Parsing to Stanford Dependencies: Trade-offs between speed and accuracy

Daniel Cer, Marie-Catherine de Marneffe
Daniel Jurafsky, Christopher D. Manning

Stanford Dependencies

Overview

About the Representation

- Widely used
- Semantically-oriented
- **Slow to extract**

Extraction Bottleneck

- Stanford lexicalized phrase structure parser

Are There Faster and Better Approaches?

- Dependency parsing algorithms
- Alternate phrase structure parsers

Organization

Brief Review of Stanford Dependencies

Properties

Extraction pipeline

Experiments Comparing Parsing Approaches

Dependency	Phrase structure
MaltParser	Berkeley
MSTParser	Bikel
	Charniak

Search Space Pruning with Charniak-Johnson

What We'll Show

Performing **dependency parsing** using a phrase structure parser followed by rule based extraction is **more accurate** and, in some cases, **faster** then using statistical dependency parsing algorithms.

What We’ll Show

Performing **dependency parsing** using a phrase structure parser followed by rule based extraction is **more accurate** and, in some cases, **faster** then using statistical dependency parsing algorithms.

For English
using the Stanford Dependency formalism

Stanford Dependencies

What We'll Show

Performing **dependency parsing** using a phrase structure parser followed by rule based extraction is **more accurate** and, in some cases, **faster** then using statistical dependency parsing algorithms.

For English
using the Stanford Dependency formalism

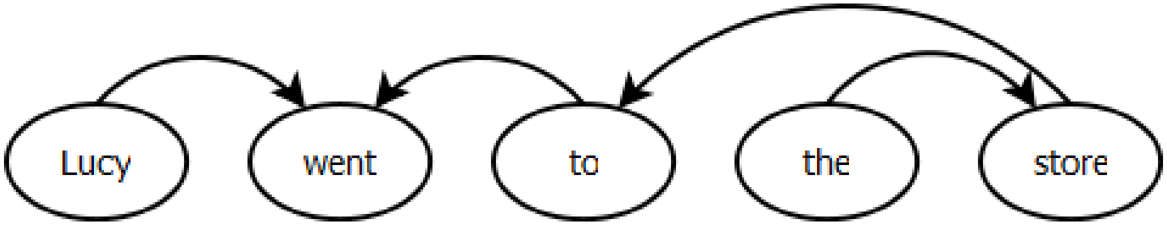
However, we suspect the results maybe more general

Stanford Dependencies

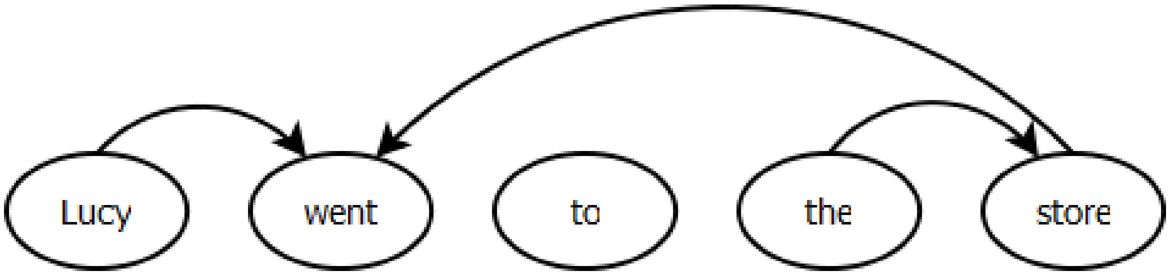
Semantically Oriented

Capture relationships between **content words**

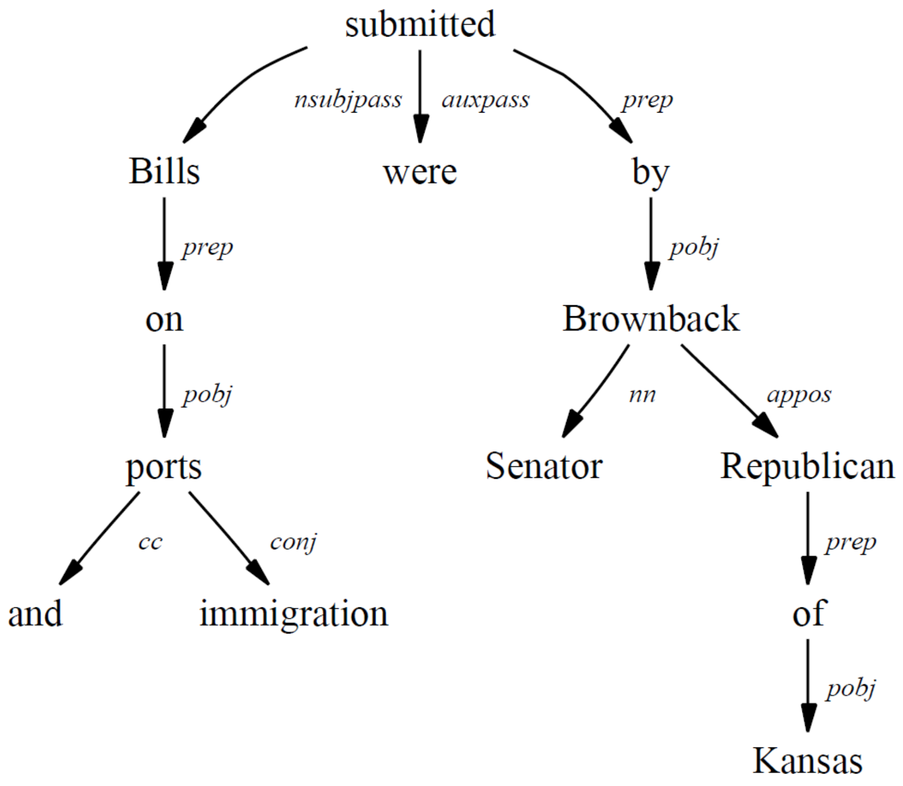
Syntactic Dependencies



Stanford Dependencies



Basic Dependencies

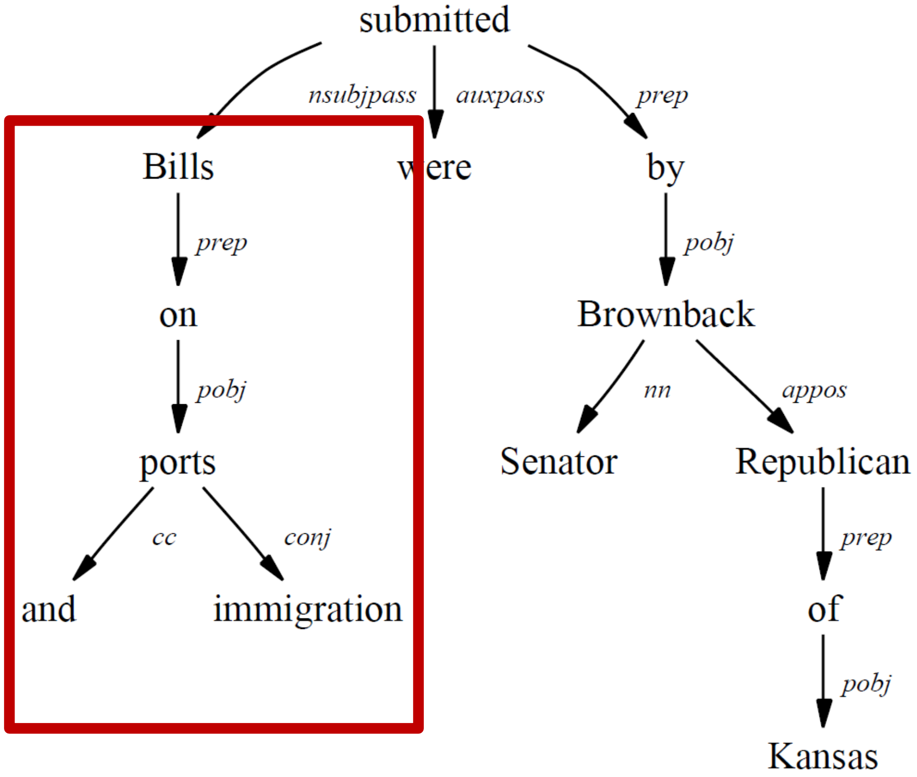


Start out by extracting syntactic heads

Results in a **projective** dependency tree

Stanford Dependencies

Basic Dependencies



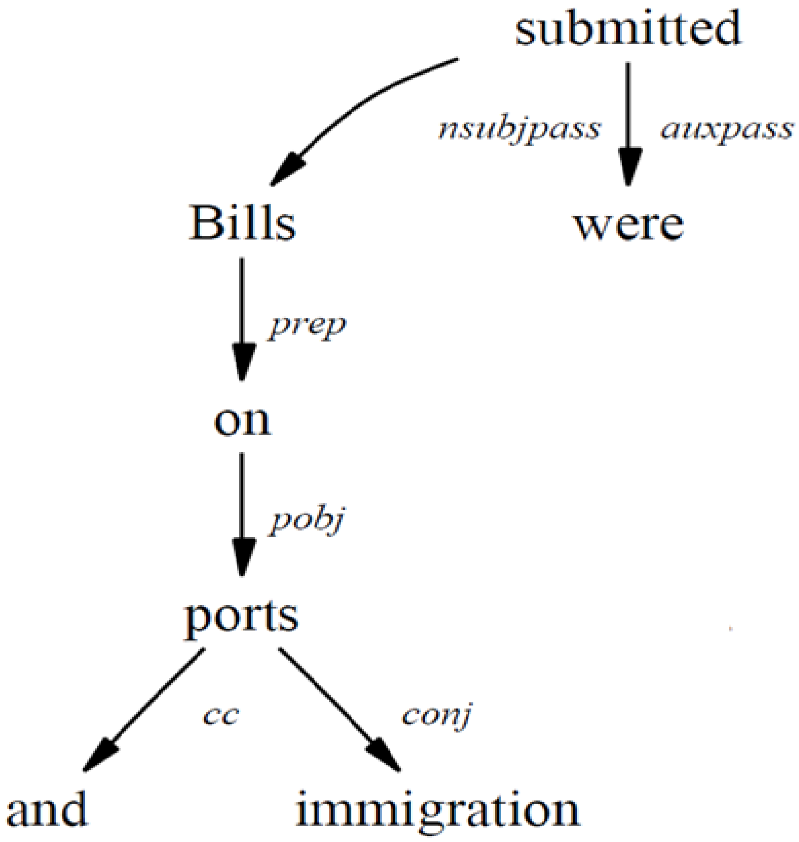
Start out by extracting syntactic heads

Results in a **projective** dependency tree

Stanford Dependencies

Collapsed Dependencies

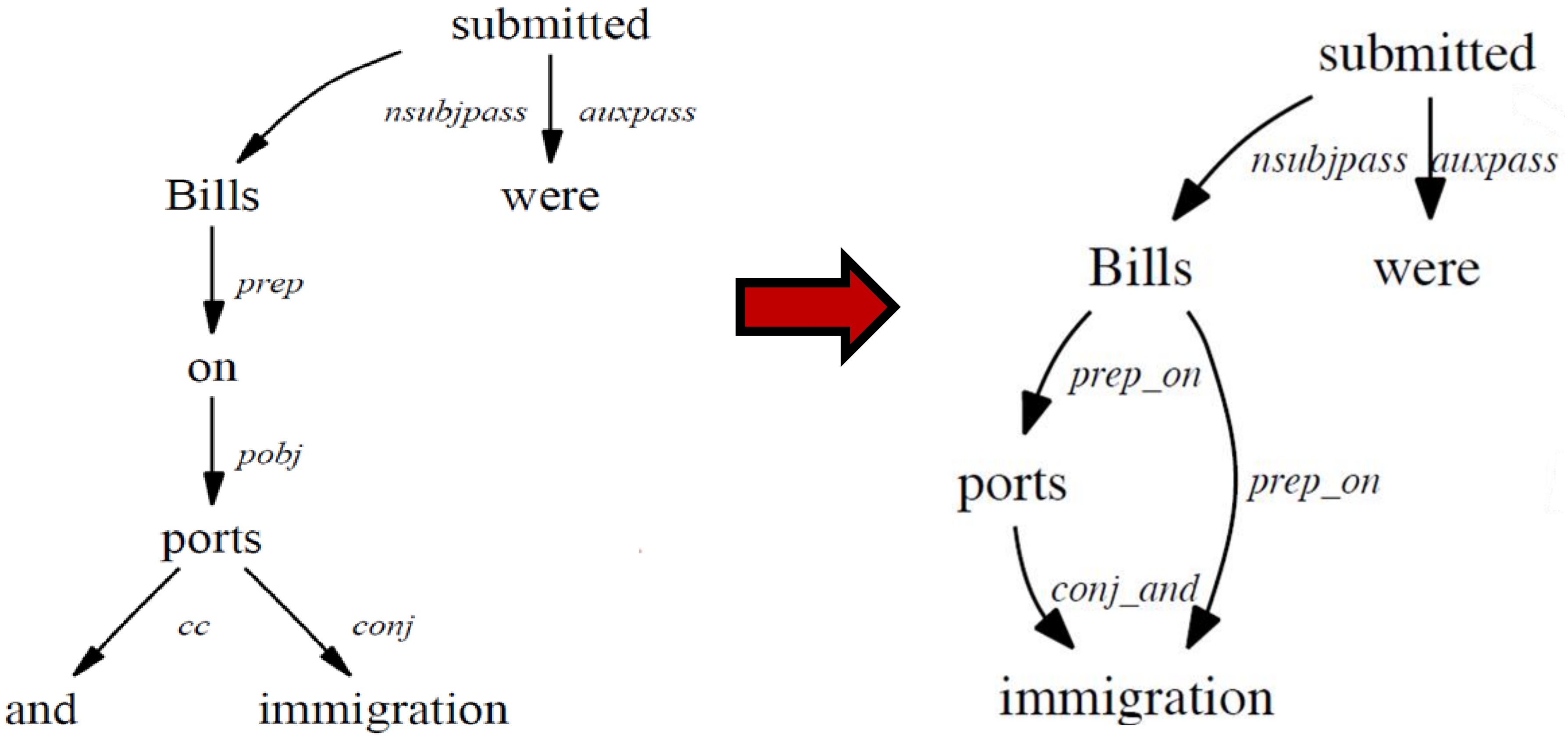
Bills on ports and immigration



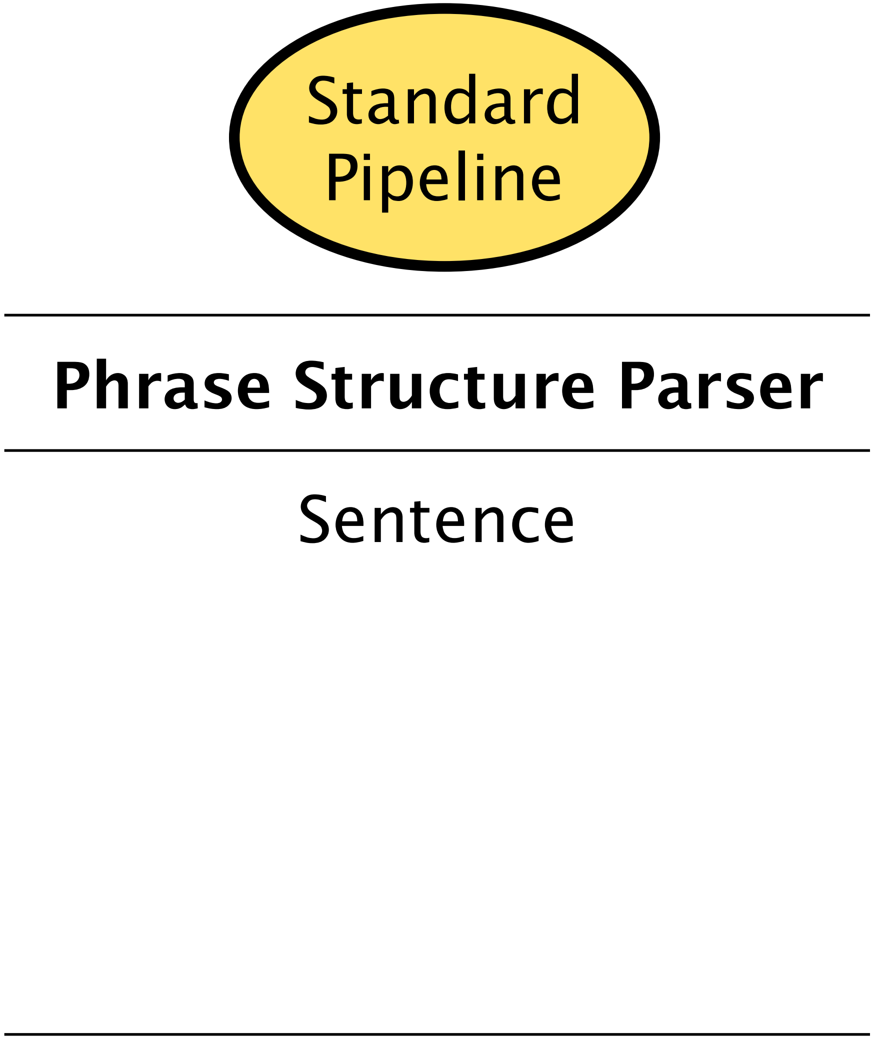
Stanford Dependencies

Collapsed Dependencies

Bills on ports and immigration



Obtaining the Dependencies



Standard Pipeline

Phrase Structure Parser

Sentence

Obtaining the Dependencies

Standard Pipeline

Phrase Structure Parser

Sentence

Constituent Parse Tree

Stanford Dependencies

Obtaining the Dependencies

Standard Pipeline

Phrase Structure Parser

Sentence

Constituent Parse Tree

Projective Basic Dependencies

Obtaining the Dependencies

Standard Pipeline

Phrase Structure Parser

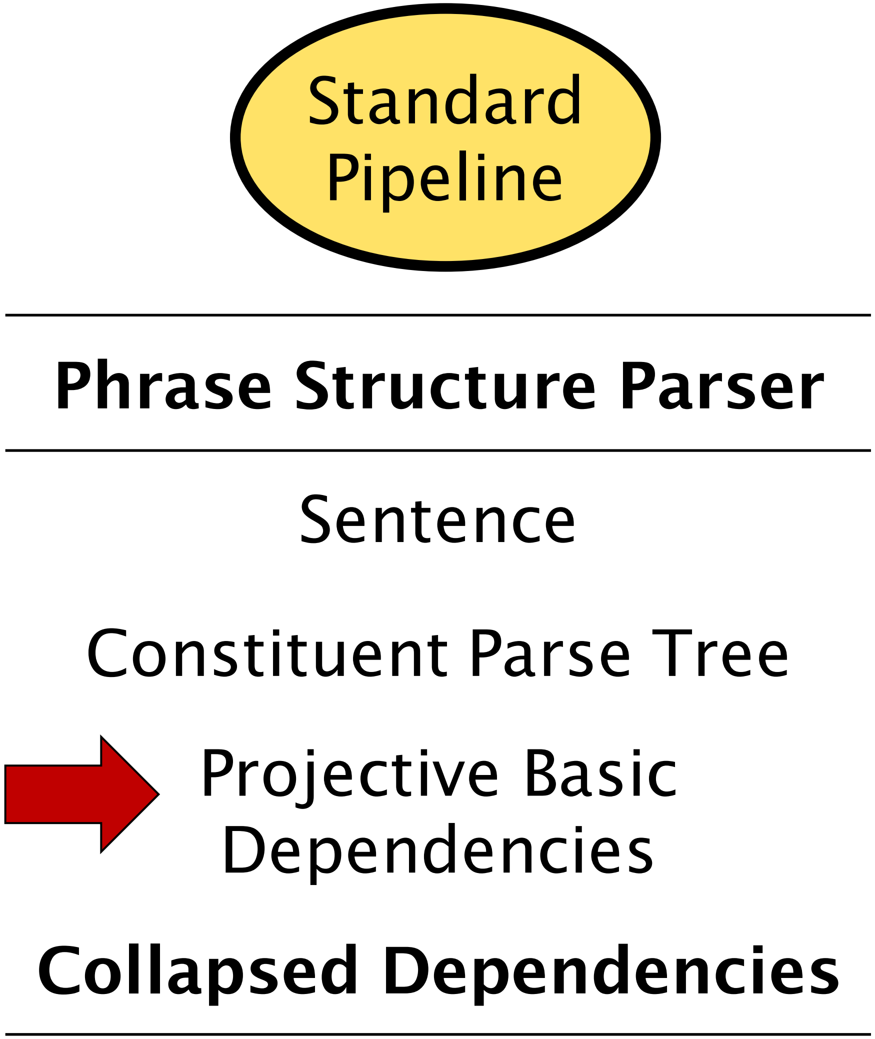
Sentence

Constituent Parse Tree

Projective Basic Dependencies

Collapsed Dependencies

Obtaining the Dependencies



Standard Pipeline

Phrase Structure Parser

Sentence

Constituent Parse Tree

Projective Basic Dependencies

Collapsed Dependencies

Obtaining the Dependencies

Standard Pipeline

Direct Pipeline

Phrase Structure Parser	**Dependency parser**
Sentence	Sentence
Constituent Parse Tree	
➡ Projective Basic Dependencies	
Collapsed Dependencies	

