SPECULATIONS ON STORY PUNS

Kim Binsted* and Graeme Ritchie
Department of Artificial Intelligence
University of Edinburgh
Scotland EH1 1HN

ABSTRACT

There is a class of joke which consists of an anecdote, which is sometimes quite long and often has no inherently humorous content, followed by a punchline which is a distorted form of some well-known phrase, proverb or quotation. Usually the punchline purports to summarise or draw a moral from the preceding story. This genre has some unusual aspects, from the viewpoint of conventional claims about the attributes of jokes. These jokes also have certain structural or formal regularities which suggest that they might be amenable to computer generation. We outline how this might be done, by decomposing the construction of such story puns into a sequence of stages; some of these are clearly manageable, others are less straightforward. We also make some observations about where such an endeavour would fit within the broader field of humour research.

1 INTRODUCTION

A particular class of joke, which we will call the *story pun*, has some unusual features from the point of view of a formal analysis of the internal semantic/pragmantic structure of jokes. At some level of abstraction, the regularities underlying story puns can be seen as analogous to those underlying certain classes of punning riddle, in that a superficial similarity between linguistic forms is the central mechanism.

In past work [BR94, BR97], we have shown that punning riddles can be generated by computational rules. Here, we sketch an architecture for a system which could generate (some simple varieties of) story puns. We also suggest that genetic algorithms might be used to generate the kind of highly constrained story required for this type of joke.

*This research has been partially funded by a PhD studentship from the Natural Science and Engineering Council of Canada.

Finally, we make some observations about what such a system would demonstrate about the nature of humour.

2 STORY PUNS

There is a particular genre of joke which is quite common within British society, and possibly elsewhere. It consists of a story which need not itself have any humorous content (and which may be quite long), concluding with a single punchline which is usually a summing up of the content or moral of the story, although it may also be a suitable last line of the narrative or an utterance attributed to a character in the story. What is peculiar to this genre is that the punchline is always a distorted form of some well-known saying, usually a proverb but sometimes a famous quotation. For example, there is a well-established proverb in the English language *"People who live in glass houses shouldn't throw stones"* (meaning that those who are vulnerable to a particular form of criticism should not themselves make that criticism of others). This has been used as the basis of the following joke:

> Once upon a time, many years ago, there was a chieftain in a remote tropical village who owned an old and battered throne of which he was very fond. One day, a visiting dignitary gave him a brand new and ornate throne, which the chieftain had to adopt immediately out of politeness. However, he could not bear to part with the old throne which had served him so well, so he stowed it away in the roof area of his grass hut, in case it should be useful in the future. Unfortunately the interior structure of his hut was too flimsy to support the weight of the large object, and it crashed through the grass ceiling, falling on the chieftain and killing him.

The moral is that people who live in grass houses shouldn't stow thrones.

Many examples of these *story puns*[1] have been constructed ([MN91] contains around 150), although few survive into common circulation. It is possible to use the construction of such jokes as a game or pastime, in which one person stipulates the original (undistorted) maxim, and another person has to construct the story pun (this was what occurred in the radio show reported in [MN91]).

The structure of such a joke is interesting, as various attributes of the main part of the story and the punchline (and the relationship between them) are different from those which are often posited as central to more conventional jokes. In particular:

- Neither story nor punchline creates ambiguity.
- The punchline does not resolve any ambiguity.
- There is no incongruity within the story.
- The punchline is always a suitable (congruous) ending.
- The punchline does not (apparently) violate any expectations set up by the story.

These jokes have some other noteworthy features:

- The *meaning* of the original, undistorted proverb plays no role in the joke at all – it is necessary only that the *wording* of the proverb be sufficiently familiar to the listener that it is evoked by the the distorted form. This is in contrast to jokes typified by the much-cited "One more drink and I'll be under the host"(Dorothy Parker), where one factor in the humorous effect is the *meaning* of the related phrase "under the table".
- If the hearer is completely unaware of the original version of the punchline, there is no humorous effect.
- If a punchline is used whose *meaning* is virtually identical to the distorted proverb, but which uses different *wording*, there is no humorous effect. This is little more than a statement that these are a subclass of puns, since puns inherently rely on their wording.
- It is essential for the humorous effect that the maxim be *distorted*.[2] A story in which a character living in a glass house came to grief as a result of throwing stones, concluding with the ungarbled proverb about such situations, would not have the same humorous effect.

