Poetry Generation

(Manurung et al., 2000; Netzer et al., 2009)

Dominikus Wetzel
dwetzel@coli.uni-sb.de

Department of Computational Linguistics
Saarland University

July 5, 2010

A Stochastic Hillclimbing Model

General model:

- view generation as explicit search trough the space (i.e stochastic hillclimbing search)
- a “state” in search space is a text with all underlying representations (from semantics to phonetics)
- a “move” can occur at any representation level
- randomness is well suited for “creativity” in poem generation

Use an evolutionary algorithm:

- iterations of two stages (evaluation and evolution)
- population: ordered set of candidate solutions
- individuals: the candidate solutions

Evolutionary Algorithm - Outline

- initialize: a collection of individuals is created, given target phonetic form and target semantics
- evaluation: each individual will be assigned a score, based on current state of representation levels
- copying: the highest ranked individuals will create copies of themselves; lower ranked individuals are replaced
- evolution: random application of operators (mutation) will change the children

Algorithm – Evaluation

Phonetics:

- presence of a regular phonetic form (rhyme, metre, alliteration, . . .)
- define a target form: score candidates on how close they are to this form
- or for alliteration: count occurrence of the same word beginnings

Algorithm – Evaluation

Syntax and Style:

- lexical choice: score interesting co-occurrences higher, reward words marked as “poetic”
- syntax: reward interesting constructions, e.g. inverse word order, clause order, topicalization
- rhetoric: rank figurative language higher, e.g metonymy

Semantics:

- define a target: consider this as “pool of ideas”
- score candidate relative to this target

Algorithm – Evolution

Mutation operators

- add: “John walked” $\longrightarrow$ “John walked to the store”
- delete: “John likes Jill and Mary” $\longrightarrow$ “John likes Jill”
- change: “John walked” $\longrightarrow$ “John lumbered”

- mutations can occur at all representation levels
- hence, mutation at one level has to update the other levels

Implementation

Grammar Formalism:

- LTAG: substitution, adjunction
- derivation tree easily allows for mutations
- extended domain of locality: predicate-argument structure, agreement of features (esp. for non-contiguous tokens)
  $\longrightarrow$ ensure e.g. rhyming across the lines