Obtaining the Dependencies

Standard Pipeline	Direct Pipeline
Phrase Structure Parser	**Dependency parser**
Sentence	Sentence
Constituent Parse Tree	
Projective Basic Dependencies	Projective Basic Dependencies
Collapsed Dependencies	

Obtaining the Dependencies

Standard Pipeline	Direct Pipeline
Phrase Structure Parser	**Dependency parser**
Sentence	Sentence
Constituent Parse Tree	
Projective Basic Dependencies	Projective Basic Dependencies
Collapsed Dependencies	**Collapsed Dependencies**

Experimental

RESULTS BY PIPELINE TYPE

Results by Pipeline Type

Method

Train Penn Treebank Sections 2 through 21

Test Penn Treebank Section 22

Dependency parsers

	Malt Parser	**MSTParser**
Algorithm	Nivre Eager, Nivre, Covington	Eisner
Classifier	LibLinear, LibSVM	Factored MIRA

RelEx CMU Link grammar parser in Stanford compatibility mode

Results by Pipeline Type

Method

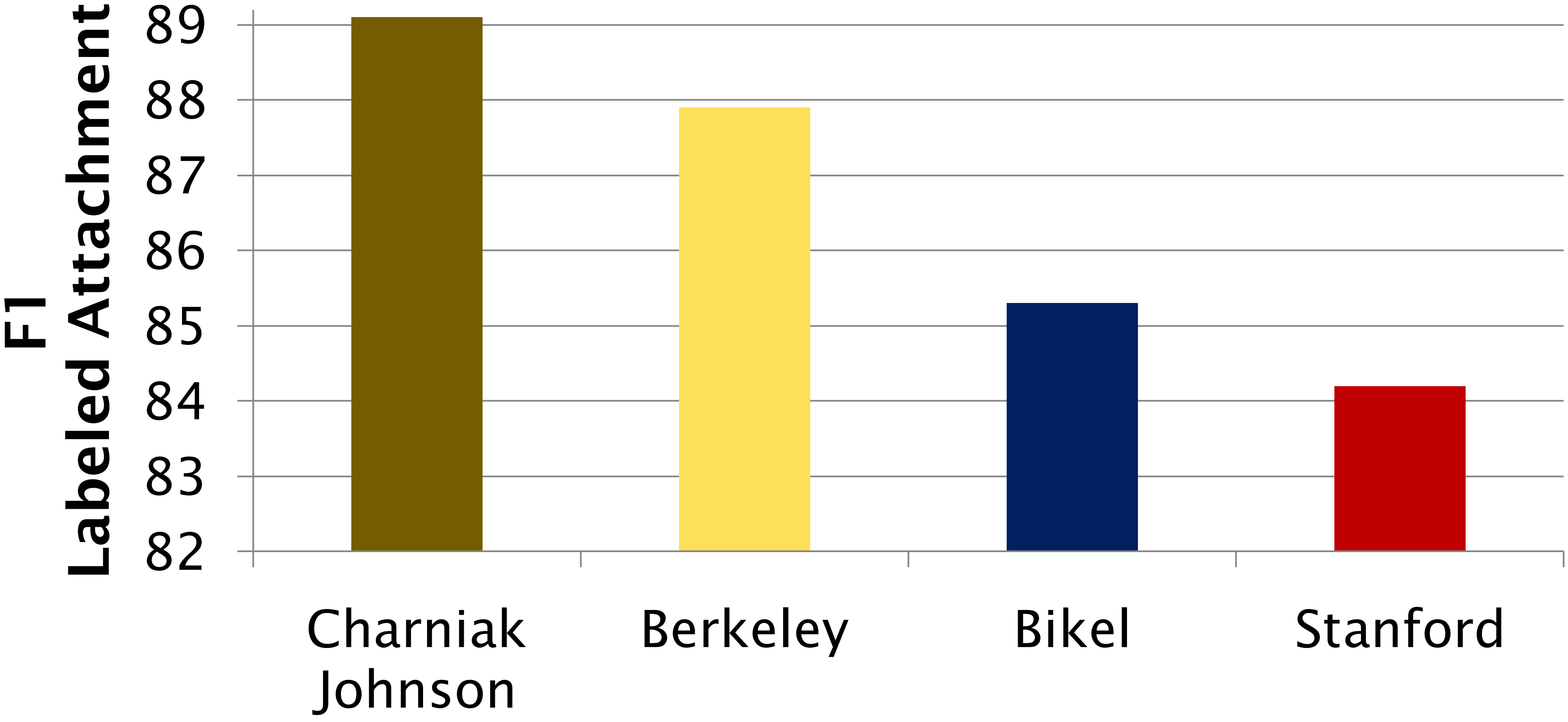
Train Penn Treebank Sections 2 through 21

