Creative Storytelling Based on Exploration and Transformation of Constraint Rules*

Carlos León[1] and Pablo Gervás[2]

Departamento de Ingeniería del Software e Inteligencia Artificial
Universidad Complutense de Madrid
[1]cleon@fdi.ucm.es, [2]pgervas@sip.ucm.es

Abstract. It is said in Boden's work that transformational creativity is the highest form of creativity. Whether this claim is true or not can be discussed, but, in principle, *transforming* the conceptual space should lead creativity processes to new areas not covered simply by exploratory creativity. In this paper Cast is presented, a new automatic storytelling system which has the ability of transforming this conceptual space by performing a state space search generating facts beyond the expected set of valid solutions and modifying the constraints for the generative process. Why this system can be considered creative is discussed, and example results from the first prototype are shown.

Keywords: Computational Creativity, Transformational Creativity, Cast, Storytelling.

1 Introduction

Boden's work considers two types of creativity regarding *conceptual space*, (the abstract location of the entities produced by creative processes): *exploratory* (exploring or searching the conceptual space) and *transformational* (changing or transforming the conceptual space) [1,2]. It is said that transformational creativity is the "highest form", that is, for a computer system to be fully creative, it has to *transform* the conceptual space.

There is not much consensus about these ideas: Bundy, in [3], explains that Boden's model does not cover all possible creative productions, and Ritchie, in [4], trying to obtain a clear definition of transformational creativity, concludes that terms used to explain what it is are still too imprecisely defined to claim that a process or an artifact has been creative in terms of high level of creativity. However, many researchers in the field agree that, even without precisely, empirically knowing the definition of creativity in general and transformational creativity in particular does not prevent us from building creative systems: those

* This research is funded by the Ministerio de Educación y Ciencia (TIN2006-14433-C02-01), Universidad Complutense de Madrid, Dirección General de Universidades e Investigación de la Comunidad de Madrid (CCG07-UCM/TIC 2803) and a grant from Caja de Burgos, Jóvenes Excelentes 2008 contest.

that *would be deemed as creative if found in humans* (as claimed, for example, by Wiggins in [5]).

Against this background, and related with Computational Creativity, automatic story generation programs follow a long tradition of creative systems, like TALE-SPIN, MINSTREL or MEXICA[6,7,8]. Typically, these systems are built with exploratory creativity processes. In order to explore new approaches to applied Computational Creativity in general and automatic story telling in particular, a story teller called CAST (*Creative Automatic Story Teller*), which tries to modify the conceptual space, is presented. Transformational creativity processes which are implemented by CAST, although not totally directed, are heavily influenced by theoretical models by Sharples ([9]), and also in previous story telling systems like MEXICA [8], KIIDS [10] and FABULIST [11].

Even considering Boden's account for creativity, nothing guarantees, in all cases, that the final quality of the generated stories will be higher than quality obtained by applying just exploratory creativity. Creative systems can be extremely complex, and we have not defined measurements for aesthetics in evaluating creative stories. However, expanding the conceptual space by modifying it opens new possibilities for generation, so possibly a new creative object (a new story, in this case), following rules not previously conceived by the system creator, could be generated. CAST uses both transformational and exploratory processes, so it should benefit from both approaches. This system does not claim to create better stories just by applying transformation. The present paper shows a way of changing the search space which is implemented in CAST, showing that it is possible to transform the constraints to get new information. Whether this is what Boden defines as transformational creativity or not is arguable, given that the definition is far from being precise enough. CAST just tries to show that transformation is *possible*, and *useful*. We will see, in the next sections, how this is done and the preliminary results that have been obtained.

