・ロト ・回ト ・ヨト

Earley Parsing in Prolog

Aleksandar L. Dimitrov

Universität Tübingen

December 19, 2007

Universität Tübingen

Self-modifying Programs in Prolog 00 000	Earley's Algorithm 00	Appendix O
Outline		

1 Self-modifying Programs in Prolog

- Modifying Predicates
- Chart-Parsing

2 Earley's Algorithm

Short Recap

3 Implementation

4 Appendix

Appendix A: Empty Determiner

Self-modifying Programs in Prolog ●0 ○○○	Earley's Algorithm 00	Appendix 0
Modifying Predicates		
Use of assert		

• Can Prolog have only store a limited set of predicates?

Aleksandar L. Dimitrov Earley Parsing in Prolog Universität Tübingen

<ロ> < 回 > < 回 > < 回 >

Self-modifying Programs in Prolog ●0 ○00	Earley's Algorithm 00	Appendix 0
Modifying Predicates		
Use of assert		

Can Prolog have only store a limited set of predicates?
 So far: Predicates from Source files

Image: A mathematical states and a mathem

Self-modifying Programs in Prolog ●0 ○○○	Earley's Algorithm 00	Appendix 0
Modifying Predicates		
Use of assert		

- Can Prolog have only store a limited set of predicates?
 - So far: Predicates from Source files
 - New: Predicates dynamically generated from

Image: A mathematical states and a mathem

Self-modifying Programs in Prolog ●0 ○○○	Earley's Algorithm 00	Appendix 0
Modifying Predicates		
Use of assert		

- Can Prolog have only store a limited set of predicates?
 - So far: Predicates from Source files
 - New: Predicates dynamically generated from
 - User input
 - Creative use of existing predicates

< 口 > < 同

Self-modifying Programs in Prolog ●0 ○○○	Earley's Algorithm 00	Appendix 0
Modifying Predicates		
Use of assert		

Can Prolog have only store a limited set of predicates?

- So far: Predicates from Source files
- New: Predicates dynamically generated from
 - User input
 - Creative use of existing predicates
- ... resulting in Self-Modifying Programs

Self-modifying Programs in Prolog ⊙ ⊙⊙⊙	Earley's Algorithm 00	Appendix O
Modifying Predicates		
Use of assert, contd.		

Are you only allowed to add to existing predicates?



A B +
 A B +
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A

3 →

Aleksandar L. Dimitrov

Earley Parsing in Prolog

Self-modifying Programs in Prolog ⊙ ⊙⊙⊙	Earley's Algorithm 00	Appendix O
Modifying Predicates		
Use of assert, contd.		

- Are you only allowed to add to existing predicates?
- Workaround: use of unknown/2

Image: Image:

Self-modifying Programs in Prolog ⊙ ⊙⊙⊙	Earley's Algorithm 00	Appendix O
Modifying Predicates		
Use of assert, contd.		

- Are you only allowed to add to existing predicates?
- *Workaround:* use of unknown/2
- How flexible is your Database?

Image: A mathematical states and a mathem

Self-modifying Programs in Prolog ⊙ ⊙⊙⊙	Earley's Algorithm 00	Appendix O
Modifying Predicates		
Use of assert, contd.		

- Are you only allowed to add to existing predicates?
- Workaround: use of unknown/2
- How flexible is your Database?
- \blacksquare \rightarrow use of abolish/1, retract/1

Self-modifying Programs in Prolog ⊙ ⊙⊙⊙	Earley's Algorithm 00	Appendix O
Modifying Predicates		
Use of assert, contd.		

- Are you only allowed to add to existing predicates?
- Workaround: use of unknown/2
- How flexible is your Database?
- \blacksquare \rightarrow use of abolish/1, retract/1
- Watch out for unification!

Self-modifying Programs in Prolog ⊙ ⊙⊙⊙	Earley's Algorithm 00	Appendix O
Modifying Predicates		
Use of assert, contd.		

