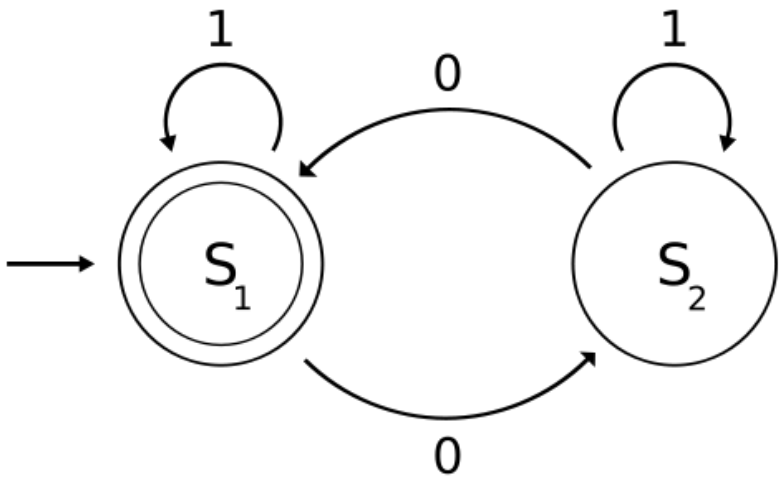
Towards Automated Analysis of CS Theory Assignments

Rochester Institute of Technologoy

August 7, 2015

Tools for CS theory students

- Computer science theory courses introduce automata and languages



Tools for CS theory students

- Computer science theory courses introduce automata and languages
- Software for analyzing, grading, and providing feedback on automata and grammars would benefit students
- Much work has already been done for regular languages[1][2]
- But many of the questions we'd like to answer about context-free languages are undecidable

[1] A. O. Bilska, K. H. Leider, M. Procopiuc, *et al.*, "A collection of tools for making automata theory and formal languages come alive," in *ACM SIGCSE Bulletin*, 1997, pp. 15–19.

[2] R. Alur, L. D'Antoni, S. Gulwani, *et al.*, "Automated grading of dfa constructions," , ser. IJCAI '13, Beijing, China, 2013, pp. 1976–1982.

Ambiguity and equivalence

Ambiguity

Given a grammar G is there a word w, $w \in \mathcal{L}(G)$, that has more than one leftmost derivation?

Ambiguity and equivalence

Ambiguity

Given a grammar G is there a word w, $w \in \mathcal{L}(G)$, that has more than one leftmost derivation?

G :

$S \rightarrow A \mid \epsilon$

$A \rightarrow \epsilon$

Ambiguity and equivalence

Equivalence

Given grammars G and G', does $\mathcal{L}(G) = \mathcal{L}(G')$?

Ambiguity and equivalence

Equivalence

Given grammars G and G', does $\mathcal{L}(G) = \mathcal{L}(G')$?

G :

$S \to A \mid \epsilon$

$A \to \epsilon$

G' :

$S \to \epsilon$

Because these problems are undecidable ...

- Can't automate grading (generally)
- Can't automate feedback (generally)

Bounded versions of problems

Ambiguity

Given a grammar G, is there a word w, $w \in \mathcal{L}(G)$, $|w| \leq k$ that has more than one leftmost derivation?

Equivalence

Given grammars G and G', does $\mathcal{L}(G) = \mathcal{L}(G')$ for all words of length $\leq k$?

CFGAnalyzer

- Axelsson, Heljanko, and Lange introduced an algorithm and a tool[3] for converting bounded instances of problems on context-free grammars into boolean formulae in conjunctive normal form

G :

$$S \rightarrow 0S0 \mid 1S1 \mid \epsilon \quad \Longrightarrow \quad (a_0 \vee b_0 \vee c_1) \wedge (a_2 \vee \neg b_1 \vee \neg d_0) \wedge \ldots$$

[3] R. Axelsson, K. Heljanko, and M. Lange, "Analyzing context-free grammars using an incremental sat solver," , ser. ICALP '08, Reykjavik, Iceland: Springer-Verlag, 2008, pp. 410–422.

CFGAnalyzer

- Axelsson, Heljanko, and Lange introduced an algorithm and a tool[3] for converting bounded instances of problems on context-free grammars into boolean formulae in conjunctive normal form
- Once the problem is expressed in CNF, a SAT solver can find satisfying assignments (if any exist)!
- A satisfying assignment would encode a witness for ambiguity or inequivalence

$$w | w \in \mathcal{L}(G), w \notin \mathcal{L}(G')$$

[3] R. Axelsson, K. Heljanko, and M. Lange, "Analyzing context-free grammars using an incremental sat solver," , ser. ICALP '08, Reykjavik, Iceland: Springer-Verlag, 2008, pp. 410–422.

