Minds & Machines (2007) 17:67–99
DOI 10.1007/s11023-007-9066-2

Some Empirical Criteria for Attributing Creativity to a Computer Program

Graeme Ritchie

Received: 28 March 2006 / Accepted: 9 May 2007 / Published online: 5 June 2007

Abstract Over recent decades there has been a growing interest in the question of whether computer programs are capable of genuinely creative activity. Although this notion can be explored as a purely philosophical debate, an alternative perspective is to consider what aspects of the behaviour of a program might be noted or measured in order to arrive at an empirically supported judgement that creativity has occurred. We sketch out, in general abstract terms, what goes on when a potentially creative program is constructed and run, and list some of the relationships (for example, between input and output) which might contribute to a decision about creativity. Specifically, we list a number of criteria which might indicate interesting properties of a program's behaviour, from the perspective of possible creativity. We go on to review some ways in which these criteria have been applied to actual implementations, and some possible improvements to this way of assessing creativity.

Keywords AI methodology · Computational creativity · Empirical criteria · Generating artefacts · Assessing output

Introduction

Since the early days of artificial intelligence, there has been a dispute about the fundamental limitations of computer programs in the general area of imitating human activities (Dreyfus 1979; Weizenbaum 1976). Boden (1992) attributes some of the central questions in this debate to the 19th century pioneer Ada Lovelace. One aspect of computer behaviour which is particularly contentious is that of *creativity*, summarised by the general philosophical question: ‘‘can computers be truly creative?’’. Recent decades have seen a growth in attempts to build computer

G. Ritchie (✉)
Department of Computing Science, University of Aberdeen, Aberdeen AB24 3UE, UK
e-mail: gritchie@csd.abdn.ac.uk

programs which tackle tasks which, when performed by humans, are deemed creative: representational paintings (Boden 1992, pp. 135–153), music (Baggi 1992; Miranda 2001), mathematical concepts (Lenat 1979; Colton 2002), stories (Meehan 1976; Turner 1994), jokes (Binsted and Ritchie 1997; Stock and Strapparava 2005), or poems (Gervás 2000, 2001, 2002; Manurung et al. 2000a, b). As well as all this concrete work in specific domains of creativity, there has also been a stream of work which tries to pin down the formal properties of (computational) creative processes in a domain-independent manner (Wiggins 2001, 2003, 2005, 2006a, b; Pease et al. 2001). Within that, a recurring theme is the question ''which computational mechanisms are most suited to producing creative behaviour?''. This strand has been heavily influenced by the much-debated ideas of Boden (1992, 1998) on the nature of creativity.

The proposals here fall, to some extent, between these two streams (specific implementations and general analyses), and may even be seen as trying to bridge between them. Our objective is to set out a way in which very general questions such as the philosophical ''can computers be truly creative?'', and the more practical ''which computational mechanisms are most suited to producing creative behaviour?'', can be connected to empirical evidence from actual working systems. The reasoning is that such general questions can be answered only if we have a way of answering the more specific question ''has this program behaved creatively on this occasion?'' (or perhaps ''to what extent has this program behaved creatively on this occasion?''). Without a way to answer this question, the more general questions can never be tested empirically. We take it as a general axiom of scientific methodology that claims should be subject to empirical refutation or corroboration.

It may be helpful to point out some issues which will *not* be tackled here. Firstly, we are not attempting to characterise the nature of the creative process, nor to build a model of creativity (unlike Wiggins (2001, 2006a, b), for example). This means that many of the suggestions raised by Boden (1992), and debated subsequently, about different types of creativity (combinational, exploratory, transformational) are not under consideration here. Also, the present paper is not itself empirical, but methodological: we are considering where one might look for evidence, rather than offering actual evidence of creativity for any particular programs. We will not be attempting to support or refute the proposition that computers can be creative, but will be considering how it would be possible to substantiate or refute such hypotheses empirically. However, this does not mean that we will be discussing such experimental matters as testing procedure, construction of suitable controls, use of appropriate statistics, etc. Although these are important issues, a logically prior debate is the decision about *what factors are to be observed*, and *how these might relate to creativity*. Once we have some idea of what to measure, then standard experimental practices can be adopted.

We shall start by setting down our basic methodological assumptions. Then we shall present our main proposals, starting with an abstract characterisation of the situation in which a potentially creative program is built and run, and proceeding to argue for various possible factors which might be relevant to a judgement about the creativity of the program. These factors will be formally stated as criteria which can be applied precisely, providing that the relevant information about the program is available. After listing these criteria, we shall look briefly at some related work which has developed since the criteria first appeared (Ritchie 2001), and then review some small studies where the criteria were applied to actual systems. After reflecting on the lessons of these studies, and we speculate on possible extensions to the framework.

Assumptions

What Kinds of Activity are Creative?

The use of the term 'creative' originated within human society, and our whole notion of 'creative' tends to manifest various social and cultural prejudices which have to be taken into consideration when considering computer behaviour. In this author's culture at least, certain activities are assumed to be more creative than others. Painting a picture, writing a poem, or creating a sculpture are often deemed creative, even when performed in a relatively ordinary manner. Mathematics, science, or engineering are rarely classed as creative, unless they are done exceptionally well. This bias does not seem helpful in a rigorous attempt to pin down the notion of creativity, particularly when applied to machines. Although there is still a tendency within AI to tacitly accept this intellectual apartheid of creative versus non-creative activities, it would be better if we could be more neutral in our formal characterisation of creative actions. In the rest of this paper, the illustrative domains mentioned will typically be areas such as poetry-writing or story-telling, but this does not mean that the formal definitions or substantive proposals relate only to those activities. We will abstract away from the particular genre of activity, and discuss only the formal properties of the production of artefacts. (See later in this section for a more detailed sketch of the type of programs considered.)

The Basis in Human Creativity

A central assumption here is that any formal definition of creativity must be based on its ordinary usage; that is, it must be *natural* and it must be *based on human behaviour*. (This is probably tacitly assumed by most research in this area, but is worth making explicit, to give later discussion a firmer foundation.) By ''natural'', we mean that any technical definition of 'creative' (or 'creativity') which is to be used in discussing the behaviour of computer programs must capture fairly accurately the original ordinary language use of the term. This is the only way to ensure that we are at least broadly considering creativity, and not some other concept, and that any claims or findings can be expressed in a meaningful way. Despite the fact that ordinary language is often imprecise or ambiguous and scientific usage should be exact, our precise formal definition should be a close approximation of the original loose term.

Similarly, the ''basis in human behaviour'' means that we should be guided by the way that the word ''creative'' is ordinarily used when talking of non-machine (human) creativity, for two reasons. Firstly, this provides us with some guidelines in firming up what we mean by creativity. Secondly, to rely on instances of machine creativity (the problem we wish to analyse) would risk circularity in claims about the nature of that process.

This orientation explains the allusions, in later sections, to instances of human creativity (as is commonplace in articles in this area).

Sources of Evidence

We should, in our pursuit of evidence that a program has behaved creatively, consider only *empirically observable factors*. In human creative activities, there are certain aspects which are knowable, such as the attributes of the artefact created, the other comparable artefacts in existence, possibly the other artefacts the creating individual was aware of. What we usually do not know is the mental or emotional processes by which the individual produced the artefact (although we may know other aspects of the action, such as the time taken). Hence, it is routine to make judgements of creativity (in humans) on the basis of what is known, often focussing on the attributes of the artefact(s). If our formal definition of achieving creativity, for analysis of computer systems, is to mimic our judgements of humans, then it too should be based only on comparably observable factors, without adding extra information about the internal workings of the computer program. This may be our most contentious working assumption, as some would argue that the inner workings of a computer program are critical in deciding its creativity; in particular, (Boden 1992, pp. 39–40) advocates just such consideration of the underlying process (for both humans and computers). We suggest that this would move away from the way human creativity is normally judged. (It also risks circularity when asking the question ''which computational mechanisms give rise to creativity?''—see below.)

There is a fine but important distinction between the production of the artefact, and the devising of the production-method; if we happen to know the method, we can treat it as an abstract artefact and consider the creativity it manifests, relative to other production-methods. This holds true for both computer and human creativity, to some extent, and starts to address the intuition that human artefacts might be assessed in terms of how much technical skill went into their production, as sometimes happens in critiques of 'conceptual' art. For example, suppose a human artist devises a highly original way of producing some physical artefact, such as a multi-media installation or a sculpture, and then employs skilled technicians to implement this method. The artist's contribution is the invention of the method (an abstract artefact), and we can assess this achievement (for its creativity, etc.). The characteristics of the final concrete result may be more attributable to the implementers.

The more finely one wishes to model judgements of creativity, the more complex matters become. For the purposes of setting up an initial framework, we shall adopt the (possibly over-simplified) assumption that the internal workings of a program are not part of the relevant data.

All this is directed at ensuring that our definition is genuinely empirical, and stated in terms of factors which are (at least in principle) observable. Moreover, our definition(s) should describe what behaviour we would regard as creative without building in, prematurely, proposals about *how* that behaviour might be achieved. If we can maintain a separation between our observational vocabulary and our theoretical models of possible mechanisms, then we can, without circularity, treat questions such as ‘‘which computational mechanisms lead to creativity?’’ as empirical issues. If we incorporate our hunches about the best way to achieve creativity into our definitions of what observable behaviour constitutes creativity, then we have, to a large extent, undermined the empirical nature of the investigation.

Colton (personal communication) has suggested that if a program run produces a set of output items which is highly-rated (by whatever measures are appropriate—see later in this section) in a relatively simple manner, it should be regarded as more creative than a comparable program run which is more laborious in producing the same results. It is unclear what importance could be attached to such information. Is it only to be used as a ‘tie-breaker’ when two different programs produce equally good results? If not, how is it to be weighed up against the ratings of the created artefacts? It is argued in Ritchie (2006, Section 7.5) that if we *define* a particular program behaviour (being ‘transformational’) as constituting creativity, then there is a potential dilemma if there is a mismatch between the implications of a program's internal and external behaviours: what if a program's transformational status supplies evidence contrary to that from the relevant attributes of its output? A similar issue has to be considered here: even if we formalise some notion of the way in which a computation has been achieved, how should we use that information, without pre-judging the issue of what computations are most effective in achieving creativity?