- Are you only allowed to add to existing predicates?
- Workaround: use of unknown/2
- How flexible is your Database?
- \blacksquare \rightarrow use of abolish/1, retract/1
- Watch out for unification!
- Use assertz/1 and asserta/1

Self-modifying Programs in Prolog ⊙ ⊙⊙⊙	Earley's Algorithm 00	Appendix O
Modifying Predicates		
Use of assert, contd.		

- Are you only allowed to add to existing predicates?
- Workaround: use of unknown/2
- How flexible is your Database?
- \blacksquare \rightarrow use of abolish/1, retract/1
- Watch out for unification!
- Use assertz/1 and asserta/1
- ... also for efficiency reasons

Self-modifying Programs in Prolog ○ ●○○	Earley's Algorithm 00	Appendix O
Chart-Parsing		
Chart Darsing with	Drolog	

Chart Parsing with Prolog

Chart Parsers operate entirely on their chart



Aleksandar L. Dimitrov Earley Parsing in Prolog

Self-modifying Programs in Prolog ••••••••••••••••••••••••••••••••••	Earley's Algorithm 00	Appendix O
Chart-Parsing		
Chart Parsing with P	rolog	

- Chart Parsers operate entirely on their chart
- Why Chart Parsing?

Aleksandar L. Dimitrov Earley Parsing in Prolog A B +
 A B +
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A

Self-modifying Programs in Prolog ○○ ●○○	Earley's Algorithm 00	Appendix O
Chart-Parsing		
Chart Parsing with P	rolog	

- Chart Parsers operate entirely on their chart
- Why Chart Parsing?
- Imagine a grammar:
 A -> BC(D)

Image: Image:

Self-modifying Programs in Prolog ○○ ●○○	Earley's Algorithm 00	Appendix O
Chart-Parsing		
Chart Parsing with P	rolog	

- Chart Parsers operate entirely on their chart
- Why Chart Parsing?
- Imagine a grammar:
 A -> BC(D)
- Input: BCD

Universität Tübingen

Self-modifying Programs in Prolog ○○ ●○○	Earley's Algorithm 00	Appendix O
Chart-Parsing		
Chart Parsing with P	rolog	

- Chart Parsers operate entirely on their chart
- Why Chart Parsing?
- Imagine a grammar:
 A -> BC(D)
- Input: BCD

Universität Tübingen

Self-modifying Programs in Prolog	Earley's Algorithm	Implementation	Appendix
00			
000			
Chart-Parsing			

Minimalistic Example

Grammar: A -> BC A -> BCD

Input: BCD



▲□▶ ▲圖▶ ▲圖▶ ▲圖▶ ▲国 ◆ ○○○

Aleksandar L. Dimitrov Earley Parsing in Prolog

Self-modifying Programs in Prolog	Earley's Algorithm	Implementation	Appendix
00			
000			
Chart-Parsing			

Minimalistic Example

Grammar: A -> BC A -> BCD Input: BCD



▲口 ▶ ▲圖 ▶ ▲ 臣 ▶ ▲ 臣 ▶ ▲ 国 ● ● ● ●

Universität Tübingen

Aleksandar L. Dimitrov

Earley Parsing in Prolog

Self-modifying Programs in Prolog	Earley's Algorithm	Implementation	Appendix
00 0●0			
Chart-Parsing			

Minimalistic Example

Grammar: A -> BC A -> BCD Input: BCD A B C D A B C D D

Aleksandar L. Dimitrov Earley Parsing in Prolog Universität Tübingen

< A

Self-modifying Programs in Prolog	Earley's Algorithm	Appendix
Chart-Parsing		

Use of assert in this context

```
Our original TD parser:
parse(C,[Word|S],S) :-
word(C,Word).
```

```
parse(C,S1,S) :-
rule(C,Cs),
parselist(Cs,S1,S).
```

Universität Tübingen

Aleksandar L. Dimitrov

Earley Parsing in Prolog

Self-modifying Programs in Prolog ○○ ○○●	Earley's Algorithm 00	Appendix O
Chart-Parsing		