As with many other forms of pun, the listener's reaction to a story pun is often to groan rather than to laugh, which emphasises that the main story portion is essentially setting up a context in which a pun can be made. In this way, story puns are comparable to real-life wit where someone achieves a humorous effect by describing some (actual) event or situation in a garbled form of a well-known maxim.

3 SOME ASSUMPTIONS

In tackling a topic as broad and as deep as humour, or even just verbally expressed humour, there are a number of different methodological strategies that could be adopted.

One possibility (perhaps exemplified by the General Theory of Verbal Humour [Att94]) is to devise a very general theory which attempts to encompass all possible phenomena in the area under consideration. Particular studies of subspecies of humour are then carried out within this overall framework, guided by its form and its principles. This is logically a very sound approach, avoiding the temptation to posit *ad hoc* devices, and integrating all work under one theory. Its drawback, in the current state of our understanding of humour, is that such a theory may be over-general to the point of vagueness, or (if more specific) may contain arbitrary details included prior to the evidence having been gathered. This sort of approach could be loosely labelled as *top-down*.

Alternatively, one could adopt a more *bottom-up* approach, in which detailed studies are made of particular humorous phenomena, using whatever theoretical constructs appear appropriate for that domain. This has the advantage that research can be fairly concrete, to the extent of even leading to testable computer programs, and hence quite fine details can be examined. The

[1] One of the current authors prefers the classification "shaggy dog story" for this genre, while the other author believes that shaggy dog stories consist of long anticlimactic stories rather than complex puns.

[2] Another genre might be possible in which the punchline is an actual *undistorted* maxim and humour is produced by the indirect or unusual way in which the proverb applies to this particular story. However, these would not be puns, and would be different from the type of joke we are considering here.

drawback, of course, is that the analyses may exist in a vacuum, unconnected to other phenomena, and the mechanisms devised may be over-specific.

In reality, some blend of these two extremes is necessary, and is normally used. One cannot devise a general theory (top-down) without at least keeping an eye on the data, and one cannot work on particular data (bottom-up) without assuming (perhaps implicitly) at least some theoretical basis. The work presented here, although sketchy and speculative, tends towards the bottom-up end of the scale. We believe that, in the field of *computational* humour, it is important that some research effort should be put into finding phenomena which are tractable, or nearly tractable, with the current state of the art, and pursuing these in some depth. A more firmly top-down approach, while valuable, might not reach actual computational implementation for many decades.

There is an analogy here with the way in which linguistic theory has developed over past decades and centuries. Much of the early work involved the detailed development of grammars (or fragments of grammar) for particular languages, in the absence of any over-arching theory of language. Although theory-development became a prominent activity in its own right under the influence of Chomsky, even modern generative linguistics started from concepts (e.g. phrase structure) which had been developed within much more specific and pre-theoretical streams of work.

4 PROPOSED SYSTEM

4.1 Overview

We have devised a relatively crude computational model of how story puns could be produced. (This is *not* a model of how such jokes could be *interpreted*.) As we will explain in section 5 below, some of the components are still underspecified in this preliminary design, but we believe that the role of each module is relatively well-defined.

Viewed as an algorithm, the model could be summarised as follows:

Choose a maxim.
Distort this maxim.
Construct a semantic representation of the distorted version.
Develop some constraints from this semantic structure.
Devise a story to meet these constraints.

We will use a worked example to illustrate how this might be carried out.

4.2 Constructing the punchline

4.2.1 The method

The original (undistorted) proverb or saying could be an input parameter for the process, as in the "game" form of joke-construction mentioned in section 2 above. Alternatively, we could assume that our proposed joke-generator would have access to a set of proverbs, quotations and other well-known sayings, and that it would initially choose one at random.

