Davis Putnam ATP

Shane Aston
Spring 2015

Davis Putnam Algorithm

- Negate Conclusion
- Convert each statement to CNF
- Turn CNFs into clauses
- Run DP algorithm

```
boolean Satisfiable(S)
begin
    if S = {} return true;
    if S = {{}} return false
    select L ∈ lit(S);
    return Satisfiable(SL) ||
    Satisfiable(SL’);
end
```

Add-ons

- Subsumption Elimination
  - [A,B] subsumes [A,B,C]
  - remove the subsumed statemetns
- Pure-Literal Elimination
  - $L \in S$ AND $\sim L \notin S$
  - remove all statements containing L
- Unit Literal
  - $[L] \in S$
  - don’t branch
- Tautological Elimination

Input Parsing

- Two formats:
  - “Classic”
    - natural language (kinda)
  - “New”

Formats

"Classic" format:	"New" format:
A implies (N or Q)	A -> (N v Q)
not(N or not A)	~(N v ~A)
A implies Q	A -> Q

CNF

- Putting in CNF is Hard…
- Two approaches:
  - Regular Expressions
  - Wolfram-Alpha

A <-> Q

~A <-> Q

A <-> (N v Q)

~(N ^ (Q v ~P) ^ J) <-> A

Regular Expressions

- Fast
- Only for simple expressions
- A
  - `re.match('(\w)$', line, re.I)`
- A <-> ~B
  - `m = re.match('(\w+) xnor NOT (\w+)$', line, re.I)`

Wolfram Alpha

- Slow…
- Any complexity

- Free Developer API
  - 2000 queries/month
  - Simple HTTP requests
  - Returns XML

Demo!