Dynamic Inspiring Sets for Sustained Novelty in Poetry Generation

Pablo Gervás
Universidad Complutense de Madrid
c/ Profesor García Santesmases s/n
Madrid, 28040, Spain
pgervas@sip.ucm.es

Abstract

One activity recognised as an interesting instance of creativity is the ability of poets to systematically come up with new poems, irrespective of how many they have already written in the past. Poets who periodically produce a new poem, different from the earlier ones and comparatively valuable, are considered to embody a more interesting type of creativity than poets who have produced only one good poem in their lifetime, or those that produce a succession of poems all built following a standard recipe so obvious that it can almost be described as a template. This paper explores the constraints imposed on computational creative processes by the requirement of novelty with respect to previous outputs. Two issues emerge as fundamental: how to evaluate novelty against a set of artifacts taken as reference, and how to adjust construction procedures so that each successive run leads to significantly different output. A possible modelling of these two issues is proposed in terms of two sets of sample artifacts: a *reference set* (against which novelty is measured) and a *learning set* (used to configure the construction procedures). Over this basic model, extended discussion is carried out to draw out interesting insights for the design of computational creative systems.

Introduction

The process of coming up with a novel poem involves a skill for producing something recognisable as a poem and the ability to recognise efforts that lead to results too similar to previous poems (in order to avoid them). Existing automatic poetry generators have mostly focused on modelling the generative skill rather than the historical evaluation function. But even if these two elements were successfully modelled in a single system, the task of modelling how they evolve over time would remain an important challenge for computational creativity. Simply by generaring new material, but also by reading material by others in between acts of creation, human authors modify the frame of reference that they employ to judge their own creations. Additionally, their technique may evolve over time, sometimes through exploration of new possibilities but quite often as a result of a conscious effort to emulate material produced by others that they have liked. Human authors who produce new material by modifying their technique are considered more creative than those that simply obtain different material using the same technique.

The present paper reviews a number of existing poetry generators focusing on their ability to model a generation skill and a validation mechanism for novelty. Two issues emerge as fundamental: how to evaluate novelty against a set of artifacts taken as reference, and how to adjust construction procedures so that each successive run leads to significantly different output. A possible modelling of these two issues is proposed in terms of two sets of sample artifacts: a *reference set* (against which novelty is measured) and a *learning set* (used to configure the construction procedures).

Previous Work

This section presents some useful references in terms of computational creativity that provide a basic vocabulary to discuss the phenomena under study, and reviews a number of automated poetry generators.

Computational Creativity

Many efforts over the recent years that address the study of creativity from a computational point of view acknowledge as a predecessor the work of Margaret Boden (1990). Boden proposed that artificial intelligence ideas might help to understand creative thought. This idea was taken up by a number of artificial intelligence researchers and gave rise to a research line that attempts to model or reproduce creative thought in computer systems. Some of Boden's ideas have had great influence in later work. One important idea was the distinction between historical and psychological views of creativity. Historical creativity (H-creativity) involves the production of ideas that have not appeared before to any one else in all human history. Psychological creativity (P-creativity) involves the production by a given person of ideas that have not occurred before to that particular person. This distinction is important because it implies that, unless a computer program is given access to historical data (and generally provided with means for social interactions with other creators), it will only be capable of P-creativity.

Wiggins (2006) takes up Boden's idea of creativity as search over conceptual spaces and presents a more detailed theoretical framework that specifies 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, the function for evaluating particular points in that 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. In fact, definitions of search space and traversal function in a creative setting are not only particular to a given creator and different from those used by others, but also constantly in flux.

Ritchie (2007) addresses another important issue in the development of creative programs, that of evaluating when a program can be considered creative. He does this by outlining a set of empirical criteria to measure the creativity of the program in terms of its output. He makes it very clear that he is restricting his analysis to the questions of what factors are to be observed, and how these might relate to creativity, specifically stating that he does not intend to build a model of creativity. Ritchie's criteria are defined in terms of two observable properties of the results produced by the program: novelty (to what extent is the produced item dissimilar to existing examples of that genre) and quality (to what extent is the produced item a high-quality example of that genre). Another important issue that affects the assessment of creativity in creative programs is the concept of inspiring set, the set of (usually highly valued) artifacts that the programmer is guided by when designing a creative program. Ritchie's criteria are phrased in terms of: what proportion of the results rates well according to each rating scheme, ratios between various subsets of the result (defined in terms of their ratings), and whether the elements in these sets were already present or not in the inspiring set.