2 Previous Work

Storytelling is crucial to understand the way humans perceive the world. The task of conceiving and composing the story that is to be told is a fundamental ingredient of storytelling. There are several approaches to automatic storytelling. The earliest attempt at addressing explicitly issues of creative activity in the context of storytelling is the development of MINSTREL [7], which tells stories about King Arthur and his Knights of the Round Table . BRUTUS [12] is a program that writes short stories about betrayal. It has a complex architecture designed to take into account knowledge about literature, but it addresses very few of the issues raised around creativity either from the point of view of writing as a cognitive activity or from the point of view of computation. MEXICA [8] implements a computer model designed to study the creative process in writing in terms of the cycle of engagement and reflection. MEXICA is a flexible tool where the user can set the value of different parameters to constrain the writing process and explore different aspects of story generation. It takes into account emotional

links and tensions between the characters as means for driving and evaluating ongoing stories. MEXICA is actually built based on Sharples' account of writing as a cycle of engagement and reflection phases. However, not all storytelling or narrative/linguistic related systems are directly based on the study of creativity. Works like FABULIST [11], TALE-SPIN [6] or UNIVERSE [13] in the narrative field, or [14] in poetry are some other examples of storytelling which follow other narrative models.

Wiggins [5] takes up Boden's idea of creativity as search over conceptual spaces and presents a more detailed theoretical framework that attempts to clarify the issue of whether creativity can be reduced to good old fashioned AI search (GOFAI). By specifying formally the different elements involved (the universe of possible concepts, the rules that define a particular subset of that universe as a conceptual space, the rules for traversing that conceptual space) Wiggins points out that the rules for traversing a conceptual space may lead to elements in the universe but outside the definition of the conceptual space. This kind of behavior would be considered a design error in an AI system, but one must keep in mind that his formal framework is intended as a characterization of the various elements that may be at play in a creative situation.

Sharples [9] presents a description of writing understood as a problem-solving process where the writer is both a creative thinker and a designer of text. Sharples proposes a cyclic process moving through two different phases: engagement and reflection. During the engagement phase the constraints are taken as given and the conceptual space defined by them is simply explored, progressively generating new material. During the reflection phase, the generated material is revised and constraints may be transformed as a result of this revision. Sharples also define two concepts which are relevant for this article: the *universe*, which is a set containing all possible stories, and the *conceptual space*, which is the result of restricting the set of coherent or valid stories. This *conceptual space* can be obtained by applying known *schemas*, or know rules about a given domain (for instance, knowing what is the process for buying a newspaper).

3 Cast: Storytelling Based on Exploration and Transformation

In this section, CAST is presented. The explanation is divided in three parts: The first one explains how the knowledge of the system is represented and managed (Section 3.1), the second one explains how the exploration is performed and what can be obtained by plain exploration in the toy domain that has been created (Section 3.2), and the third one shows how transforming constraints for the generation can lead the system to new results, not obtainable by just exploration (Section 3.3).

3.1 Representing Knowledge: Schemas and Facts

Basic knowledge in CAST is defined as a big set of logic predicates, in which the name of the predicate is a property which links two objects, in the form

$chase(policeman, criminal), want(criminal, money)$. It is also possible to define these facts with instantiation of logic variables: Instead of defining each possible group of $property(object, value)$, we can define facts with logic rules. For a very simplistic instance: $chase(?x, ?y) \leftarrow works(?x, policestation) \wedge steals(?y, money)$.

Constraints in CAST are stored in pieces of information called *schemas*. Schemas in CAST are sets of rules which define particular knowledge about something in the domain. For instance, to create a constraint which sets that a policeman can not be a friend of a criminal, we could write a small logic constraint, as a schema, like $friend(?x, ?y) \wedge \neg chase(?x, ?y)$. The symbols with a ? mark are logic variables, and can be bound to literals like *policeman* or *criminal*.

3.2 Creating the Story by Exploring the Conceptual Space

To explore the conceptual space, CAST performs a state space search in which a partial story is completed by the addition of new facts during the exploration. We have a knowledge base $\langle K, C \rangle$, where K is the set of logic facts, and C the set of schemas (in the form explained in Section 3.1), and a set with user restrictions in the form of logic rules for the story generated which we call I. Both sets have to be created by external systems, by hand or by analyzing texts, for instance. In these example they have been created by hand for explanation purposes. K stores a set of logic facts about a toy domain where there is a policeman, a criminal and a warehouse, and constraints in C are basic contraints about the world. We also have a partial initial story s_0, which is the empty set of facts.