Use of assert in this context

```
Our new TD chart parser:
parse(C,[Word|S],S) :-
word(C,Word).
```

```
parse(C,S1,S) :- chart(C,S1,S).
```

```
parse(C,S1,S) :-
rule(C,Cs),
parselist(Cs,S1,S),
asserta(chart(C,S1,S)).
```

Aleksandar L. Dimitrov

Earley Parsing in Prolog

Earley's Algorithm

▲□▶▲圖▶▲圖▶▲圖▶ ▲国▶

Aleksandar L. Dimitrov Earley Parsing in Prolog

Self-modifying Programs in Prolog 00 000	Earley's Algorithm ●0	Appendix O
Short Recap		
Short Recap		

- Earley is a Chart Parser
- The chart is consulted and modified by three main components:

メロト メタト メヨト

Self-modifying Programs in Prolog 00 000	Earley's Algorithm ●○	Appendix O
Short Recap		
Short Recan		

- Earley is a Chart Parser
- The chart is consulted and modified by three main components:
 - The Predictor (Top-Down element)

A B +
 A B +
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A
 A

Self-modifying Programs in Prolog 00 000	Earley's Algorithm ●0	Appendix O
Short Recap		
Short Recap		

- Earley is a Chart Parser
- The chart is consulted and modified by three main components:
 - The Predictor (Top-Down element)
 - The Scanner

メロト メロト メヨト

Self-modifying Programs in Prolog 00 000	Earley's Algorithm ●○	Appendix O
Short Recap		
Short Docon		



- Earley is a Chart Parser
- The chart is consulted and modified by three main components:
 - The Predictor (Top-Down element)
 - The Scanner
 - The Completer (Bottum-Up elment)

< □ > < 同 >

∃ >

Self-modifying Programs in Prolog 00 000	Earley's Algorithm ●○	Appendix O
Short Recap		
Shart Dacan		



- Earley is a Chart Parser
- The chart is consulted and modified by three main components:
 - The Predictor (Top-Down element)
 - The Scanner
 - The Completer (Bottum-Up elment)
- Let's see an example...

< □ > < 同 >

.∋...>

Self-modifying Programs in Prolog 00 000	Earley's Algorithm ○●	Appendix O
Short Recap		
A Short Example		

The dog chases the cat Predict:

· []



Universität Tübingen

Aleksandar L. Dimitrov

Earley Parsing in Prolog

Self-modifying Programs in Prolog 00 000	Earley's Algorithm ○●	Appendix O
Short Recap		
A Short Example		

The dog chases the cat • Predict: $S \rightarrow NP VP$

I.



Aleksandar L. Dimitrov

Earley Parsing in Prolog

Self-modifying Programs in Prolog 00 000	Earley's Algorithm ○●	Appendix O
Short Recap		
A Short Example		

• Predict: $S \rightarrow NP VP$

Predict:

・ロト・日本・日本・日本・ 日本・ シック

Universität Tübingen

Aleksandar L. Dimitrov

Earley Parsing in Prolog

Self-modifying Programs in Prolog 00 000	Earley's Algorithm ○●	Appendix O
Short Recap		
A Short Example		

- Predict: $S \rightarrow NP VP$
- Predict: $NP \rightarrow D N$

・ロ・・聞・・ヨ・・ヨ・ しょうくの

Aleksandar L. Dimitrov Earley Parsing in Prolog

Self-modifying Programs in Prolog 00 000	Earley's Algorithm ○●	Appendix O
Short Recap		
A Short Example		

- Predict: $S \rightarrow NP VP$
- Predict: $NP \rightarrow D N$

Scan:

・ロト・白ト・白ト・白ト 一日 うべの

Aleksandar L. Dimitrov Earley Parsing in Prolog

Self-modifying Programs in Prolog 00 000	Earley's Algorithm ○●	Appendix O
Short Recap		
A Short Example		

- Predict: $S \rightarrow NP VP$
- Predict: $NP \rightarrow D N$

Scan: Accept the

くロ> <四> < 回> < 回> < 回> < 回> < 回> < 回> < ○ののの
</p>

Aleksandar L. Dimitrov Earley Parsing in Prolog

Self-modifying Programs in Prolog 00 000	Earley's Algorithm ○●	Appendix O
Short Recap		
A Short Example		

- Predict: $S \rightarrow NP VP$
- Predict: $NP \rightarrow D N$
- Scan: Accept the
- Complete:

Image: Image:

3 →

Self-modifying Programs in Prolog 00 000	Earley's Algorithm ○●	Appendix O
Short Recap		
A Short Example		

- Predict: $S \rightarrow NP VP$
- Predict: $NP \rightarrow D N$
- Scan: Accept the
- Complete: Nothing.