Jennings (2008) introduced computationally plausible modeling of the fact that most human creativity takes place with the creator embedded in a broader society of other creators and critics, and that this context affects significantly the creation of new artifacts. To capture the way in which humans react to these constraints, Jennings defines the concept of *creative autonomy*, which requires that a system be able to evaluate its creations without consulting others, that it be able to adjust how it makes these evaluations without being explicitly told when or how to do so, and that these processes not be purely random. The model he proposes relates the evaluation of a systems creations to its perception of how other members of its social context are likely to evaluate them. Changes in how this evaluation is carried out may be triggered by the need to align personal evaluations with other members of the society or as a side effect of trying to justify past evaluations. Creative autonomy is therefore argued to emerge out of the interactions with multiple critics and creators, rather than from meditative isolation.

Automatic Poetry Generators

A number of existing automated poetry generators are reviewed focusing on the basic techniques for text creation that have been used as underlying technologies.

The generate & test paradigm of problem solving has been widely applied in poetry generators. Because metric restrictions are reasonably easy to model computationally, very simple generation solutions coupled with an evaluation function for metric constraints are likely to produce acceptable results (given an assumption of poetic licence as regards to the content).

The WASP system (Gervás, 2000) draws on prior poems and a selection of vocabulary provided by the user to generate a metrically driven recombination of the given vocabulary according to the line patterns extracted from the original poems. The WASP automatic poet used a set of construction heuristics obtained from formal metric constraints to produce a poem from a set of words and a set of line patterns provided by the user. The system followed a generate and test method by randomly producing word sequences that met the formal requirements. Output was impeccable from the point of view of formal metrics, but clumsy from a linguistic point of view, and it made little sense.

An example of poem output by WASP is given below:

> Todo lo mudará la edad hermosa.
> Marchitará la luz el vuelo helado
> del gesto. Se escogió en color airado
> con no hacer mudanza por su rosa.

This is a metrically correct *cuarteto*, a bit stilted from a grammatical point of view, and clearly driven by the underlying choice of vocabulary and patterns, which recall very specific examples of Spanish sixteenth century classics. The actual meaning[1] emerges from the construction process as a surprise.

An initial work by Manurung (1999), based on chart generation, focuses on the generation of poetry in English, starting from a semantic representation of the meaning of the desired poem. A very important driving principle in this case is to respect the unity between form and meaning that is considered to provide the aesthetical backbone of real poetry. This implies that poems to be generated must aim for some specific semantic content, however vaguely defined at the start of the composition process. The approach relied on chart generation, taking as input a specification of the target semantics in first order predicate logic, and a specification of the desired poetic form in terms of metre. Words that subsume the input semantics are chosen from a lexicon, and a chart is produced incrementally to represent the set of possible results. At each stage, the partial solutions are checked semantically to ensure that no sentences incompatible with the original input are produced. Additionally, partial results are checked for compatibility with the desired poetic form. Because the search space is pruned of invalid partial solutions at each stage, the approach is generally efficient. This corresponds to a systematic generate & test approach, trying all possibilities and making sure that no partial constituent is generated twice by the system. It also allows the user to control the input in terms of meaning. This has the advantage of restricting somewhat the probability of obtaining non-sensical output, but it also limits the degree of freedom of the system. The amount of creativity that system can exercise on the semantics of its output is limited. This consti-

[1] *Beautiful age will alter all. // The frozen flight of the gesture // will make light wilt. // It was chosen in angry colour // on not adapting for its rose.*

tutes a significant restriction on the extent of poetic licence allowed.

An example of poem output by Manurung's initial system is given below:

> the cat is the cat which is dead
> the bread which is gone is the bread
> the cat which consumed
> the bread is the cat
> which gobbled the bread which is gone

This is produced by the system from input specifying a limerick target form and target semantics {`cat(c), dead(c), bread(b), gone(b), eat(e,c,b), past(e)`} but disregarding the rhyme scheme.