Example

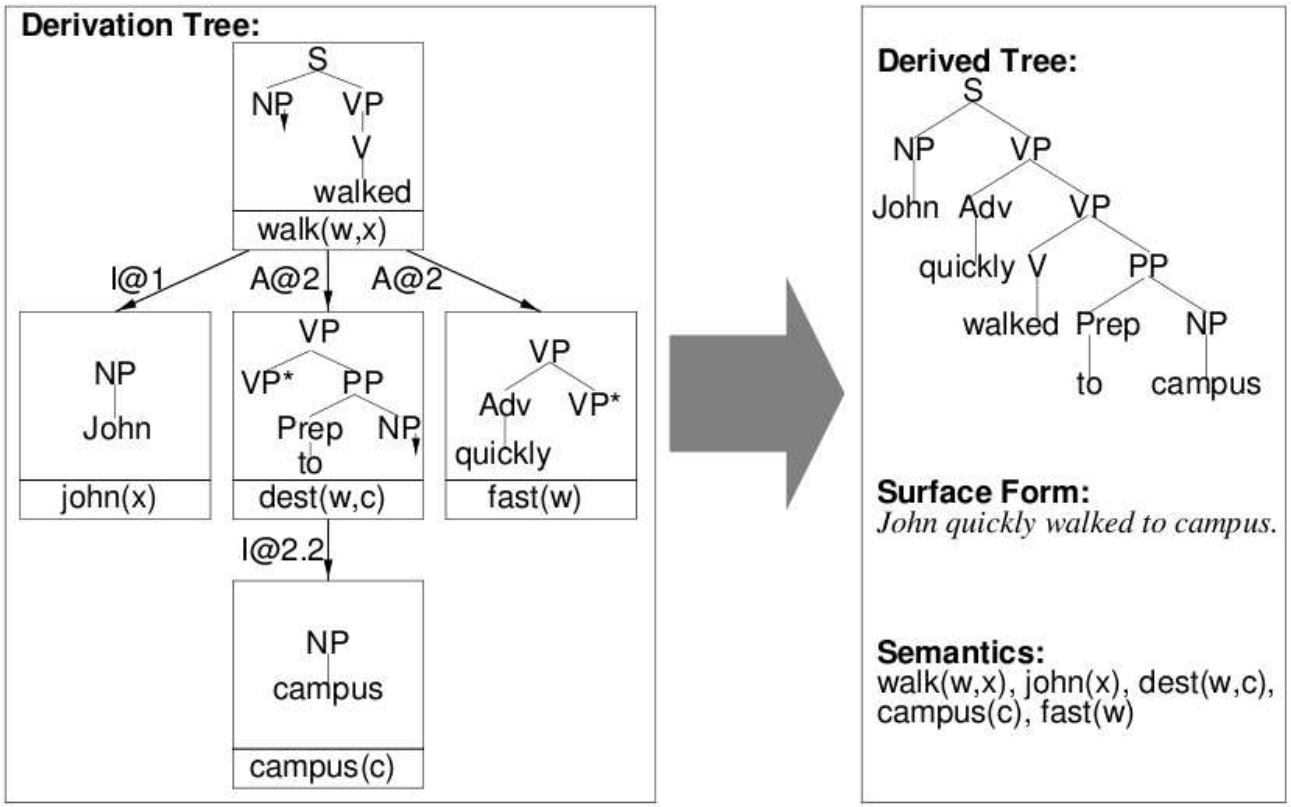


Figure: Derivation Tree and Derived Tree with “semantic pool”

Implementation - Operators

Semantic explorer:

- introduce random propositions into “semantic pool of individuals”
- could be extended with the use of a knowledge-base

Semantic realizer:

- randomly select and realize propositions from the semantic pool
- determine all lexical items for the selected proposition
- identify all elementary trees suitable for the lexical items
- determine all substitution/adjunction nodes in the derivation tree for these elementary trees
- randomly choose one of these nodes and insert

Syntactic paraphraser:

- randomly select an elementary tree from a derivation tree and apply paraphrase
- Example: active to passive: replace root node (predicate-argument) and update addresses of subject and object

Two Haikus

cherry tree
poisonous flowers lie
blooming

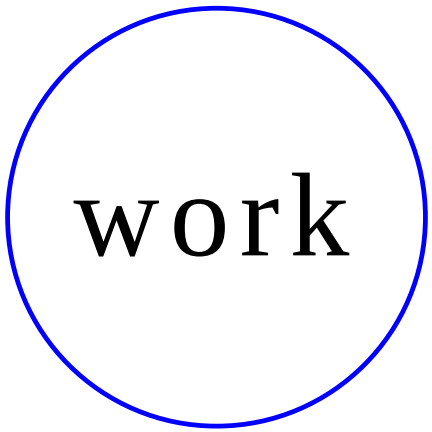
---

blind snakes
on the wet grass
tombstoned terror

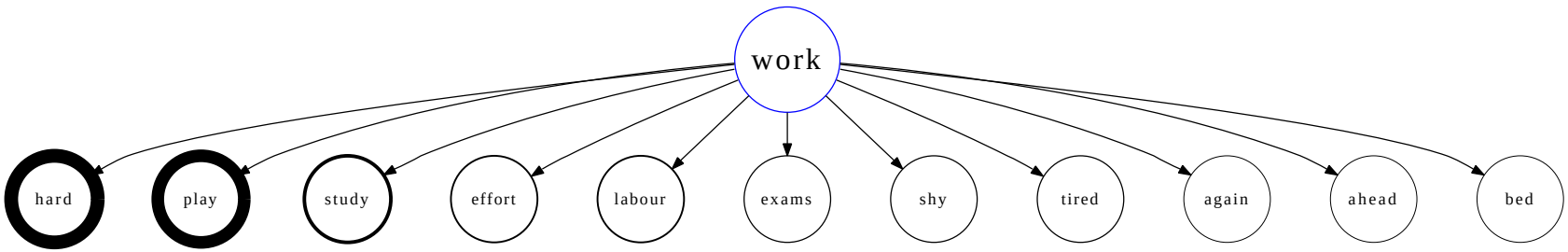
Form of poetry - Haiku

- Haikus originated in Japan and have a very restricted form
- English Haikus: three lines, 5-7-5 syllables
- functional words may be dropped