Test Penn Treebank Section 22

Phrase Structure Parsers

- Charniak
- Charniak Johnson Reranking
- Bikel
- Berkeley
- Stanford

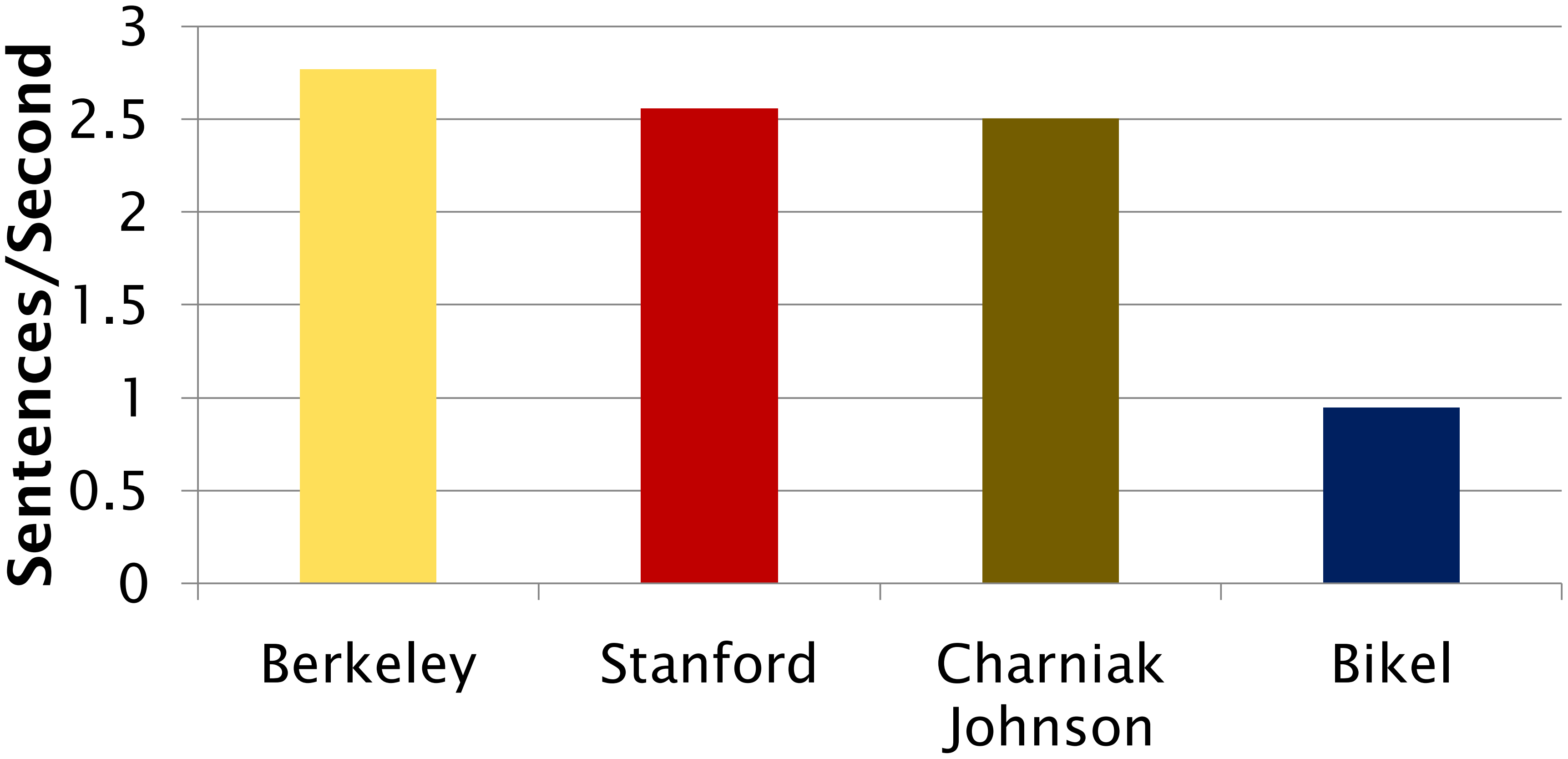
Phrase Structure Parser **Accuracy**



Best: Charniak Johnson Reranking Parser

Results by Pipeline Type

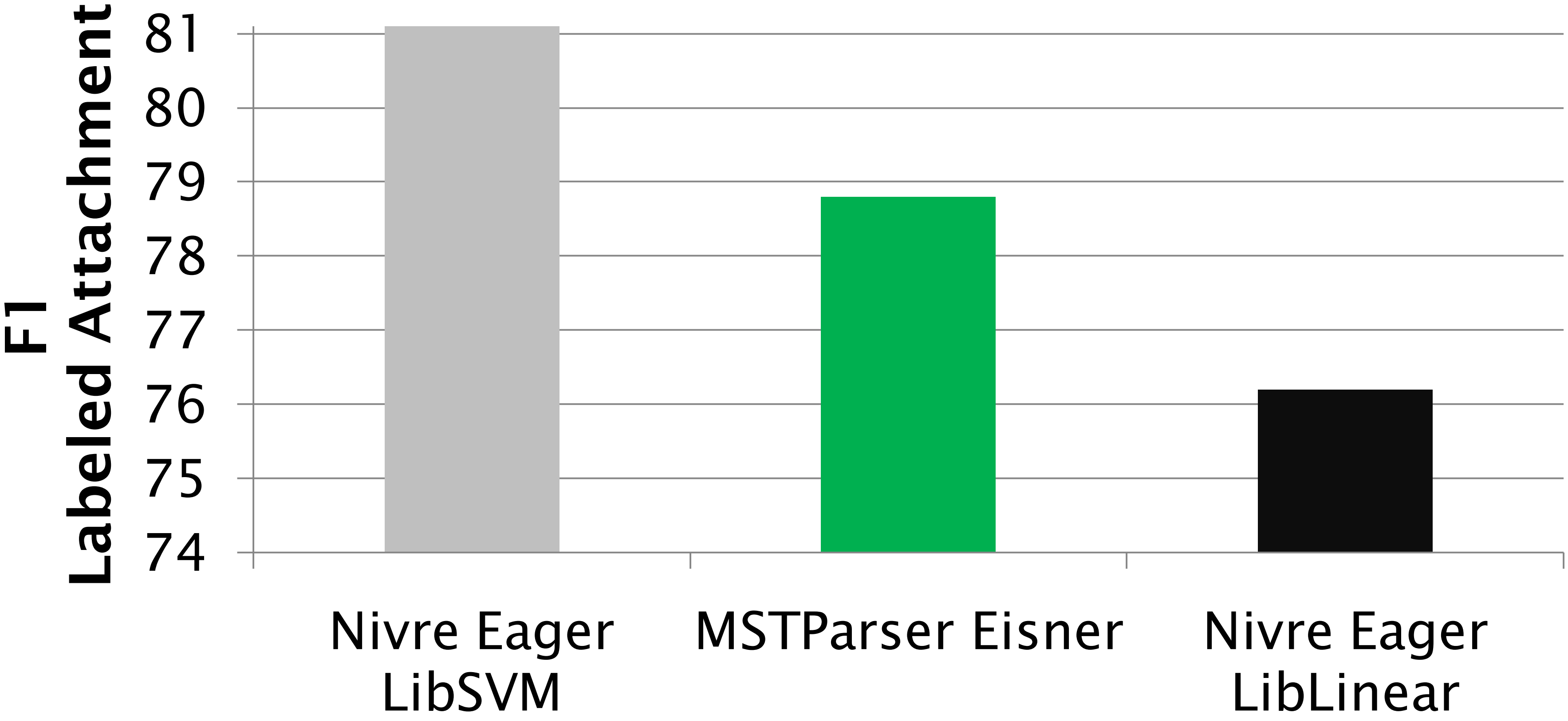
Phrase Structure Parser **Speed**



Parse times similar except for Bikel

Results by Pipeline Type

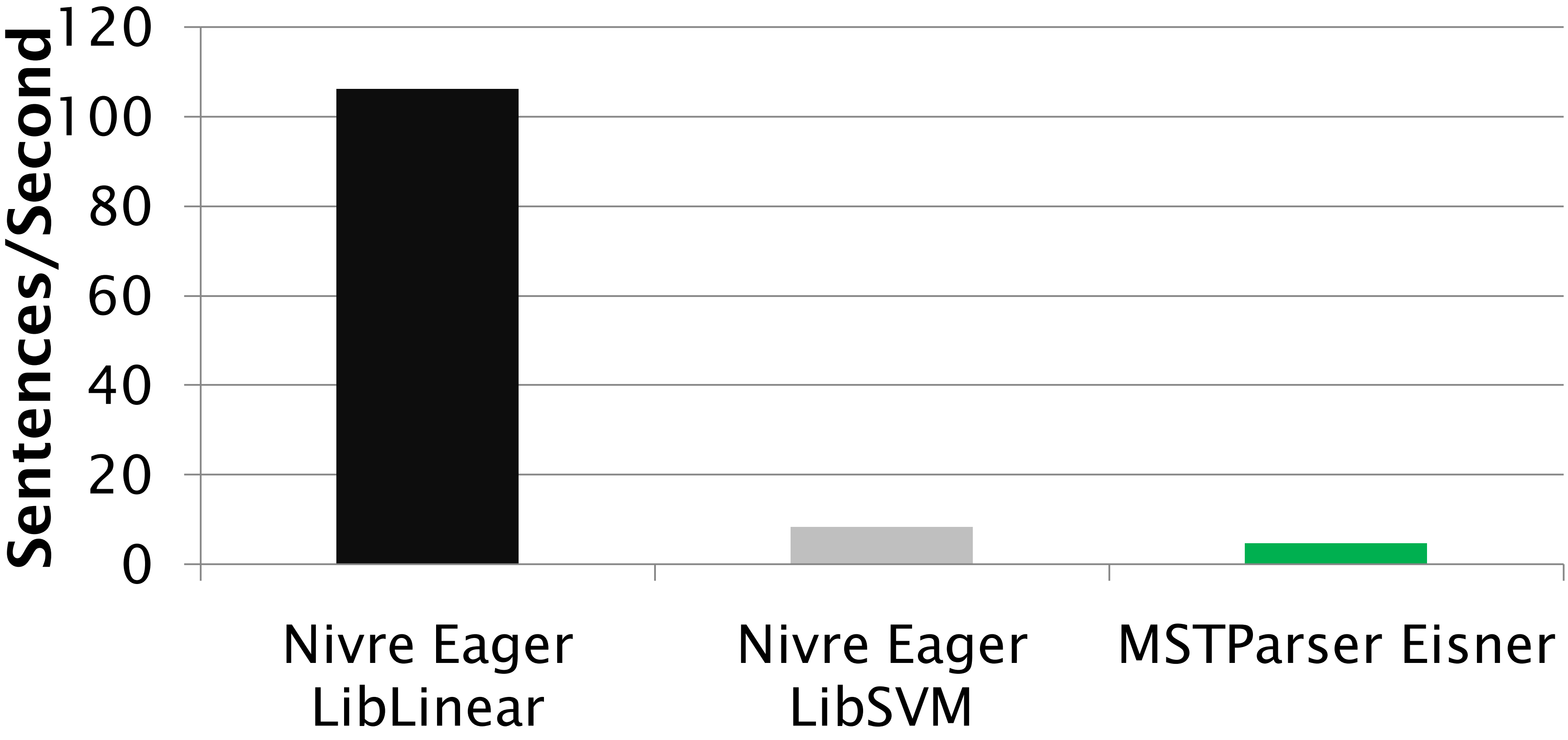
Dependency Parser **Accuracy**



Best: Nivre Eager LibSVM

Results by Pipeline Type

Dependency Parser **Speed**



Fastest: Nivre Eager LibLinear

Comparison of

SPEED AND ACCURACY TRADE-OFFS

Speed and Accuracy Trade-Offs

Worst vs. Best **Accuracy**

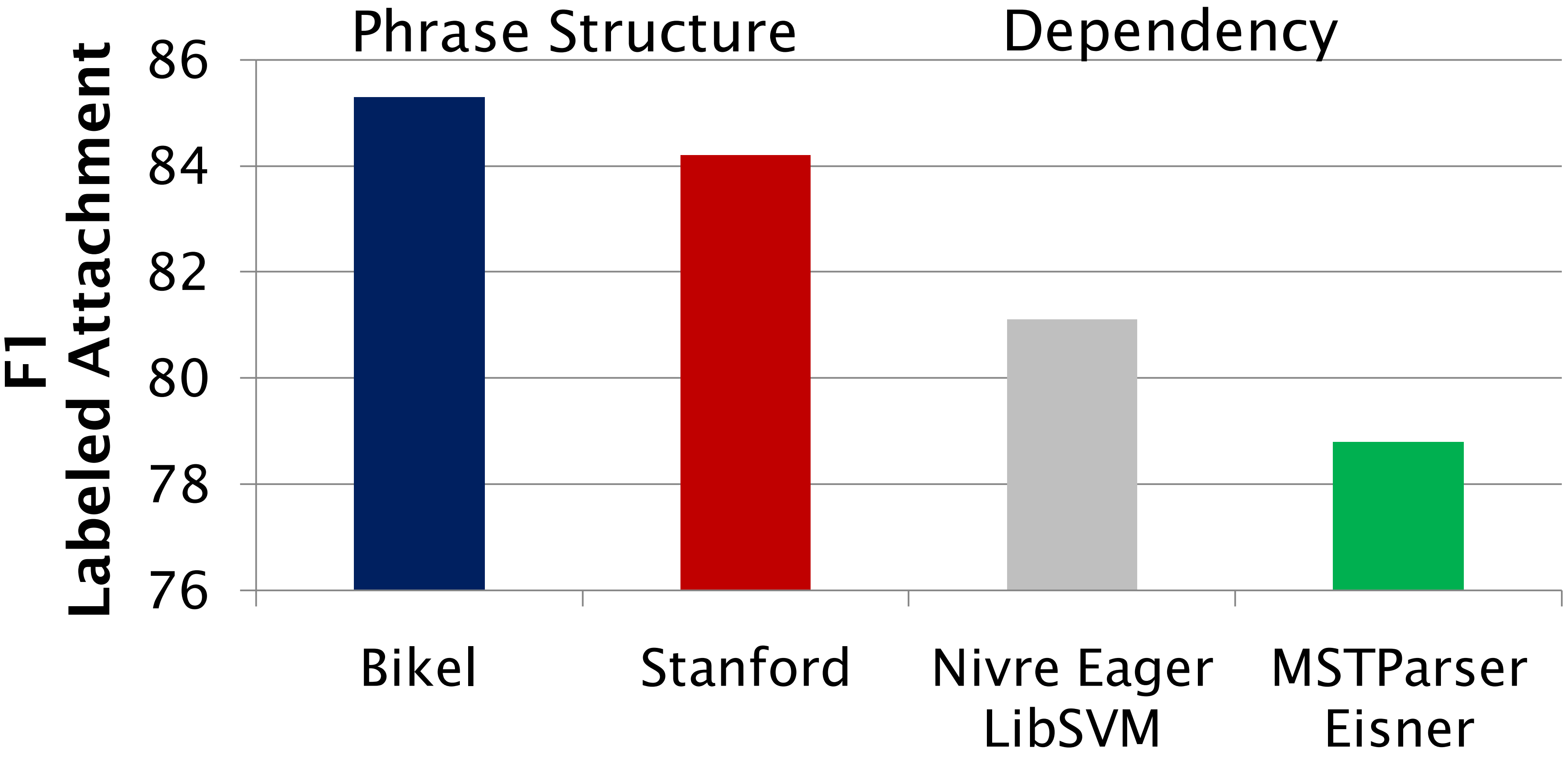
Worst Phrase Structure Parser

vs.

Best Dependency Parser

Speed and Accuracy Trade-Offs

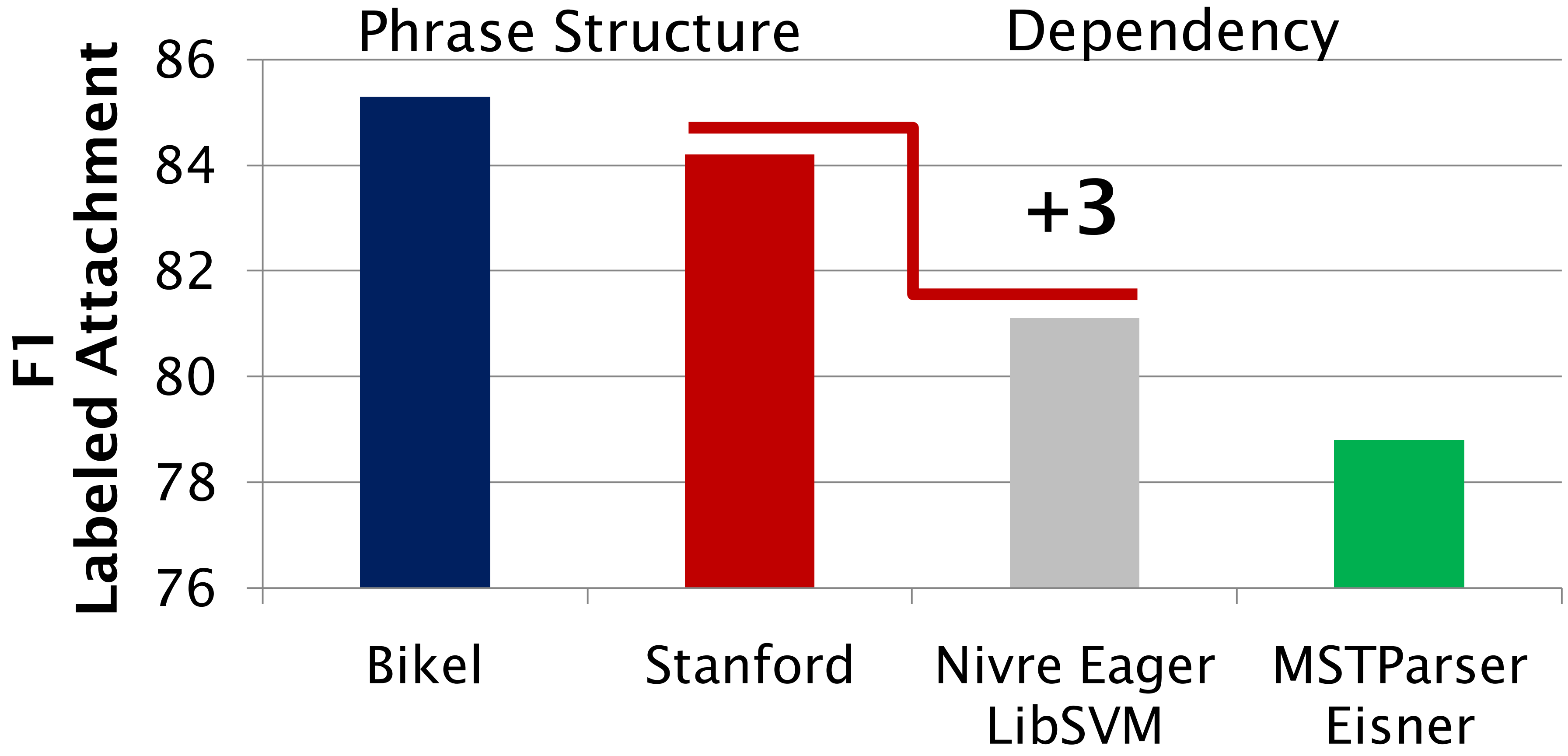
Worst vs. Best **Accuracy**



Worst phrase structure parser better than Best dependency parser

Speed and Accuracy Trade-Offs

Worst vs. Best **Accuracy**



Worst phrase structure parser better than Best dependency parser

Speed and Accuracy Trade-Offs

Best vs. Best **Accuracy**

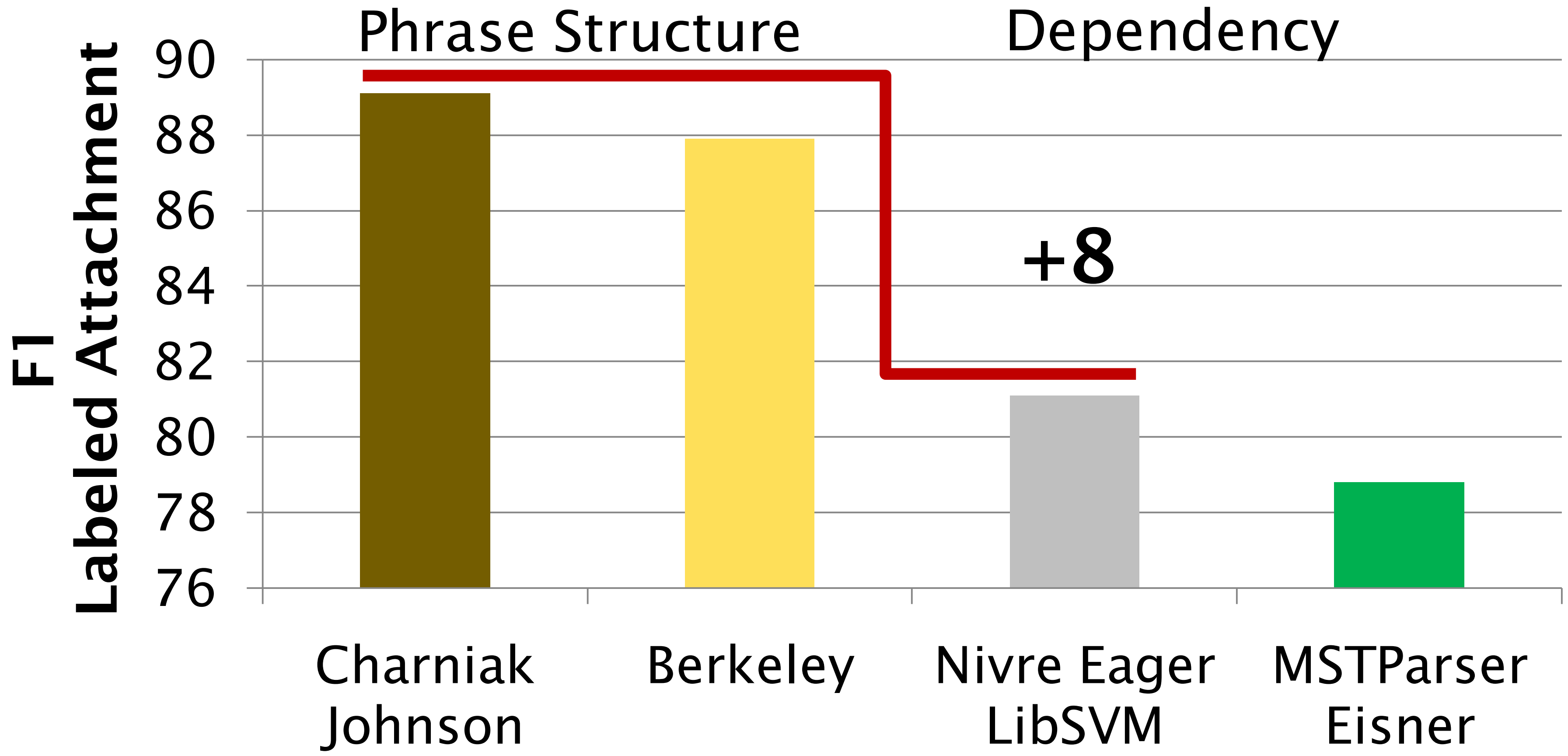
<u>Best</u> Phrase Structure Parser

vs.

<u>Best</u> Dependency Parser

Speed and Accuracy Trade-Offs

Best vs. Best **Accuracy**



Eight point difference between
Best phrase structure and Best dependency parser

Speed and Accuracy Trade-Offs

Worst vs. Best **Speed**

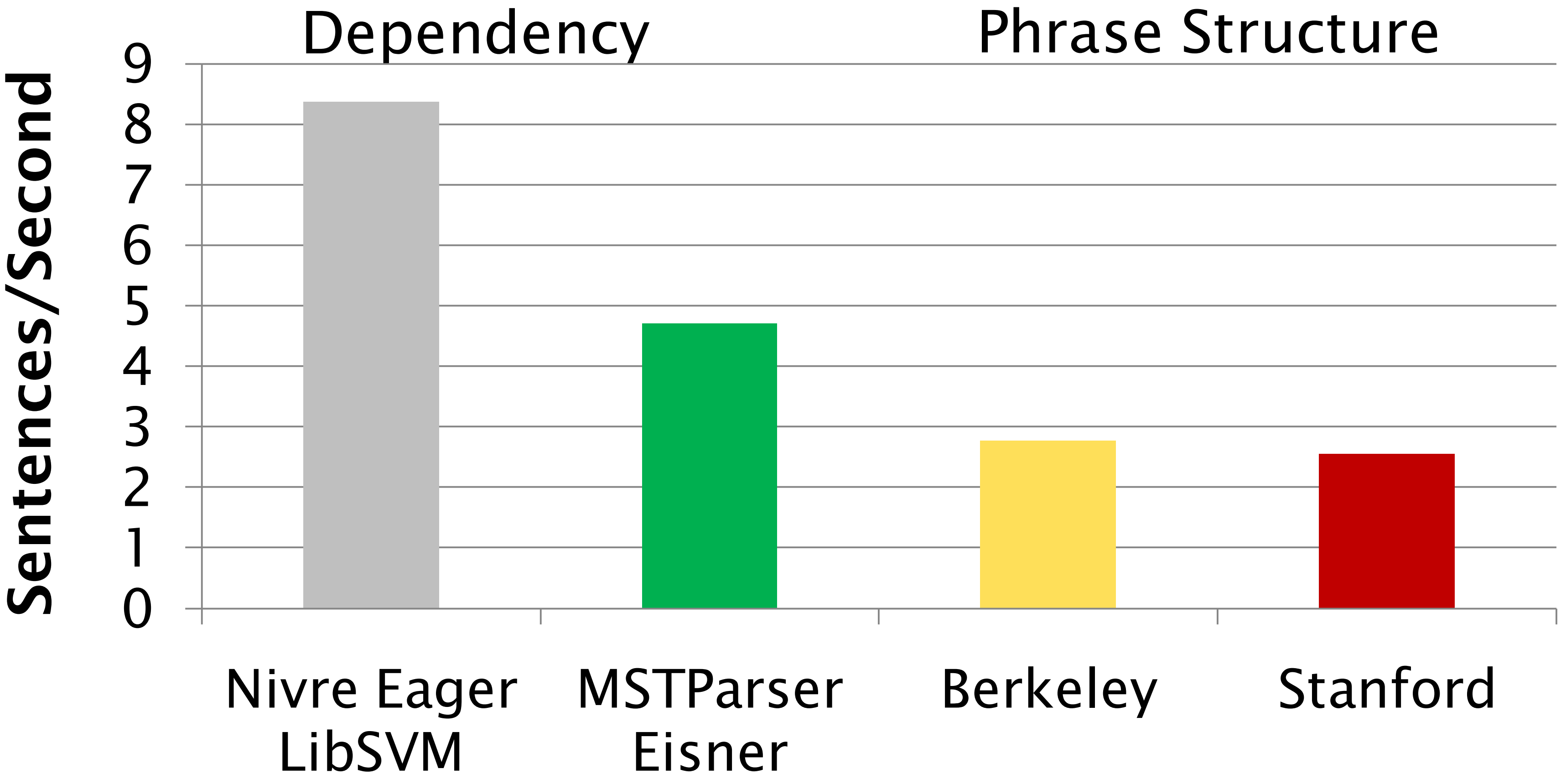
Worst Dependency Parser

vs.