Manurung went on to develop in his Phd thesis (Manurung, 2003) an evolutionary solution for this problem. Evolutionary solutions seem particularly apt to model this process as they bear certain similarities with the way human authors may explore several possible drafts in parallel, progressively editing them while they are equally valuable, focusing on one of them when it becomes better valued, but returning to others if later modifications prove them more interesting. Manurung's evolutionary solution is demonstrated in MCGONAGALL, a proof-of-concept system for a model of poetry generation as a state space search, solved using evolutionary algorithms. Manurung tests his system exhaustively, hoping to demonstrate separately its abilities as a form aware generator (come up with poems matching a given target form), as a tactical generator (come up with valid realizations for a target semantics) and as a poetry generator (combine both to come up with poems matching the target form and the target semantics as closely as possible). Results seem to indicate that acceptable output from a purely linguistic point of view is easily achievable for the form aware generator, achieved with difficulty for the tactical generator, and extremely difficult to achieve for the poetry generator.

An example of poem output by MCGONAGALL in its form aware mode is given below:

> They play. An expense is a waist.
> A lion, he dwells in a dish.
> He dwells in a skin.
> A sensitive child,
> he dwells in a child with a fish.

Manurung's results explain why most automatic poetry generators restrict themselves to operating in form aware mode.

Another important tactic that human authors are known to use is that of reusing ideas, structures, or phrasings from previous work in new results. This is very similar to the AI technique of Case-Based Reasoning (CBR). Some poetry generators have indeed explored the use of this technique as a basic generation mechanism. ASPERA, an evolution of the WASP system (Gervás, 2001) used CBR to build verses for an input sentence by relying on a case base of matched pairs of prose and verse versions of the same sentence. Each case was a set of verses associated with a prose paraphrase of their content. An input sentence was used to query the case base and the structure of the verses of the best-matching result was adapted into a verse rendition of the input. This constituted a different approach to hardening the degree of poetic licence required to deem the outputs acceptable (the resulting verses should have a certain relation to the input sentence). ASPERA is described as following a classic Retrieve-Reuse-Revise-Retain CBR cycle (Aamodt and Plaza, 1994). The Revise-Retain stages involve carrying out an analysis of any validated poems in order to add the corresponding information to the system data files, to be used in subsequent computations. The ASPERA system requires user supervision to carry out these steps, with a human revising successive outputs to decide which should be retained.

An example of poem output by ASPERA is given below:

> Ladrará la verdad el viento airado
> en tal corazón por una planta dulce
> al arbusto que volais mudo o helado.

It is interesting to observe that this poem[2] shows certain similarities to the output of the WASP system shown above (the first verse of this *terceto* matches the structure of the second verse of the *cuarteto*). This suggest that the same classic verse must have been used to provide the line pattern used by WASP and the prose-verse version of the sentence used by ASPERA. There are also similarities in lexicon and rhymes.

In 1984 William Chamberlain published a book of poems called "The Policeman's Beard is Half Constructed" (Chamberlain, 1981). In the preface, Chamberlain claimed that all the book (but the preface) had been written by a computer program. The program, called RACTER, managed verb conjugation and noun declension, and it could assign certain elements to variables in order to reuse them periodically (which gave an impression of thematic continuity). Although few details are provided regarding the implementation, it is generally assumed that RACTER employed grammar-based generation. The poems in Chamberlain's book showed a degree of sophistication that many claim would be impossible to obtain using only grammars, and it has been suggested that a savvy combination of grammars and carefully-crafted templates may have been employed, enhanced by heavy filtering of a very large number of results.

An example of poem output by RACTER is given below:

> More than iron
> More than lead
> More than gold I need electricity
> I need it more than I need lamb or pork
> or lettuce or cucumber
> I need it for my dreams

This poem shows how structural repetition (in this instance of constructions *more than* and *I need*) can be fundamental for aesthetic effect.

The use of n-grams to model the probability of certain words following on from others has proven to be another useful tecnique. An example of poetry generation based

[2] *The angry wind will bark the truth // in such a heart for a sweet plant // to the bush that you fly mute or frozen.*

on this is the cybernetic poet developed by Ray Kurzweil. RKCP (Ray Kurzweil's Cybernetic Poet)[3] is trained on a selection of poems by an author or authors and it creates from them a language model of the work of those authors. From this model, RKCP can produce original poems which will have a style similar to the author on which they were trained. The generation process is controlled by a series of additional parameters, for instance, the type of stanza employed. RKCP includes an algorithm to avoid generating poems too close to the originals used during its training, and certain algorithms to maintain thematic coherence over a given poem. Over specific examples, it could be seen that the internal coherence of given verses was good, but coherence within sentences that spanned more than one verse was not so impressive.