Construction of possible target punchlines from a given maxim is not too problematic. The punchline should be "similar" to the original in some sense. The notion of "similarity" is essentially that which is used in punning riddles (cf. [PG84]), where puns can be based on inexact matches between words. Tactics that are suitable include metathesis (spoonerism) (e.g. "throw stones" $\rightarrow$ "stow thrones") and substitution of a phonetically similar segment (e.g. "glass" $\rightarrow$ "grass"). There might be a large number of phrases or sentences which could be produced from the same original saying, so heuristics might have to be developed to choose the more productive or suitable ones. On the other hand, the next step in the algorithm, analysing the punchline, might succeed only on a few distorted maxims, so that pre-selection becomes unnecessary; that is, a set of possible punchlines could be passed to the next phase, letting the analyser filter out unviable variants.

4.2.2 Worked example

The maxim chosen for this example is "Look before you leap". Substituting similar sounding words, for example, could produce several different distorted maxims, such as:

- Hook before you heap
- Rook before two sheep
- Leak before you loop
- Book before you reap

All of these could potentially work in a story pun. We will use the last one, "Book before you reap", for this example.

4.3 Analysing the punchline

4.3.1 The method

The punchline, as constructed, is still just a sequence of words, as the garbling process operated at a very low level, manipulating phonetic information. For the punchline to influence the construction of a suitable story, there must be a representation of its *content*. To produce this, there must be a module capable of scanning a sentence and constructing a symbolic representation of its meaning in some suitable formalism (e.g. some type of logic).

4.3.2 Worked example

There are many different formalisms in use in computational language processing. For the sake of this example, we will use a logic-like language in which operators representing the speech-act or illocutionary force of the utterance take as their arguments propositions (similar to those of predicate logic).[3] It is quite possible that in developing a complete system, criteria for choosing or devising a suitable formalism would become apparent.

The semantic content of "Book before you reap" could be represented as:

$$\text{IMPER}(\ book_V(\text{hearer},\ \text{Y},\ \text{Time1})\ \&\ reap_V(\text{hearer},\ \text{X},\ \text{Time2}),\ \text{Time1} < \text{Time2}).$$

A possible simplification that could be considered would be to restrict the model to morals stated as commands, so that any variant of the punchline which could not be analysed as a simple imperative (e.g. "Rook before two sheep") would be eliminated.

4.4 Constructing the story

4.4.1 Simplifications

Story generation is very hard, and there is no prospect of having a good quality completely computer-generated fictional story in the near future. Generating a story which is guaranteed to be summed up by a given punchline is therefore the hardest part of this preliminary design. In order to make some progress, we believe it is both valid and feasible to constrain the type of story in various ways, so that our generator will be operating in a much smaller space of possibilities. There are various genres of story which seem to have certain stereotypical attributes, involving the types of characters, the types of events, the types of goals that characters have, the situations that characters may be in, etc. In particular, fables (e.g. [Len67]), fairy stories (e.g. [CR80]) and Greek myths (cf. [SPT96]) all appear promising. These relatively stylised genres also have the advantage of comparatively straightforward narrative devices – they make little use of flashbacks, multiple viewpoints, etc.

In the program of research we are outlining here, we would select one such genre as our area of study. For the sake of the worked example, we will choose the moral fable genre, since story puns bear a certain resemblance to moral fables, in that they build up to and justify a one-sentence moral.

4.4.2 Representing the story

Once we have adopted a manageable genre of story, the next issue is that of story representation. Although there is a considerable literature on literary analysis of fables (cf. [Car85]), creative writing (e.g. [EP90]), and story generation (e.g. [Mee76] and [Kan90]), there is no consensus on how to represent, formally, the content of a story. We would have to postulate some abstract representation of the story, using some knowledge representation formalism (e.g. a traditional logic, or something of the "KL-ONE" family [BS85]), based on the ideas proposed in these existing disciplines.

This representation, for generality, would have to contain information not only about events, situations and characters in the world being described; it would also have to represent the narrative presentation (e.g. the order in which events are described), although our simplifications (section 4.4.1 above) might render this rather trivial for the examples to be studied.