Each new state in the search adds a subset σ_K of K, which will contain several facts. To choose this subset, we have to define a function $\varphi(s, K)$ which will return the new subset to be added to s, the partial story. There are virtually infinite possible definitions of $\varphi(s_i, K)$, so, for the sake of simplicity, we define it as a function which returns a random set of facts from K which were not present in s_i. Obviously, this definition is far from being the best possible one, but it will serve for our purpose of showing how transformation could improve the generation of a new story. In Section 4 how choosing a best $\varphi(s_i, K)$ could improve the story generation is discussed. The system, on each step, will apply the next operator:

$$s_{n+1} = s_n \cup \varphi(s_n, K) \tag{1}$$

Then the search will start generating different partial stories. On each step (each partial story s_i), before expanding the children nodes, the system has to determine if the partial story is *coherent* as a story. Intuitively, this function would return that $kill(criminal, policeman), eat(policeman, sandwich)$ is not coherent, because (considering these facts to be sequential) a dead man can not eat anything. To define this function, $\lambda(s_i, C)$, we use the set of schemas C. $\lambda(s_i, C)$ will return *true* if each fact of the set of the partial story $s_i$is valid with the schema base. For instance, if $C = \{friend(?x, ?y) \wedge \neg chase(?x, ?y)\}$, and $s_i = \{chase(policeman, criminal) \wedge friend(policeman, criminal)\}$, $\lambda(s_i)$ will return $false$.

Algorithm 1 Exploratory algorithm for Story Generation

1. $s_0 = \{\}$
2. while $\omega(s_i, I) = \bot$/* *while the partial story is not coherent with the user* */
   (a) repeat $s_{n+1} = s_n \cup \varphi(s_n, K)$ while $\lambda(s_{n+1}, C) = \bot$/* *look for a new fact* */
   (b) if it has been possible, *continue*, expand the new state
   (c) if there is no possible fact, return *failure*, discard the fact and continue exploring

$$\lambda(s_i, C) = \begin{cases} \bot \leftarrow \exists c \in C \mid c \wedge s_i \rightarrow \bot \\ \top \leftarrow otherwise \end{cases} \tag{2}$$

For each s_i, in each state of the search, we have another function which decides whether the new partial story s_i is valid as a story as the user wants it to be, that is, what kind of story the user wants. If it is valid, then it is returned as a solution. This function must be controlled by the user requirements, as the constraints set for the story, what we call I. The function is called $\omega(s_i, I)$. For instance, to set that the story should be about a policeman which captures a criminal. Formally, it could be expressed as a set of facts which have to be true in the story, the definition is the next one:

$$\omega(s_i, I) = \begin{cases} \top \leftarrow I \subset s_i \\ \bot \leftarrow otherwise \end{cases} \tag{3}$$

All this description of the exploration process just defines a generative search where we check the validity of the partial story with two functions: One of them is defined in the knowledge base, and the other one is defined by the user. Obviously, with this description, we can generate a story which will be a subset of K.

The algorithm, then, could follow a simple state space search. Algorithm 1 gives a global idea of how the process is performed.

So, having defined $\langle K, C \rangle$ and I and the functions $\varphi(s_i, K)$, $\lambda(s_i, C)$ and $\omega(s_i, I)$, we could obtain this story about a criminal who steals something, and a policeman chasing him (we assume that the order of the facts is the sequence of them in time, first the left column from top to bottom, and next the right one):

$$\begin{array}{cc} getin(criminal, bank) & shoot(criminal, police) \\ catch(criminal, money) & shoot(police, criminal) \\ go(criminal, warehouse) & kill(policeman, criminal) \\ chase(policeman, criminal) & feel(policeman, good) \end{array}$$

This story is simple, but coherent. However, nothing special has been performed to create it, and it is not creative. It is just a set of valid facts together, and all the information needed for this story to be created has been set by the

designed of the system and the functions. Although defining the previous functions in a different, more sophisticated way, could create better stories (in the sense of more complex, or more detailed), the process would not be creative (although the functions could be computed with creative processes).