What Kind of Program?

We will be putting forward an analysis of what goes on when a program carries out potentially ‘creative’ activity. The raises the questions: what types of program are we considering, and can we define this class of program without first defining ‘‘creative’’, which might run the risk of circularity?

The broad class of program under consideration operates in the following situation:

- There is some (usually culturally-defined) class of artefacts which the program is to generate. This class of artefacts exists prior to the program being constructed, and is not defined in terms of the workings of the program.
- The class is, in principle, extremely large, possibly infinite.
- Given an item (human or computer generated), there may not be a precise definition of whether it is in that class. That is, membership of the class is ill-defined in some way, being either fuzzy, or subjective, or not subject to algorithmic assessment. Moreover, humans are able to judge (possibly with disagreements) the extent to which an artefact belongs to a particular class.
- Given an item, humans can rate the (usually subjective) ‘quality’ of the item.

These attributes are intended to capture what it is that computer-generated poems, artistic pictures, stories, etc. have in common, and to distinguish them from more orderly constructs such as solutions to equations, the results of numerical calculations, the output of a compiler, or the documents found by an information retrieval system. It is noticeable that the devising of mathematical concepts or conjectures is, historically, regarded within AI as being (potentially) creative (Lenat 1976; Colton 2002), even though such objects are more well-defined than poems or paintings. (Colton (personal communication) has pointed out that the HR program (Steel et al. 2000; Colton et al. 2000) produces only well-formed items of the target class, namely mathematical conjectures.) However, in all other respects they conform to the sketch given above. This means that one could choose to apply our framework to mathematical discovery programs, but would find that some parts of it gave trivial or uninteresting results. (See also Ritchie (2006, Section 4.2) for further discussion.)

Given the essential reliance on human judgement to assess the output of a program, we should make explicit that we have in mind an arrangement whereby a computer program produces items of some sort, and these are appraised (in some suitably controlled way) by human judges. This puts the computer program on the same level, and makes it subject to similar standards, as human creators.

P-Creativity and H-Creativity

Boden (1992) makes the important distinction between *H-creativity* (producing an idea/artefact which is wholly novel within the culture, not just new to its creator) and *P-creativity* (producing an idea/artefact which is original as far as the creator is concerned, even though it might have been proposed or generated elsewhere in the culture, perhaps much earlier in history). This is a very useful distinction, because it clarifies what evidence is or is not relevant. Boden points out that when studying the process of being creative (within a single human or in a computer program) it is *P-creativity* that is at issue, since we are interested in how a single agent can come up with something that is novel *relative to its initial state of knowledge*. A P-creative discovery may prove to be of little use to society because it repeats something that was already known, but that does not render the intellectual or artistic feat of producing the idea/artefact uncreative, viewed in isolation. The mechanisms of creation are what we are interested in here; that is, we shall largely ignore the H (historical) variant, and consider only the P (personal) notion of creativity.

Essential Properties

In discussions of creativity, there seems to be widespread support for the idea that two important, perhaps essential, criteria for deciding whether creativity has occurred are:

Novelty *To what extent is the produced item dissimilar to existing examples of its genre?*

Quality *To what extent is the produced item a high quality example of its genre?*

As noted earlier, discussions of creativity have to be rooted in human creativity, where assessments of the novelty or quality of an artefact are assumed to be applied only to genuine examples of the artefact class (stories, melodies, poems, pictures, etc.). That is, philosophical discussions of creativity (typified by Boden) usually do not consider the prior test:

Typicality *To what extent is the produced item an example of the artefact class in question?*

Since computer creativity is at a relatively basic stage, this more prosaic question has to be asked of items before appreciation of novelty and quality arise. (Novelty and typicality may well be related, since high novelty may raise questions about, or suggest a low value for, typicality; we shall return to this matter later.) For example, poetry generators are currently incapable of reliably producing texts which would consistently be rated as poems by human judges, and joke generators are similarly variable in quality. It therefore makes sense to consider the most fundamental goal of a potentially creative program: is it even doing the job it is supposed to do, by producing artefacts which are of the required sort?

In our formulation below, we have chosen to make typicality and quality primitive elements, describing the (very important) concept of novelty in terms of other constructs, including typicality. The intention is that both typicality and quality will usually be assessed by human judgement and may therefore be partly or wholly subjective.

Judgements by humans will be based on their whole cultural experience and knowledge, and hence are likely to reflect historical comparisons of the artefacts. It might seem that this means human verdicts can assess only H-creativity, not P-creativity. However, if we avoid using human judgements about concepts such as 'originality', 'novelty' or 'creativity', and instead try to confine these verdicts to more basic notions such as typicality and quality, we may be able to keep the focus on P-creativity. Novelty will be taken into account elsewhere in our framework, not treated as a primitive judgement.

In passing, one could consider that the difference between P-creativity and H-creativity ('P-creativity and H-creativity' above) is primarily definable in terms of novelty—an achievement which is P-creative but not H-creative may meet similar standards of quality, but not be greatly novel when judged in a wider context. More specifically, the difference lies in the source or basis of the novelty judgement: the individual, or a whole culture through history. In a sense, there are analogous notions of P-novel and H-novel, which play corresponding roles in defining P-creative and H-creative. We shall refer back to this when discussing Koza et al.'s proposals.

The Framework

In the next section, we shall set out our central idea: a set of criteria which can be applied to a situation where a (potentially creative) program has created some output data (artefacts), and which give some indicators of the extent to which that program has been creative on that occasion. Before we can state the criteria precisely, we need to outline, at a suitable level of abstraction, exactly the type of situation to which our criteria apply. That is, we need a formal statement of the entities and relations that are involved when a program 'creates', so we can set up conditions whose fulfilment might count as evidence of creativity. That formal statement is the topic of this section.

Basic Items

A creating program operates to produce artefacts in some medium. The 'medium' is essentially the output data type of the program. At the level of abstraction adopted here, we can ignore (at least provisionally—see 'Possible Extensions' below) the internal structure of the entities that the program produces, and simply postulate a set, possibly infinite, of *basic items*. This is *not* a definition of what would count as a 'successful' or 'valid' output for the program, merely a statement of the data type which it produces (e.g. strings of words, arrays of pixels). For example, the basic item set for a program intended to produce simple verbal jokes might be the set of finite sequences of words and punctuation symbols.

Rating the Output

We want any assessment of items produced by a program to be as faithful as possible to the two notions of novelty and quality stated in the previous section, taking into account the remarks we made about typicality.

We shall take basic items as being *possible* instances of the intended class of artefacts. More subtly, they may be instances to some degree. We will therefore represent a class of artefacts (the target of the creative exercise) as a mapping from the basic items to the interval [0,1]. (This is equivalent to treating the class as a fuzzy set, but that perspective will not be developed here.) That is, the extent to which an output basic item is a poem/story/picture/joke/melody/etc. will be expressed as this numeric score. This is the notion of typicality introduced in the previous section, and takes one step towards allowing us to capture the novelty criterion. We will decompose the intuitive idea of novelty into two separate factors. Firstly, items which gain low scores on the typicality mapping will be deemed to be dissimilar to the norm for that class. That is, we assume that this mapping encodes the notion of established norms for the artefact class, so that high-scoring items are very much part of the class, and low-scoring ones are implicitly dissimilar from the past practice (in society or culture) which has established the class. This gives an abstract notion of novelty with respect to the genre. Secondly, in the next section, we shall try to formalise the notion of a program producing items which are different from those which guided its original construction. This is a more local, specific form of novelty with respect to a subset of already known artefacts. In this way, we do not treat novelty as a single primitive attribute, but decompose it (slightly) into other factors, which could be loosely glossed as *untypicality* and *innovation*.

As noted earlier, a useful distinction can be made between properties which measure to what extent an item meets the criteria for membership in the intended artefact class (is it a poem/joke/conjecture/etc.?) and further properties whose presence indicate that the artefact is a *good* instance of this type of artefact (a good poem, a funny joke, an elegant or profound conjecture, etc.). This latter evaluation will also be formalised as a mapping from basic items to [0,1]. This attempts to capture the second informal property, quality, introduced earlier.

Both typicality and quality would normally be determined by human assessment of basic items (potential artefacts). For example, the output of the JAPE joke-generator was evaluated by human judges against two standards: ‘‘is this item a joke?’’ and ‘‘how funny is this item?’’ (Binsted et al. 1997). These correspond directly to the typicality rating and the value rating of our framework, so (given the complete raw data from that evaluation) these criteria could be evaluated.

These two mappings—for class membership and quality—may themselves be based on further definitions (e.g. a checklist of properties, perhaps with weightings attached). At present, we have no firm proposals on what this information should be, but we shall call it a *rating scheme*, and list it separately so that the distinction can be made in our later definitions, abstracting away from its internal details. We shall also assume an operation *APPLY* which, given a rating scheme, creates a mapping to [0,1]. Notationally, we shall usually make the abbreviation of using the name of a rating scheme as if it were the function which APPLY would create; that is, writing $\mathtt{rat(X)}$ as short for $\mathtt{APPLY(rat)(X)}$. The set of possible rating schemes for a set A will be written as ‘‘$\mathcal{RAT}(A)$ ’’.

The Objects Generated

We can now use a rating scheme as representing a class of basic items.

Definition 1 An *artefact class* consists of a set $\mathcal{B}$ and a rating scheme for $\mathcal{B}$

Here we are using a single rating scheme to capture both inherent, measurable properties of a basic item, such as syllable counts, and more subjective aspects. In particular, discussions of creativity sometimes argue that the *expectations* of the audience are relevant—an artefact which exceeds or violates the expectations of the perceiver may be rated more highly. Here, those aspects are packed into the notion of an artefact class, on the grounds that expectations are in a sense a subjective notion of what typifies a particular genre. This should suffice at least as a first approximation.

As noted above, we also need a rating scheme to represent the quality of the generated artefact.