An example of poem output by RKCP (after Lord Byron) is given below:

> Oh! did appear
> A half-formed tear, a Tear.
> By the man of the heart.

This example shows how extreme brevity can help to convey the idea of interesting underlying semantics even when none have actually been involved in the construction process.

Modelling Example-Driven Creation

Attempts at modelling the task of text creation computationally tend to focus on a static representation. The conceptual models used correspond to those of a single act of creation, or the creation of a single text. This section outlines how a model that considers the dynamics involved in a sequence of acts of creation might be described in terms of some classic concepts of computational creativity.

With a view to exploring the phenomena that we are interested in, we need to make a few assumptions about the various ingredients that might be involved, and how we are to refer to them. We take up Ritchie's terminology (Ritchie, 2007) to describe the *inspiring set*, the set of (usually highly valued) artifacts that the programmer is guided by when designing a creative program. In our case, this will be a collection of texts that the program is aware of at the start.

However, this set of poems may be used in two different ways. On one hand, it can be used to inform the production mechanism that is used. Some of the possible mechanisms for producing text used by the automatic poetry generators reviewed above rely on a collection of text either to act as case base, from which to extract a grammar, or on which to train an n-gram model. We will refer to the set of texts used for this purpose as the *learning set*. On the other hand, it can be used to inform the evaluation metric used for checking novelty of results. In most cases, this will take the form of checking new results against this previous collection to test for p-creativity. We will refer to the set of texts used for this purpose as the *reference set*.

[3] http://www.kurzweilcyberart.com/poetry/rkcp_overview.php3

Instances of Inspiring Sets in Poetry Generators

ASPERA and RKCP clearly rely on a set of poems that they use as inspiring set. In both cases this inspiring set is used as learning set. They differ in the way in their approach to dynamic updating of this learning set, and on whether they use it as a reference set as well.

ASPERA includes a mention of how the results of the system can be integrated into the creative process. Because this system applies a CBR solution, subsequent results may be added to the case base to provide additional material for later runs. In this case, the set of poems used to create the case base constitutes the learning set, in the sense that it determines what outputs can be constructed. The Revise-Retain stages constitute instances of expanding the learning set. However, one must take into account the fact that this task must be carried out by a human.

No mention is made of how ASPERA avoids replicating partly or completely previous solutions. Indeed, in traditional CBR systems replicating previous solutions is considered an advantage rather than a drawback. This may be a disadvantage when they are applied for creative purposes. Under this light it seems fair to say that ASPERA does not use a reference set.

In the case of RKCP, the selection of poems by a given author from which the language model used by system is built constitutes an instance of a learning set. RKCP introduces an innovation by allowing the possibility of maintaining separate models for the styles of different authors. In this case, each one of these models comes from a different learning set, and the system can switch from one to another to achieve variety. Yet the set of models remains fixed.

The RKCP is said to include an algorithm to avoid generating poems too close to the originals used during its training. No details are provided as to how this algorithm operates. It is unclear whether it achieves this by checking against a set of reference texts or by constraining the construction process in some way. Yet it is fair to say that in this instance the learning set is also being used as a reference set. There seems to be no mechanism for ensuring that successive system outputs differ from one another at least as much as they differ from the reference set.

The chart approach employed in Manurung (1999) is aimed at ensuring that no partial constituent is generated twice by the system. This can be seen as a different practical solution to the problem of avoiding redundant outputs. The use of a chart structure to store intermediate results is another alternative to comparing each candidate with a store of previously produced items, with no added cost of re-indexing after each production. However, this system applies that solution only to avoid redundancy during exploration towards a specific single output. No mention is made of keeping records of previous outputs to be considered when producing new ones. For this additional purpose, a chart would be impractical.

Dynamic Inspiring Sets as Configuration Resources

The review of existing automatic poetry generators has shown that is is possible to define both the construction process and the validation of output novelty with respect to a set

of texts, referred to as inspiring set. The concept of an inspiring set of poems as means of configuring a poetry generator presents the advantages of corpus-based approaches to natural language processing. Within this paradigm, the performance of a processing system is determined by the corpus on which it was trained. If the system needs to change, this is done by changing the corpus and retraining the system.