4.4.3 Constraining the story

Most AI work on story generation has focussed on methods which would actually construct a story from some basic material, such as information about the characters and their goals. An alternative approach, which we would like to explore, is to attempt to define declarative constraints or preferences describing the formally represented

[3] This general idea is reminiscent of the *procedural semantics* of [Woo78].

structure of the story. That is, using ideas from the various disciplines mentioned in Section 4.4.2 above, we would formulate a set of heuristic rules which, given a formally represented story (or candidate for being a story) would evaluate its merit as a story. We do not believe that there is really some abstract measure of "quality" that can be used to rate real stories. However, there are some very crude guidelines which can be used to separate those structures which barely count as stories from those which are at least well-formed stories. The basis for this belief is that literature on creative writing can be viewed as attempts to do just this. For example, when a textbook advises "Never switch viewpoints in the middle of a scene" (e.g. [Dib, Bic93]), it is formulating a heuristic that could be applied to a symbolic representation of a story to make some estimate of its competence or success. (Notice that it is crucial that we have some formal, abstract representation of the story, as it would be completely infeasible to apply such abstract heuristics to the bare textual form of a story.)

A constraint-based (or "evaluative") approach such as this has the advantage that the very heterogeneous requirements of the story pun can all be accommodated. A story pun has certain requirements which might not arise in more conventional story generation:

- The plot must (somehow) conform to (the meaning of) the pre-determined final punchline or "pseudo-moral".

- Enough suitable concepts and vocabulary items must be introduced in the course of the story to make the final punchline relatively natural.

- Nevertheless, for the story pun to be humorously effective, there should not be so much preparation within the story that the punchline can be easily predicted.[4]

All these requirements are of a very varied nature, so some very general scheme is needed for their representation. (We do not suggest that formalising these requirements would be trivial.)

[4] It is easier to succeed with a story pun if the audience does not know in advance the approximate shape of the punchline; it is a sign of the skills of Muir and Norden [MN91] that their stories amuse even when the original (undistorted) proverb is known in advance.

4.4.4 Producing the story

Let us assume that we have managed to achieve the various goals listed above – we have selected a narrow, stylised genre of story, we have devised a formal representation for the content (including presentation) of such stories, and we have developed a set of heuristics which will contribute to the rating of how well a story meets our needs. How then can we *produce* such a story? Although we may have ways of telling how good a story is once we have it fully represented, we need some way of creating the structures. Clearly, the space of possibilities is very large.

Here we introduce another major speculation. Let us assume that our formal representation is such that we can define a compressed encoding of it into a sequence (tuple) of values from some limited range. That is, suppose we can re-code a story-stucture into the form

$$< v_1, v_2, \ldots, v_n >$$

where each v_i is selected from some (fairly small) set of possible atomic values (and where the full structure can be reconstructed from this sequence of values if needed). This might well be possible, given the simplifications we have made already. In the narrow, stylised story genres, there are often relatively few options which go to establish the basic parameters of the story.

We are now in a position which is exactly appropriate for the application of a technique known as *evolutionary algorithms* or *genetic algorithms* [Mit96]. These methods (a highly sophisticated variant on what is sometimes called "generate-and-test") have proved to be very useful in problem areas where the search space for solutions is very large, but there are clear constraints on what constitutes a 'good' solution. *Genetic algorithms* are based on a loose analogy with the way that evolution can occur through genetic variation and natural (Darwinian) selection. They require the following:

1. a class of structures or items which we wish to produce (e.g. abstract representations of stories), regarded as analogous to organisms in an evolutionary setting;

2. some measure of "fitness" which rates how good an item is (e.g. our heuristics of story quality), and hence whether the item should be given preference in a "survival of the fittest" process.

3. a way of encoding an item as a tuple, so that the tuples can be treated as analogous to chromosomes and can be mutated or combined in ways which regard them simply as sequences of symbols.
4. a way of decoding a tuple as an item, so that fitness can be assessed after any mutation or combination takes place.

This technique works in the following way. Initially, a large number of such tuples (the pseudo-chromosomes) are generated at random (perhaps within some very minimal constraints on well-formedness). Then the following cycle is repeated many times.

1. Each item is evaluated for its "fitness", usually by decoding the tuple representation (chromosome) into the full structure and then applying various evaluation functions. This gives some rating of the relative "goodness" or "fitness" of the various items.
2. Some manipulation of the tuples occurs, based on their fitness, in a manner analogous to mutation or sexual reproduction in genes. For example, the two "fittest" tuples might be combined (reproduction); or a single tuple might have a small number of its elements changed randomly (mutation). This creates a new "population" of chromosomes.
3. The cycle then repeats.