Definition 2 A *value-based artefact class* consists of a triple ($\mathcal{B}$, *typ*, *val*), where $\mathcal{B}$ is a set (the basic items) and $typ, val \in \mathcal{RAT}(\mathcal{B})$ (the *typicality ratings* and the *value ratings* respectively).

Inspiring Set, Program and Results

The origins of a generating program are pertinent to assessing its creativity, as is often acknowledged by worries about ‘results being pre-programmed in’. However, it is beyond the scope of this paper to formalise this part of the process. For the moment, we shall adopt a very simplified framework, as follows. The construction of the program is influenced (either explicitly or implicitly) by some subset of the available basic items. This subset, which we will call the *inspiring* set, could be all the relevant artefacts known to the program designer, or items which the program is designed to replicate, or a knowledge base of known examples which drives the computation within the program.

The motivation for including the inspiring set (notated *I*) in our formal account is that creativity could be viewed as depending on the extent to which the program does or does not replicate (or closely imitate) the instances which guided its design.

Where *I* consists of a wide variety of examples with which the program designer was acquainted, it may be very difficult to define its exact extent, so that some of our formal proposals involving this set will be hard to apply in practice. However, some of the studies we shall discuss later show how *I* may be a concrete set of instances which drive the creating program’s computations.

It has been suggested (by a reviewer of this paper) that it is also necessary to consider the case where there is no inspiring set. This is different from the analyst not knowing what inspiring set had been involved, or the results being completely different from the inspiring set; this condition involves there being no such items at all ($I = \emptyset$). For there to be no inspiring set, a program (of the general sort outlined in ‘Assumptions’) would have to come into being without the designer being guided by any exemplars of the artefacts to be created, and with no (semi-)automated process (e.g. machine learning, case-based reasoning) which was based on exemplars. Perhaps this could be the case where a program was designed to create what we have called ‘‘basic items’’, possibly randomly, with no previous choice of artefact class. In such a situation, it is conceivable that a program creating random strings of words and punctuation might happen to produce a poem or a story. This is not typical of the situations focussed on here; however, our formalisation does allow for an empty inspiring set as a special case.

As noted above, we will not offer a dissection of how a (potentially creative) program comes into being. For example, the relationship of the inspiring set to the *program* is of interest. The program design/construction could be human-crafted or automated; parts of the process could even be random. There is also the question of input parameters used for different runs of a program. The creativity of the program could in principle be assessed according to these aspects. As our proposals constitute an initial framework, we have confined our attention to a narrower range of phenomena, and have not attempted to describe these possibilities within our criteria (but see ‘Possible Extensions’, later). For our purposes, a program produces a set of basic items, *R*.

Evidence for Creativity

Preliminaries

The various formal constructs set out above allow us to state some criteria which could be applied in deciding how successful a potentially creative program is, or has been, in a particular run. We do not consider the idea that the creativity of a program can be considering independently of the sets of results that it produces, i.e. the outputs from its runs. We do not claim that all these criteria are essential to an assessment of creativity, nor that they are the only such criteria that could be considered; rather, they are a first draft of a general catalogue of relevant factors (see ‘Discussion’, later).

These criteria tackle the question of appraising the output of a generating program in isolation, without knowledge of the program’s construction or internal workings. This is comparable to the assessment which people routinely make of human-created artefacts (see ‘Assumptions’ above).

In the formal criteria listed below, we assume a value-based artefact class $(\mathcal{B}, typ, val)$, an inspiring set *I* (a subset of $\mathcal{B}$), and a program which has produced a set of results *R* (also a subset of $\mathcal{B}$). That is, our criteria apply to the results of a particular run of the program, or of a set of runs where the results have been aggregated without maintaining links to the runs which created them.

It should be possible to generalise these ideas to cover a sequence of ‘‘runs’’ where a sequence of separate result sets is produced, so as to assess the creativity of a program in general rather than a single run of that program. That elaboration is not explored here (but see ‘Possible Extensions’ below).

The criteria involve the ratings *typ* and *val*, the result set (*R*) and the inspiring set (*I*), combined and compared in various ways.

For convenience, we employ the following notation:

$T_{\alpha,\beta}(X) def{=}\{x \in X | \alpha \leq typ(x) \leq \beta\}$: The subset of X falling in a given range of typicality.

$V_{\alpha,\beta}(X) def{=}\{x \in X | \alpha \leq val(x) \leq \beta\}$: The subset of X falling in a given range of quality.

$AV(F, X) def{=}\frac{\sum_{x \in X} F(x)}{(X)}$: The average value of a function *F* across finite set *X*.
$ratio(X, Y) def{=}\frac{(X)}{(Y)}$: The relative sizes of two finite sets *X*,*Y*, where $Y \neq \emptyset$.

The Criteria

As noted earlier when we introduced the notion of typicality, the first goal which a potentially creative computer program must meet, before aspiring to novelty or quality, is to produce items which are judged as being instances of the intended class. This factor is represented here by the *typ* scores of result items, so one way to capture the success of the program on this rudimentary goal would be to look at the average value of this score across the result set (notation as defined above):

Criterion 1 $AV(typ, R) > \theta$, *for suitable* θ.

Here, θ is some threshold to be chosen in the particular situation, indicating the score at which items are deemed to have reached acceptable typicality. It is not obvious or trivial to define such a threshold. We shall return to this matter in 'Discussion' below.

Still focussing on this elementary notion of success, another possible condition to examine is whether a significant proportion of the result set are indeed examples of the intended genre, by scoring well on typicality:

Criterion 2 *ratio* $(T_{\alpha,1}(R), R) > \theta$, *for suitable* α, θ.

In this criterion, there are two parameters to be chosen, α and θ. The expression $T_{\alpha,1}(R)$ can be thought of as 'created artefacts which conform to the established definition of, or norms for, the class'.

These two criteria can be seen as conflicting with the novelty requirement, but, as discussed in 'Assumptions', merely succeeding in conforming to the norms of the chosen genre is an achievement for a computer program. It is therefore worthwhile including some tests for this level of success. Branching out into producing untypical items—as suggested by novelty—is a more advanced level of creativity, which we will attempt to capture in further criteria, below.

When we widen the scope of the assessment to consider quality (i.e. are the generated artefacts rated as being good, in some sense?), then there are two analogous criteria to the above typicality conditions—average quality being above some threshold, and a large proportion of the results being above some quality threshold, with $V_{\gamma,1}(R)$ meaning 'the high-quality artefacts':

Criterion 3 $AV(val, R) > \theta$, *for suitable* θ.

Criterion 4 *ratio*$(V_{\gamma,1}(R), R) > \theta$, *for suitable* γ, θ.

A more subtle criterion would be to confine attention to those items which met some typicality threshold (i.e. were genuine instances of the artefact class) and then ask what proportion of these also scored well in terms of quality:

Criterion 5 *ratio* $(V_{\gamma,1}(R) \cap T_{\alpha,1}(R), T_{\alpha,1}(R)) > \theta$, *for suitable* α, γ, θ.

As Boden (1992) makes clear, a higher rating of creativity should be accorded to the production of artefacts which do not conform closely to the norms of the genre (typicality), but which nevertheless are rated highly when judged on their merits (quality). We can model this judgement in various ways, depending on what we choose to compare the set of untypical high-valued items ($V_{\gamma,1}(R) \cap T_{0,\beta}(R)$) with. One possibility would be to compare with the entire set of outputs (R):

Criterion 6 *ratio* $(V_{\gamma,1}(R) \cap T_{0,\beta}(R), R) > \theta$, *for suitable* β, γ, θ.

This criterion asks whether a large proportion of the program's output falls into the (supposedly desirable) category of untypical but high-valued. Alternatively, we could compare with the set of all untypical items ($T_{0,\beta}(R)$):

Criterion 7 *ratio* $(V_{\gamma,1}(R) \cap T_{0,\beta}(R), T_{0,\beta}(R)) > \theta$, *for suitable* β, γ, θ.

That is, what proportion of the untypical items are of good quality? If there are no untypical items, (i.e. $T_{0,\beta}(R) = \emptyset$ for whatever value of β is deemed appropriate), then this criterion could not be used. A comparison could also be made with the set of typical high-valued items ($V_{\gamma,1}(R) \cap T_{\alpha,1}(R)$):

Criterion 8 *ratio* $(V_{\gamma,1}(R) \cap T_{0,\beta}(R), V_{\gamma,1}(R) \cap T_{\alpha,1}(R)) > \theta$, *for suitable* α, β,γ, θ.

This criterion, as originally stated in Ritchie (2001), has certain formal flaws. Firstly, unlike most of the other criteria, the left hand side value does not range between 0 and 1, but is unbounded. Secondly, in cases where there are no high-value highly typical values ($V_{\gamma,1}(R) \cap T_{\alpha,1}(R)$ is empty), the ratio involves division by zero. The revised version below avoids these undesirable properties, but makes a slightly different comparison—high-value untypical items and all high-value items:

Criterion 8a *ratio* $(V_{\gamma,1}(R) \cap T_{0,\beta}(R), V_{\gamma,1}(R)) > \theta$, *for suitable* α, β,γ, θ.

We have now introduced all the threshold parameters which will be used in these criteria: α (threshold to achieve high typicality), β (limit of untypicality), γ (threshold to achieve good quality), and θ (general comparison level in all criteria).

Now we turn to another aspect of novelty, namely the extent to which the program reproduces known examples. Earlier, we proposed criteria 1 and 2 to allow consideration of a lower level of attainment than is sometimes considered in more philosophical discussions. Similarly, it is useful to consider a way of defining 'mere replication', as this may, particularly in the earlier stages of development of a program, be a useful goal to achieve. Ritchie and Hanna (1984) comment that even if the AM program (Lenat 1976) had not produced anything original, but had 'merely' shown a computational route by which many interesting (known) concepts could in principle be reached, that would have been a useful finding. The next criterion formalises that idea—replicating a large proportion of the inspiring set I:

Criterion 9 *ratio* $(I \cap R, I) > \theta$ *for suitable* θ.