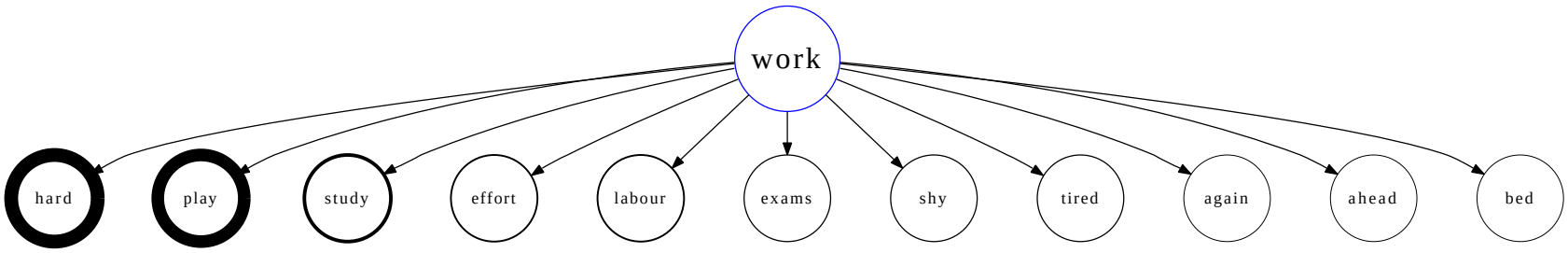
Word association norms (WANs)



Word association norms (WANs)



Word association norms (WANs)



- collection of cue words with sets of free associations
- associations obtained by collecting immediate responses to cue words
- 42% of English WANs do *not* occur within a window of 10 words in a large balanced corpus

Algorithm – Dataset

WAN corpus:

- from University of South Florida
- ca. 5,000 cue words, ca. 10,500 target words
- since 1973, with more than 6,000 participants

Haiku corpus:

- ca. 3,600 English Haikus
- varying resources: amateur sites, children's writings, translations of Japanese Haikus, from sites of Haiku Associations

Content selection corpus:

- Google n-gram (1 TB) $\longrightarrow$ diverse data
- entire text of Project Gutenberg $\longrightarrow$ easier to POS-tag

Algorithm – Overview

1. theme selection:
   set overall theme
2. syntactic planning:
   determine specific syntactic structure
3. content selection:
   select fitting lines from corpus
4. filter over-generation:
   remove lines with unintended properties
5. ranking:
   establish a ranking between all generated Haikus

Algorithm – Theme selection

Heuristics:

- begin with user-supplied seed word
- randomly choose direction ($P(cue) = 0.5, P(target) = 0.5$)
- then, randomly choose a neighbour (based on relative frequencies)
- repeat the two previous steps for all chosen words (level = 3)
- repeat the three previous steps n-times (n = 8)

Result:

- collects associated words close enough to the seed word
- but also distant enough to be interesting

Algorithm – Syntactic planning

Preprocessing:

- POS-tag the Haiku corpus
- extract patterns from each Haiku line
- patterns are POS sequences with lexicalized tokens:
  $\longrightarrow$ e.g. *DT_the JJ NN*
- from each Haiku line, take the top-40 patterns (total: 120)

During generation:

- choose a first-line pattern randomly (according to relative frequencies)
- choose the second and third-line pattern conditioned on the previous line (also based on relative frequencies)

Algorithm – Content selection

Preprocessing:

- POS-tag n-grams from Google corpus/Project Gutenberg

During generation:

- find those n-grams that match the selected patterns
- only take those which contain one of the selected theme words (both stemmed)
- first line has to contain the seed word

Algorithm – Filter over-generation

- filter candidates with “undesired” properties: e.g. repeating content words in two different lines