Image: Image:

-

Self-modifying Programs in Prolog 00 000	Earley's Algorithm ○●	Appendix O
Short Recap		
A Short Example		

- Predict: $S \rightarrow NP VP$
- Predict: $NP \rightarrow D N$
- Scan: Accept the
- Complete: Nothing. Why?

-

Self-modifying Programs in Prolog 00 000	Earley's Algorithm ○●	Appendix O
Short Recap		
A Short Example		

- Predict: $S \rightarrow NP VP$
- Predict: $NP \rightarrow D N$
- Scan: Accept the
- Complete: Nothing. Why?
- Predict:

Self-modifying Programs in Prolog 00 000	Earley's Algorithm ○●	Appendix O
Short Recap		
A Short Example		

- Predict: $S \rightarrow NP VP$
- Predict: $NP \rightarrow D N$
- Scan: Accept the
- Complete: Nothing. Why?
- Predict: Nothing.

Image: A mathematical states and a mathem

-

Self-modifying Programs in Prolog 00 000	Earley's Algorithm ○●	Appendix O
Short Recap		
A Short Example		

- Predict: $S \rightarrow NP VP$
- Predict: $NP \rightarrow D N$
- Scan: Accept the
- Complete: Nothing. Why?
- Predict: Nothing. Why?

< □ > < 同 >

-

Self-modifying Programs in Prolog 00 000	Earley's Algorithm ○●	Appendix O
Short Recap		
A Short Example		

- Predict: $S \rightarrow NP VP$
- Predict: $NP \rightarrow D N$
- Scan: Accept the
- Complete: Nothing. Why?
- Predict: Nothing. Why?
- Scan:

< □ > < 同 >

-

Self-modifying Programs in Prolog 00 000	Earley's Algorithm ○●	Appendix O
Short Recap		
A Short Example		

- Predict: $S \rightarrow NP VP$
- Predict: $NP \rightarrow D N$
- Scan: Accept the
- Complete: Nothing. Why?
- Predict: Nothing. Why?
- Scan: Accept dog

< A

Self-modifying Programs in Prolog 00 000	Earley's Algorithm ○●	Appendix O
Short Recap		
A Short Example		

- Predict: $S \rightarrow NP VP$
- Predict: $NP \rightarrow D N$
- Scan: Accept the
- Complete: Nothing. Why?
- Predict: Nothing. Why?
- Scan: Accept dog
- Complete:

< 口 > < 同

Self-modifying Programs in Prolog 00 000	Earley's Algorithm ○●	Appendix O
Short Recap		
A Short Example		

- Predict: $S \rightarrow NP VP$
- Predict: $NP \rightarrow D N$
- Scan: Accept the
- Complete: Nothing. Why?
- Predict: Nothing. Why?
- Scan: Accept dog
- Complete: Complete NP

Self-modifying Programs in Prolog 00 000	Earley's Algorithm ○●	Appendix O
Short Recap		
A Short Example		

- Predict: $S \rightarrow NP VP$
- Predict: $NP \rightarrow D N$
- Scan: Accept the
- Complete: Nothing. Why?
- Predict: Nothing. Why?
- Scan: Accept dog
- Complete: Complete NP
- etc. . .

_ ∢ ∩ೌ

Self-modifying Programs in Prolog 00 000	Earley's Algorithm ○●	Appendix O
Short Recap		
A Short Example		

- Predict: $S \rightarrow NP VP$
- Predict: $NP \rightarrow D N$
- Scan: Accept the
- Complete: Nothing. Why?
- Predict: Nothing. Why?
- Scan: Accept dog
- Complete: Complete NP
- etc. . .