However, there is general agreement that producing more than just the inspiring set is a symptom of creativity, and this can be described by considering the ratio of the whole result set to the subset which are replications:

Criterion 10 *ratio* $(R, I \cap R) > \theta$, *for suitable* θ.

This is another criterion whose original statement (Ritchie 2001a) has formal flaws. Its left hand side value does not range between 0 and 1, and, in cases where there are no replicated values ($I \cap R$ is empty), the ratio involves division by zero. A revised version could be formed by inverting the ratio, but this would lead to a criterion with ‘‘less than’’ as its central operator, inelegantly different from the other criteria. A better revision is the following:

Criterion 10a $(1 - ratio(I \cap R, R)) > \theta$, *for suitable* θ.

Let us turn now to novel results (i.e. items not in the inspiring set): $R - I$. Simply producing unknown items is not interesting unless they have some significant properties. Firstly, they could, on average, be exemplars of the chosen genre:

Criterion 11 $AV(typ, (R - I)) > \theta$, *for suitable* θ.

Another possible success condition would be that novel items were, on average, highly valued:

Criterion 12 $AV(val, (R - I)) > \theta$, *for suitable* θ.

To be more demanding, we could ask whether novel and highly typical items form a significant proportion of the results:

Criterion 13 $ratio(T_{\alpha,1}(R - I), R) > \theta$, *for suitable* α, θ.

Similarly, we could ask if novel and high quality items are a high proportion of the results:

Criterion 14 $ratio(V_{\gamma,1}(R - I), R) > \theta$, *for suitable* γ, θ.

The above are the only criteria originally presented in [Ritchie 2001], but there are a few more which suggest themselves on the basis of the last few in the list. We could similarly consider whether highly typical items formed a high proportion of the novel items:

Criterion 15 $ratio(T_{\alpha,1}(R - I), (R - I)) > \theta$, *for suitable* α, θ.

Similarly, the proportion of high-valued items amongst the novel items could be considered:

Criterion 16 $ratio(V_{\gamma,1}(R - I), (R - I)) > \theta$, *for suitable* γ, θ.

The last two conditions to consider are the occurrence within the novel results of items which are of good quality and highly typical (i.e. demonstrate creativity within existing norms) or are of good quality and untypical (i.e. demonstrate some original deviation from past practice):

Criterion 17 $ratio(V_{\gamma,1}(R - I) \cap T_{\alpha,1}(R - I)), (R - I)) > \theta$, *for suitable* α, γ, θ.

Criterion 18 $ratio(V_{\gamma,1}(R-I) \cap T_{0,\beta}(R-I), (R-I)) > \theta$, *for suitable* β, γ, θ.

All of criteria 11–18 would be inapplicable in the case where $(R-I) = \emptyset$. In the case $I = \emptyset$ (see 'The Framework', earlier), criteria 9 and 10 would have to be avoided, but other criteria involving I should still function in an orderly fashion.)

Related Proposals

The central ideas of the previous section were first presented in Ritchie (2001), and have been followed by a number of elaborations (Colton et al. 2001; Pease et al. 2001). We shall review this subsequent work here, as well as some separate proposals by (Koza et al. 2003), before considering some applications of the ideas to actual systems.

Fine Tuning

Colton et al. (2001) (CPR) provide a small elaboration of the basic framework outlined above. Their aim is to define more precisely what would count as *fine-tuning* a program—that is, arranging the input knowledge of a program so as to produce particular output: ''A program which has been carefully tailored in order to produce very specific artefacts cannot be claimed to be a good program on the grounds that it produces those artefacts.''

The basic framework already allows a crude indication of such a situation, namely where the inspiring set and the result set are identical, or almost so. Such a scenario might be rated poorly in creative terms (it would score badly on criterion 10/10a, and, *a fortiori*, on criteria 13 and 14), but it could still be interesting from a research point of view, since—as noted earlier—it might well demonstrate, and test by implementation, a mechanistic route from very simple data to interestingly complex output.

CPR refine this approach. They posit a program working with some set K of input knowledge, and then consider the effect, on the set of high-valued output from the program, of removing some knowledge K'; i.e. starting from $(K-K')$ instead. The output items in question would, in the notation used above, be $V_{\gamma,1}(R)$ for some chosen γ. CPR notate these as V_K (where the full knowledge set is used) and $V_{(K-K')}$ where K' is omitted, and then define:

- K' is *creatively irrelevant* if $V_K = V_{(K-K')}$
- K' is *creatively useful* if $V_{(K-K')} \subset V_K$
- K' is *creatively destructive* if $V_K \subset V_{(K-K')}$

If I is, as before, the inspiring set, then any output artefacts belonging to I are 'reinventions', and high-valued output artefacts not in I are what CPR call the *creative set*. They suggest defining K' (the change in input knowledge) as being *fine-tuned* if $V_K - V_{(K-K')}$ intersects with I (i.e. includes some reinventions) and does not overlap with the creative set set produced with input knowledge K (i.e. includes none of the high-valued new inventions).

CPR also propose a definition of a degree of fine-tuning for a set of knowledge K', in the case where there are at least some high-valued new items in $V_K - V_{(K-K')}$.

$$ft(K') = \frac{|I \cap (V_K - V_{(K-K')})|}{|(V_K - V_{(K-K')}) - I|} \tag{1}$$

(These formulae have been set-theoretically simplified from CPR's presentation)

They offer two possible definitions of how fine-tuned a program P is, starting from an inspiring set I and a knowledge set K:

m_1: Take the union of all knowledge sets K' (subsets of K) which are fine-tuned (by the definition above). Divide the size of this set by the size of K.
m_2: Compute the maximum value of $ft(K')$ for all subsets K' of K.

CPR observe:

> If m_1 is greater than 0 or m_2 greater than 1, we can claim that P using K has been fine-tuned to some extent. If m_1 is 1, P using K is completely fine-tuned, in the sense that every item of knowledge in K contributes to some subset (which non-redundantly contributes to V_K) which is fine-tuned. If m_2 is greater than 1, then there is at least one such subset of K which is used more to replicate known artefacts than to find new ones of value.

CPR also suggest that this perspective allows the comparison of the effects of different pieces of knowledge. If P can be run with K set either to $K_0 \cup K_1$ or $K_0 \cup K_2$, then the various fine-tuning measures can be compared for $K' = K_1$ and $K' = K_2$.

Other Formalisations

Pease et al. (2001) (hereinafter ''PWC'') acknowledge the basic framework in 'Evidence for creativity' above, and offer some further formal definitions in the area of computational creativity. Although the basic framework attempts to be relatively neutral with respect to substantive theories of creativity, building in as few claims as possible about what gives rise to creativity, PWC are bolder, incorporating some relatively specific views about creative mechanisms into the constructs they propose. We shall therefore discuss their ideas in two groups: very general extensions, close to the neutral spirit of the framework, and more specific proposals embodying particular ideas about creativity.

General Extensions

Subsets of inspiring set. PWC suggest that a fine distinction might usefully be made between two possible forms of I: I_S (all items known to the program designer) and I_W (the items the designer knows have influenced the program design). However, they do not develop this refinement or incorporate it into the basic framework.

Novelty as complement of typicality. PWC suggest novelty could be formalised as the fuzzy complement (with respect to the set R of results) of the fuzzy set defined by the *typ* mapping, but this is not connected to other proposals.

Quality and affect. PWC point out that the quality of an artistic item is sometimes related to the emotional response it evokes. Assuming that a number of human subjects numerically rate each item for either positive or negative response, PWC offer (without much discussion) 5 possible measures of quality: total of all ratings, average of all ratings, total of positive ratings, total positive minus total negative, and average of positive ratings.

More Specific Proposals

Program process: randomness, meta-level and complexity. PWC delve further into the process whereby a program produces an artefact (cf. 'Assumptions' earlier). They offer a formula for the ''randomness'' of an artefact, based on a probability distribution for particular outputs given particular inputs, and a distance measure between artefacts. Also, they present 5 possible ways of computing the *complexity* of an output artefact (and one for the complexity of a program), mostly based on comparing sets of knowledge items used (cf. 'Fine tuning' above). They say these could yield ''a useful measure of novelty'', but do not explain why complexity and novelty are synonymous. They quote (Bundy 1994) as suggesting that the complexity of a concept space might be relevant to creativity, but their measures mainly assess the complexity of a single artefact, not of a concept space.

PWC suggest that meta-level processing is central to creativity, by proposing to partition a program's knowledge into either *object* or *meta*, and labelling the degree of novelty of an output item according to which types of knowledge have been used in generating it (along with some complexity conditions).

Surprise. PWC offer a formula for the surprise of an artefact, depending on a similarity measure for events, and a probability distribution over events.

PWC–Overall

Although PWC's proposals at first glance appear to be an elaboration of our basic framework, on closer inspection it is not clear where they would fit into, or alongside, our formalisation. The basic framework is a first attempt at setting out which factors might be worth measuring in order to determine whether a program has behaved creatively. It has deliberately been stated conservatively, with as few substantive claims about creativity as possible while considering how established ideas such as ''novelty'' and ''quality'' might be taken into account. PWC, on the other hand, have taken some quite specific ideas about creativity, and suggested some preliminary formalisations for them. They move from treating novelty as a relatively primitive notion (similarity to existing artefacts, or to existing norms) to loading it with specific ideas about what constitutes creativity. It is unclear whether there is still a difference, for PWC, between novelty of an artefact and an artefact being evidence of creativity: novelty seems to have become the central construct.

As their paper is relatively short, they do not give full supporting arguments for the decisions they have made in designing their formula, some of which are not completely obvious. PWC also do not link the various proposals together, so the ideas remain relatively unconnected.

Koza et al.'s Guidelines

Koza et al. (2003) discuss the extent to which genetic programming has been successful as an approach to machine intelligence. They are not primarily concerned with characterising creativity *per se*, but the issues which they consider have some overlap with the concerns here. In contrast to our proposals, Koza et al. are looking for empirical ways which might show that a *method* (of problem-solving or of discovery) is *successful* (rather showing that a *program* is *creative*).[1]

Koza et al. claim that ''genetic programming now routinely delivers high-return human-competitive machine intelligence'', and then say in more detail what they mean by ''routine(ly)'', ''high-return'', ''human-competitive'' (and ''machine intelligence'', though that turns out to be less closely related to the creativity factors that our basic framework focusses upon).