3.3 Creating the Story by Transforming the Constraints in the Conceptual Space

In the previous section we have seen that exploring the conceptual space can lead to coherent results, but the quality is limited. Of course, there are better ways of exploring the conceptual space. What we present in this section is how it is possible to adapt the state space search for it to be able to modify the constraints of the $\lambda(s_i, C)$ function.

We have seen that for each modification of the partial story, a set of new facts is chosen, and then, if the facts are *coherent*, a new partial story is generated, and then the search for a valid story continues. CAST creates many non-coherent partial stories, and, just by exploratory processes, they are simply discarded. These non-coherent facts *break* the conceptual space, but that does not necessarily mean that they lead to a non-valid story.

Thus, a new option arises for the process: Adapting the set of constraints to let the conceptual space grow, and in this way letting the system create a set of stories, which is different that the set which plain exploratory methods like the previously explained one can create.

In order to make this ideas possible in CAST, the previous algorithm has to be modified: The C set will change during the generation process. Then, we have to replace the $\lambda(s_i, C) : \langle set, set\rangle \rightarrow boolean$ function, with:

$$\lambda'(s_i, C_i) : \langle set, set\rangle \rightarrow \langle boolean, set\rangle$$

which receives the state from the state space search, and returns *true* if the new s_i (the new partial story) can be made coherent, and a new set of constraints, which is a transformation of the previous one. Of course, the transformation of the constraint rules could be done even if the new state is coherent, but the process would be the same.

The definition of a new λ' function involves creating a model of how transversing the constraints creates acceptable stories. While creating such a model and precisely defining what an "acceptable" story is, are beyond the scope of this paper, it is interesting to give some ideas which could be useful for discussion (Section 3.3).

We could define λ' as a function which just considers a percentage of not-valid facts (as defined in Section 3.2) as valid ones, using a random function. For instance, if we allow to add 10% of non-valid facts. This is semantically equivalent to the next expression (j is the state which follows the state i):

$$\lambda'(s_j, C_j) = \begin{cases} \bot, C_i \leftarrow (\exists c \in C \mid c \wedge s_i \rightarrow \bot) \wedge random(0..1) \geqslant param \\ \top, C_i \leftarrow otherwise \end{cases} \quad (4)$$

This approach can be considered as noise in the system, and can lead to unexpected results. Some of the can be considered coherent, and somehow original. At least, not possible by plain exploration. In this story (chosen in a set of generated stories with these process, many of them not coherent), a corrupt policeman betrays a criminal:

$$\begin{array}{cc} getin(criminal, bank) & getin(policeman, car) \\ catch(criminal, money) & go(policeman, warehouse) \\ friend(criminal, policeman) & go(criminal, warehouse) \\ getin(criminal, car) & kill(policeman, criminal) \end{array}$$

And some others make no sense at all. Again, discussion about if this partial randomness in the system can be considered artistic or surrealist is not the purpose of this paper.

Other option is to try to apply *meta-rules*. We call *meta-rules* those rules which apply to constraints and modify them. Currently, there is no standard way of creating them, nor a global description of how they are defined. They have been designed by hand, and the authors are aware they are not necessarily the best ones. However, it can be interesting to show a high level example of them.

Let us suppose we have the domain which outputs the story in Section 3.2. In a particular state i,

$$s_i = \left\{ \begin{array}{cc} getin(criminal, bank) & catch(criminal, money) \\ go(criminal, warehouse) & chase(policeman, criminal) \end{array} \right\}$$

and C_i is still the same set than C_0 (the initial constraints set). Then, the exploration operators create a new fact: $friend(policeman, criminal)$. As we have the restriction $friend(?x, ?y) \wedge \neg chase(?x, ?y)$, the new fact would be considered as not coherent. However, we apply the next meta-rule:

> If there is some restriction about a fact, and the schema forbidding the inclusion of that fact is related with no more than one fact in the partial story, that schema can be removed.

