Towards Automata and Grammars that Process DAGs

F. Drewes

CS Research Day 2015

UMEÅ UNIVERSITET

Automata and Grammars for Linguistic Structures

- Grammars and automata are useful because of their algorithms.
- One application area is Natural Language Processing.
- Conflicting goals:

$$\left.\begin{array}{c}\text{We want expressivity}\\ \updownarrow \\ \text{We want simplicity \& efficiency}\end{array}\right\}\ \text{eat \& keep the cake???}$$

$\Rightarrow$ much of what we are doing is trying to push the limits.

Current “Hot Topic”

Develop good ways to represent meaning together with suitable algorithms for dealing with those representations.

What Can It be Used For?

The practical goal is to create a basis for improved language technology:

- machine translation (e.g., Google translate)
- speech recognition and generation (e.g., Siri)
- advanced search in text or movie archives

Current solutions are often very useful, but are far from perfect.

They mainly rest on the use of vast amounts of resources, but this approach has intrinsic limitations.

In short. . .

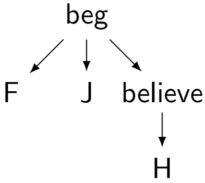
Language technology that does not take meaning into account is, literally, bound to be meaningless.

Relation to Other Research (1)

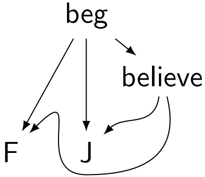
- Traditional structures representing language are (syntax) trees.
- To describe meaning this is not enough.

  Example: Frank begs the jury to believe him.

syntax: a tree



meaning: a DAG (directed acyclic graph)



- Currently, a corpus of Abstract Meaning Representations is built by computational linguists in the US and UK.
- Needed: the formal machinery to process such structures.

Relation to Other Research (2)

Previous formal language research to build upon:

- Grammars and automata on strings and trees
  – extremely useful for capturing syntax; efficient algorithms
- Transducers for syntax-directed translation
  – good for long distance dependencies; nice algorithmic properties
- Weighted automata computing “goodness”, probability, etc
  – add the quantitative aspect, algorithms often generalize the unweighted case
- (Context-free) graph grammars for more complex structures
  – nice structural properties & decidability results, but high computational complexity; unclear how to make weighted; most of the generality not needed (DAGs are sufficient).

Relation to Other Research (2)

Previous formal language research to build upon:

- Grammars and automata on strings and trees
  – extremely useful for capturing syntax; efficient algorithms
- Transducers for syntax-directed translation
  – good for long distance dependencies; nice algorithmic properties
- Weighted automata computing “goodness”, probability, etc
  – add the quantitative aspect, algorithms often generalize the unweighted case
- (Context-free) graph grammars for more complex structures
  – nice structural properties & decidability results, but high computational complexity; unclear how to make weighted; most of the generality not needed (DAGs are sufficient).

We need something in between, right here!

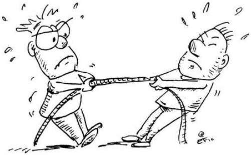
The Research

Develop a theory of automata and grammars on DAGs

One size fits all cannot reasonably be hoped for.

- Develop a variety of formalisms that complement each other.
- Grammars and automata, weighted and unweighted, more and less powerful ones.
- Ultimate aim: Develop a small set of formal models that cover the relevant aspects but are as efficient as possible.

Efficiency!!!



Coverage!!!

From a Recent Paper

DAG automata (with D. Chiang, D. Gildea, A. Lopez, G. Satta)

- A simple type of weighted finite automata working on DAGs.
- Generalizes well-known types of automata that are successfully being used in NLP. Good
- Recognized languages are NP-complete. Uggh, bad
- Their path languages are not even context-free. Hmmm. . .
- Efficient algorithm for computing the weight of DAGs of bounded treewidth. Good, but. . .
- Class of context-free graph languages closed under intersection with DAG automata. Good, but. . .
- Emptiness decidable in polynomial time. Good, but what about finiteness?

Where to Go from Here?

It's a promising start, but most questions are open:

- Cut away what is not needed (like non-regular path languages).
- Include things that are needed (like unbounded degree).
- Develop better algorithms (if possible).
- Find good grammatical formalisms (complementing the automata).
- Study suitable restrictions of known devices (like graph grammars).
- Investigate the relationship between all of these.
- Address the problem of automatic training/learning.
- Implement and try it out on real data.

Where to Go from Here?

It's a promising start, but most questions are open:

- Cut away what is not needed (like non-regular path languages).
- Include things that are needed (like unbounded degree).
- Develop better algorithms (if possible).
- Find good grammatical formalisms (complementing the automata).
- Study suitable restrictions of known devices (like graph grammars).
- Investigate the relationship between all of these.
- Address the problem of automatic training/learning.
- Implement and try it out on real data.

Thanks for listening!