Koza et al. term a method as *routine* if it can be applied to a new problem, or a wide range of problems, with relatively little specialised adjustment. That is, routine application is an indication of generality.

A method is *high-return*, on the other hand, if most of what it does is automated, without a large investment of hand-crafted effort. Koza et al. view this as a *ratio* (''artificial-to-intelligence'') which is high if value added by the problem-solving method is significantly greater than the contribution of the human designers.

A computed result (what we are here calling an ''artefact'') is *human-competitive* if it meets one or more of eight guidelines which Koza et al. provide. In the terminology we have been using here, these tests can be viewed as various combinations of quality (e.g. ''The result solves a problem of indisputable difficulty in the field''), and/or H-novelty (e.g. ''The result is publishable in its own right as a new scientific result, independent of the fact that the result was mechanically created'') and/or P-novelty (e.g. ''The result is equal to or better than a result which was considered an achievement in its field at the time it was first discovered''.

It might be possible to form a number of alternative definitions of ''creative'' for computer programs by combining some of Koza et al.'s notions (since no single definition in their repertoire corresponds directly with what we are focussing on). For example a program which achieves human-competitiveness criterion [A] (''The result was patented as an invention in the past, is an improvement over a patented invention, or would qualify today as a patentable new invention'') and is also high-return might be deemed to be creative.

The lack of a need for specialised tuning (as in ''routine'') does sound as if it might affect judgements of creativity, but this intuition is more naturally captured by their next attribute, ''high-return''. That attribute is predicated of a *method*, but a similar perspective could be taken on a specific program (as, in fact, Koza et al. do, when they cite the Deep Blue chess program as an example of a highly successful program with a disappointingly low ''AI ratio''). The desirability of being ''high-return'' is similar to the idea that the creativity of a program is less if the program

[1] They are also very much focussed on what is sometimes called 'knowledge-poor' programming, where a very general method does not need much hand-crafted domain knowledge to succeed. This focus is underlined by the fact that they group together, for comparison with the method being assessed, ''live human players'' and ''human-written computer programs''.

designer has injected a great deal of hand-crafted knowledge. The question is: how can this be made precise, or even quantified? The use of the ‘‘inspiring set’’ in our criteria, and Colton et al.’s definition of ‘‘fine-tuning’’ are very preliminary attempts to get a hold on this idea.

The use of patentability (to show human-competitiveness) is an indicator of both and quality and novelty. For example, guidelines for US patents[2] state that ‘‘the claimed invention as a whole must accomplish a practical application. That is, it must produce a ‘‘useful, concrete and tangible result’’. This is a clear quality requirement. These guidelines also stipulate the need for patent examiners to ‘‘conduct a thorough search of the prior art’’, an H-novelty constraint. (Koza et al. broaden this to P-novelty, by allowing something that was patented in the past to qualify as human-competitive.) The Patent Act 1977 in the UK[3] demands that an idea be ‘‘novel’’, ‘‘involve an inventive step’’ and ‘‘have industrial application’’. An invention has an inventive step ‘‘if it is not obvious to a person skilled in the art’’; a further guideline (for biotechnology) glosses this as situations where known theory would make the outcome obvious.

Patentability (past or present) is quite a high standard for computer programs, in the current state of the art (although Koza et al. cite some examples of patented inventions by programs). Moreover, it would not, on its own, take into account the contribution by the computer program (as opposed to knowledge or ideas supplied by the programmer); hence Koza et al.’s use of ‘‘high-return’’. Also, the focus (at least in the USA and the UK) on practicality means that patentability is inapplicable in many of the areas explored by research into computational creativity—the UK’s 1977 Act explicitly excludes from eligibility for patenting: ‘‘a discovery, scientific theory or mathematical method; a literary, dramatic, musical or artistic work or any other aesthetic creation whatsoever; a scheme, rule or method for performing a mental act, playing a game or doing busines, or a program for a computer; the presentation of information’’.

On the whole, there is a overlap in concerns between Koza’s framework and ours, but the methodological aims are slightly different.

Applications of the Criteria

A Poetry Generator

Gervás (2002) analyses one of his poetry-generating programs, WASP (Gervas 2000), using the criteria in ‘Evidence for Creativity’ above. WASP’s inspiring set is a particular 16th century Spanish classical sonnet, which is taken as establishing the allowable line lengths (in syllables) and stress patterns. The program then tries to create four-line stanzas (*cuartetos*) that are part of the sonnet form. The generation uses a set of line patterns (enforcing constraints of syllable count and stress patterns, and based on the part-of-speech sequences occurring in the lines of the sonnet

[2] http://www.uspto.gov/web/offices/pac/dapp/opla/preognotice/guidelines101_20051026.pdf

[3] http://www.patent.gov.uk/patentsact1977.pdf

chosen as starting point) and a set of vocabulary (the words of the original sonnet and ‘‘a number of additional words’’, all annotated with syntactic and stress information). Where more than one word can satisfy the metrical constraints, a random choice is made, so that the process is non-deterministic and may yield different output on different runs from the same input.

For testing, 14 different initialisations were used, and 12 runs were made with each. The 168 resulting items were evaluated by volunteers, who scored each item on syntactic correctness (0–5) and ‘‘aesthetic qualities’’. The syntactic correctness score and the number of lines in the stanza (since some may fall short) are combined to give a *typ* score, the aesthetic verdict providing the *val* score. Gervás explores the effect of combining syntax and line-count scores with different weights, carrying out assessments for weightings of 50/50, 30/70 and 70/30.

Gervás gives a table of values for the first 14 criteria (i.e. those listed in Ritchie (2001)), but observes that as none of the inspiring set I appear in the result set R, criteria 11–14 replicate the values of criteria 1–4, and criteria 9 & 10 give pathological results (0.00 and division by zero); see Table 1. Gervás also experiments with varying values for the thresholds α,β (the boundary *typ* value between typical and atypical items) and γ (the *val* value at which an item is deemed valuable). He considers 5 different possibilities: equal thresholds, being either high (0.7), medium (0.5) or low (0.3); high α,β and low γ; low α,β and high γ. He notes that these thresholds affect only criteria 2, 4, 5, 6, 7, 8 (ignoring 9–14 for reasons stated earlier), and offers Table 2 to summarise the effects (using the 50/50 weighting for *typ* calculation).

Gervás observes that threshold changes can cause large changes in the overall rating for the criteria, and suggests that perhaps thresholds should be set beforehand depending on the domain and/or the objectives of the system. He also speculates that perhaps the typicality scale should have *two* thresholds: a low one below which artefacts are deemed atypical, and a high one above which they are typical; presumably artefacts in between are neither typical nor atypical. He does not say exactly where these boundaries should be inserted in the definitions of the criteria, but a plausible scheme would be to use the higher threshold where α is used, and the lower one for β.

Table 1 Applying criteria to WASP data

Crit.	Informal meaning	50/50	70/30	30/70
1	Average typicality	0.71	0.67	0.75
2	Typical results/results	0.54	0.48	0.79
3	Average quality	0.47	0.47	0.47
4	Good results/results	0.24	0.24	0.24
5	Good typical results/typical results	0.36	0.34	0.29
6	Good atypical results/results	0.05	0.08	0.01
7	Good atypical results/atypical results	0.12	0.16	0.06
8	Good atypical results/good typical results	0.28	0.52	0.05

Table 2 Varying thresholds for WASP evaluation

Crit.	High	Medium	Low	α, β high γ low	α, β low γ high
2	0.54	0.88	0.89	0.54	0.89
4	0.24	0.50	0.68	0.68	0.24
5	0.36	0.57	0.77	0.89	0.28
6	0.05	0.00	0.00	0.21	0.00
7	0.12	0.00	0.00	0.45	0.00
8	0.28	0.00	0.00	0.44	0.00

Gervás also considers how the Colton et al. measure of 'fine-tuning' could be applied to WASP, and concludes that *ft* will always have a value of 0, since there are no re-inventions within the output set. However, he points out that this is because the fine-tuning measure depends, with excessive simplicity, on a created item being *identical* to an inspiring set element (in order for it to count as a re-invention). Gervás rightly observes that where artefacts are *extremely similar* to elements of *I*, that should contribute to a verdict of fine-tuning (see 'Possible Extensions').

Pereira et al. (2005) give a resume of the WASP study, and go on to carry out similar examinations of two other programs, Divago and Dupond.

A Concept Generator

Divago (Pereira 2005) generates new concepts using *conceptual blending* (Fauconnier and Turner 1998), with a genetic algorithm to explore the space. It starts from a pair of concepts, such as (*horse, bird*) (from a knowledge base of known concepts), and tries to find possible blends of these which meet a pre-specified goal (e.g. ''something which flies and is a transport means''). Pereira et al. describe experiments in applying Divago in three domains: horse–bird combinations, interpretations for noun–noun compounds such as ''pet fish'', and novel creatures (from a KB of three). In applying the creativity criteria to the results, they do not use human judgements (as in the WASP study) to determine ratings for either *typ* or *val*. Instead, they devise ways of computing these automatically from the knowledge sources available to Divago, by making certain assumptions:

(a) The concepts in the original KB count as *I*.
(b) Typicality is measured by closeness to *I* (i.e. the KB defines the norms for artefacts), using an edit-distance.
(c) Value is measured by how well the artefact meets the input goal (a factor which is already used to guide Divago's search).

Using the settings $\alpha = \beta = \gamma = 0.5$, the results for the original 14 criteria are as given in Table 3 (here, criterion 10 is recomputed as 10a). Pereira et al. make a number of observations about these findings:

- ''The system is better at producing good items than typical ones.''
- As the system is guided in its search by maximising *Relevance*, any typicality achieved is an incidental side-effect of this process.

