* corresponde al conjunto de atributos {*HASBEARD: 1, TYPE: person*} . Se puede considerar que el `ATTRIBUTE-SET` representa el contenido semántico del correspondiente `WORD-STRING`.

Ocasionalmente un `ATTRIBUTE-SET` puede contener el mismo `ATTRIBUTE` dos veces (por ejemplo si son dos nodos con el mismo `NAME` y `VALUE` pero diferente `ID`). Esto puede suceder cuando una descripción contiene información repetida. Por ejemplo, en *the white-haired man with a white beard* existen dos instancias de *hairColour*, una para el pelo del hombre, y otra para el color de su barba. En este caso tenemos diferentes instancias del mismo atributo.

La representación semántica de `ATTRIBUTE-SET` es plana en el sentido de que no refleja la estructura sintáctica de una descripción o la manera en que los atributos son realizados en relación los demás. Para información sobre estos aspectos se proporciona la representación `ANNOTATED-WORD-STRING` que se describe a continuación.

B.1.5. El Nodo `ANNOTATED-WORD-STRING`

La `ANNOTATED-WORD-STRING` contiene el mismo texto que el nodo `WORD-STRING`, pero con las subcadenas anotadas de manera adicional representando los atributos a los que corresponden. La idea es relacionar partes del texto con partes relevantes de la representación de la entidad objetivo. Así esta notación indica cómo los atributos individuales se corresponden con trozos del texto.

El conjunto de atributos en `ANNOTATED-WORD-STRING` es exactamente el mismo que el de `ATTRIBUTE-SET`, exceptuando algún `META-ATTRIBUTE` adicional. Otra diferencia es que en `ANNOTATED-WORD-STRING` los atributos relacionados con el pelo pueden ir anidados. Veamos una descripción más detallada de estas dos cuestiones:

1. `META-ATTRIBUTE:` Este elemento se usa para expresiones de lugar. `META-ATTRIBUTE` engloba la parte del texto relevante respecto al lugar (por ejemplo *in the centre of the screen* y tiene como nodos hijo o bien un atributo *x-dimension*, un atributo *y-dimension*, o ambos (tendrá ambos cuando la expresión es una combinación de las dimensiones vertical (Y) y horizontal (X)). Por ejemplo, la expresión *in the centre of the screen*, donde el centro se refiere tanto a columnas como filas, se anotaría como sigue:

```
<META-ATTRIBUTE NAME="location">
<ATTRIBUTE ID="..." NAME="x-dimension" VALUE="3" />
<ATTRIBUTE ID="..." NAME="y-dimension" VALUE="2" />
in the centre of the screen
```

```
</META-ATTIRIBUTE>
```

2. **Anidamiento:** Un nodo `ATTRIBUTE` *hairColour* es siempre hijo de un nodo `ATTRIBUTE` *hasHair*. Esto se debe a que el uso del color del pelo siempre conlleva que *hasHair* tiene valor 1. Por ejemplo, la expresión *white-haired* se anotaría como sigue:

```
<ATTRIBUTE ID="..." NAME="hasHair" VALUE="1">
<ATTRIBUTE ID="..." NAME="hairColour" VALUE="light" />
white-
</ATTRIBUTE>
haired
</ATTIRIBUTE>
```

   Hay que tener en cuenta, sin embargo, que el atributo *hasHair* sí puede aparece sin el atributo hijo *hairColour*.

B.1.6. `ATTRIBUTE`s que no aparecen en la representación `DOMAIN`

En ocasiones los participantes en el experimento usaron propiedades para referirse a los objetos que no estaban representadas en `DOMAIN`, como por ejemplo sería la propiedad *broad-shouldered*. Estos casos fueron anotados como sigue:

```
<ATTRIBUTE ID="..." NAME="other" VALUE="other">
broad-shouldered
</ATTRIBUTE>
```