Best Phrase Structure Parser

Speed and Accuracy Trade-Offs

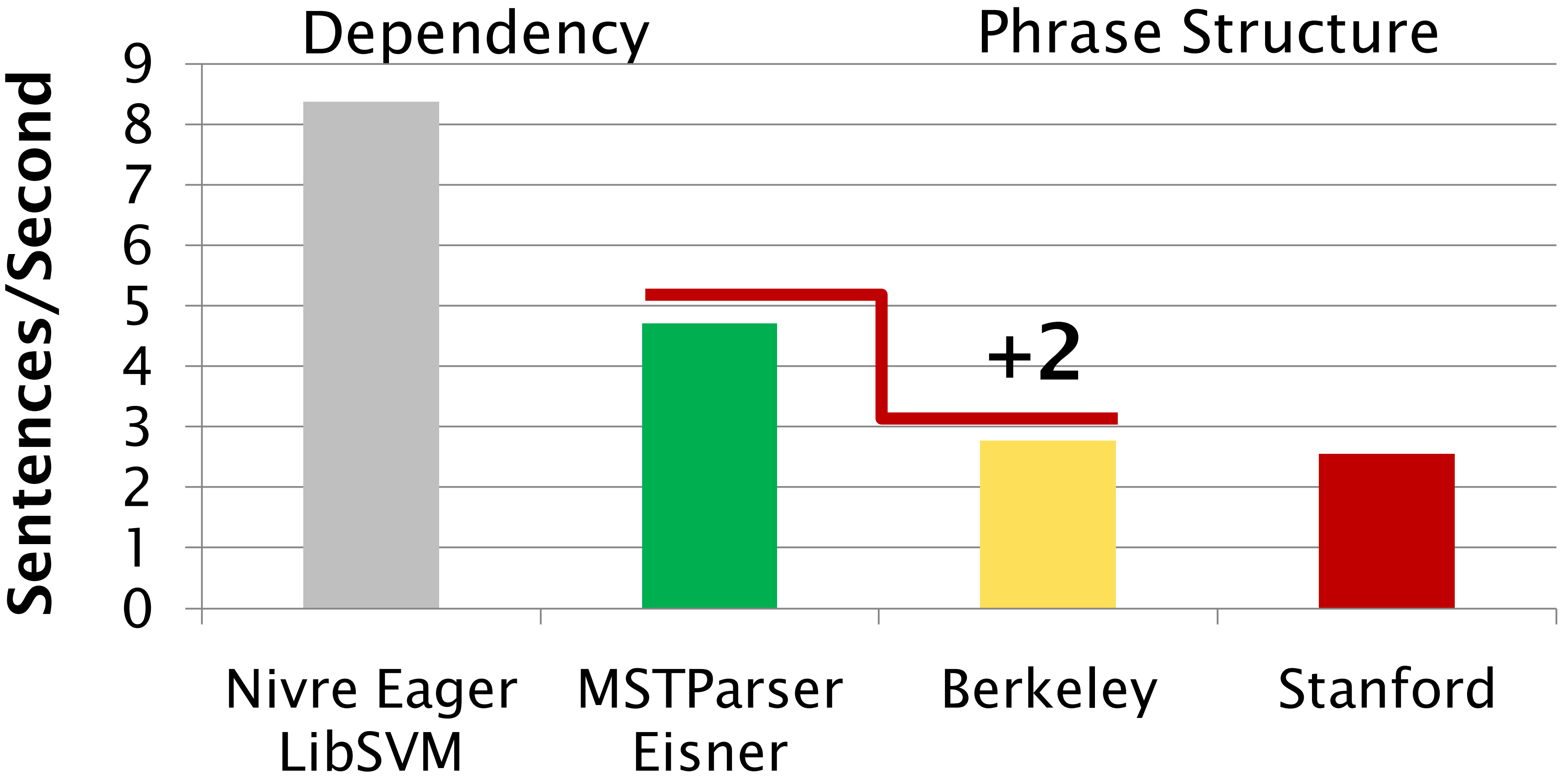
Worst vs. Best **Speed**



Worst dependency parser better than

Best phrase structure parser

Worst vs. Best **Speed**



Worst dependency parser better than

Best phrase structure parser

Best vs. Best **Speed**

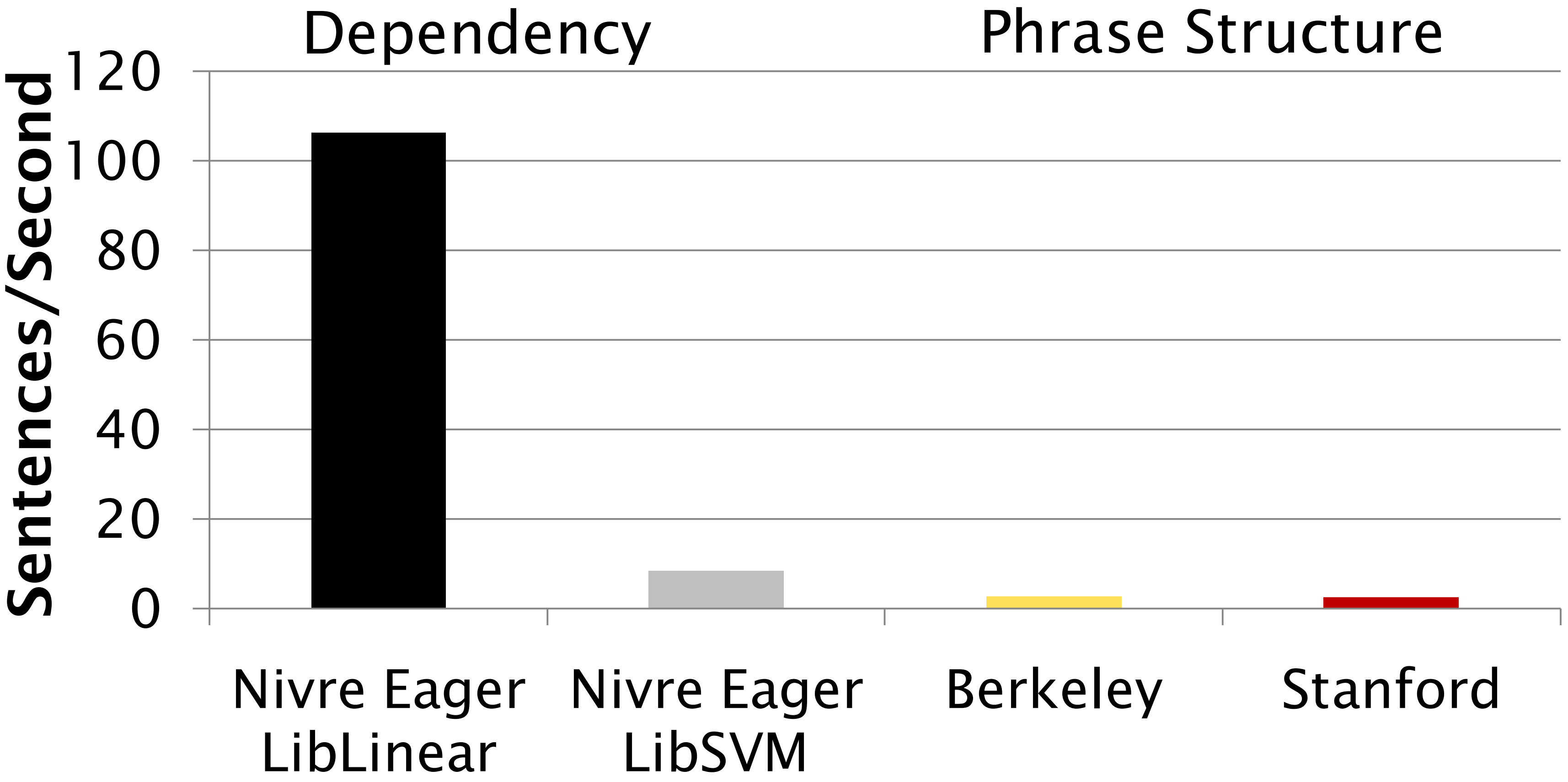
Best Dependency Parser

vs.

Best Phrase Structure Parser

Speed and Accuracy Trade-Offs

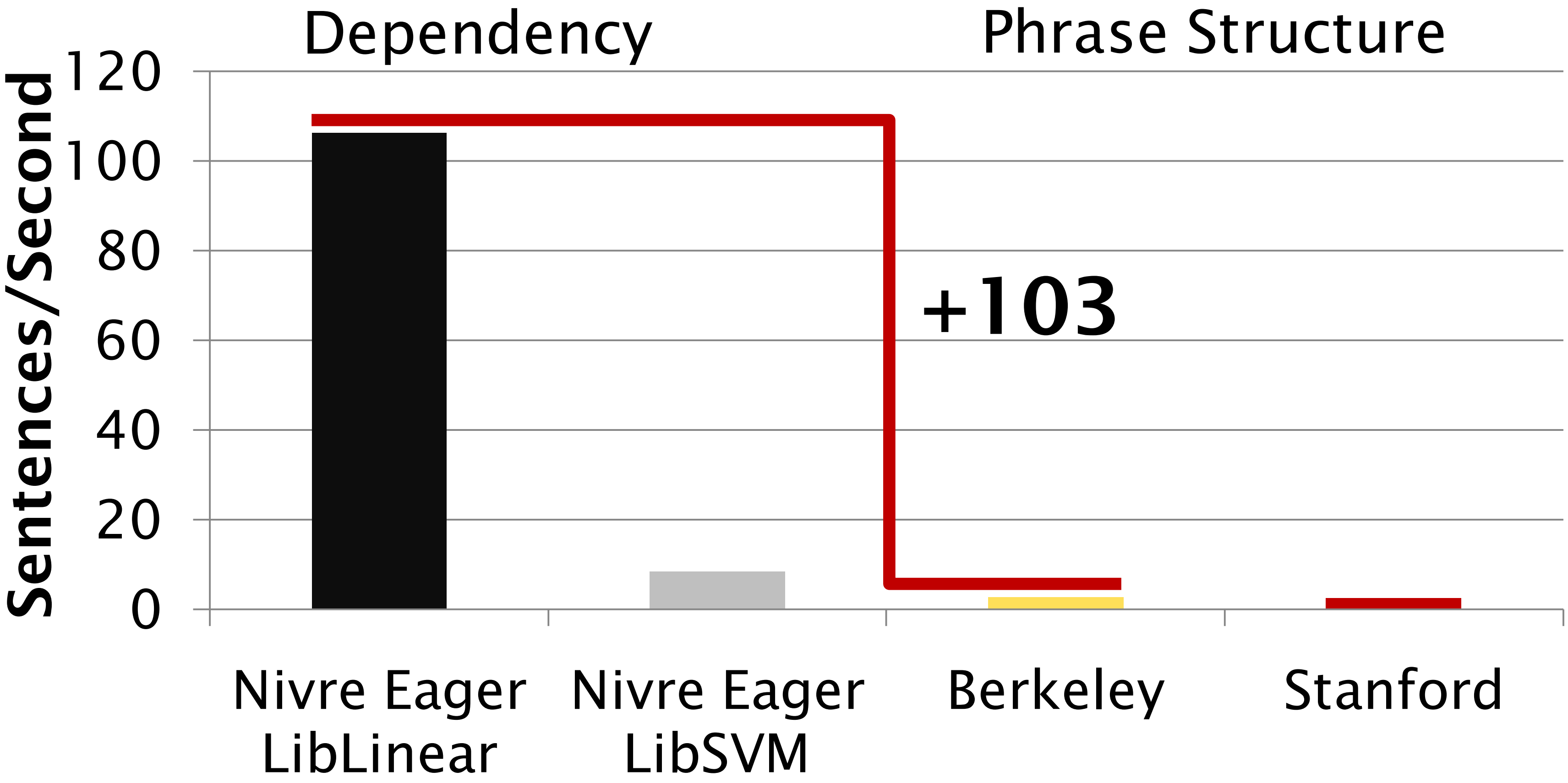
Best vs. Best **Speed**



103 sentences/second difference between
Best dependency and Best phrase structure parser

Speed and Accuracy Trade-Offs

Best vs. Best **Speed**



103 sentences/second difference between
<u>Best</u> dependency and <u>Best</u> phrase structure parser

Speed and Accuracy Trade-Offs

Out-of-the-Box Summary

Accuracy

Use Phrase Structure Parsers

Best choice: Charniak Johnson Reranking

Speed

Use Dependency Parsers

Best Choice: Nivre Eager* with LibLinear

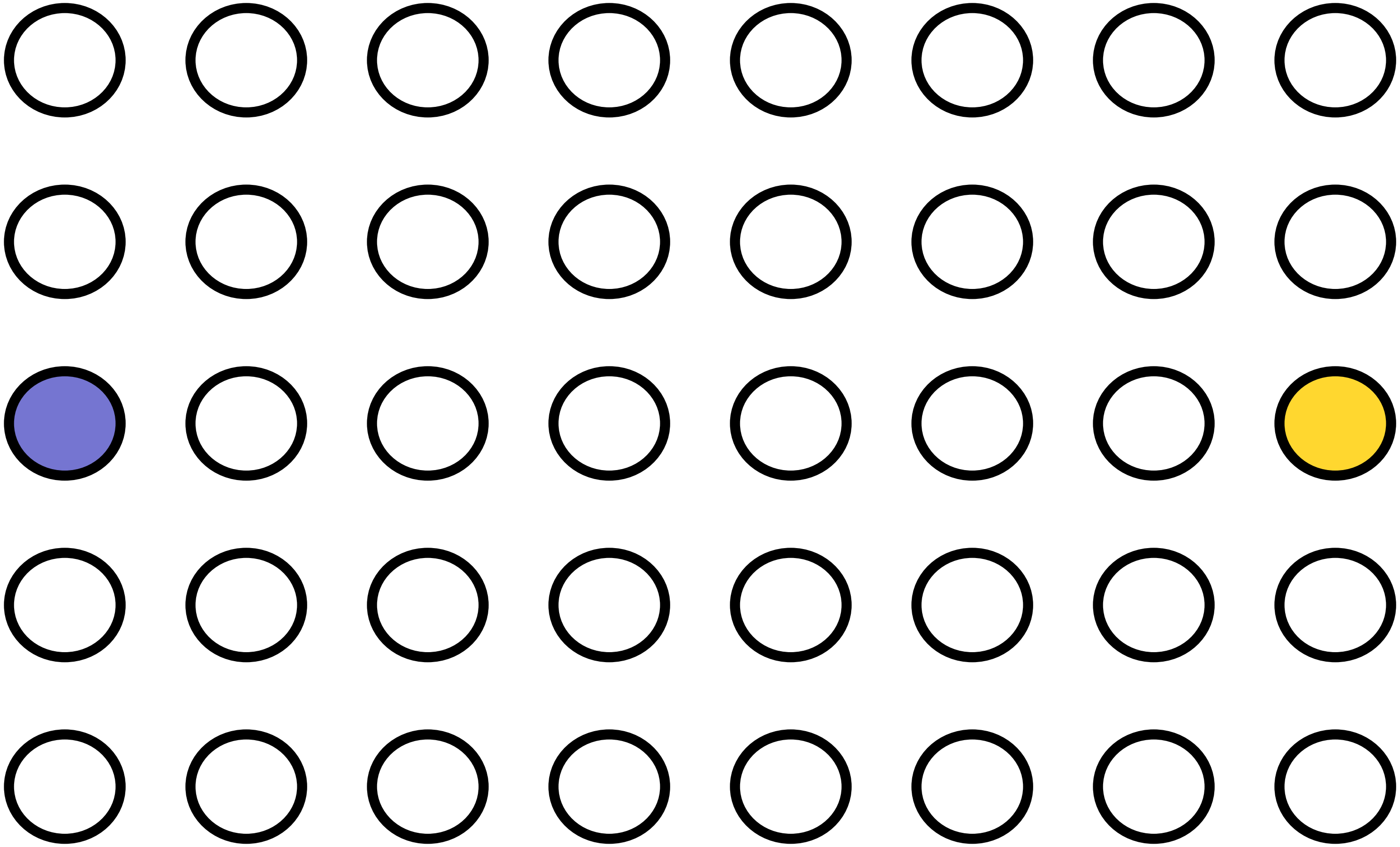
* Actually, any parser in the MaltParser package will do.