Then, the $\lambda'(s_i, C_i)$ function returns:

$$\left\langle s_j = \left\{ \begin{array}{cc} getin(criminal, bank) & catch(criminal, money) \\ go(criminal, warehouse) & chase(policeman, criminal) \\ friend(policeman, criminal) & \end{array} \right\}, \right\rangle$$
$$C_j = C_i - \{friend(?x, ?y) \wedge \neg chase(?x, ?y)\}$$

Then, $shoot(policeman, criminal)$ is going to be added, but it is not coherent because there is a constraint against this inclusion:$shoot(?x, ?y) \wedge chase(?x, ?y) \wedge \neg friend(?x, ?y)$. The system can not find a meta-rule for modifying that constraint, so the new fact can not be added.

Finally, following the process, this is the story that has been obtained. It is coherent, and it was not possible to obtain it just by exploration:

$$
\begin{array}{cc}
getin(criminal, bank) & friend(policeman, criminal) \\
catch(criminal, money) & shoot(criminal, policeman) \\
go(criminal, warehouse) & kill(criminal, policeman) \\
chase(policeman, criminal) & go(criminal, warehouse)
\end{array}
$$

Meta-rules applied in CAST are useful for the examples and domains we are working on, but they can lead to incorrect stories (not coherent ones), if the definition of the meta-rules does not cover all necessary cases. These "necessary cases" are dependent on the domain, obviously: Sets of meta-rules covering adaptations for a given domain are useful only inside that domain. As it has been shown in the example, the rules are absolutely *ad hoc*, and further research is necessary to define them in a better way. This toy domain has been designed for exemplifying how the system works. It is possible, of course, to create more complex ones. There are many more possible definitions for λ'. The only purpose of explaining these results is to show that transformation is possible, and there are many possible ways for doing it. A generative system following these ideas can not forget the importance of the exploration (which have been done randomly in this article), because the results from exploration are the basis for transformation in a system like CAST. Next section discusses the main benefits, drawbacks and characteristics of this approach.

4 Discussion

CAST is intended to focus on transformation of constraint rules, and some possible approaches have been shown. The main question to ask is if CAST is a transformational creativity application. By trying to apply ideas present by Ritchie in [4] we can find three sets of generated objects (stories in this paper) about what is generated: The *conceptual space*, which would be in this case all "typical items", or stories following very obvious schemas; the *possible items*, or the full set of all possible stories; and all items in the *medium*, which contains all possible elements written with characters (if we consider the medium to be written text on paper, or on the computer screen).

Following Sharples [9] we have the *universe* (every possible object generated), and a set of *schemas*. They both create the *conceptual space*. Although both definitions for the *conceptual space* is not exactly equivalent, we can, for this discussion, relax the exact meanings, considering it to be the *expected stories*. It is difficult to define what "expected" means in this context, and for this discussion we assume an intuitive definition for it: stories which follow a very classical or typical plot, or character behavior.

We have seen, in Section 3.3, two examples of stories which could not be generated by the exploratory process in Section 3.2. However, it can be argued that this is a problem of the exploration algorithm, which is extremely simple. Of course it is, but it could be replaced with a more complex one and, in principle, the transformation processes could be applied. The point, then, is whether transformation in general, and the presented approaches as instances for it, are

useful, and really creative. Following the descriptions of what the *conceptual space* can be, it is changing. Bypassing the rules or modifying schemas really modify what can be generated, so the set of possible stories for the system is different. From this point of view, the system can be considered creative.

Relative to other systems, MINSTREL [7] also performs adaptation from knowledge present in a database. However, in MINSTREL, a TRAM is selected, and applied to a given narrative problem. Then, the narrative problem is mapped into the TRAM, and, applying it as a solution, it is "mapped-back" into the problem domain. CAST's transformations do not work in this way: Instead, CAST is able to apply any modification, not necessarily one related with a "similar" schema. Obviously, this approach can obtain a bigger set of new information, but can also produce bad results because it does not have any restriction beyond testing the validity, and the test functions are still not fully developed.