Apéndice C

Conocimiento General de los Dominios de Sapper

En este apéndice se muestra una pequeña parte del conocimiento extraído del sistema Sapper que ha sido utilizado para el trabajo presentado. La información se presenta en formato Prolog, aunque existe también una traducción a hechos individuales que es la que ha sido usada realmente.

```
%
% A_GALAXY_FAR_FAR_AWAY

:- defconcept(a_galaxy_far_far_away,

[[isa, 0.5, place, galaxy]]).

%
% A_LONG_TIME_AGO

:- defconcept(a_long_time_ago,

[[isa, 0.5, time_period]]).

%
% ADVISE_MERLIN_ARTHUR

:- defconcept(advise_merlin_arthur,

[[isa, 0.5, event],
 [predicate, 0.5, help, advise],
 [agent, 0.5, merlin],
 [patient, 0.5, king_arthur]]).
```

```
%
% ADVISE_OBI_WAN_LUKE

:- defconcept(advise_obi_wan_luke,

[[isa, 0.5, event],
 [predicate, 0.5, help, advise],
 [agent, 0.5, obi_wan_kenobi],
 [patient, 0.5, luke_skywalker]]).

%
% ARTOO_DEETOO

:- defconcept(artoo_deetoo,

[[isa, 0.5, droid, robot],
 [attr, 0.5, round, inarticulate, short]]).

%
% ATTACK_REBEL_ALLIANCE_DEATH_STAR

:- defconcept(attack_rebel_alliance_death_star,

[[isa, 0.5, event],
 [predicate, 0.5, attack],
 [agent, 0.5, han_solo, luke_skywalker, rebel_alliance],
 [instr, 0.5, x_wing_fighter],
 [target, 0.5, death_star]]).

%
% BECOME_ARTHUR_KING

:- defconcept(become_arthur_king,

[[isa, 0.5, event],
 [predicate, 0.5, assume_position, assume_power, ascend_throne],
 [agent, 0.5, king_arthur],
 [patient, 0.5, albion]]).

%
% BLOOD

:- defconcept(blood,

[[isa, 0.5, body_fluid, liquid],
 [attr, 0.5, organic, 0.4, viscous, 0.5, red]]).
```

```
%
% BRAIN

:- defconcept(brain,

[[isa, 0.5, organ],
 [control, 0.5, mind, rationality, thought, intelligence],
 [attr, 0.5, important, central]]).

%
% BRAIN_SURGEON

:- defconcept(brain_surgeon,

[[isa, 0.5, profession, doctor, person, surgeon],
 [attr, 0.1, blood, 0.9, medical_school, 0.85, educated, 0.9, precise,
   0.8, skilful, clean, 0.6, influential, 0.75, well_paid,
   0.5, operating_theatre],
 [affect, 0.5, life, 0.8, patient, 0.05, corpse, 0.7, human_flesh, mind,
   0.9, brain],
 [perform, 0.5, hypocratic_oath, pre_op, 0.9, surgery],
 [control, 0.5, disinfectant, scalpel],
 [part, 0.5, surgical_glove, surgical_mask, white_smock]]).

%
% BURGLARY

:- defconcept(burglary,

[[isa, 0.5, crime, activity],
 [attr, 0.5, dangerous, illegal],
 [part, 0.5, lock_pick]]).

%
% BUSINESS

:- defconcept(business,

[[isa, 0.5, social_act],
 [depend, 0.5, currency, business_skills, communication_skills,
   social_convention]]).

%
% BUTCHER

:- defconcept(butcher,
```

```
[[isa, 0.5, profession, person],
 [attr, 0.8, blood, -0.5, educated, -0.2, well_paid, 0.1, skilful,
   -0.6, precise, 0.5, abatoire, -0.8, clean],
 [affect, 0.5, livestock, 0.8, carcass, 0.5, meat],
 [perform, 0.5, slaughter],
 [control, 0.5, livestock, cleaver],
 [part, 0.5, white_apron]]).

%
% CAMELOT

:- defconcept(camelot,

[[isa, 0.5, castle, place],
 [attr, 0.5, albion, white],
 [part, 0.5, round_table]]).

