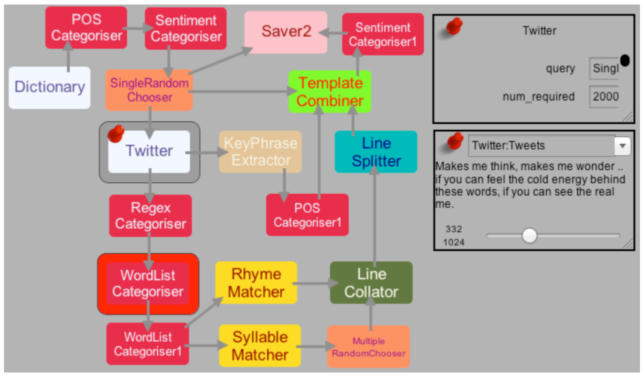
Towards a Flowcharting System for Automated Process Invention

Simon Colton and John Charnley

Computational Creativity Group, Department of Computing, Goldsmiths, University of London
www.doc.gold.ac.uk/ccg



Twitter

query Singl

num_required 2000

Twitter:Tweets

Makes me think, makes me wonder .. if you can feel the cold energy behind these words, if you can see the real me.

332
1024

Figure 1: User-defined flowchart for poetry generation.

Flowcharts

Ironically, while automated programming has had a long and varied history in Artificial Intelligence research, automating the creative art of programming has rarely been studied within Computational Creativity research. In many senses, software writing software represents a very exciting potential avenue for research, as it addresses directly issues related to novelty, surprise, innovation at process level and the framing of activities. One reason for the lack of research in this area is the difficulty inherent in getting software to generate code. Therefore, it seems sensible to start investigating how software can innovate at the process level with an approach less than full programming, and we have chosen the classic approach to process design afforded by flowcharts. Our aim is to provide a system simple enough to be used by non-experts to craft generative flowcharts, indeed, simple enough for the software itself to create flowcharts which represent novel, and hopefully interesting new processes.

We are currently in the fourth iteration of development, having found various difficulties with three previous approaches, ranging from flexibility and expressiveness of the flowcharts to the mismatching of inputs with outputs, the storage of data between runs, and the ability to handle programmatic constructs such as conditionals and loops. In our current approach, we represent a process as a script, onto which a flowchart can be grafted. We believe this offers the best balance of flexibility, expressiveness and usability, and will pave the way to the automatic generation of scripts in the next development stage. We have so far implemented the natural language processing flowchart nodes required to model aspects of a previous poetry generation approach and a previous concept formation approach.

The Flow System

In figure 1 we present a screenshot of the system, which is tentatively called *Flow*. The flowchart shown uses 18 sub-processes which, in overview, do the following: a negative valence adjective is chosen, and used to retrieve tweets from Twitter; these are then filtered to remove various types, and pairs are matched by syllable count and rhyme; finally the lines are split where possible and combined via a template into poems of four stanzas; multiple poems are produced and the one with overall most negative valency is saved. A stanza from a poem generated using 'malevolent' is given in figure 2. Note in figure 1 that the node bordered in red (WordList Categoriser) contains the sub-process currently running, and the node bordered in grey (Twitter) has been clicked by the user, which brings up the parameters for that sub-process in the first black-bordered box and the output from it in the second black-bordered box. We see the 332nd of 1024 tweets containing the word 'cold' is on view. Note also that the user is able to put a thumb-pin into any node, which indicates that the previous output from that node should be used in the next run, rather than being calculated again.

It's our ambition to build a community of open-source developers and users around the Flow approach, so that the system can mimic the capabilities of existing generative systems in various domains, but more importantly, it can invent new processes in those domains. Moreover, we plan to install the system on various servers worldwide, constantly reacting in creative ways to new nodes which are uploaded by developers, and to new flowcharts developed by users with a variety of cultural backgrounds. We hope to show that, in addition to creating at artefact level, software can innovate at process level, test the value of new processes and intelligently frame how they work and what they produce.

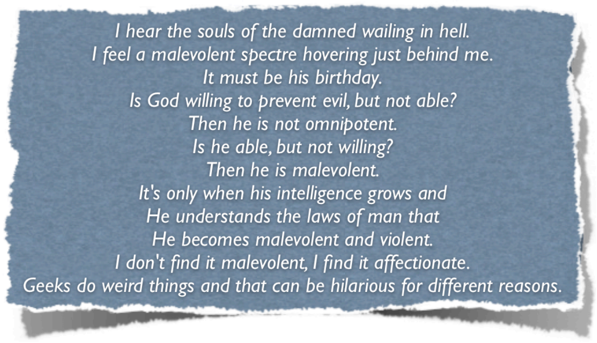

I hear the souls of the damned wailing in hell.
I feel a malevolent spectre hovering just behind me.
It must be his birthday.
Is God willing to prevent evil, but not able?
Then he is not omnipotent.
Is he able, but not willing?
Then he is malevolent.
It's only when his intelligence grows and
He understands the laws of man that
He becomes malevolent and violent.
I don't find it malevolent, I find it affectionate.
Geeks do weird things and that can be hilarious for different reasons.

Figure 2: A stanza from the poem *On Being Malevolent*.