Table 3 Applying criteria to Divago data

Crit.	Informal meaning	Horse–Bird	Noun–Noun	Creature
1	Average typicality	0.443	0.543	0.343
2	Typical results/results	0.273	0.563	0.333
3	Average quality	0.504	0.782	1.0
4	Good results/results	0.636	0.781	1.0
5	Good typical results/typical results	1.0	0.778	1.0
6	Good atypical results/results	0.364	0.344	0.667
7	Good atypical results/atypical results	0.5	0.786	1.0
8	Good atypical results/good typical results	0.5	0.786	2.0
9	Results in *I*/*I*	0.00	0.036	0.0
10a	1-(Results in *I*/Results)	1.0	0.938	1.0
11	Average typicality of new results	0.406	0.513	0.308
12	Average quality of new results	0.483	0.831	1.0
13	Typical new results/new results	0.273	0.500	0.333
14	Good new results/results	0.636	0.781	1.0

- The apparently better results for the *Creatures* domain can be explained in terms of properties of the data and the search spaces involved in the three domains.

A Paraphrase Generator

Pereira et al. also carried out a similar study with the Dupond system (Mendes et al. 2004), a program which generates approximate paraphrases of sentences by replacing selected words with synonyms or hypernyms, in a manner with an element of randomness. Human subjects were presented with 192 items which consisted of a human-written sentence and three Dupond-generated variants of it: one using first hypernyms (H1), one using synonyms with fewer senses (L), and one using synonyms with more senses (M). The subjects classified the generated sentences relative to the human source sentence, on 3-point scales, for being more original (O), having the same meaning (S), being more comprehensible (U).

As in the Divago study (above), *typicality* was taken to be the opposite of *originality* (the O ratings). Value was measured in two alternative ways: either the S rating or the U rating.

Once again, the parameters α, β, γ were all set to 0.5, in the absence of any principled reason for any other values. Full criteria scores with $val = S$ are provided, but only for criteria 3–8 for $val = U$; all these are as in Table 4, with the U values in brackets.

Pereira et al.'s Summing Up

Pereira et al. rightly observe that it is not realistic to compare scores achieved by their different systems, as the applications are so different, although they believe that more general claims can be made such as ‘‘Divago tends to be more creative

Table 4 Applying criteria to Dupond data

Crit.	Informal meaning	H1	L	M
1	Average typicality	0.559	0.49	0.554
2	Typical R/R	0.75	0.475	0.65
3	Average quality	0.295(0.146)	0.495 (0.238)	0.426 (0.294)
4	Good R/R	0.1(0)	0.5(0.05)	0.4 (0.025)
5	Good typical R/typical R	0.1(0)	0.474 (0.053)	0.462(0.038)
6	Good atypical R/R	0.025(0)	0.3 (0.025)	0.125 (0)
7	Good atypical R/atypical R	0.091(0)	0.545(0.045)	0.333(0)
8	Good atypical R/good typical R	0.333(DIV BY 0)	1.333(1.0)	0.417(0)
9	Results in *I*/*I*	0.015	0.015	0.026
10a	1-(Results in *I*/Results)	0.985	0.985	0.975
11	Average typicality of new R	0.559	0.49	0.554
12	Average quality of new R	0.295	0.495	0.426
13	Typical new R/new R	0.75	0.475	0.65
14	Good new R/R	0.1	0.5	0.4

than Dupond in a variety of criteria'', and that one can argue that a particular approach (e.g. genetic algorithms) leads to more creativity. Even these conclusions may be over-interpretation of the findings, given the degree of arbitrariness in the choice of constants (α, γ) (which Pereira et al. remark on) and the varying interpretations of notions such as ``typicality'' and ``quality''. As Pereira et al. comment, care is needing in deciding how to assess these basic factors.

They also remark that the criteria involve typicality and value rather than novelty and usefulness, conceding that value and usefulness might be equated but that typicality and novelty are opposites. This perspective could be contested. The criteria do not include novelty *as a primitive construct*, but do address novelty in terms of the comparison with the inspiring set and also the notion of atypicality; that is, novelty is decomposed into other, more basic, relations.

These authors conclude that although the criteria represent ``consensual properties'' and ``principles generally accepted'', the problem lies in the instantiation of the criteria and the lack of canonical problems.

A Melody Generator

(Haenen and Rauchas 2006) describe a program which generates melodies, with some assessment using our criteria (as well as some statistical comparisons of human judgements about the computer output). Each output was generated using the characteristics of exactly one famous melody. Human judges were asked to rate generated melodies on two scales: typicality (1–3) and quality (1–5). Typicality was defined as being similarity to other melodies in a set made up of the 21 'inspiring' items (famous melodies), the 21 generated melodies, and 6 randomly generated melodies. To evaluate the formulae for the criteria, α (typicality threshold) was set to 2/3, equivalent to the central score (``somewhat typical''), and two values of γ

(quality threshold) were explored: 3/5 (the central score, ''indistinct melody, not very interesting'') and 3.87/5 (where 3.87 was the average rating given by the judges to human-composed melodies). Haenen and Rauchas report that 18 of 21 artefacts exceeded the α value, 16 exceeded $\gamma = 0.667$ and 8 exceeded $\gamma = 0.773$, and remark that ''no results from the inspiring set appeared in the result set''; they therefore do not discuss criteria 9 to 4 in detail (cf. the WASP study, above). (*I* was in effect a different singleton for each generated artefact.) Haenen and Rauchas, following the practice of Gervás and Pereira et al., list the values for the criteria (evaluated for the set of 21 generated artefacts), for their values of α and γ. They comment on the proportion exceeding the thresholds (cf. criteria 2, 4), and the fact that one piece was rated as atypical but of good quality (cf. criteria 6, 7, 8). From the latter artefact, they conclude that the program ''is able to produce creative melodies''.

Haenen and Rauchas, in their short paper, do not discuss their use of the criteria at length (and their statistical measures of the human judgement data may be more illuminating), leaving some questions. Were there any implicit inspiring items in the way that the program mechanisms were devised? Is similarity to the set of items used in the study (inputs and outputs) a suitable yardstick of typicality? What, if anything, can be inferred from the 8 criteria values listed for these results?

Discussion

Use of the Criteria

Probably the most central issue arising from the applications described in the previous section is the intended usage of the criteria. Those who have applied them to particular systems, as reviewed above, have treated all the 14 original criteria as having equal status, and as providing a 14-point checklist or yardstick which will provide a quantitative profile of the creativity level of the program. This is not how they were intended. The aim of the original presentation of the criteria (Ritchie 2001) was to show how to make precise *some* factors which are of interest when assessing a potentially creative program, in order to illustrate a range of possibilities which would-be assessors of programs could select from, add to, or modify in a systematic way. There is no consensus on what counts as creative, particularly when considering programs. A framework such as the one outlined here allows for multiple definitions of creativity (or definitions of different styles or levels of creativity). As mentioned in 'Assumptions', for a computer to manage even to produce 'normal' or 'typical' exemplars of a genre (thus scoring well on criteria 1 and 2) is a worthwhile task, but it is a different level of achievement from producing highly-valued but untypical artefacts (criteria 6, 7, 8). This emphasises that the set of criteria listed here should be considered as a repertoire from which one might draw. The fact that (as observed in Ritchie (2001)) different criteria seem to lead in different directions with respect to the underlying intuition about creativity is not a problem. Rather, one can define different variants by suitable choices and combinations of criteria.

More formally, the various parameters α, β, γ, θ in the above definitions are a source of flexibility. Also, whatever criteria are formally defined (those given above being an illustrative set) can be used selectively, or put into different logical combinations, or various weights could be attached to them (so that the overall creativity rating for a program's behaviour could then be the weighted sum of the results of specified criteria).

If we could settle on a set of criteria such as those listed in 'Evidence for Creativity' above, then we could formally define a *creativity judgement system* as being a set of values for the various parameters involved (α, β, ...). However, it is premature to frame such a definition. We need to refine our ideas about suitable criteria (and suitable parameters) before attempting standardisation in this way.

As already remarked, the criteria do not all pull in the same direction, as some reward typicality and others give high scores to atypicality. Criteria 1 and 2 are not even real evidence of creativity (nor originality, novel, surprise or quality), but mere baseline measures to see if the program is at least succeeding at its core task of creating the right kind of artefact. Similarly, criteria 9 merely checks whether the program can at least replicate the kind of artefacts on which it is based. Criteria 3 and 4 reward value (quality) without regard for typicality, and so might be deemed rather blunt probes. Criterion 5 looks for items which are both typical and good, and so could be said to reflect Boden's ''exploratory'' creativity (although Boden characterises this, informally, in terms of a process—exploration—rather than in terms of a resulting characteristic—typicality). Criteria 6,7 and 8 focus on untypical but good, which (with the same caveat) might correspond to Boden's ''transformational'' creativity. All of 1–7 take no account of past artefacts (particularly, the inspiring set) except insofar as this is encoded into the notion of typicality. Criterion 10/10a looks for avoidance of the inspiring set, which in itself is not very interesting, if the resulting artefacts have poor typicality or quality ratings; this is a clear example of a criterion which might be a helpful component of a profile of the behaviour of the program, but in isolation gives no evidence of interesting creativity. Criteria 11–18 then concentrate on the novel results (outside the inspiring set), but again there are different emphases available: 11, 13, 15, searching for typicality alone (i.e. successful but unadventurous creation), with 12, 14, and 16 testing quality without regard for typicality, 17 seeking the conjunction, and 18 demanding what is sometimes viewed as the most creative combination: novelty (both w.r.t. I and as low typicality) and high quality. This shows that not all these criteria are equally demanding. Application of the criteria might be made more subtle by separating them into subsets which represent different facets of program behaviour:

Basic success: 1,2, 9
Unrestrained quality: 3,4
Conventional skill: 5
Unconventional skill: 6,7,8a
Avoiding replication: 10a
Basic success, avoiding replication: 11,13,15
Unrestrained quality, avoiding replication: 12,14,16

Conventional skill, avoiding replication: 17
Unconventional skill, avoiding replication: 18

It is interesting to note that the authors who have made use of the criteria do not use them as boolean conditions to be either satisfied or not, despite the fact that they were originally stated in this way. Since the 14 criteria were stated as quantities to be compared with θ, the later writers have simply cited these quantities to give some idea of the behaviour of the program. This avoids the need to make an arbitrary choice for the threshold θ, and allows a subtler statement of the extent to which a program is meeting a criterion. This may be a better way to regard the 'criteria': as a vector of values, all (in the revised set of 18) in [0,1], which constitute a profile of the program's behaviour.