_ ∢ ∩ೌ

Self-modifying	Programs	Prolog
00		

Implementation

Appendix 0

The Outer Predicates

We only need to modify parse sligthly

・ロト・日本・日本・日本・ 白々の

Aleksandar L. Dimitrov Earley Parsing in Prolog

Self-modifying	Programs	Prolog
000		

Implementation

< □ > < 同 >

Appendix 0

The Outer Predicates

- We only need to modify parse sligthly
- ... and introduce process which is just a wrapper around the three magicians...

Self-modifying	Programs	Prolog
000		

Implementation

< □ > < 同 >

Appendix 0

The Outer Predicates

- We only need to modify parse sligthly
- ... and introduce process which is just a wrapper around the three magicians...

Self-modifying Programs in Prolog 00 000	Earley's Algorithm 00	Implementation	Appendix O

The three Magicians'

• ... make an *exhaustive* search on their current domain

Aleksandar L. Dimitrov Earley Parsing in Prolog Universität Tübingen

・ロト ・回ト ・目ト

Self-modifying	Programs	Prolog	

The three Magicians

- ... make an *exhaustive* search on their current domain
- ... store everything they find on the main chart

Aleksandar L. Dimitrov Earley Parsing in Prolog

Self-modifying	Programs	Prolog

Implementation

< □ > < 同 >

Appendix 0

The three Magicians

- ... make an *exhaustive* search on their current domain
- ... store everything they find on the main chart
- ... but first operate on what they found themselves

Self-modifying	Programs	Prolog
000		

Implementation

The three Magicians

- ... make an *exhaustive* search on their current domain
- ... store everything they find on the main chart
- ... but first operate on what they found themselves (asserta)
- All of them rely on store/3

Self-modifying Programs in Prolog 00 000	Earley's Algorithm 00	Implementation	Appendix 0
The store predicate			

• Consider $NP \rightarrow NP$ Conj NP

・ロト・日本・日本・日本・ 日本・ シック

Aleksandar L. Dimitrov Earley Parsing in Prolog

Self-modifying Programs in Prolog 00 000	Earley's Algorithm 00	Implementation	Appendix O
The store predicate			

- Consider $NP \rightarrow NP$ Conj NP
- How does Earley avoid left recursion?

・ロト・イラト・イミト ・ミー 少へぐ Universität Tübingen

00	
000	

The store predicate

- Consider $NP \rightarrow NP$ Conj NP
- How does Earley avoid left recursion?
- store doesn't store anything that's already there!
- But how do we parse recursive structures such as The dog and the cat and the elephant?

Self-modifying	Programs	Prolog	
000			

Earley's Algorithm

Implementati

Appendix

What do we do about empty determiners?

What about them?

Aleksandar L. Dimitrov Earley Parsing in Prolog

Self-modifying	Programs	Prolog
00		
000		

Earley's Algorithm

Implementation

Appendix

What do we do about empty determiners?

What about them?

We could: modify the predictor

・ロト・「聞・・言・・言・」 しゃくの

Aleksandar L. Dimitrov Earley Parsing in Prolog

Self-modifying	Programs	Prolog
00 000		

Earley's Algorithm

Implementati

Image: Image:

Appendix

What do we do about empty determiners?

- What about them?
- We could: modify the predictor
- Or recast $D o \emptyset$ as word and modify the scanner

Self-modifying	Programs	Prolog
00 000		

Earley's Algorithm

< □ > < 同 >

What do we do about empty determiners?

- What about them?
- We could: modify the predictor
- Or recast $D \to \emptyset$ as word and modify the scanner
- Method 1: modifying the predictor...

Self-modifying	Programs	Prolog
00 000		

Earley's Algorithm

< □ > < 同 >

What do we do about empty determiners?

- What about them?
- We could: modify the predictor
- Or recast $D o \emptyset$ as word and modify the scanner
- Method 1: modifying the predictor...
- What about Method 2?