Algorithm – Ranking

- highly associative (WAN) Haikus should have a high rank
- content selection has introduced new content words
- count number of 1st and 2nd degree associations
- give more weight to 2nd degree associations

Evaluation

"Turing-Test" setup:

- humans should indicate how much they liked the presented Haiku (scale of 1-5)
- decided whether it has been produced by human/computer

Test data:

- AUTO: 10 random human Haikus, 15 generated Haikus (top ranking)
  $\rightarrow$ seed words were manually identified content words from the 10 human Haikus
- SEL: 9 human award-winning Haikus, 17 manually selected generated Haikus
  $\rightarrow$ a generated Haiku contained at least one content word of a human Haiku

Evaluation

Test subjects:

- AUTO: 40 people
- SEL: 22 people
- age 18–74, native/fully fluent English speakers, most did *not* have academic background in literature

Results

		Human Poems	Gaiku
AUTO	avg. % classified as Human	72.5%	37.2%
	avg. grade	2.86	2.11
SEL	avg. % classified as Human	71.7%	44.1%
	avg. grade	2.84	2.32

Table: Turing test results

- AUTO: overall 66.7% correct classifications
- SEL: overall 61.4% correct classifications
- avg. grade significantly higher for human Haikus

Conclusion

(Manurung et al., 2000):

- use of stochastic hillclimbing search, with evolutionary algorithm
- requires lots of resources (pronunciation dictionary, hand-crafted grammar, knowledge-base, etc.)
- here: implementation only in its beginning
  $\longrightarrow$ see PhD thesis: (Manurung, 2004)
- restricted to “classic” poems

(Netzer et al., 2009)

- Haikus have higher word associations than prose and newswire
- use of Word Association Norms (WAN) to identify content patterns
- easily accessible resources (Haiku and WAN corpus, Google n-gram/Project Gutenberg)
- no manual work needed
- here: restricted to Haikus

The End

Thank you!

References

Manurung, H. M. (2004). *An Evolutionary Algorithm Approach to Poetry Generation*. PhD thesis, University of Edinburgh. College of Science and Engineering. School of Informatics.

Manurung, H. M., Ritchie, G., and Thompson, H. (2000). Towards A Computational Model of Poetry Generation. In *In Proceedings of AISB Symposium on Creative and Cultural Aspects and Applications of AI and Cognitive Science*, pages 79–86.

Netzer, Y., Gabay, D., Goldberg, Y., and Elhadad, M. (2009). Gaiku: Generating Haiku with Word Associations Norms. In *CALC '09: Proceedings of the Workshop on Computational Approaches to Linguistic Creativity*, pages 32–39, Morristown, NJ, USA. Association for Computational Linguistics.

Difficulties compared to NLG

Communicative goal:

- NLG: given a communicative goal, produce a string conveying the message
- PGen: communicative goal not necessarily well defined

Modular vs. integrated:

- NLG: modular stages, i.e. content determination, text planning, surface realization
- PGen: strong interdependent connections between semantics, syntax and lexical level

Rich resources

- PGen: wide coverage grammar, rich lexicon, a knowledge-base due to large quantity of phonetic, syntactic and semantic constraints

Evaluation:

- PGen: people are more tolerant towards work of art

Implementation - Evaluators

The metre evaluator:

- divide stress pattern into feet of descending/falling rhythm
- compares this pattern to phonetic target form
- count number of feet: penalize if too short/long (strong)
- count number of weak syllables: penalize disagreement, but less

WANs and WordNet relations

Are Haikus more associative than prose or newswire?

- WAN associativity: two nodes are connected iff one is a cue for the other
  $\longrightarrow$ count word pairs with distance ≤ 2, normalize by all word pairs where both words are in the WAN corpus
- WordNet relatedness: count word pairs with distance ≤ 3
- Corpora: 200 Haikus, random 12-word sequences from Project Gutenberg and NANC newswire corpus

Source	Avg. Assoc. Relations	Avg. WordNet Relations
News	0.26	2.02
Prose	0.22	1.4
Haiku	0.32	1.38

Table: WAN and WordNet relations