Making Use Of

CHARNIAK JOHNSON SEARCH SPACE PRUNING

Example Search

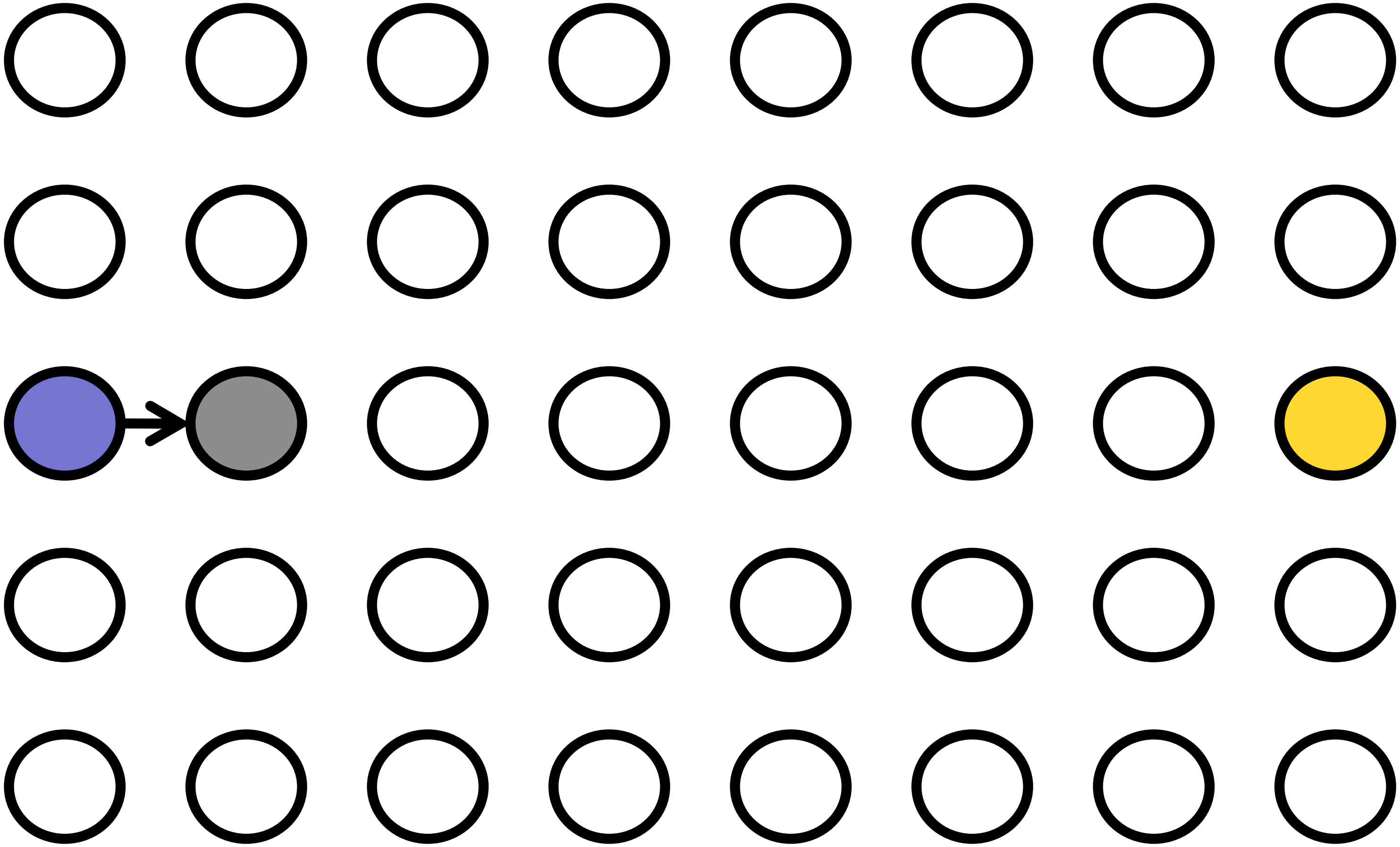
Best First Search



Charniak Johnson Search Space Pruning

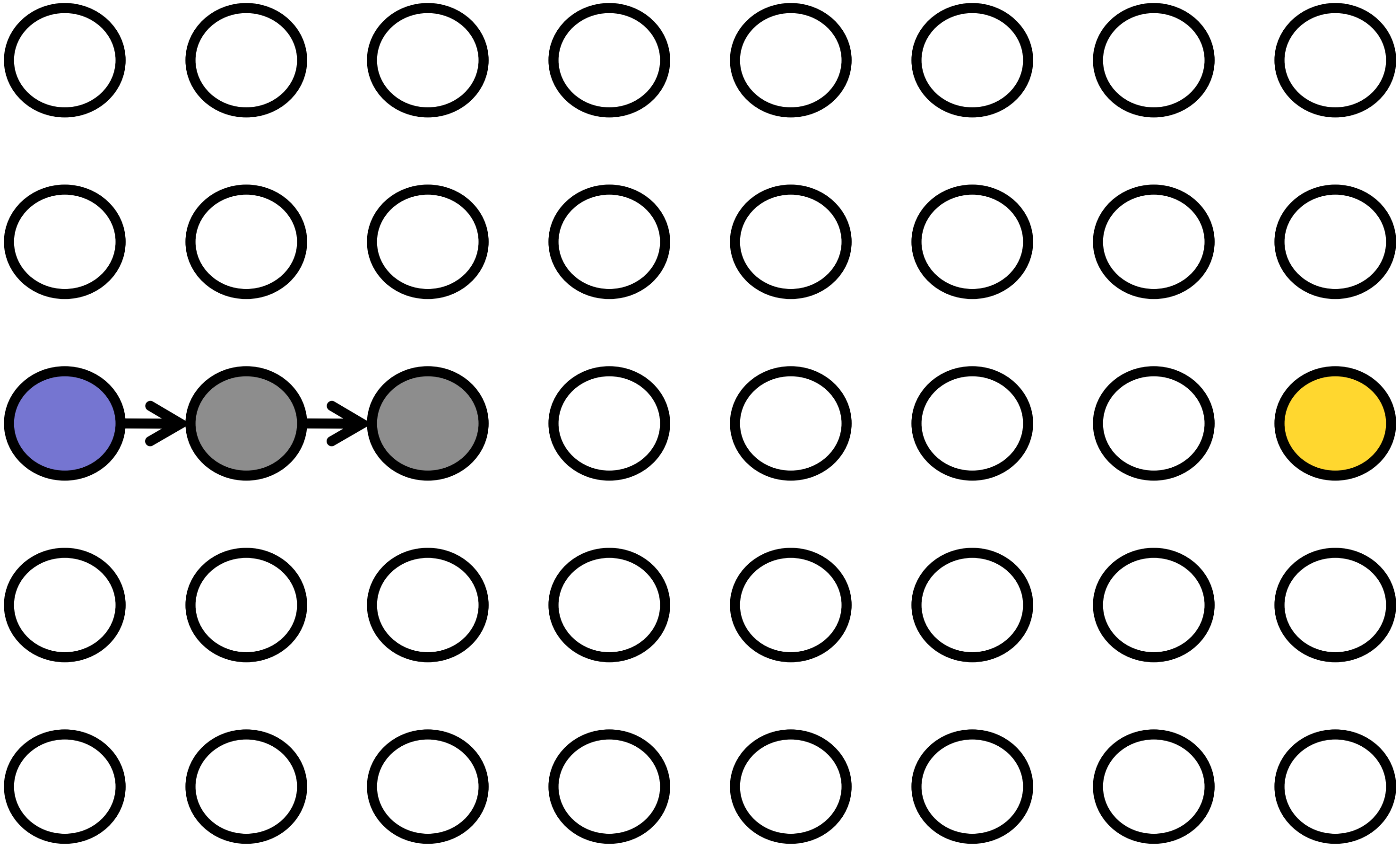
Example Search

Best First Search



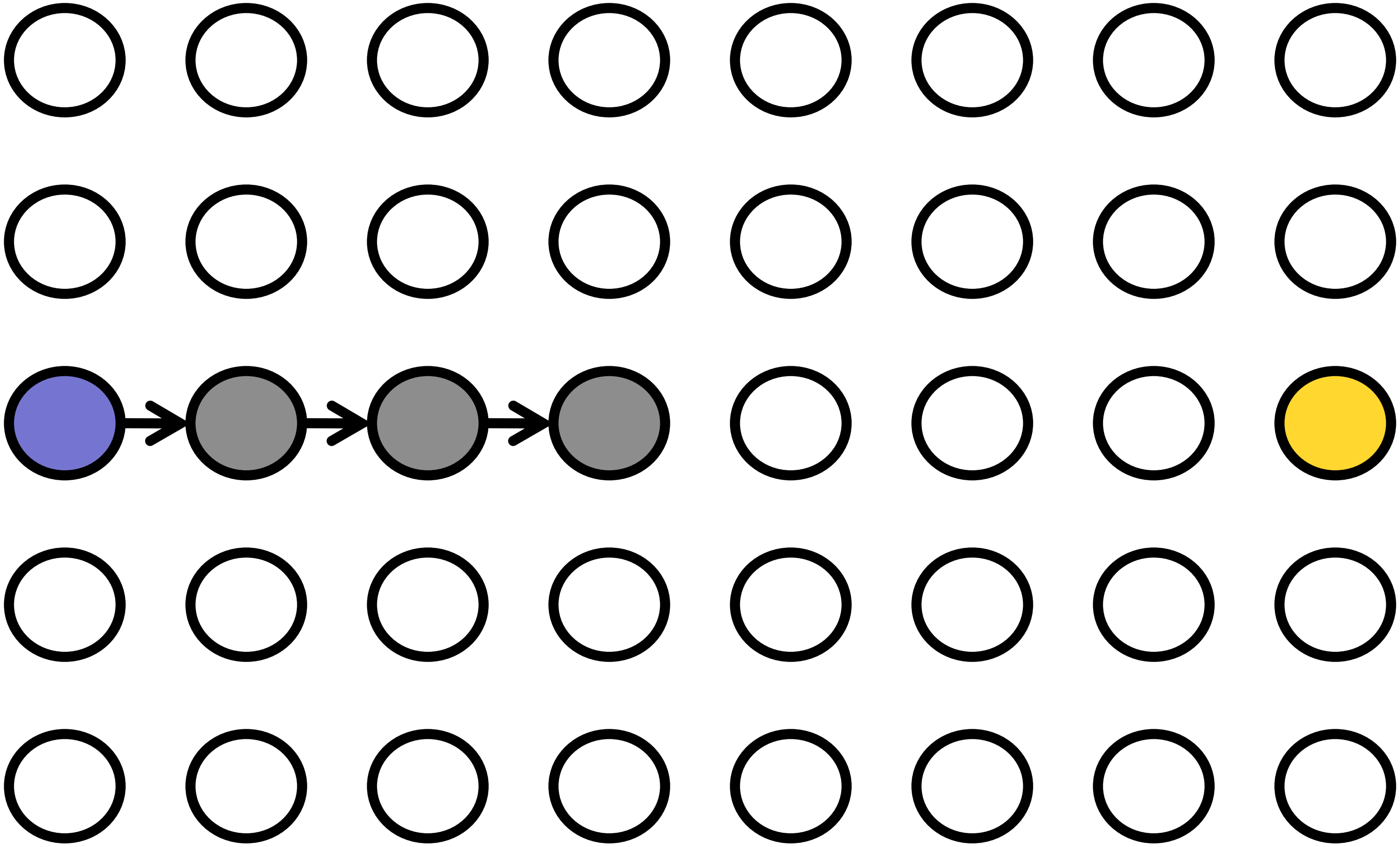
Example Search

Best First Search



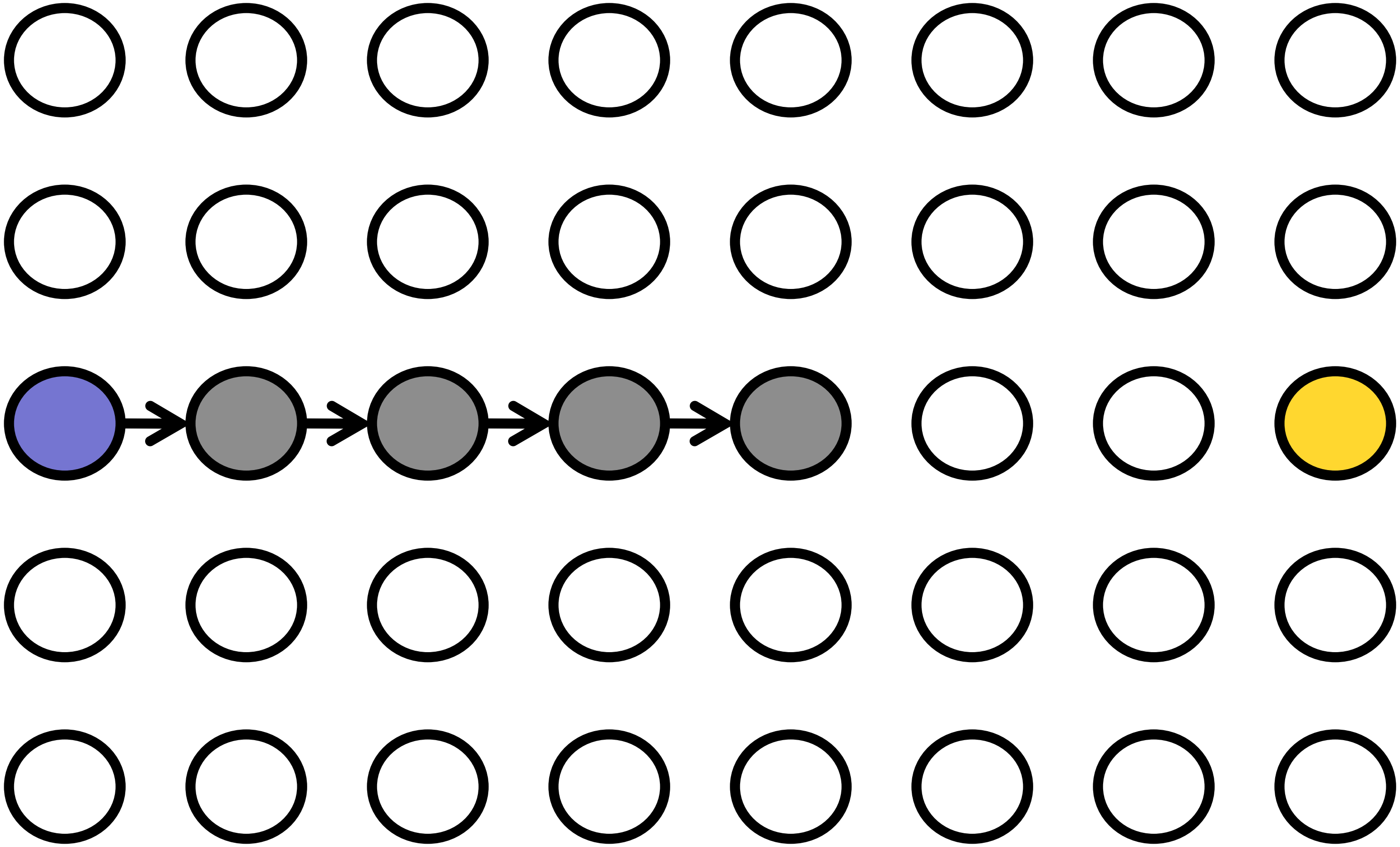
Example Search

Best First Search



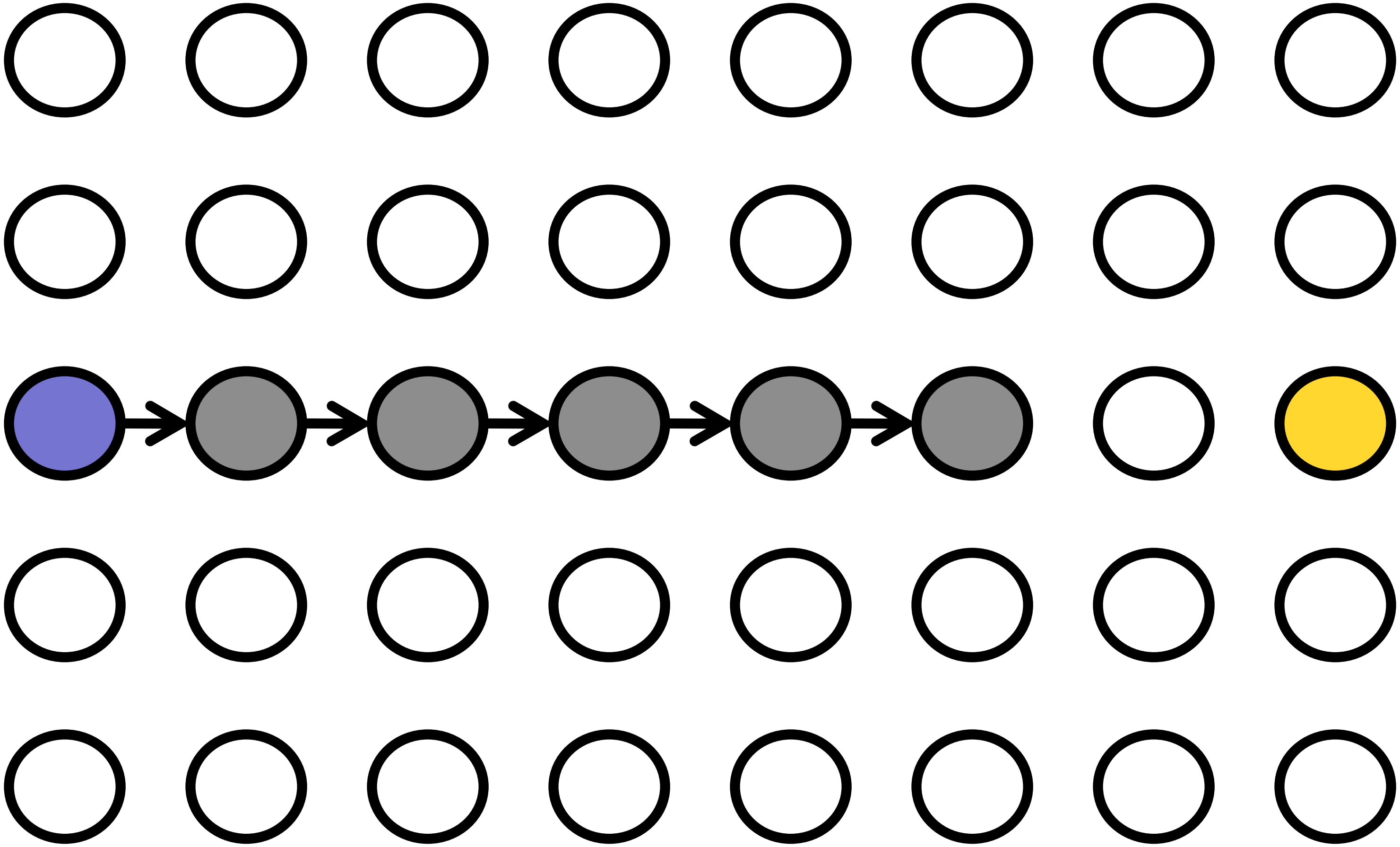
Example Search

Best First Search



Example Search

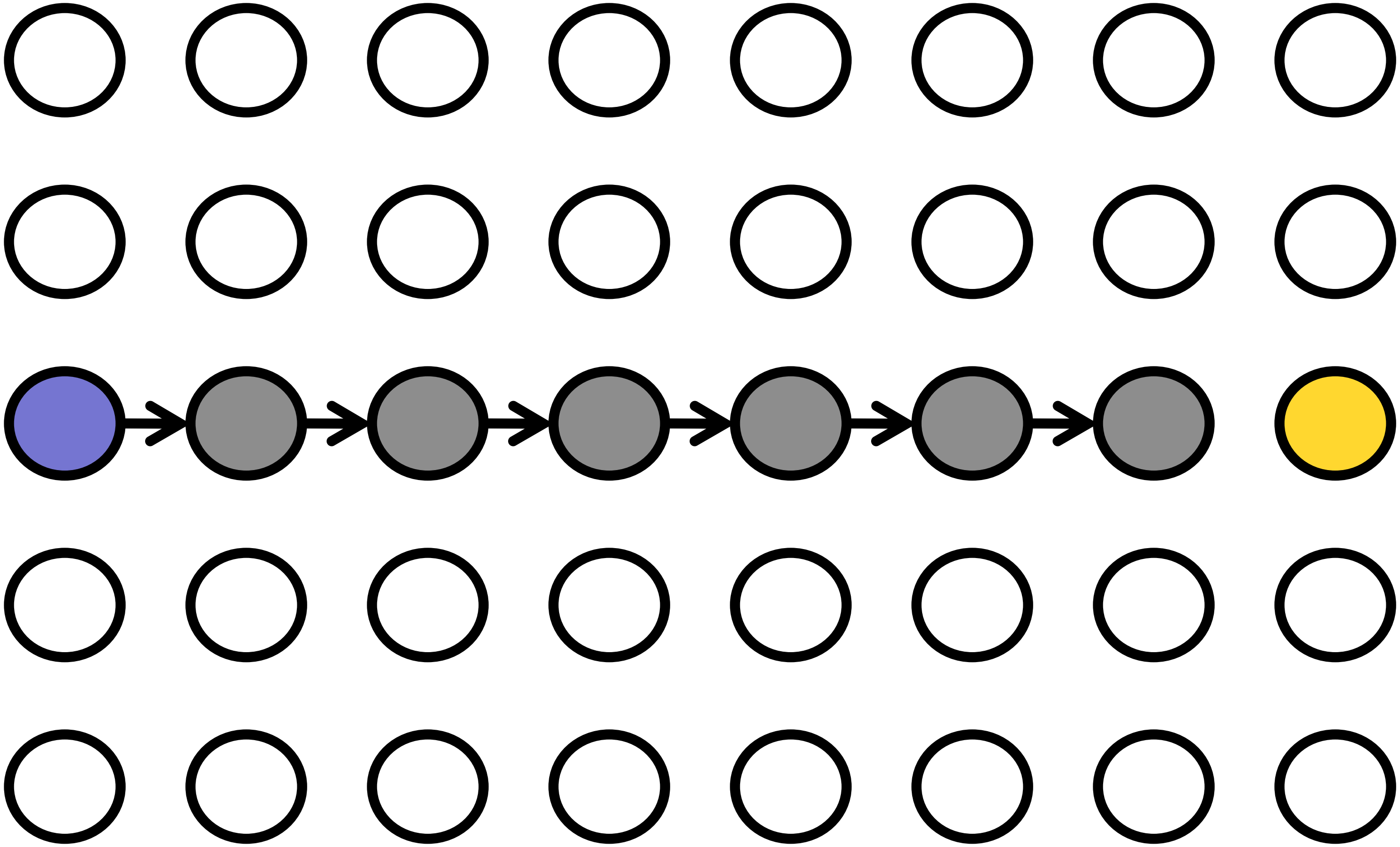
Best First Search



Charniak Johnson Search Space Pruning

Example Search

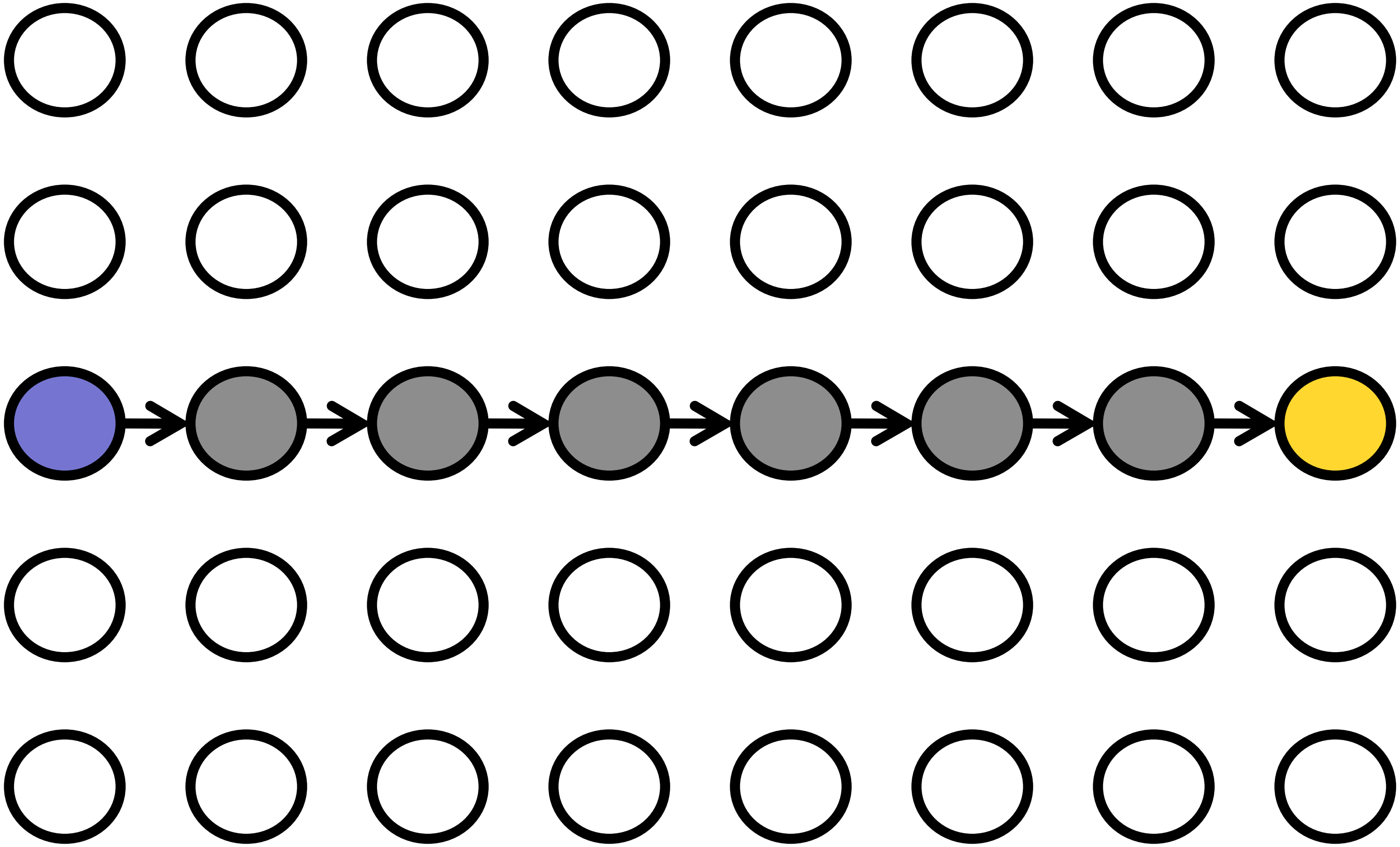
Best First Search



Charniak Johnson Search Space Pruning

Example Search

First Complete Parse!