This process can be run either for some predetermined number of cycles, or until some criterion is met (e.g. that the fitness of the best item has not improved on several successive cycles).

This computational method, although over twenty years old in its conception [Hol75], has become feasible only recently with increasing computer power. It has worked with extraordinary effectiveness in certain areas where the problem has a large search space, a clear notion of evaluation of "goodness", and the possibility of a quasi-chromosomic encoding. (See, for example, [BN95] for a range of applications, including scheduling, layout, stock control, financial planning, traffic management.) Whether it would serve the cause of story generation as well remains to be seen.

4.4.5 Worked example

Work in this area has not advanced far enough to give even a tentative representation of the story, let alone in a form appropriate for use with genetic algorithms. Instead, we will informally outline some of the constraints from the genre and from the parsed punchline, and give a story that satisfies those contraints.

Our chosen genre, the moral fable, has a typical setting, plot structure, cast of characters, and prop list. Although few of these are necessary features, they help to make the genre recognizable, and are not overly restrictive.

- **Characters:** All characters are either animals (e.g. fox, sheep etc) or human. If human, the character must have one of a limited number of occupations (e.g. farmer, doctor, fishwife etc). Both animals and humans have their own simple characteristics, beliefs, goals and relations (e.g. fox is cunning, sheep is stupid, fox wants to eat sheep, farmer wants to protect sheep etc). These characteristics do not change over the course of the story (i.e. there is no character development). Any character can converse with any other character. Characters can lie.
- **Objects:** Each character is associated with a small set of objects. Some objects may be associated with more than one character (e.g hay is both eaten by sheep and mown by farmers).
- **Actions:** Some acts can be achieved by any character (e.g speaking), others cannot (e.g. only farmers can mow, and only when they have a field of hay).
- **Setting:** There is a limited number of settings (e.g. the forest, a farm, the city, a house etc). Each character is typically found in one setting, and can venture into a few others. For example, a fox is typically found in the forest, can go onto a farm, but would not be found in the city. This constrains the characters that can interact.
- **Plot structure:** A character attempts to do something (consistent with its goals). It fails. The reason for its failure, and/or advice on how to avoid a similar failure, is summarized in the moral.

The punchline also constrains the story, in that the fable must justify the punchline as its moral. We assume that the system has access to lexical information. Our chosen punchline, "Book before you reap", might constrain the plot structure of the fable as follows:

> A character attempts to reap something, and fails. The reason for its failure is that it failed to book some location before attempting to reap.

From the cast of characters, objects, settings etc, we select those that fit into this outline. Let us assume that, in 'fable world', only farmers can reap, and that the only object that can be reaped is hay. There are no constraints from the punchline on the location, so we choose one consistent with farmers, hay and reaping: a field. This gives us the basic plot:

> A farmer tries to reap hay, but fails. The reason for his failure is that he did not book a field before attempting to reap.

Although we suggest that the generation of the final version of the story could be done with some suitable search mechanism, such as genetic algorithms, here it is done by hand.

> Farmer Joe needed some hay to feed his sheep. He picked up his scythe and went to the field to cut the hay. Unfortunately, when he got there, he found another farmer already working in the field. He had forgotten to make a reservation!
>
> Moral: Book before you reap.

Note that the character has been given a name, a few objects and characters (the sheep, the scythe, the other farmer) not strictly necessary to the story have been introduced, and none of the key words in the punchline are explicitly mentioned in the story. Also, the style is relatively sophisticated, using punctuation, adverbs, etc. A good story generator, whether based on genetic algorithms or not, must be able to manipulate the basic story in ways consistent with 'good' creative writing heuristics, as discussed in Section 4.4.3 above.

5 HOW FEASIBLE IS THIS?

Let us summarise what would be needed to make this work.

Punchline construction. This initial phase of the process seems straightforward. Given some basic lexical information (including phonetics), distorting a sentence into a similar-sounding sentence is not difficult.