%
% CHEWBACCA

:- defconcept(chewbacca,

[[isa, 0.5, alien, wookie, creature],
   [attr, 0.5, inarticulate, hairy, strong, tall],
 [perform, 0.5, flying]]).

%
% CLEAVER

:- defconcept(cleaver,

[[isa, 0.5, instrument],
 [attr, 0.3, sharp, 0.2, clean, 0.6, heavy, 0.7, dangerous,
   0.8, blood]]).

%
% COMPUTER

:- defconcept(computer,

[[isa, 0.5, device],
 [part, 0.5, monitor, cpu, keyboard]]).

%
% CORPSE

:- defconcept(corpse,

[[isa, 0.5, body, person],
```

```
 [part, 0.5, torso, arm, leg, head],
 [attr,  0.95, dead, -0.9, healthy, pleasant]]).

%
% CRIMINAL

:- defconcept(criminal,

[[isa, 0.5, vocation],
 [attr, 0.5, dextrous, skilful, illegal, unethical, dishonest],
 [control, 0.5, grapevine, outlaw_gang, lock_pick, thermal_lance],
 [part, 0.5, black_glove, stocking_mask],
 [affect,  -0.7, police_force, victim, bank, door_lock, wall_safe],
 [perform, 0.5, bank_raid, burglary]]).


%
% DARTH_VADER

:- defconcept(darth_vader,

[[isa, 0.5, magician, jedi_knight, person, man],
 [control, 0.5, death_star, the_force, light_saber],
 [attr, 0.5, evil, heavy_breath, black],
 [part, 0.5, mask, cloak],
 [disconnect, 0.5, princess_leia]]).

%
% DEATH_STAR

:- defconcept(death_star,

[[isa, 0.5, military_installation, fortress, space_station, place],
 [attr, 0.5, powerful, evil]]).

%
% EXCALIBUR

:- defconcept(excalibur,

[[isa, 0.5, weapon, sword],
 [attr, 0.5, hand_held, narrow, long, magical, powerful]]).

%
% MILITARY_GENERAL

:- defconcept(military_general,

[[isa, 0.5, soldier, person],
```

```
 [prototype, george_patten, napoleon],
 [attr, 0.6, old, 0.7, influential, control, powerful,
   0.8, intelligent, educated, military_school, warlike, arrogant,
   agressive, battlefield],
 [control, 0.5, army, nerve_gas, atomic_bomb, command_centre,
   snub_fighter, bomber_plane, soldier],
 [depend, 0.5, army],
 [create, 0.5, military_propaganda, plan],
 [affect, 0.5, army, -0.7, enemy_army, enemy_soldier, 0.8, soldier],
 [perform, 0.5, bombing_raid, 0.1, cavalry_charge],
 [part, 0.5, military_uniform]]).

%
% GUINNEVERE

:- defconcept(guinnevere,

[[isa, 0.5, royal_personage, queen, woman, person],
 [attr, 0.5, young, popular, beautiful]]).

%
% HACKER

:- defconcept(hacker,

[[isa, 0.5, vocation, person],
 [attr, 0.5, suspicious, -0.8, social, 0.8, intelligent, 0.5, unfit,
   pale, freaky],
 [control, 0.5, logic_probe, internet, game_program, computer],
 [affect, 0.5, internet],
 [create, 0.5, game_program],
 [perform, 0.5, hacking, programming],
 [part, 0.5, blue_jeans_and_sneakers]]).

%
% HAN_SOLO

:- defconcept(han_solo,

[[isa, 0.5, captain, pilot, pirate, man, person],
 [attr, 0.5, cynical, skilful, brave, handsome, young],
 [control, 0.5, millenium_falcon],
 [perform, 0.5, flying],
 [connect, 0.5, chewbacca]]).

%
% HYPOCRATIC_OATH

:- defconcept(hypocratic_oath,
```

```
[[isa, 0.5, promise, oath],
 [attr, 0.5, professional, ethical]]).

%
% KING_ARTHUR

:- defconcept(king_arthur,