Example Search

After the First Complete Parse

Count edges expanded so far

Then Expand

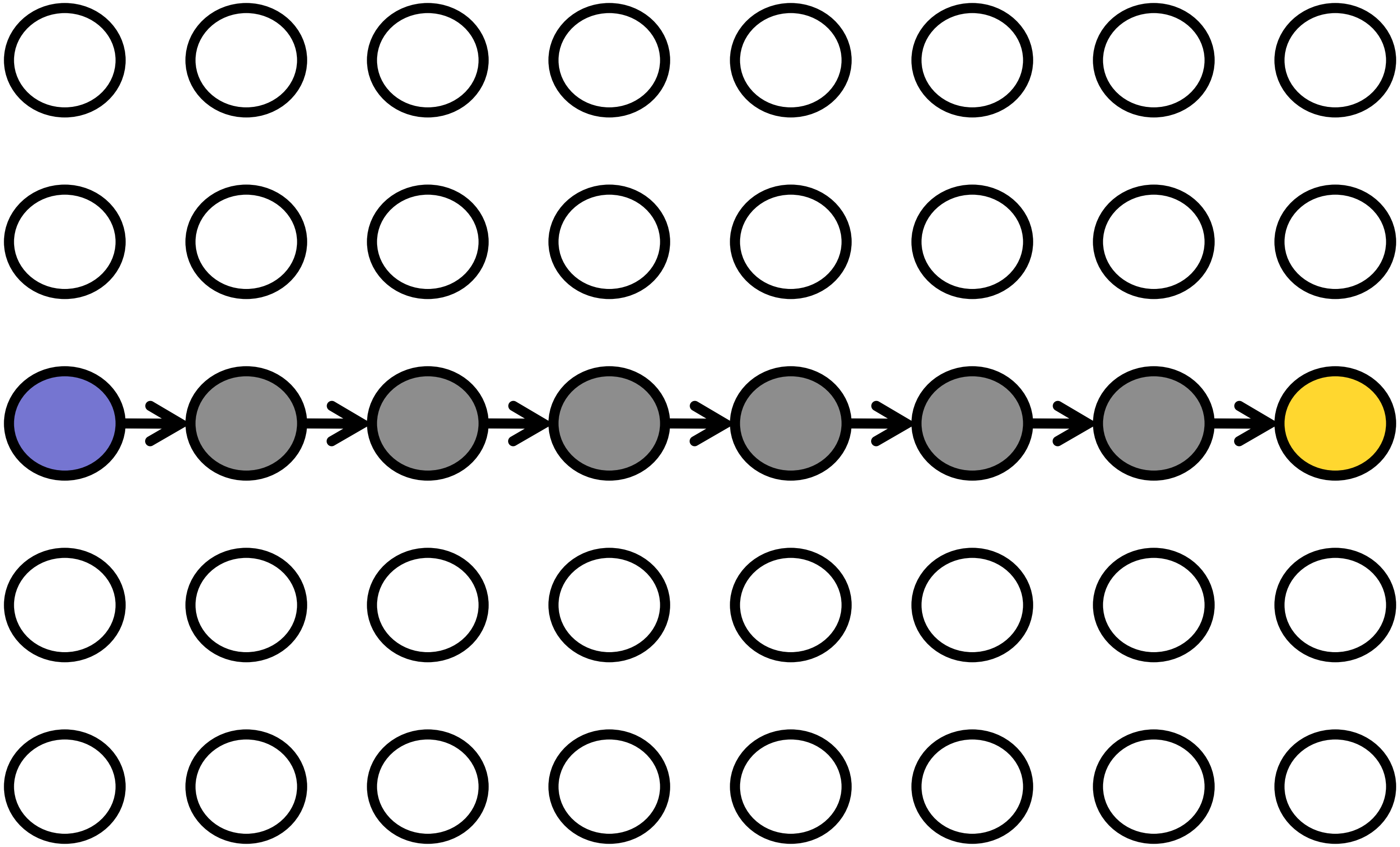
Edge count x Pruning constant more edges