Punchline analysis. It is completely possible, in theory, to analyse a sentence of English into a representation of its literal meaning. In practice, it would (in the current state of the art) not be trivial to construct a working analyser which would provide correct analyses of arbitrary sentences with any level of syntactic complexity, formulated from a wide vocabulary. Although we are assuming that the story domain could be simplified, the punchlines come from a very unconstrained source, namely existing proverbs and sayings. Not only does this widen the vocabulary greatly, it could introduce some quite unusual grammatical forms.

Representing the story. To do this completely would require significant research, but (given the use of a narrow, stylised genre), it should be feasible to develop some (comparatively crude) formal representation. It has to be recognised, however, that taking short-cuts here could have a deleterious effect on the story-evaluation issue.

Evaluating a story This is the most difficult (and perhaps most interesting) part. It would involve significant research, with a synthesis of ideas from a number of subdisciplines.

Producing the story. If the previous steps, particularly the formulation of an evaluation measure, are achieved, then the use of genetic algorithms should be feasible. The only further prerequisite is the design of a quasi-chromosomic (tuple) encoding for story structures.

6 DISCUSSION

Let us imagine that we manage to construct a program along the lines outlined here. What would the import of this be for a (computational) theory of humour?

What is noticeable about the proposed design is that none of the stages purport to embody or rely upon a theory of humour in any way. Each is a non-humorous process. It is the overall effect which (we hope) will be humorous. This is not paradoxical, as it is quite acceptable for the whole to create an effect that is not attributable to any one of its parts. However, nothing in the design process was driven by any theory of what was or was not funny. Rather, it was based on observing regularities in a process which was deemed humorous, and attempting to model these regularities (cf. remarks in Section 3 above on the "bottom-up" approach). If it were successful, we would have created a model of a class of jokes without explicit use of a theory of humour. Hence, it is not clear what such a working program would prove or disprove. It may be an experiment, loosely speaking, but what hypothesis does it test?

It is important to realise that a computer program (or even an algorithmic but unimplemented model) may embody theoretical claims or assumptions, even although these are not explicitly represented and even although the system designers were not consciously aware of them. Any design has implicit assumptions, and these may depend upon some unarticulated theory of how humour works. It is commonplace for novice research workers in artificial intelligence (typically, students) to build computer programs without a clear abstract or theoretical model, and to try (with difficulty) to determine afterwards what assumptions constitute their *de facto* theory. We are perhaps in that position. We have to confess ignorance of what theory of humour has led to the outline proposal in Section 4 above.

This may be a symptom of the very early stage of the development of formal (or computational) models of humour. At present, we have very little theoretical framework on which to base further steps into the unknown. In this situation, one possible (and perhaps essential) first step is to investigate and formalise what might be termed the *major structural* aspects of humorous texts. That is, a joke genre (such as story puns) will be based on certain (fairly gross) relationships between formal entities. These can be seen as *necessary* characteristics for a text to qualify as a member of the genre. However, some of the texts which meet these gross structural criteria will not be very funny at all, whereas others will be extremely funny. Study of why there are these variations in funniness should take us much closer to the essential aspects of humour, and so might seem to be more important than the modelling of the "major structural" aspects. However, the formal statement of the major structural aspects provides a foundation upon which more detailed and subtle studies can be based. Until the undergrowth has been cleared away, it is difficult to focus clearly on the fine detail.

A loose analogy can be made with the role of syntax in a theory of language. Even someone who believes that the most essential aspect of language is its meaning is likely to concede that it is unrealistic to attempt to develop a theory of natural language semantics in complete isolation from syntax. Indeed, the more clearly grammatical issues are understood, the less there will be unwanted distractions in the study of meaning.

Although the methodological points argued above have been stated with respect to a non-existent program (i.e. the story pun generator whose content we have outlined), they could largely be illustrated with respect to a working program — the JAPE riddle-generator ([BR94, BR97]). That program generates punning riddles which (at their best) are of a standard comparable to those circulating amongst schoolchildren. It was designed entirely by observing the formal regularities within such riddles, and modelling these patterns in abstract rules. Although it could be argued that the design is influenced by, or illustrates, ideas about ambiguity, any relationship to a general theory of humour is very nebulous. Nevertheless, it clears the way for more fundamental questions to be asked, such as what types of ambiguity enhance humorous effect.