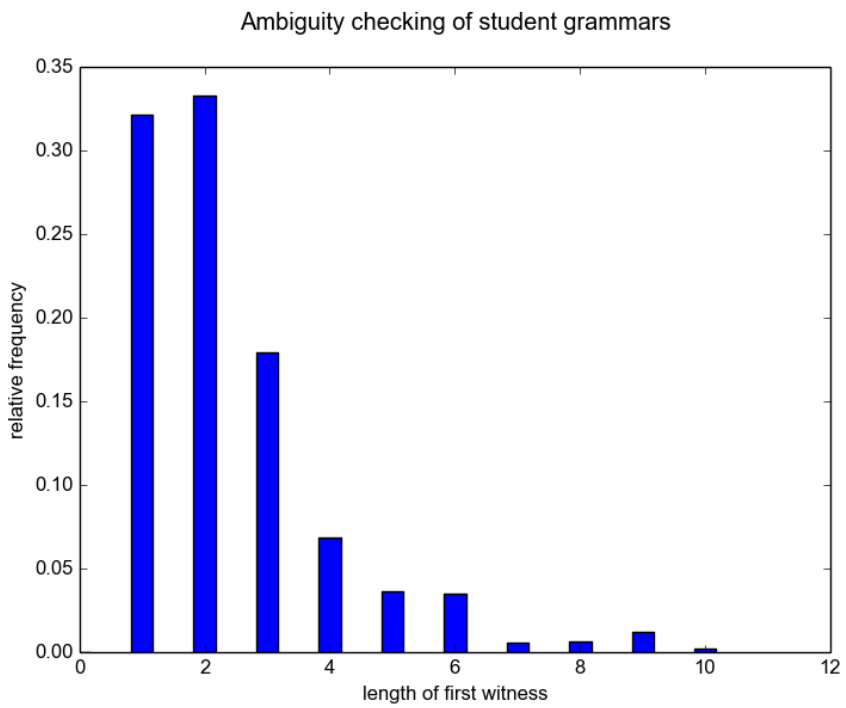
CFGAnalyzer

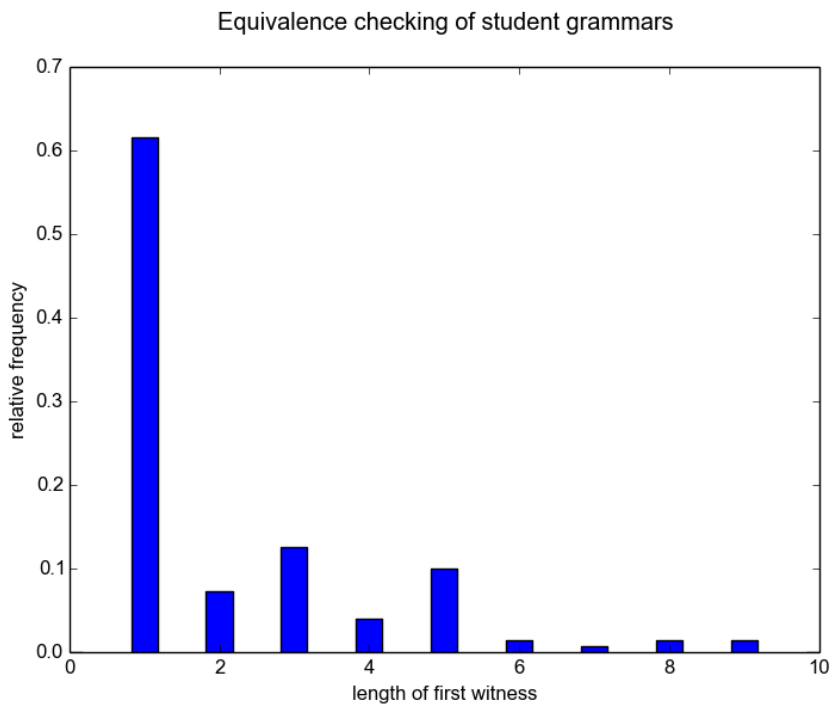
- Axelsson, Heljanko, and Lange introduced an algorithm and a tool[3] for converting bounded instances of problems on context-free grammars into boolean formulae in conjunctive normal form
- A modification to this algorithm would find all witnesses, giving a student a sense of how wrong their solution might be or how ambiguous their grammar is

$$W = \{w | w \in \mathcal{L}(G), w \notin \mathcal{L}(G')\}$$

[3] R. Axelsson, K. Heljanko, and M. Lange, "Analyzing context-free grammars using an incremental sat solver," , ser. ICALP '08, Reykjavik, Iceland: Springer-Verlag, 2008, pp. 410–422.