Pruning constant = T parameter /10

Charniak Johnson Search Space Pruning

Example Search

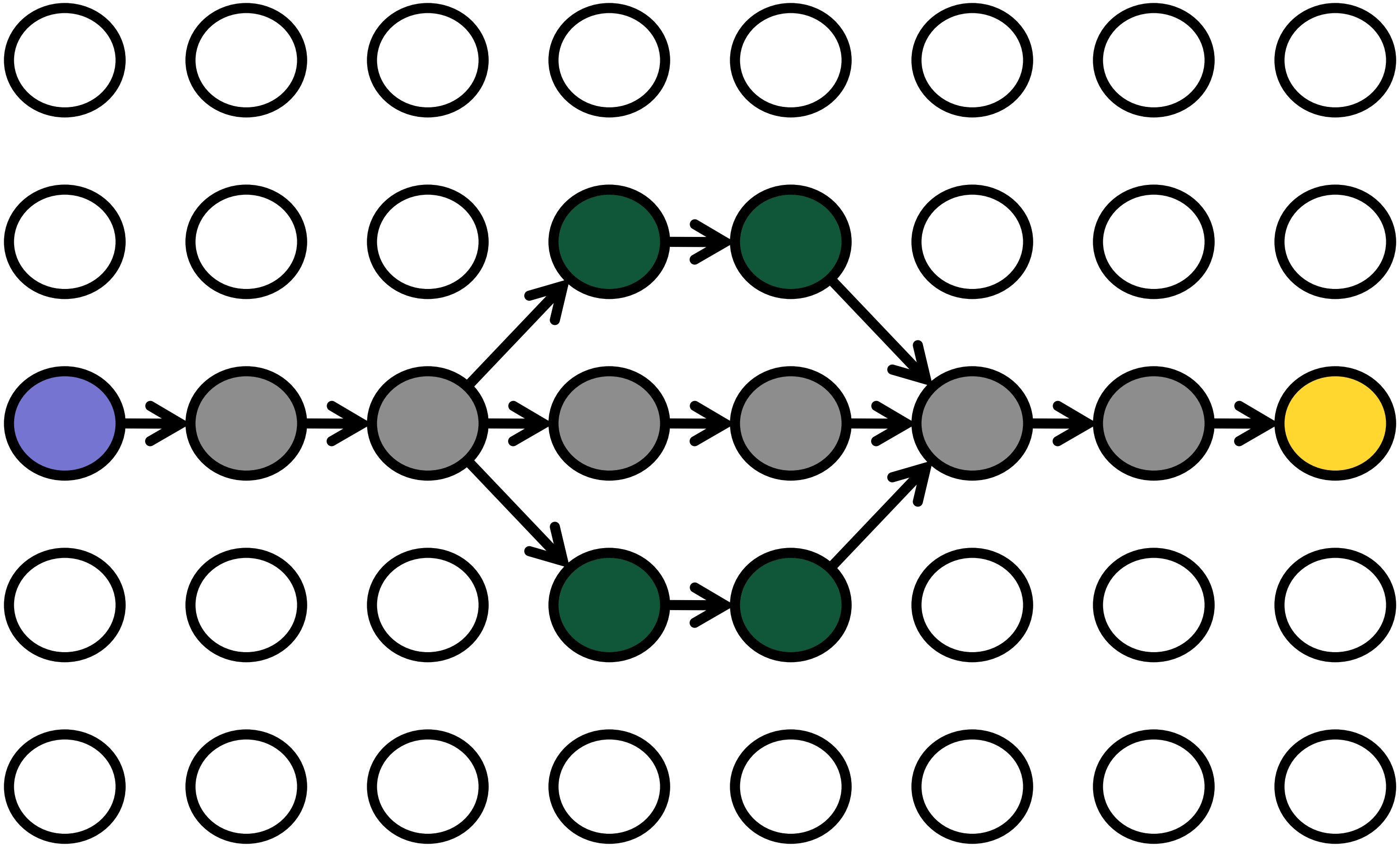
Expand edge count x Constant more edges



Charniak Johnson Search Space Pruning

Example Search

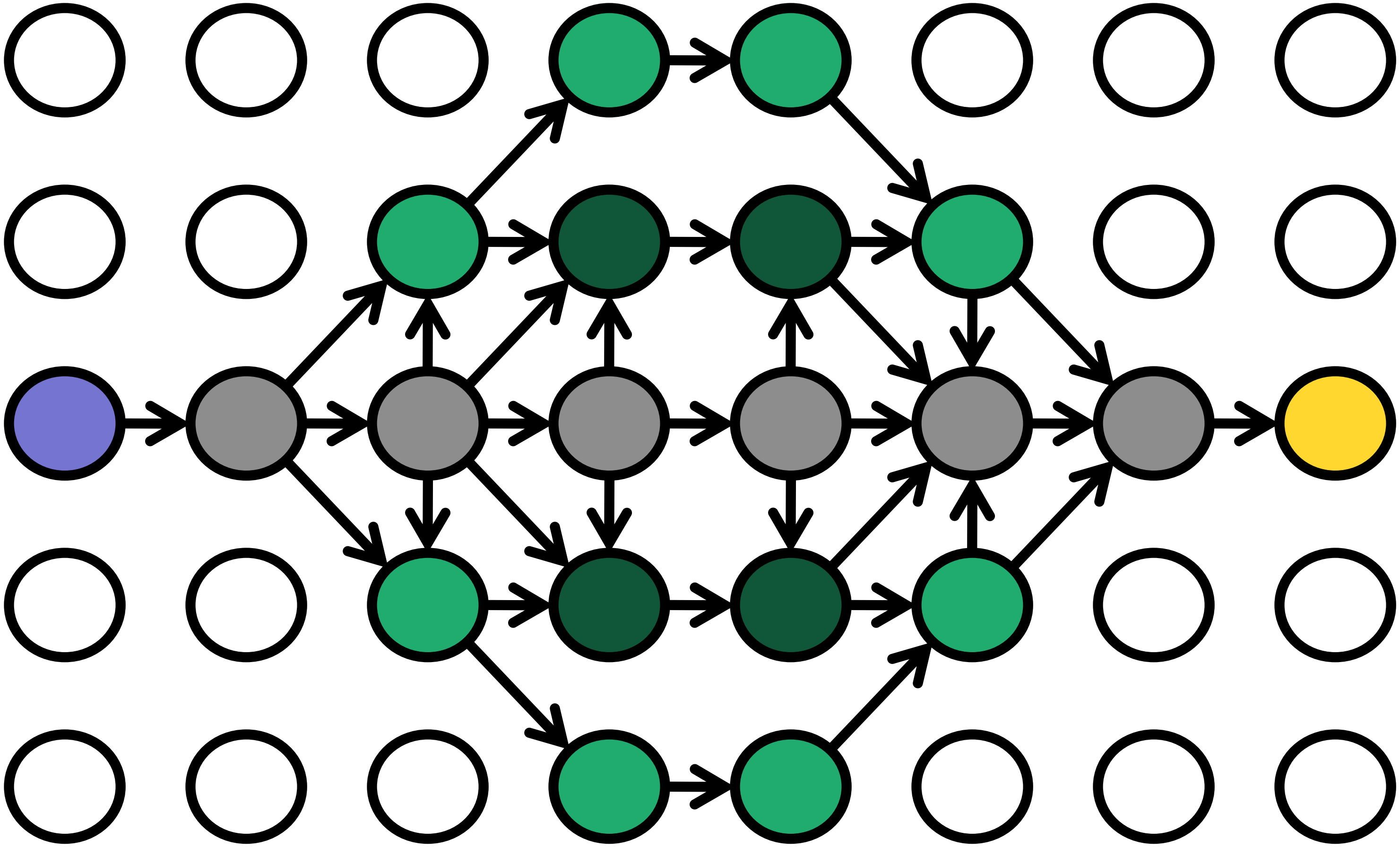
Expand edge count x Constant more edges



Charniak Johnson Search Space Pruning

Example Search

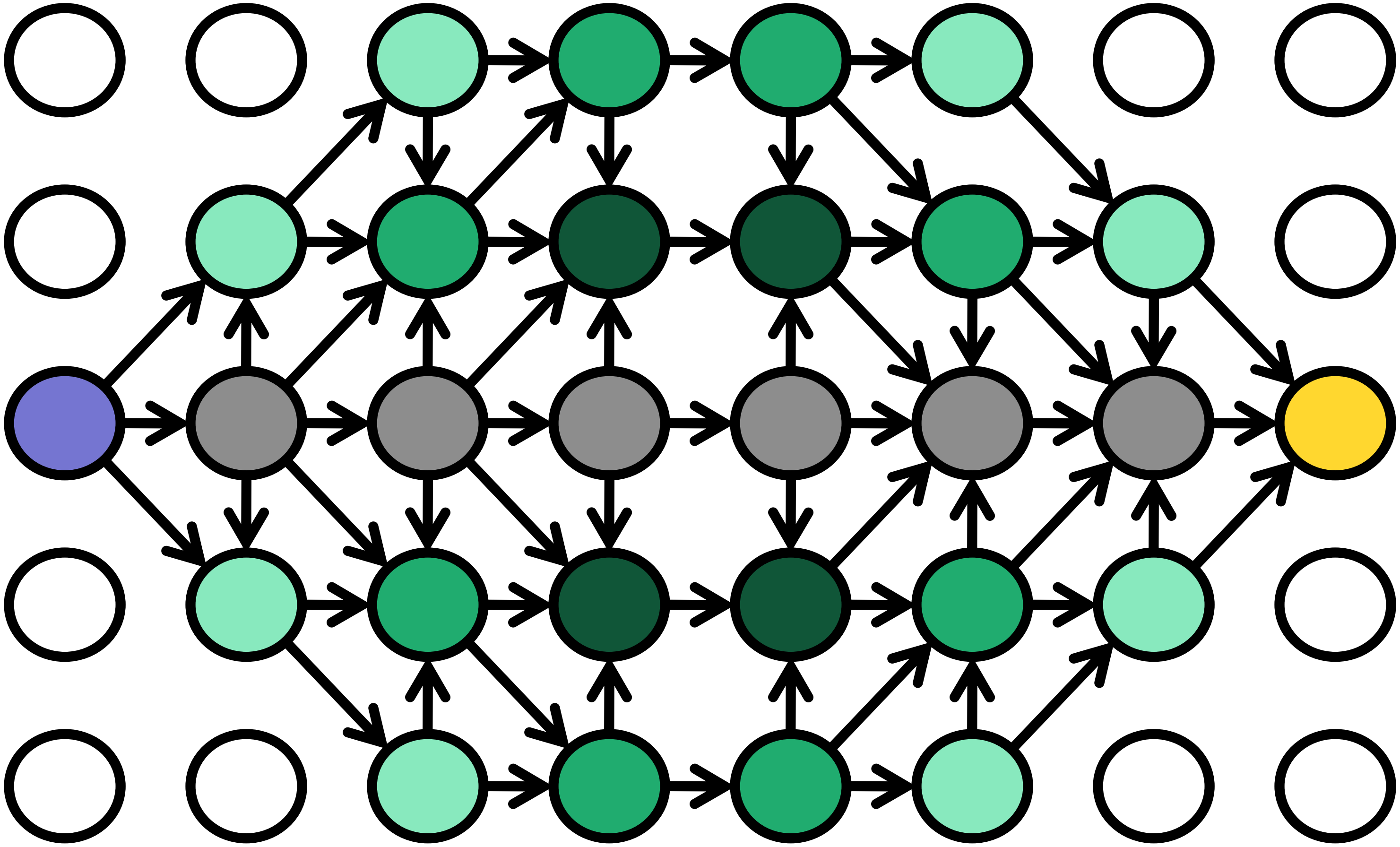
Expand edge count x Constant more edges



Charniak Johnson Search Space Pruning

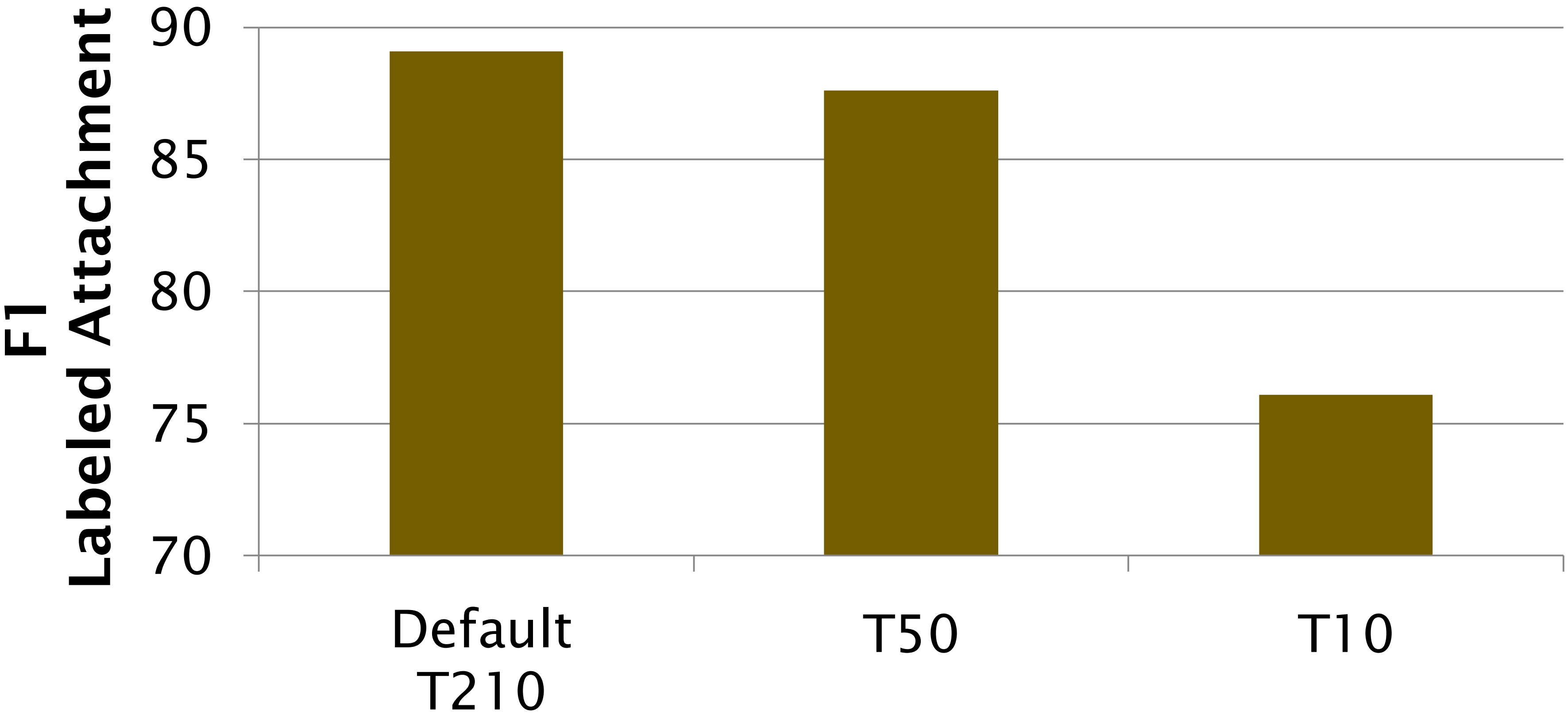
Example Search

Expand edge count x Constant more edges



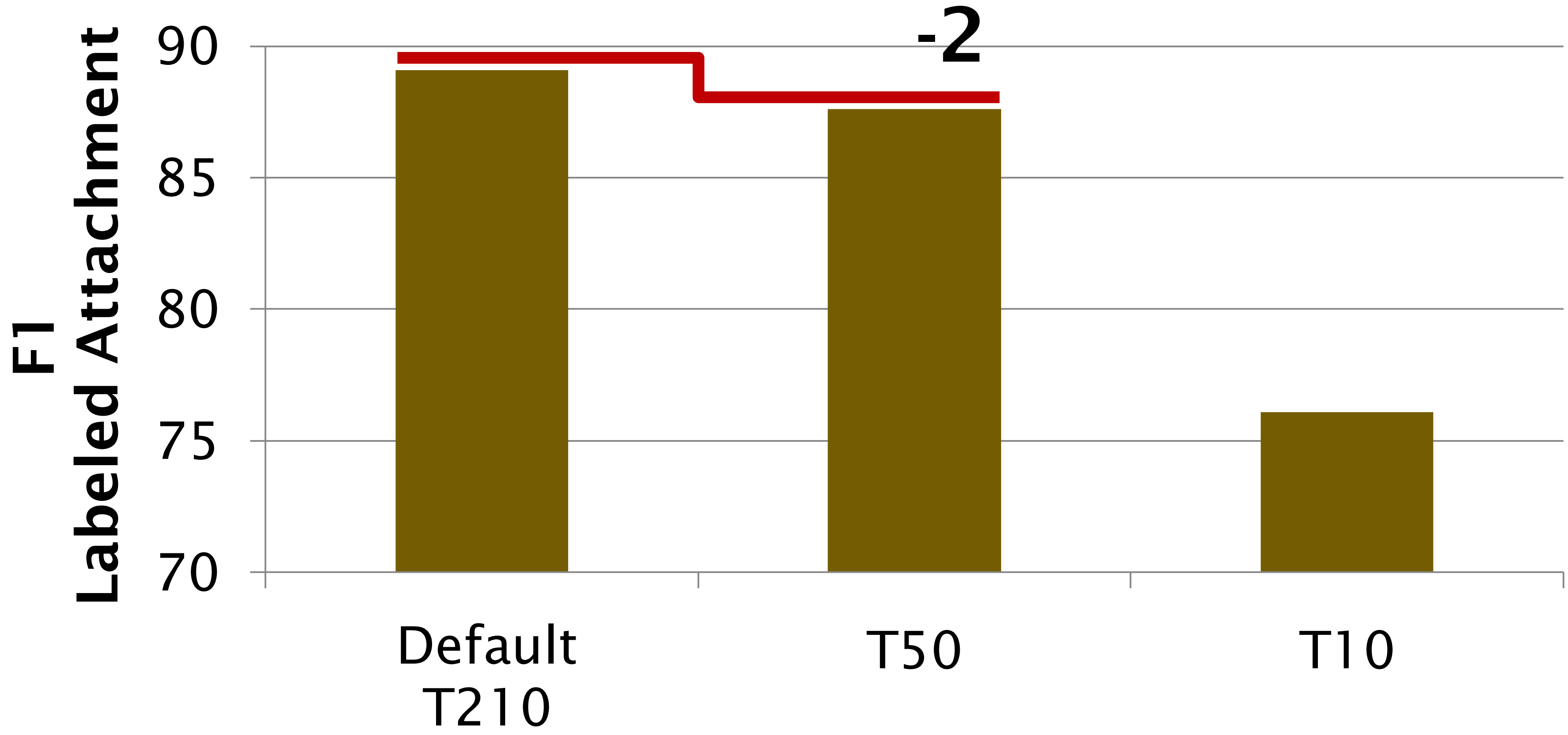
Charniak Johnson Search Space Pruning

Pruning Effects on **Accuracy**



Minimal loss of accuracy for moderate pruning

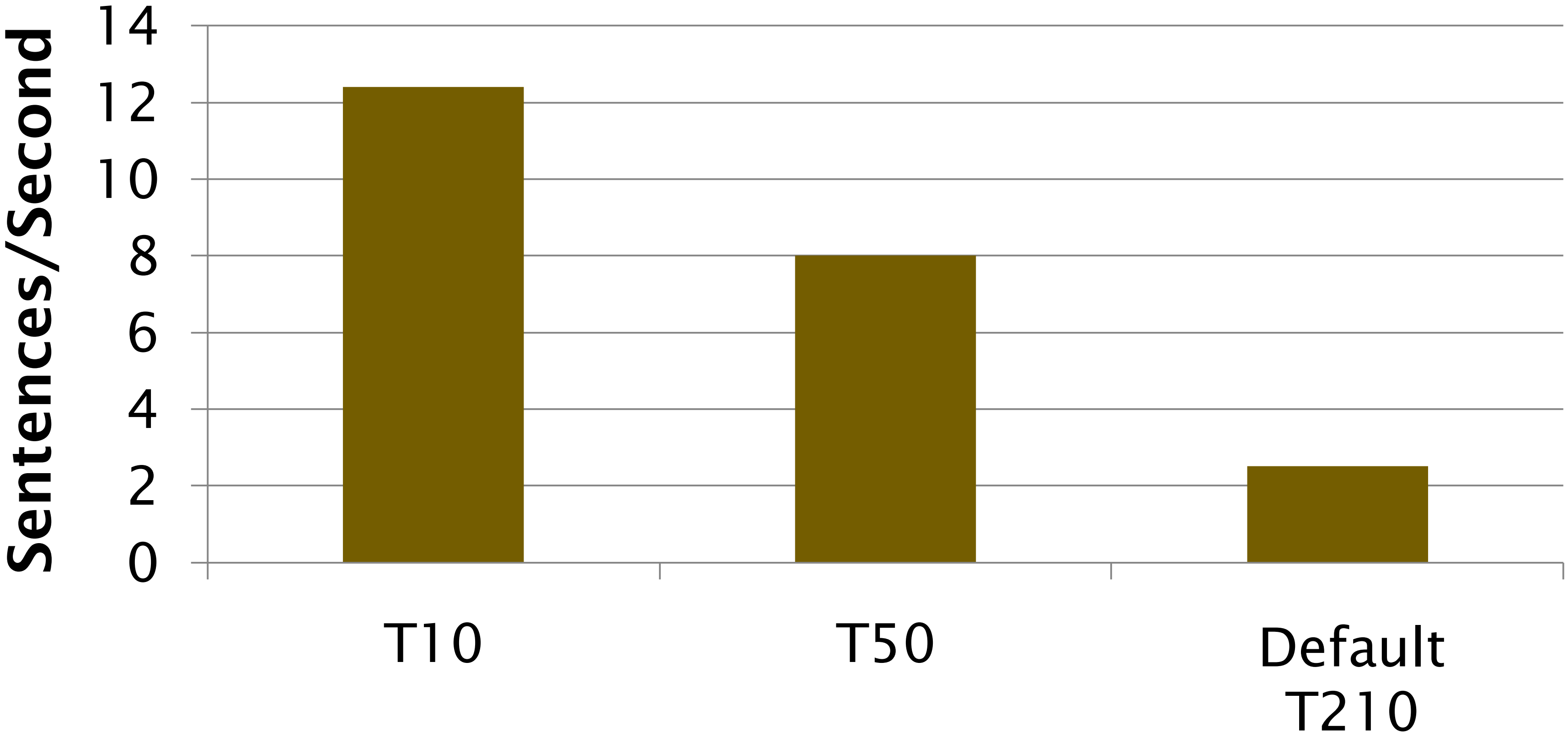
Pruning Effects on **Accuracy**



Minimal loss of accuracy for moderate pruning

Charniak Johnson Search Space Pruning

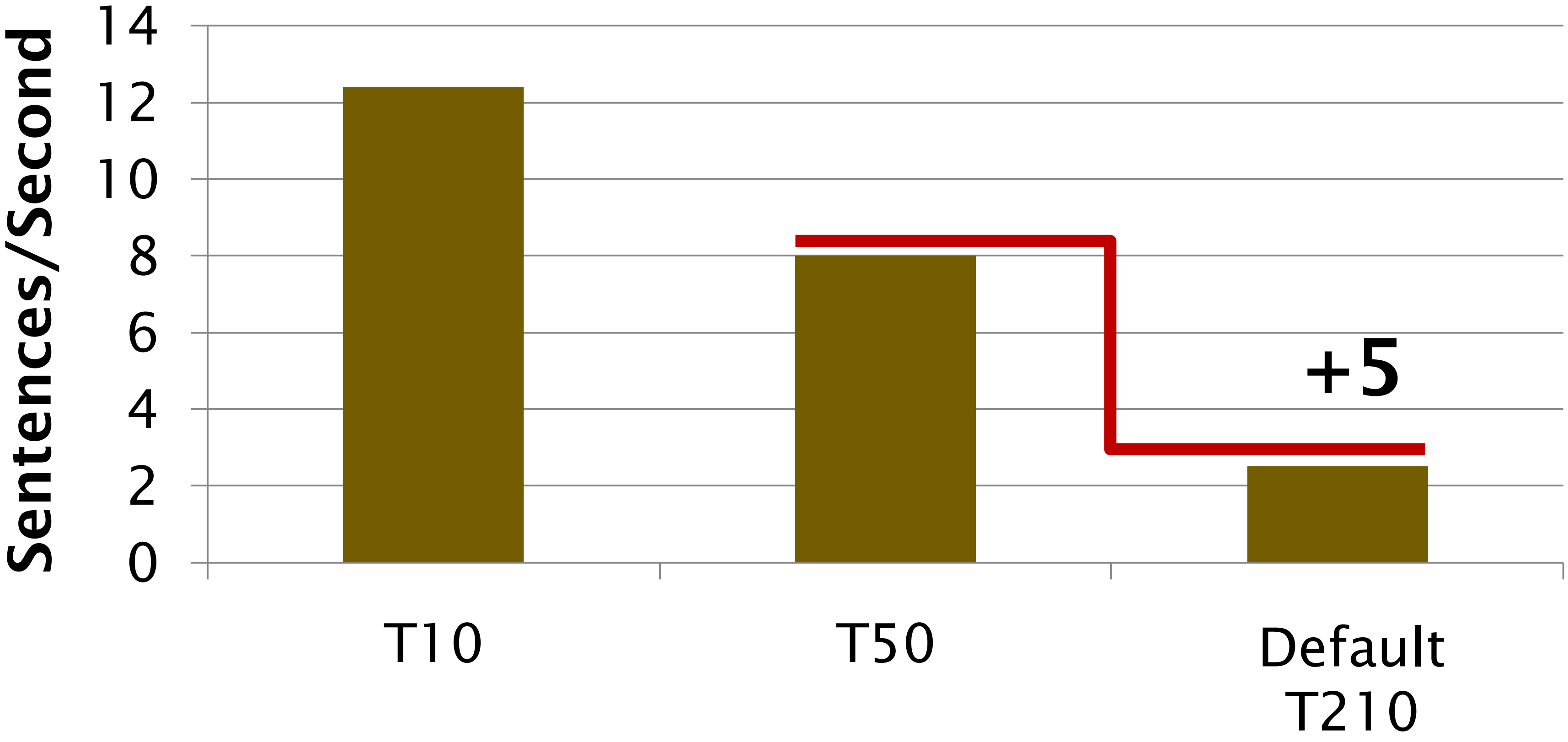
Pruning Effects on **Speed**



3x speed gains with moderate pruning

Charniak Johnson Search Space Pruning

Pruning Effects on **Speed**



3x speed gains with moderate pruning

Speed and Accuracy Trade-Offs

Best vs. Best **Accuracy**

Best Phrase Structure Parser

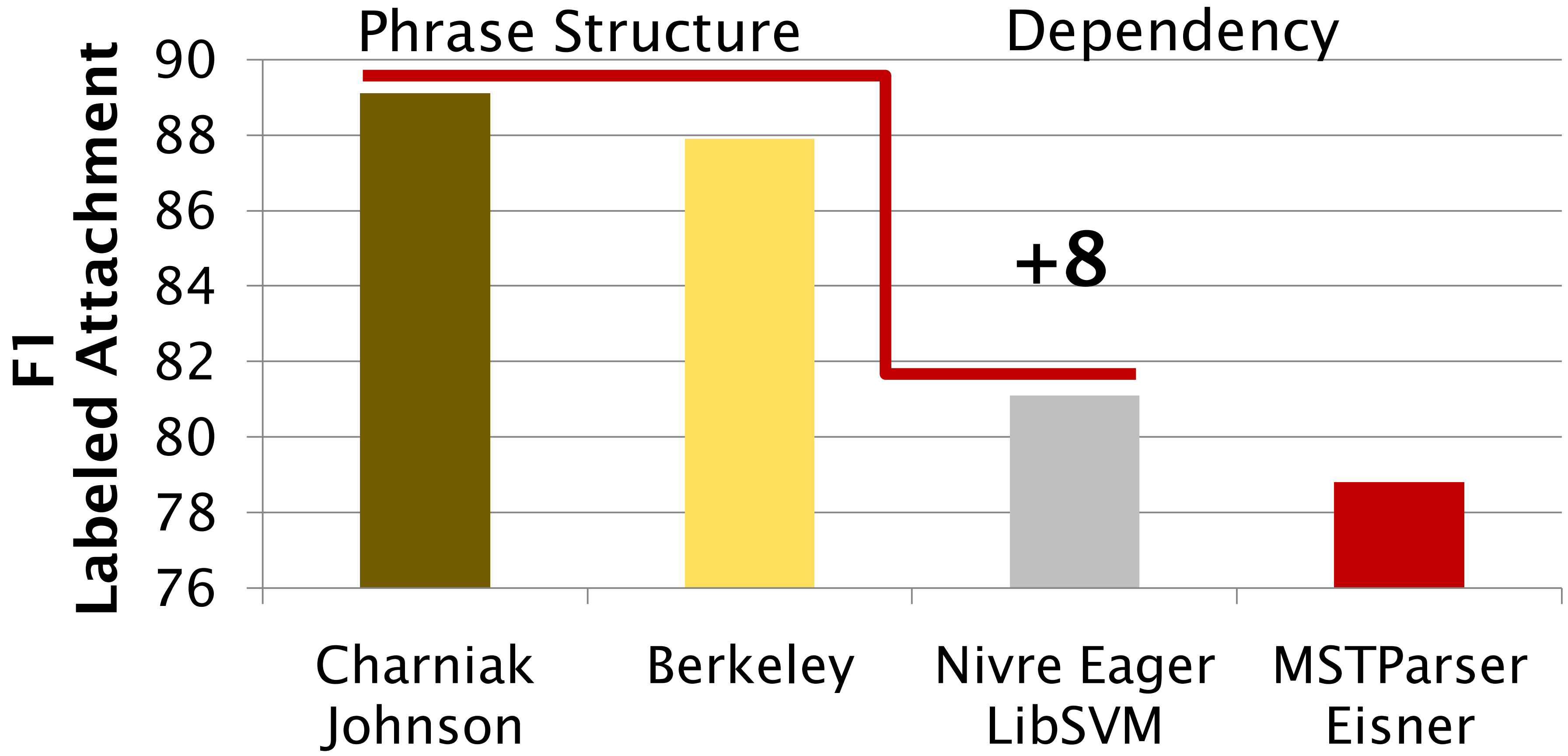
vs.