More generally, there is the point that was already made in Section 3 above — we have to try out lots of small scale studies just to stimulate ideas about what the ingredients might be of a final theory of humour.

7 WHAT NEXT?

We have argued that:

1. Story puns form a well-defined genre of joke with characteristics quite different from most told jokes.

2. There are formal regularities in story puns which suggest a generator could be decomposed into independent stages.

3. All of these proposed modules could be constructed, at least in some simple form, using currently known techniques.

4. This outline design, whether implemented or not, has no apparent links to any theory of humour in general, but might be a useful preliminary step in formalising structural aspects of one subclass of joke.

The next steps, for those with time, inclination and funding, would be to try implementing such a system and then to experiment with the various factors involved, to see what makes one story pun funnier than another.

References

[Att94] Salvatore Attardo. *Linguistic Theories of Humour*. Mouton de Gruyter, Berlin, 1994.

[Bic93] J. Bickham. *Scene and structure*. Writers Digest Books, Cincinatti, Ohio, 1993.

[BN95] J. Biethahn and V. Nissan, editors. *Evolutionary Algorithms in Management Applications*. Springer, Berlin, 1995.

[BR94] Kim Binsted and Graeme Ritchie. An implemented model of punning riddles. In *Proceedings of the Twelfth National Conference on Artificial Intelligence (AAAI-94)*, Seattle, USA, 1994. Available as Research Paper 690, Department of Artificial Intelligence, University of Edinburgh.

[BR97] Kim Binsted and Graeme Ritchie. Computational rules for generating punning riddles. *Humor*, 1997. To appear.

[BS85] R.J. Brachman and J.G. Schmolze. An overview of the KL-ONE knowledge representation system. *Cognitive Science*, 9(2):171–216, 1985.

[Car85] Pack Carnes. *Fable scholarship: an annotated bibliography*. Garland, New York, London, 1985.

[CR80] Patricia Conroy and Sven Rossel, editors. *Tales and stories of Hans Christian Andersen*. University of Washington Press, Seattle, 1980.

[DCT95] Ansen Dibell, Orson Scott Card, and Lewis Turco. *How to Write a Million*. Robinson Publications, London, 1995.

[Dib] Ansen Dibell. Plot. In *[DCT95]*.

[EP90] George Evans and Vince Powell. *Get writing: a practical guide to creative writing*. BBC Books, London, 1990.

[GSJLW86] B.J. Grosz, K. Sparck Jones, and B. Lynn Webber, editors. *Readings in Natural Language Processing*. Morgan Kaufmann Publishers, Inc., San Mateo, CA, 1986.

[Hol75] John Holland. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, 1975.

[Kan90] Mark Kantrowitz. Natural language text generation in the oz interactive fiction project. Technical Report CMU-CS-90-158, School of Computer Science, Carnegie Mellon University, 1990.

[Len67] R. T. Lenaghan, editor. *Caxton's Aesop*. Harvard University Press, Cambridge, 1967.

[Mee76] James Meehan. *The metanovel : writing stories by computer*. PhD thesis, Yale University, Department of Computer Science, 1976.

[Mit96] Melanie Mitchell. *An introduction to genetic algorithms*. MIT Press, Cambridge, Mass, 1996.

[MN91] Frank Muir and Dennis Norden. *The Utterly Ultimate 'My Word' Collection*. Mandarin Humour Classics, 1991.

[PG84] W.J. Pepicello and Thomas A. Green. *The Language of Riddles*. Ohio State University, 1984.

[SPT96] N. M. Sgouros, G. Papakonstantinou, and P Tsanakas. A framework for plot control in interactive story systems. In *Proceedings of 13th National Conference on Artificial Intelligence (AAAI-96)*, pages 162 – 167, Portland, OR, USA, 1996.

[Woo78] W.A. Woods. Semantics and quantification in natural language question answering. In M. Rubinoff and M.C. Yovits, editors, *Advances in Computers*, pages 2–64. Academic Press, New York, NY, 1978. Reprinted in [GSJLW86], pp. 205-248.