[[isa, 0.5, hero, farmboy, king, man, person],
 [attr, 0.5, young, popular, humble, brave, handsome],
 [part, 0.5, suit_of_armour],
 [control, 0.5, excalibur],
 [depend, 0.5, excalibur],
 [connect, 0.5, merlin, lancelot, guinnevere],
 [disconnect, 0.5, morgana_lefay],
 [up, 0.5, uther_pendragon]]).

%
% KNIGHT

:- defconcept(knight,

[[isa, 0.5, warrior, person, man],
 [control, 0.5, horse, sword]]).

%
% KNIGHT_OF_ROUND_TABLE

:- defconcept(knight_of_round_table,

[[isa, 0.5, warrior, knight, person, man],
 [control, 0.5, horse, sword]]).

%
% LANCELOT

:- defconcept(lancelot,

[[isa, 0.5, champion_of_king, knight, knight_of_round_table,
   man, person],
 [attr, 0.5, popular, skilful, brave, handsome, young],
 [control, 0.5, horse, sword],
 [part, 0.5, suit_of_armour],
 [perform, 0.5, jousting]]).

%
% LIGHT_SABER
```

```
:- defconcept(light_saber,

[[isa, 0.5, weapon],
 [attr, 0.5, hand_held, narrow, long]]).

%
% LIVESTOCK

:- defconcept(livestock,

[[attr,  -0.8, intelligent],
 [isa, 0.5, collection],
 [part, 0.5, carcass, cow]]).

%
% LOVE_LUKE_PRINCESS_LEIA

:- defconcept(love_luke_princess_leia,

[[isa, 0.5, event],
 [predicate, 0.5, love],
 [agent, 0.5, luke_skywalker],
 [patient, 0.5, princess_leia]]).

%
% LUKE_SKYWALKER

:- defconcept(luke_skywalker,

[[isa, 0.5, farmboy, hero, man],
 [control, 0.5, x_wing_fighter, fathers_light_saber],
 [connect, 0.5, artoo_deetoo, see_threepio, chewbacca, han_solo,
   princess_leia, obi_wan_kenobi],
 [disconnect, 0.5, the_emporer, darth_vader]]).

%
% MAGICIAN

:- defconcept(magician,

[[isa, 0.5, vocation],
 [perform, 0.5, jug_trick, illusion, magic],
 [depend, 0.5, audience],
 [part, 0.5, formal_glove, black_tuxedo],
 [attr, 0.5, nightclub, charming, dapper],
 [control, 0.5, audience, magic_wand, stage_assistant]]).

%
% MAN
```

```
:- defconcept(man,

[[isa, 0.5, person],
 [substance, 0.5, human_bone, human_flesh],
 [part, 0.5, human_hair, human_skin, hand, ear, nose, eye, face]]).

%
% MERLIN

:- defconcept(merlin,

[[isa, 0.5, magician, man, person],
 [attr, 0.5, good, powerful, religious, mystical, wise, old],
 [perform, 0.5, counsel, magic],
 [control, 0.95, magic, 0.6, magic_wand]]).

%
% MILLENIUM_FALCON

:- defconcept(millenium_falcon,

[[isa, 0.5, space_ship],
 [attr, 0.5, rebel_alliance]]).

%
% OBI_WAN_KENOBI

:- defconcept(obi_wan_kenobi,

[[isa, 0.5, magician, jedi_knight, person, man],
 [attr, 0.5, good, powerful, wise, old],
 [control, 0.95, the_force, 0.5, light_saber],
 [disconnect, 0.5, the_emporer, darth_vader]]).

%
% OPERATING_THEATRE

:- defconcept(operating_theatre,

[[isa, 0.5, place, theatre],
 [part,  0.5, operating_table, scalpel],
 [attr,  0.8, clean, sterile, 0.7, bright]]).

%
% POLICEMAN

:- defconcept(policeman,
```

```
[[isa, 0.5, element, person],
 [attr, 0.5, honest, ethical, legal]]).

%
% PRECISE

:- defconcept(precise,

[[isa, 0.5, accuracy]]).

%
% PRINCESS_LEIA

:- defconcept(princess_leia,

[[isa, 0.5, royal_personage, princess, person, woman],