It is important to compare Sharples account for writing with CAST. In MEXICA Sharples' model directs the main algorithm. The main idea is that creative writing is produced by alternation of two processes: *engagement* and *reflection*, as explained in Section 2. CAST is not fully directed by Sharples' ideas, but it performs a sort of engagement and reflection process. The exploration generates knowledge without limits, and the transformation step checks for validity and adapts the constraints, accepts the new information or discards the content. Then, the exploration stage can be considered to be engagement, and it can be argued that checking the coherence and adapting the constraints set is a kind of reflection. However, this is not totally true, because in the checking stage, the information can be totally discarded, which does not literally follow Sharples' model.

Are meta-rules more useful than defining better constraint rules? It is more difficult, in general, to define meta-rules than defining constraint rules. On the other hand, constraints rules are domain dependent and meta-rules are not (if they are well defined, of course). The size of the set of valid stories that can be generated (which can be considered a measurement for the quality of the system) depends both on the constraint rules and on the meta-rules. It can be concluded, then, that meta-rules, covering general knowledge about how to transform, are useful for the system as itself, only if it is possible to create valid ones (general for many domains, applicable for every one of them). If this is true, the constraints database can be really domain dependent, and would not have to store general content. It is necessary further research and discussion to claim a formal, well demonstrated conclusion.

This paper's proposal is about trying a new option for this objective: Transforming story generation rules. It does not substitute the exploratory methods, (in fact, it depends on them), it just allow for a new, additive option. Although it is difficult to create meta-rules for this to allow a storytelling system create *exceptional* stories, it gives a new option. Even if, for a single generated story, it creates a new single, useful schema, the transformation can be worth the extra processing.

5 Conclusions and Future Work

CAST, a new storytelling system based on exploration and transformation has been presented and some results of the system are shown on the paper. The results of an example story is also explained in the paper, and the relation with transformational creativity is discussed.

Although this paper is focused on transformation of the rules, exploration can not be left as an unimportant matter. Choosing appropriate content for adding to the story can make the transformation more powerful, because probably it would not have to take into account completely meaningless additions. Next versions of CAST will be focused on better exploration approaches and, of course, a more deep study about transformation options.

References

1. Boden, M.: Computational models of creativity. Handbook of Creativity (1999) 351–373
2. Boden, M.: Creative Mind: Myths and Mechanisms. Routledge, New York, NY, 10001 (2003)
3. Bundy, A.: What is the Difference between Real Creativity and Mere Novelty? Behavioural and Brain Sciences (1999)
4. Ritchie, G.: The Transformational Creativity Hypothesis. **24**(3) (2006) 241–266
5. Wiggins, G.: Searching for Computational Creativity. New Generation Computing, Computational Paradigms and Computational Intelligence. Special Issue: Computational Creativity **24**(3) (2006) 209–222
6. Meehan, J.: The Metanovel: Writing Stories by Computer. PhD thesis (1976)
7. Turner, S.R.: Minstrel: A computer model of creativity and storytelling. Technical Report UCLA-AI-92-04, Computer Science Department, University of California (1992)
8. Pérez y Pérez, R.: MEXICA: A Computer Model of Creativity in Writing. PhD thesis, The University of Sussex (1999)
9. Sharples, M.: An account of writing as creative design. The Science of Writing (1996)
10. Gervás, P., Díaz-Agudo, B., Peinado, F., Hervás, R.: Story Plot Generation Based on CBR. Knowledge-Based Systems. Special Issue: AI-2004 **18** (2005) 235–242
11. Riedl, M.: Narrative Planning: Balancing Plot and Character. PhD thesis, Department of Computer Science, North Carolina State University (2004)
12. Bringsjord, S., Ferrucci, D.: Artificial Intelligence and Literary Creativity: Inside the mind of Brutus, a StoryTelling Machine. Lawrence Erlbaum Associates, Hillsdale, NJ (1999)
13. Lebowitz, M.: Creating characters in a story-telling universe. Poetics **13** (1984) 171–194
14. Gervás, P.: An Expert System for the Composition of Formal Spanish Poetry. Journal of Knowledge-Based Systems **14**(3-4) (2001) 181–188