In computational work carried out so far on automated poetry generators, inspiring sets have always been configured in a static way. A given set of texts is taken as inspiring set and used as learning set and/or reference set, with smaller or larger numbers of poems produced with no thought to how the act of producing new poems might affect either. Ideally we would want to explore the possibilities of having the creation process itself provide feedback to these sets, with a view to identifying whether such feedback might capture some of the patterns observed in human creativity. If this approach is applied in creative endeavours, it presents the additional problem of requiring some means for managing the progressive evolution of these inspiring sets, including the need to periodically retrain the system. This issue is discusses with respect to the technical solutions outlined earlier.

CBR approaches include the means for systematically updating the case base, which constitutes an instance of a learning set. A case base can also be employed as a reference set if the retrieval stage is configured to always select slightly dissimilar cases, forcing the reuse stage to apply heavy adaptation.

In a similar vein n-gram based approaches provide means for enforcing difference with the inspiring set by controlling the probability of the resulting chains. To model the possibility of progressively extending the inspiring set, a procedure would have to be introduced for periodically retraining the models employed.

The Problem of Managing Inspiring Sets

From the point of view of collecting intuitions on the various challenges involved in modelling human creativity, the issues outlined so far point to a candidate requirement that had not been identified as part of creative systems developed in the past: the need for appropriate management of the set of inspiring artifacts.

Where these artifacts are used as a reference set, appropriate management of the reference set would become a must for any system that intends to be aware of whether its results in successive runs are indeed new with respect to previous output. This would correspond to the way in which human creators seem to be aware of the state of the art in their fields, at least to the extent of knowing when solutions being explored are indeed new. This requirement would be tightly related to Boden's concept of p-creativity.

The most straightforward solution to apply would be to test every result for similarity against previously known texts. This requires each new result to be compared with an ever increasing set of previous results. Elaborate means of indexing or clustering the set of reference texts would help reduce the complexity of checking for novelty. However, the improvements obtained in such a way would have to be offset with the difficulties involved in having to re-index the reference set each time a new result is added.

From an engineering point of view, it would make sense to resort to indexing the set of reference artifacts. The task of indexing would slow down the production of successive results, but it would speed up the task of exploring the conceptual space in search for new candidate solutions. This would match the intuition that human creators may take some time between the creation of one piece and setting out to produce a new one. During this time a creator may be digesting the results of his last creation, or searching for inspiration for the next. The task of re-indexing the reference set could be identified as part of either or both of these processes. In contrast, during the act of creation itself human creators shift very rapidly through candidate solutions, with no lingering on past material. This would match more closely an engineering model based on prior indexing than one based on systematic comparison.

Where the set of already known artifacts is used to inform the construction procedure, different procedures will arise depending on the particular technique employed. In CBR solutions, management of the learning set would involve periodically re-indexing a growing case base. For n-gram based approaches, management of the learning set would consist of periodically retraining the set of models employed, or possibly more radical periodical rearrangements involving re-clustering of the learning set and training of a new set of models based on the resulting set of clusters. With respect to human performance, the periodical maintenance of the learning set would mirror evolution of author's technique.

Intuitions for Computational Creativity

The issue of considering successive system results as part of the reference set is fundamental for being able to discern whether any given result is p-creative as defined by Boden. Not many previous creative systems consider this. A notable exception is the MEXICA storytelling system (Pérez y Pérez, 1999), which does include a mechanism for checking its results with previous outputs.

The subdivisions of the inspiring set described above could be related to some of the formal elements described in Wiggins' framework (Wiggins, 2006). The learning set of any such system actually determines the conceptual space, and the production mechanism derived from it constitutes the operational equivalent of a traversal function. In a similar vein, the reference set will undoubtedly play a fundamental role in any evaluation function designed to consider novelty, as that of any system aiming to be creative should. The possibilities discussed above of dynamically updating either the learning set or the reference set would capture the intuitions that definitions of search space, traversal function and evaluation function may be constantly in flux.