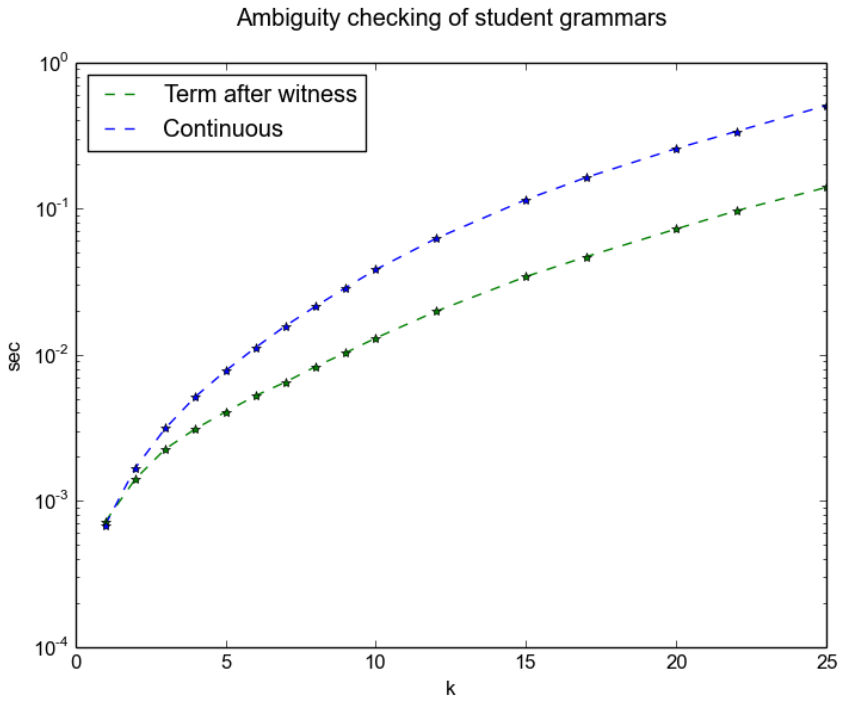
What is a reasonable value for the bound?

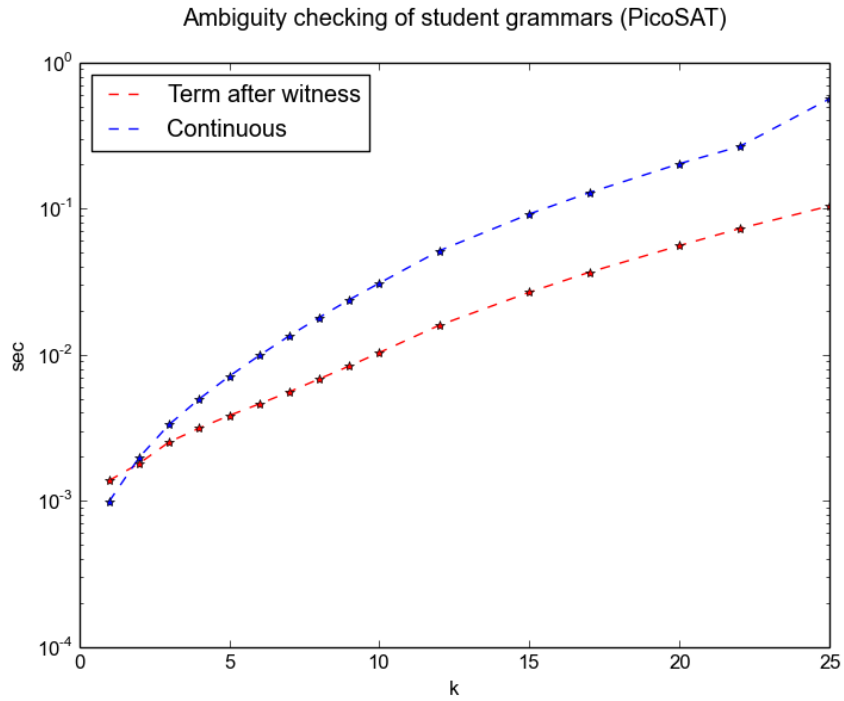




What proportion of the benchmark data has a shortest witness of length n?

Integration of PicoSAT





Execution times for grammars with witnesses of length k in zChaff[4] vs. PicoSAT[5]

[4] M. W. Moskewicz, C. F. Madigan, Y. Zhao, *et al.*, "Chaff: engineering an efficient sat solver," , ser. DAC '01, Las Vegas, Nevada, USA: ACM, 2001, pp. 530–535.

[5] A. Biere, "Picosat essentials," *JSAT*, vol. 4, no. 2-4, pp. 75–97, 2008.

Future work

- Integrate the CFGAnalyzer tool, along with its modifications, into JFLAP
- Explore additional methods of assigning partial credit (edit distance?)
- Test on more student samples!

References I

[1] A. O. Bilska, K. H. Leider, M. Procopiuc, O. Procopiuc, S. H. Rodger, J. R. Salemme, and E. Tsang, "A collection of tools for making automata theory and formal languages come alive," in *ACM SIGCSE Bulletin*, 1997, pp. 15–19.

[2] R. Alur, L. D'Antoni, S. Gulwani, D. Kini, and M. Viswanathan, "Automated grading of dfa constructions," , ser. IJCAI '13, Beijing, China, 2013, pp. 1976–1982.

[3] R. Axelsson, K. Heljanko, and M. Lange, "Analyzing context-free grammars using an incremental sat solver," , ser. ICALP '08, Reykjavik, Iceland: Springer-Verlag, 2008, pp. 410–422.

[4] M. W. Moskewicz, C. F. Madigan, Y. Zhao, L. Zhang, and S. Malik, "Chaff: engineering an efficient sat solver," , ser. DAC '01, Las Vegas, Nevada, USA: ACM, 2001, pp. 530–535.

[5] A. Biere, "Picosat essentials," *JSAT*, vol. 4, no. 2-4, pp. 75–97, 2008.

Thank you!

Questions?