 [attr, 0.5, independent, beautiful, feisty, young],
 [control, 0.5, rebel_alliance]]).

%
% REBEL_ALLIANCE

:- defconcept(rebel_alliance,

[[isa, 0.5, collection, alliance, military_force],
 [part, 0.5, space_rebel],
 [mode, 0.5, sympathetic, heroic, good]]).

%
% SCALPEL

:- defconcept(scalpel,

[[isa, 0.5, instrument],
 [attr, 0.8, sharp, surgical, clinical, clean, sterile, 0.5, narrow,
   0.3, dangerous, 0.4, blood, 0.8, precise, -0.2, heavy]]).

%
% SCIENTIST

:- defconcept(scientist,

[[isa, 0.5, vocation, profession],
 [create, 0.5, science_presentation, scientific_law, scientific_theory,
   mathematical_model, scientific_literature],
 [depend, 0.5, scientific_law, -0.5, god, 0.5, scientific_literature],
 [perform, 0.5, science_presentation, scientific_method, experiment],
 [attr, 0.8, progressive, 0.3, opinionated, 0.5, intelligent, educated,
   objective, rational],
```

```
 [affect, 0.5, laboratory_rat, 0.3, society],
 [part, 0.5, surgical_glove, white_smock],
 [control, 0.5, laboratory_assistant]]).

%
% STORM_TROOPER

:- defconcept(storm_trooper,

[[isa, 0.5, warrior, man, person],
 [attr, 0.5, evil]]).

%
% SURGEON

:- defconcept(surgeon,

[[isa, 0.5, profession, doctor, person],
 [depend,  -0.7, cardiac_arrest, 0.5, medical_textbook],
 [attr, 0.7, intelligent, 0.85, educated, 0.9, precise, 0.6, dextrous,
   0.8, skilful, clean, careful, 0.6, influential, 0.75, well_paid,
   0.5, operating_theatre, 0.8, delicate, 0.25, blood, 0.9, medical_school],
 [affect,  -0.7, cancer, 0.5, life, 0.05, corpse, 0.8, patient,
   0.5, circulatory_system, 0.7, human_flesh],
 [perform, 0.5, hypocratic_oath, pre_op, 0.9, surgery],
 [control, 0.5, circulatory_system, radiation_therapy, disinfectant,
   scalpel],
 [part, 0.5, surgical_glove, surgical_mask, white_smock]]).

%
% SURGERY

:- defconcept(surgery,

[[isa, 0.5, activity, operation],
 [attr, 0.5, surgical, clinical],
 [part, 0.6, preparation, 0.2, blood, 0.5, anaesthetic, operating_table,
   pre_op, scalpel],
 [affect,  0.05, corpse, 0.8, patient],
 [effect,  0.05, dead, 0.2, blood, 0.5, worry, fear, discomfort, pain]]).

%
% SWORD

:- defconcept(sword,

[[isa, 0.5, weapon, instrument],
 [attr, 0.5, blood, sharp, narrow, 0.3, heavy, 0.7, dangerous,
   0.5, hand_held, long],
```

```
 [substane, 0.5, steel]]).

%
% THE_EMPEROR

:- defconcept(the_emperor,

[[isa, 0.5, magician, jedi_knight, person, man],
 [attr, 0.5, evil, powerful],
 [control, 0.5, darth_vader, the_empire, the_force],
 [disconnect, 0.5, obi_wan_kenobi, princess_leia]]).

%
% THE_EMPIRE

:- defconcept(the_empire,

[[isa, 0.5, collection],
 [mode, 0.5, belligerant, evil],
 [part, 0.5, storm_trooper]]).

%
% THE_FORCE

:- defconcept(the_force,

[[isa, 0.5, power],
 [attr, 0.5, arcane]]).

%
% WEAPON

:- defconcept(weapon,

[[isa, 0.5, instrument],
 [attr, 0.8, dangerous]]).

%
% X_WING_FIGHTER

:- defconcept(x_wing_fighter,

[[isa, 0.5, vehicle, aircraft, space_ship],
 [attr, 0.5, rebel_alliance]]).
```