Best Dependency Parser

with Pruning

Pruning Speed and Accuracy Trade-Offs

Best vs. Best **Accuracy**

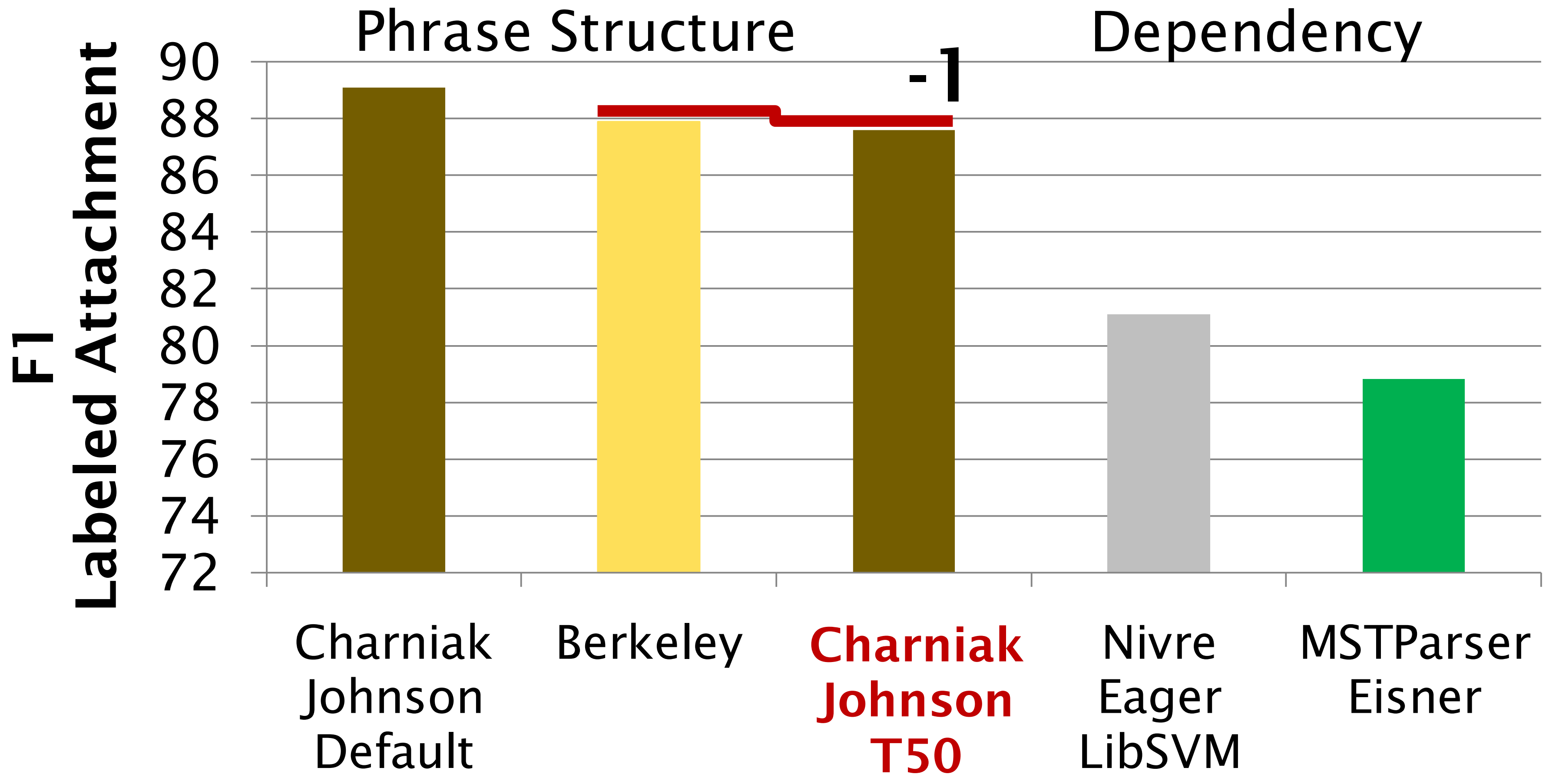


Remember this?

Let’s insert Charniak Johnson T50

Pruning Speed and Accuracy Trade-Offs

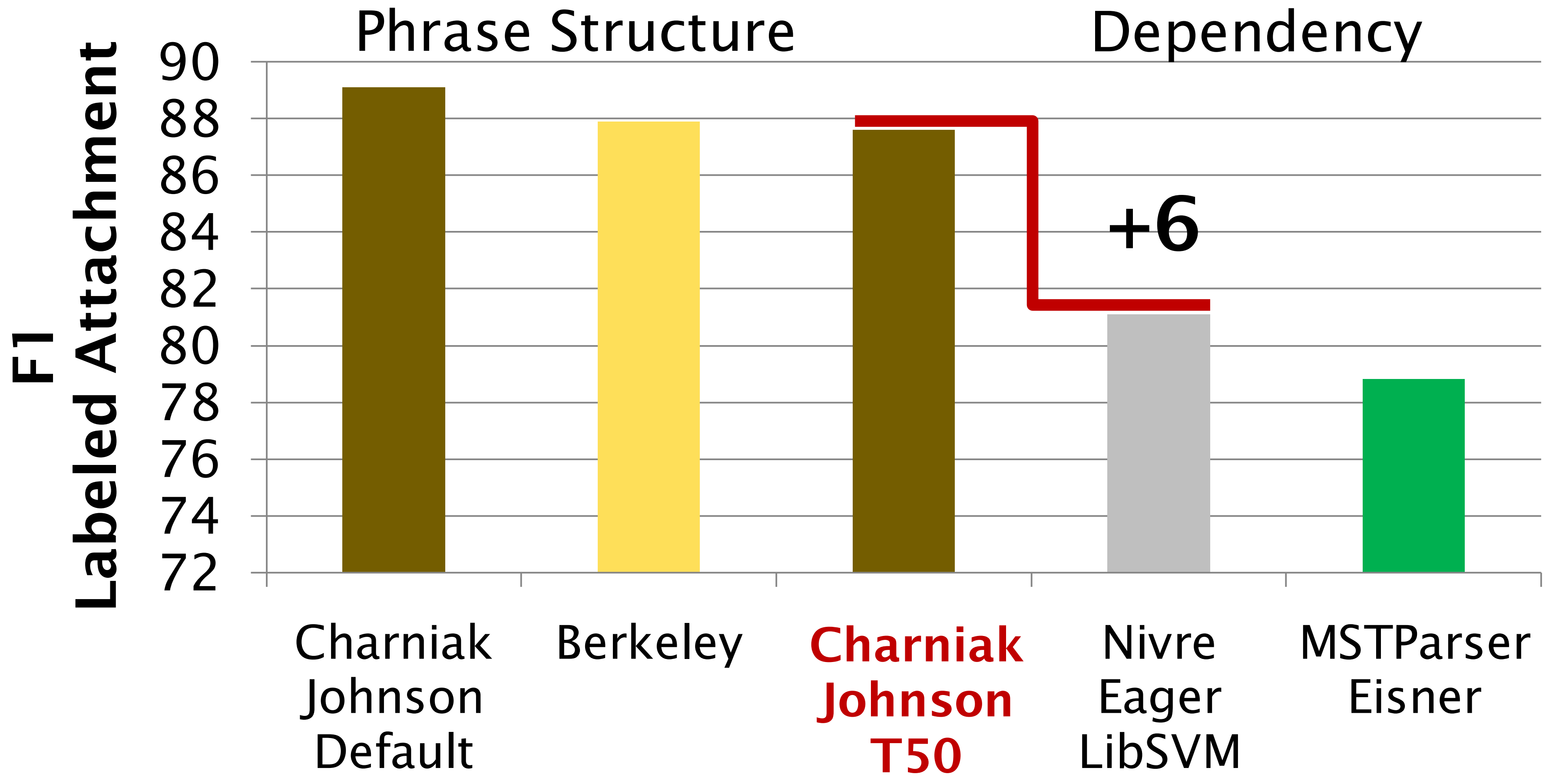
Best vs. Best **Accuracy**



About as good as Berkeley

Pruning Speed and Accuracy Trade-Offs

Best vs. Best **Accuracy**



Much better than best dependency parser

Speed and Accuracy Trade-Offs

Best vs. Best **Speed**

Best Dependency Parser

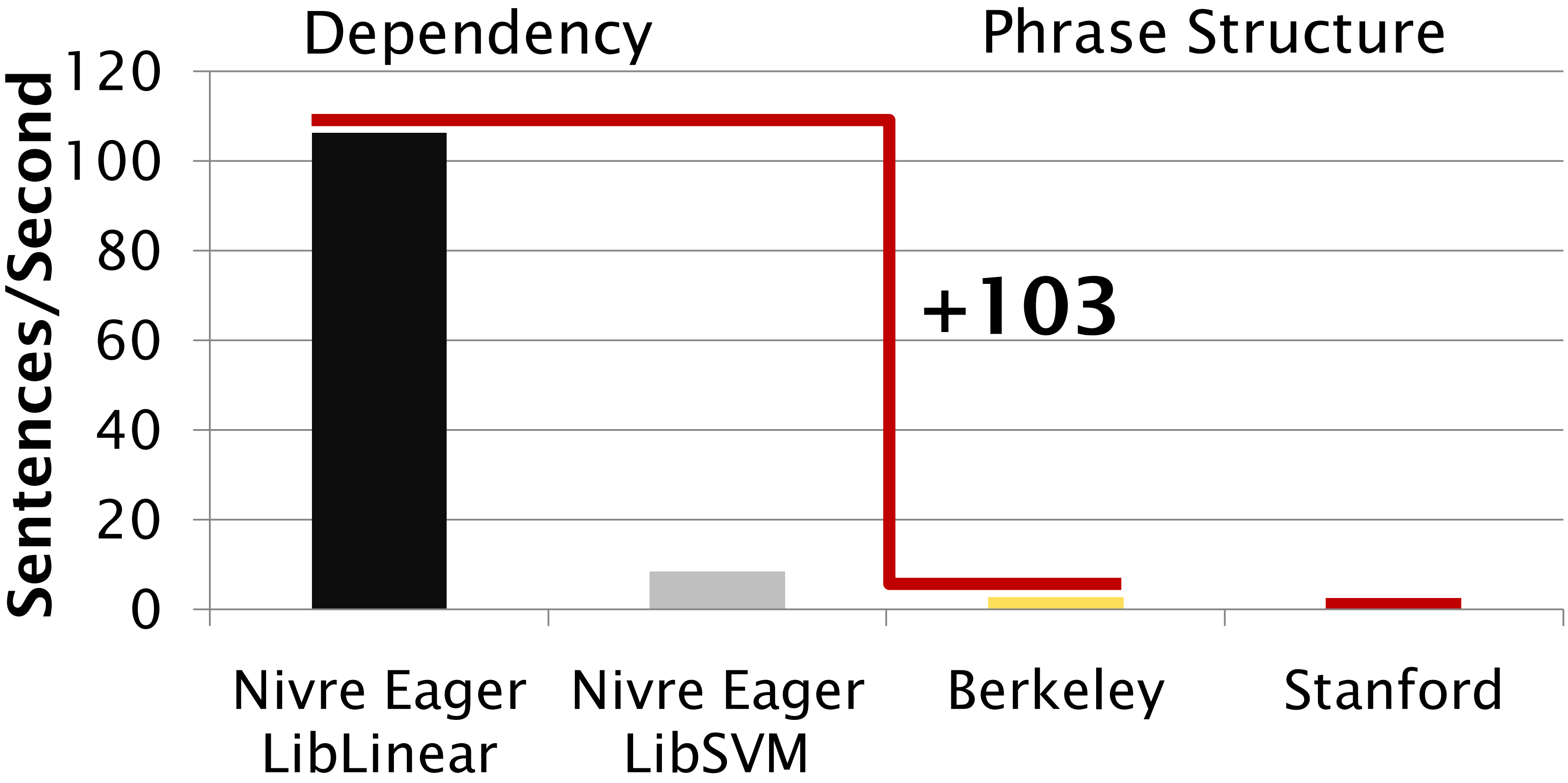
vs.

Best Phrase Structure Parser

with Pruning

Speed and Accuracy Trade-Offs

Best vs. Best **Speed**

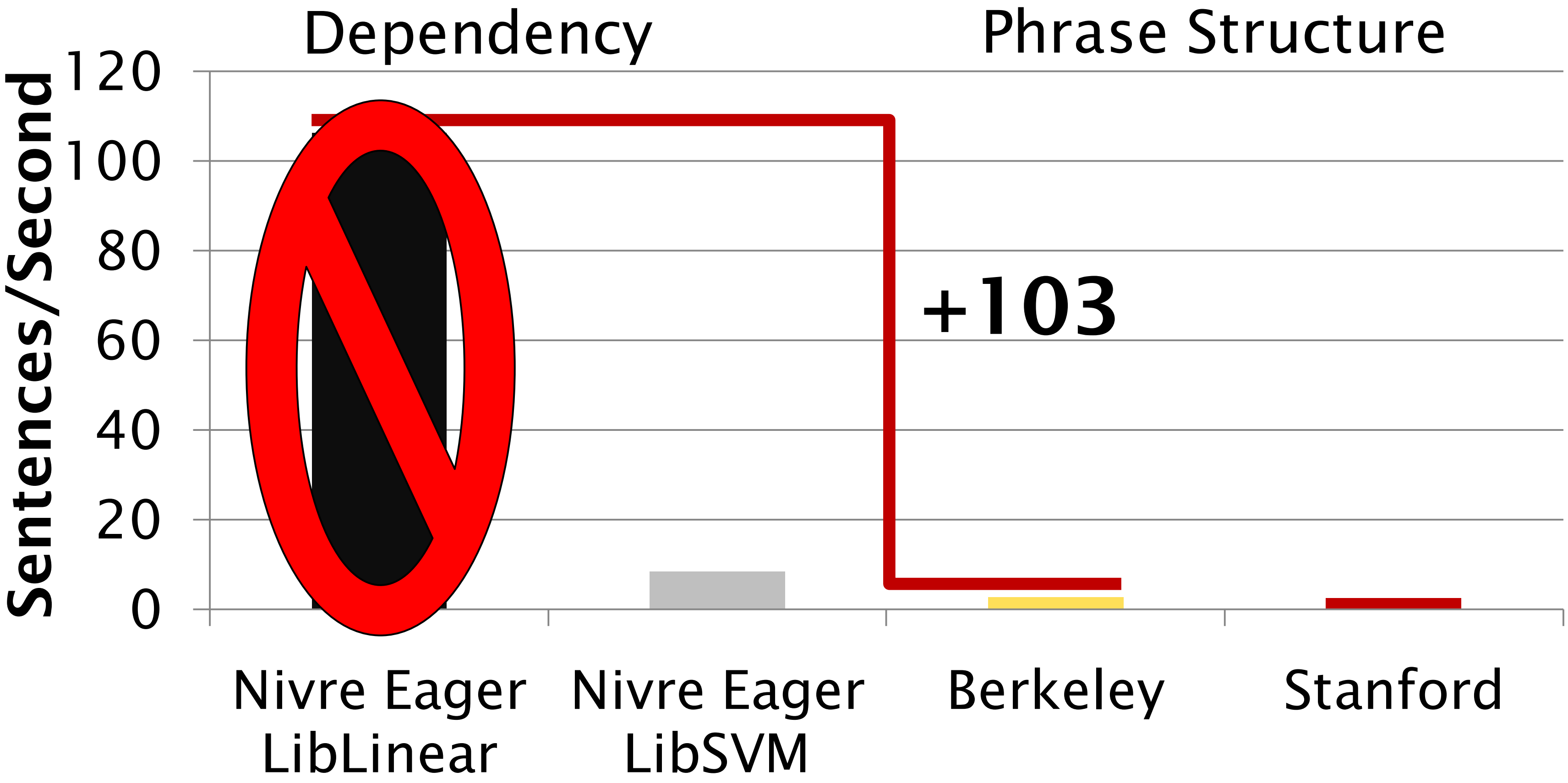


Remember this?

Let's insert Charniak Johnson T50

Speed and Accuracy Trade-Offs

Best vs. Best **Speed**

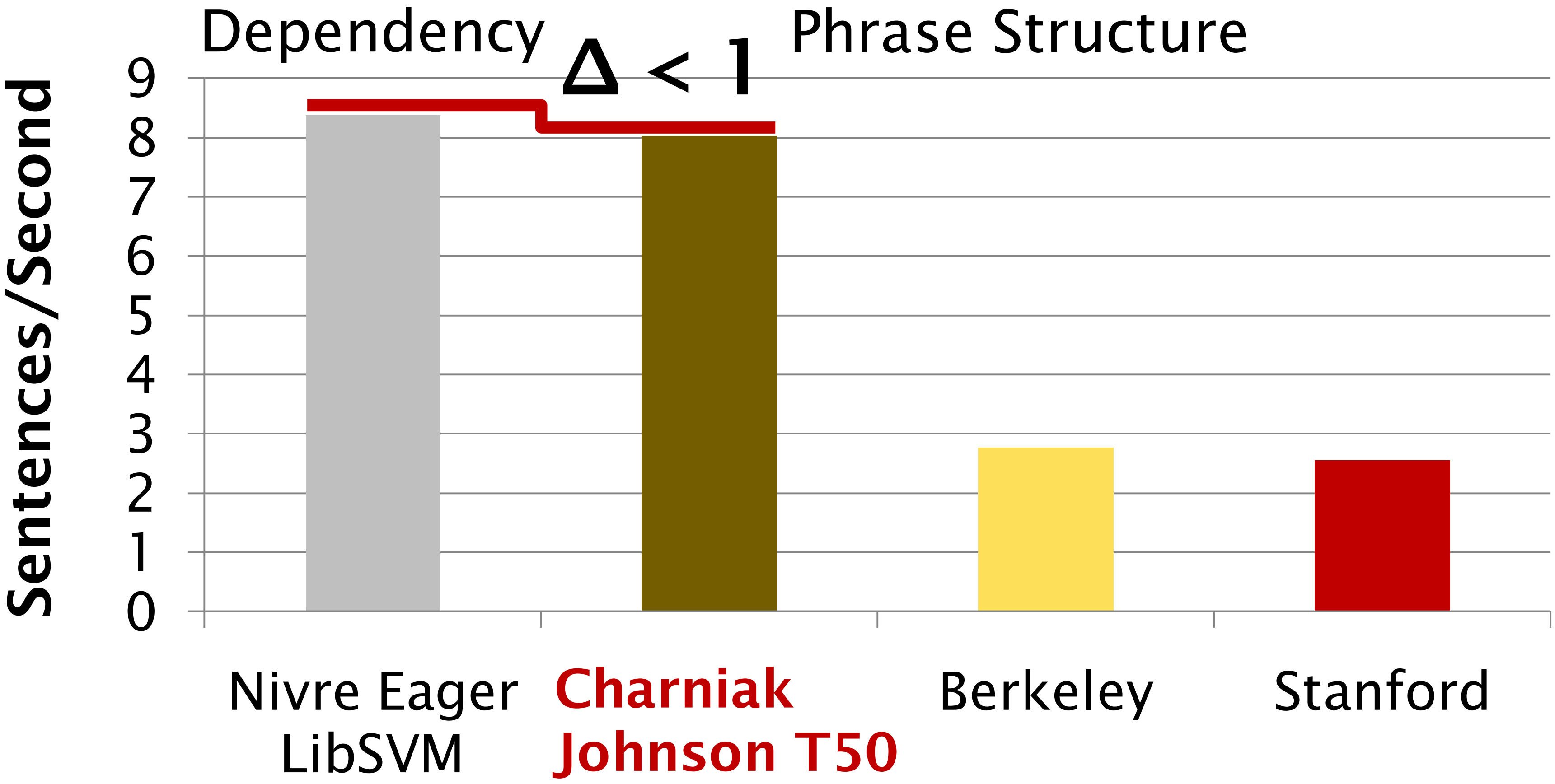


Let’s insert Charniak Johnson T50

…Excluding the 100+ sentences/second parser

Speed and Accuracy Trade-Offs

Best vs. Best **Speed**



Nearly as fast as Nivre Eager using LibSVM

Speed and Accuracy Trade-Offs

Pruning Summary

Charniak Johnson T50

Accuracy similar to
best phrase structure parsers

Speed similar to
most dependency parsers

Parsing to Stanford Dependencies

Conclusion

For Accuracy

Charniak Johnson

89.1 Labeled Attachment F1

For Speed

Nivre Eager with LibLinear

+100 Sentences/second

For A Good Balance

Charniak Johnson T50

87.6 Labeled Attachment F1

Speed similar to most dependency parsers

Thanks to
Joakim Nivre
Mihai Surdeanu