The Meaning of the Criteria

Another notable aspect of the case studies surveyed in 'Applications of the Criteria' above is that some of these researchers have used the formulae of the original criteria while making slight changes to the definitions of the primitive constructs, which means that they are altering the meanings of the criteria.

In the Divago study, typicality (*typ*) is measured as closeness to *I*, which means that what were two separate factors in the original criteria are now very closely related, both in effect measuring avoidance of replication, instead of having *typ* indicate success at constructing exemplars of some independently defined genre of artefact. The Dupond study also tinkers with the *typ* rating, by setting it to embody the opposite of a subjective judgement of being more original than the source sentence. This may well have been a interesting factor to measure, but again it is a different concept from a measurement of the extent to which an artefact falls within some target class of artefacts. Haenen and Rauchas treat typicality as similarity to the complete set of items involved in their study, including randomly generated artefacts. It is not clear that this is the same intuitive notion as 'typical example of the intended artefact class', and also it means that similarity to the inspiring set (something which does not show up explicitly in their use of the criteria) becomes a component of typicality, as in the Divago study. (Haenen and Rauchas also asked their human judges to guess which melodies were human-written and which computer-generated. This could have been used to give a computer piece a *typ* value equal to the proportion of judges who thought it to be human-written, thereby sticking closer to the original idea of human-judged conformity to cultural norms.)

Quality (*val*) is defined in the Divago study as closeness of fit to the program's input goal, a more subtle change of emphasis. While this does not alter the formal meaning of the criteria—they were deliberately general and unconstrained about how value ratings might be arrived at—it does shift the perspective by which we view the Divago program. It is no longer aiming for some hazy, subjectively defined notion of a 'good' artefact (as generators of melodies, stories, jokes, poems, pictures, etc. usually do—see 'Assumptions'), it is now searching in a space where success can be completely defined algorithmically. It is more like a generator of class timetables or of chip designs, where the space may be very large, but complete success is not only possible but can be unambiguously detected if it occurs. It is not completely obvious whether these elaborate criteria are appropriate in this simpler situation.

The point is not that any of these modifications are wrong, or that inappropriate factors have been considered. Rather, the problem is that the authors have changed the meaning of the formulae while retaining the notation, without emphasising the fact that they are focussing on slightly different factors. A cleaner approach would have been to define different formulae or criteria, presenting a rationale for each one, so that the set of attributes being measured would be more explicit. Formal criteria like this offer a means of stating a particular view of what is interesting to measure, but it is important to argue out the case for looking at particular properties, rather than presenting the changes as mere minor amendments to formal definitions which were originally justified on other grounds. Such redefinitions disguise what is being measured.

Parameter Values

The question of how to determine the various thresholds α, β, γ, θ is far from trivial. (The role of θ could be eliminated, if, as suggested earlier, we treat the criteria not as truth-valued conditions, but formulae providing values in [0,1].) One possibility (following the way that Haenen and Rauchas (2006) chose a value for γ) is to use values determined by similar assessments of human-created artefacts. That is, if we assume that *typ* and *val* correspond to subjective judgements, by human subjects, of the qualities of artefacts, we could compute values such as the mean, standard deviation, maximum, and minimum for the judgements of *human generated* artefacts, and then use these in some way to define the thresholds for the assessment of computer artefacts,

Possible Extensions

The ideas presented here are just a first step in building up a methodology for assessing potentially creative programs. There are a number of ways in which they could perhaps be elaborated to capture more subtle aspects of creative processing.

Similarity

One important weakness of the framework (raised by Ritchie (2001), Pease et al. (2001), Pereira et al. (2005)), is that it does not handle the notion of *similarity* very cleanly. In particular, the present criteria compare the result set with the inspiring set only in terms of overlap in membership, but make no allowance for the idea that output items could vary in the extent to which they are similar to those in the inspiring set (as commented by Gervás, above). If we could define some form of distance metric between basic items, then a more subtle approach would be possible.

The question of similarity/distance measures is a vast topic which it is not feasible to explore here, but it is worth making one methodological point. Since our whole approach is based upon observing and measuring empirical aspects of generating programs, with as little theorising as possible about how creativity happens, any similarity measure used as part of the criteria should be based on observable properties of generated artefacts. These could come, for example, from ratings given by human subjects, either from direct judgements of similarity between artefacts, or indirectly via ratings on a set of scales which could be combined in some way to compute similarity/distance. Alternatively, a similarity measure could be based on a theoretical model of the domain in question, but it would be important to ensure that the similarity between artefacts could be determined objectively without prejudging questions of novelty or creativity.

Regardless of how the similarity measure is defined, if we have a measure for the distance between any two items, it should be possible to devise a suitable definition of the distance of an item from a set (such as the inspiring set), or even the overall distance of one set of items (e.g. the results) from another set (e.g. the inspiring set). Then we could either rephrase those criteria which mention I to be more general and distance-based, or it might be clearer to devise additional distance-based criteria.

For example, suppose that our distance function has (in some way) allowed the definition, for any set S of basic items, of:

> $N_\delta(S)$: the δ-neighbourhood of S, being the set of all items (including members of S) which lie within distance δ of the set S.

Then we can define counterparts of criteria 9 and 10/10a by replacing $I \cap R$ with $N_\delta(I) \cap R$, and counterparts of 11–18 by replacing $R - I$ with $R - N_\delta(I)$ for some suitable value of δ. (These could be criteria 9b, 10b, 11b, etc.)

There is another way in which a distance measure could throw light on aspects of creativity. If we have a way of locating artefacts within a multi-dimensional space (based either on similarity ratings or on assignment of properties to artefacts), then we can consider the extent to which a program's output is clustered. If the generated items are all very similar to each other, and so are restricted to a narrow part of the available space, that would be a further interesting factor in giving a 'profile' of the performance of the program (if we adopt the perspective mentioned earlier, that the criteria could be seen as characterising the behaviour of the program). Whether narrow clustering indicates greater creativity than wide exploration does is a question for debate. The suggestion by (Pereira et al. 2005) (see 'Possible Extensions' below) that *repetition* is a sign of lower creativity could be generalised to a proposal that *near-repetition* (as measured by similarity) is a symptom of less creativity.

The Contribution of the Designer

It would be interesting to develop further the issue mentioned in 'The Framework' above: how does the program come into being, and what is its exact relationship to the inspiring set? The question of making the inspiring set concrete and measurable has been tackled in some specific ways in the studies reported in 'Applications of the Criteria', usually by focussing on the items in some knowledge base which drives the creative process. The way in which the program is designed, and how input data are chosen, are still relatively uncharted.

Ritchie (2001) sketches a formalisation of some of the relevant stages: the selection of the inspiring set I from $\mathcal{B}$, taking into account *typ* and *val*; the devising of both an algorithm for the program, and a tuple of input parameter domains which show what arguments the program takes; the initialisation of parameter values for a particular run of a program. All of these could be appraised from a creativity point of view. At present, we offer no formal criteria for these subprocesses, but this appears to be an area where our style of analysis could be applied.

It could be argued that this relaxes the strict assumption that we should not, when assessing creativity, be examining *how* the creation occurred. However, the extension sketched here would still not delve into the internal workings of the program; rather, it would aim to clarify the role of the program designer, and the extent to which pre-programmed knowledge might be biasing the outcomes. This is the question that Koza et al. describe as the 'artificial-to-intelligence' ratio or colton et al. raise as 'fine tuning'.

A further, more advanced scenario is the situation where a user (maybe even the program's designer) intervenes during the running of the program, using knowledge of what the program has already done in order to guide it in particular directions (as in (Lenat 1976)). Such interventions are also relevant to the question of the extent of the program's own creativity. However, this is much more difficult to formalise and assess. It would need some explicit model of the computation process which allowed the idea of knowledge being added at intermediate stages, while not (to conform to our core assumptions) taking undue account of the steps made within the program. This is beyond the current framework.

Self-Rating of Output

Boden (1992, p. 83) suggests that one facet of human creativity is an ability to recognise the worth of a created entity. At a very crude level, we could allow for a generating program which assigns a rating to each item that it produces, indicating either the typicality or value (or both) that the program allocates to that item. Some programs (e.g. AM (Lenat 1979)) include a mechanism of this sort. It should be feasible to devise further formal definitions (in addition to those in 'Evidence for Creativity') which make use of this form of data. These additional criteria should then capture the intuitive idea that the program's rating of its own output ought (if the program is to be deemed creative) to have a high correlation with corresponding external ratings (gleaned, for example, from human judgements).

Multiple Runs

Pereira et al. criticise the criteria for describing a single set of program results, rather than behaviour over time as a number of runs occur. They suggest that if a system *repeats itself* in later runs, that is a sign of less creativity. There are three logically separate issues here: *repetition, non-determinism*, and *input parameters*.

The issue of repetition arises even without admitting multiple runs. We have assumed throughout our discussions that the output R is a *set* of results. If we generalise R to be a *bag*, then we can consider matters such as repetitions. If we define R to be a *sequence*, we can also consider the order in which artefacts are produced, although it is less obvious what the implications for creativity judgements might be.

In the case of multiple program runs, if the program retains information from one run to the next (via a persistent store of some sort), then the sequence of runs may be regarded as instalments of one run, with results accumulating. In that case, repetition could be handled much as in the single-run case, if we merge or concatenate the output from the runs. On the other hand, if the program starts completely afresh each time, then the question is whether the program behaves identically (or similarly) on each run. If the program is sufficiently non-deterministic that it may produce different results on successive runs, then a comparison could made between the different result-sets for similarity, if we wished to explore Pereira et al.'s intuition that variety is more creative.