The reference set in its various configurations described above plays an important role in the definition or identification of the novelty of the results of a program that aims at being creative, as described by Ritchie (2007). No mention has been made in this paper of the quality of the results, which is their other important observable property. An ap-

propriate evaluation of the quality of results would play a fundamental role in all the processes described above. Only results that have been rated above a given threshold of quality should be considered for updating either the reference set or the learning set. This constitutes a less obvious but quite significant role of whatever evaluation function is used to determine quality in a creative system. If the most obvious role of the quality function is to select actual system outputs, by doing so it actually guides the development of an individual style for the system, through the task of filtering what results get added to reference or learning set. The choices taken in the past are fed back into system operation and affect choices taken in the future.

The dynamic handling of the reference set and the learning set may provide a simple way of implementing in computational systems the kind of social interaction described by Jennings as fundamental for human creativity (Jennings, 2008). Systems defined in terms of dynamic inspiring sets as described in this paper would automatically adjust their novelty evaluation functions just by being exposed to new artifacts as they are added to the reference set. They would also adjust their production mechanisms by exposure to new artifacts as they are added to their learning set. This would allow for easy modelling of the interplay between learning to emulate the work of creators one admires (as driven by its inclusion in the individual's learning set) and trying to steer away from the work one dislikes (by including it in the reference set). Such a model would capture much of the way in which human creators learn their trade.

Conclusions

Computational approaches to creativity in the past have mostly considered static views of the creative process, in the sense that they have focused on a single act of creation rather than what it is that enables a creator to achieve a sequence of acts of creation, each one producing a different artifact that innovates with respect to previous ones and possibly is the result of a different process of construction or composition. The present paper has explored how such a dynamic view of the creative process may be modelled in terms of two subdivisions of the concept of inspiring set: a learning set from which the production mechanisms are learnt, and a reference set against which candidate results have to be checked to ensure novelty. This approach implies that each individual creative act must be considered in a larger context. From an engineering point of view this presents the problem of how to manage the reference set in a computationally tractable manner. It also introduces the problem of when and how each of these sets is updated, or when they are processed to re-train the production mechanisms or re-index the reference set.

As future work it would be very valuable to consider possible ways of implementing some of these ideas in experimental systems. Such systems need not attempt to model full-blown literary generators. Even the simpler task of generating coherent text as output could serve as a test for some of the ideas presented in this paper. The mechanisms for ensuring novelty with respect to a set of source materials apply just as well to simple text as to more literary efforts. The ability to identify when a production mechanism is reaching its limits and begins to produce less novel material can also be tested over such simple set ups.

Acknowledgments

The research reported in this paper is partially supported by the Ministerio de Educación y Ciencia (TIN2006-14433-C02-01, TIN2009-14659-C03-01), Universidad Complutense de Madrid and Dirección General de Universidades e Investigación de la Comunidad de Madrid (CCG07-UCM/TIC 2803).

References

Aamodt, A., and Plaza, E. 1994. Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AI Communications* 7(i).

Boden, M. 1990. *The creative mind*. London: Weidenfeld and Nicholson.

Chamberlain, W. 1981. *The Policeman's Beard is Half Constructed*. New york: Warner Books.

Gervás, P. 2000. WASP: Evaluation of different strategies for the automatic generation of Spanish verse. In *Proceedings of the AISB-00 Symposium on Creative & Cultural Aspects of AI*, 93–100.

Gervás, P. 2001. An expert system for the composition of formal Spanish poetry. *Journal of Knowledge-Based Systems* 14(3-4):181–188.

Jennings, K. E. 2008. Developing creativity. artificial barriers in artificial intelligence. In *In Proceedings of International Joint Workshop on Computational Creativity 2008*, 1–10.

Manurung, H. M. 1999. Chart generation of rhythm-patterned text. In *Proc. of the First International Workshop on Literature in Cognition and Computers*.

Manurung, H. M. 2003. *An evolutionary algorithm approach to poetry generation*. Ph.D. Dissertation, University of Edimburgh, Edimburgh, UK.

Pérez y Pérez, R. 1999. *MEXICA: A Computer Model of Creativity in Writing*. Ph.D. Dissertation, The University of Sussex.

Ritchie, G. 2007. Some empirical criteria for attributing creativity to a computer program. *Minds & Machines* 17:67–99.

Wiggins, G. 2006. Searching for Computational Creativity. *New Generation Computing, Computational Paradigms and Computational Intelligence. Special Issue: Computational Creativity* 24(3):209–222.