The third consideration here is the effect of input parameters. We have largely glossed over the idea that a program is affected by parameters (apart from 'fine tuning'). If successive runs of a program are carried out with different parameters, then this allows a study of the effects of these parameters on the output.

Random Generation

Since the formalisation says little about *how* artefacts are generated, there is a sense in which it does not exclude the random production of basic items. The level of abstraction of the framework means that it has no way to distinguish random generation from any other approach. Although random generation is not a hugely interesting case from an AI point of view, it might, in certain discussions, be an interesting baseline for comparison purposes. To allow explicit formal comparisons, we would need a more detailed definition of the available space of basic items and how this space could be randomly explored. One way to do that might be to decompose the notion of ''basic item'' so as to make explicit the internal structure of an item (e.g. as an array of pixels, or as a sequence of words). The other requirement would be a suitable definition of 'random combination of atomic parts'.

This might connect to the ideas of Wiggins (2006a), where a formalisation is given of the notion of a potentially creative program exploring a space of concepts. Some exploration strategies could be classed as 'random', or suitable baselines in some other sense. If these were specified in sufficient detail to implement, or at least to predict their outcomes, then the result set of such a benchmark strategy could be compared with the result set of a supposedly creative program, in terms of typicality and value. Statistical tests could be applied to determine how distinct these sets of ratings were.

Conclusion

We have proposed an approach to the assessment of creativity (in programs) in which relevant factors (such as the quality of produced artefacts) are made explicit and are defined precisely. This allows the definition of a number of formulae relating these factors, in ways which should show how successful the program has been at certain aspects of the creative endeavour. ‘Possible Extensions’ lists just some of the ways in which they could be improved. As noted in ‘Use of the Criteria’, we do not see this list of criteria as forming a definite standard of creativity. Instead, we offer it as an initial draft of a set of possible measures to be applied to a program, to illuminate what it is and is not achieving. We hope that this framework will stimulate discussion of what exactly should be considered when debating the issue of creativity, and may contribute, in the longer term, to a sounder basis for attributing creativity to programs.

The usefulness of some of the criteria, or the correctness of the particular formalisation, may be disputed, but anyone who believes that questions about the creative behaviour of programs are to be tested empirically should put forward some comparably detailed ways in which creativity could be assessed.

Acknowledgements I would like to thank Geraint Wiggins, Simon Colton and Alison Pease for useful discussions of this material, and Chris Thornton for comments on a draft.

References

Baggi, D. (Ed.). (1992). Readings in computer generated music. New York: IEEE Computer Society Press.

Binsted, K., Pain, H., & Ritchie, G. (1997). Children’s evaluation of computer-generated punning riddles. *Pragmatics and Cognition, 5*(2), 305–354.

Binsted, K., & Ritchie, G. (1997). Computational rules for generating punning riddles. *Humor: International Journal of Humor Research, 10*(1), 25–76.

Boden, M. A. (1992). The creative mind (2nd ed.). London: Abacus. First published 1990.

Boden, M. A. (1998). Creativity and artificial intelligence. *Artificial Intelligence, 103*, 347–356.

Bundy, A. (1994). What is the difference between real creativity and mere novelty? *Behavioral and Brain Sciences, 17*(3), 533–534. Open Peer Commentary on Boden (1992).

Colton, S. (2002). *Automated theory formation in pure mathematics. Distinguished dissertations.* London: Springer-Verlag.

Colton, S., Bundy, A., & Walsh, T. (2000) Agent based cooperative theory formation in pure mathematics. In G. Wiggins (Ed.), *Proceedings of AISB 2000 symposium on creative and cultural aspects and applications of AI and cognitive science* (pp. 11–18). Birmingham, UK.

Colton, S., Pease, A., & Ritchie, G. (2001). The effect of input knowledge on creativity. In R. Weber & C. G. von Wangenheim, (Eds.), *Case-based reasoning: Papers from the workshop programme at ICCBR 01*, Vancouver.

Dreyfus, H. L. (1979). *What computers can’t do*. New York: Harper Row, revised edition. First edition 1972.

Fauconnier, G., & Turner, M. (1998). Conceptual integration networks. *Cognitive Science, 22*(2), 133–187.

Gervás, P. (2000). WASP: Evaluation of different strategies for the automatic generation of Spanish verse. In G. A. Wiggins (Ed.), *Proceedings of the AISB 00 symposium on creative & cultural aspects and applications of AI & cognitive science* (pp. 93–100). Society for the Study of Artificial Intelligence and Simulation of Behaviour.

Gervás, P. (2001). Generating poetry from a prose text: Creativity versus faithfulness. In G. A. Wiggins (Ed.), *Proceedings of the AISB 01 symposium on artificial intelligence and creativity in arts and science* (pp. 93–99). Society for the Study of Artificial Intelligence and Simulation of Behaviour.

Gervás, P. (2002). Exploring quantitative evaluations of the creativity of automatic poets. In C. Bento, A. Cardoso, & G. Wiggins (Eds.), *2nd workshop on creative systems, approaches to creativity in artificial intelligence and cognitive science, ECAI 2002*. Lyon, France.

Haenen, J., & Rauchas, S. (2006). Investigating artificial creativity by generating melodies, using connectionist knowledge representation. In *Proceedings of 3rd joint workshop on computational creativity, ECAI* (pp. 33–38). Italy: Riva del Garda.

Koza, J. R., Keane, M. A., Streeter, M. J., Mydlowec, W., Yu, J., & Lanza, G. (2003). *Genetic programming IV: Routine human-competitive machine intelligence*. Kluwer Academic Publishers/ Springer.

Lenat, D. (1976). *An artificial intelligence approach to discovery in mathematics as heuristic search*. Memo AIM-286, Department of Computer Science, Stanford University.

Lenat, D. (1979). On automated scientific theory formation: A case study using the AM program. In J. Hayes, D. Michie, & L. Mikulich (Eds.), *Machine intelligence 9* (pp. 251–283). Chichester: Ellis Horwood.

Manurung, H. M., Ritchie, G., & Thompson, H. (2000a). A flexible integrated architecture for generating poetic texts. In *Proceedings of the fourth symposium on natural language processing (SNLP 2000)* (pp. 7–22). Thailand: Chiang Mai.

Manurung, H. M., Ritchie, G., & Thompson, H. (2000b). Towards a computational model of poetry generation. In G. A. Wiggins (Ed.), *Proceedings of the AISB 00 symposium on creative & cultural aspects and applications of AI & cognitive science* (pp. 79–86). Society for the Study of Artificial Intelligence and Simulation of Behaviour.

Meehan, J. (1976). *The metanovel: Writing stories by computer*. PhD thesis, Yale University, Department of Computer Science.

Mendes, M., Pereira, F. C., & Cardoso, A. (2004). Creativity in natural language: Studying lexical relations. In *Proceedings of LREC workshop on language resources for linguistic creativity*. Evaluations and Language Resources Distribution Agency.

Miranda, E. (2001). *Composing music with computers*. Amsterdam: Focal Press/Elsevier.

Pease, A., Winterstein, D., & Colton, S. (2001). Evaluating machine creativity. In R. Weber, & C. G. von Wangenheim (Eds.), *Case-based reasoning: Papers from the workshop programme at ICCBR 01* (pp. 129–137). Vancouver.

Pereira, F. C. (2005). *A computational model of creativity*. PhD thesis, Universidade de Coimbra.

Pereira, F. C., Mendes, M., Gervás, P., & Cardoso, A. (2005). Experiments with assessment of creative systems: An application of Ritchie's criteria. In P. Gervás, T. Veale, & A. Pease (Eds.), *Proceedings of the workshop on computational creativity, 19th international joint conference on artificial intelligence* (*Technical Report* 5-05, pp. 37–44). Departamento de Sistemas Informáticosy Programación, Universidad Complutense de Madrid.

Ritchie, G. (2001). Assessing creativity. In *Proceedings of the AISB symposium on artificial intelligence and creativity in arts and science* (pp. 3–11). York, England.

Ritchie, G. (2006). The transformational creativity hypothesis. *New Generation Computing, 24*, 241–266.

Ritchie, G. D., & Hanna, F. K. (1984). AM: A case study in AI methodology. *Artificial Intelligence, 23*, 249–268.

Steel, G., Colton, S., Bundy, A., & Walsh, T. (2000). Cross-domain mathematical concept formation. In G. Wiggins, (Ed.), *Proceedings of AISB 2000 symposium on creative and cultural aspects and applications of AI and cognitive science* (pp. 3–10). Birmingham, UK.

Stock, O., & Strapparava, C. (2005). The act of creating humorous acronyms. *Applied Artificial Intelligence, 19*(2), 137–151.

Turner, S. R. (1994). *The creative process: A computer model of storytelling*. Hillsdale, NJ: Lawrence Erlbaum Associates

Weizenbaum, J. (1976). *Computer power and human reason*. San Francisco: Freeman.

Wiggins, G. (2001). Towards a more precise characterisation of creativity in AI. In R. Weber, & C. G. von Wangenheim (Eds.), *Case-based reasoning: Papers from the workshop programme at ICCBR 01*. Vancouver: Navy Center for Applied Research in Artificial Intelligence.

Wiggins, G. (2003). Categorising creative systems. In *Proceedings of third (IJCAI) workshop on creative systems: Approaches to creativity in artificial intelligence and cognitive science*.

Wiggins, G. (2005). Searching for computational creativity. In P. Gervás, T. Veale, & A. Pease (Eds.), *Proceedings of the IJCAI-05 workshop on computational creativity* (*Technical Report* 5-05, pp. 68–73). Departamento de Sistemas Informáticos y Programación, Universidad Complutense de Madrid.

Wiggins, G. (2006a). A preliminary framework for description, analysis and comparison of creative systems. *Knowledge-Based Systems, 19*, 449–458.

Wiggins, G. A. (2006b). Searching for computational creativity. *New Generation Computing, 24*(3), 209–222.