# Learning Dependency-Based Compositional Semantics


## Percy Liang, Michael I. Jordan and Dan Klein



**Human Language Technology (HLT) 2011**



Krunal Shah (shahkr@seas.upenn.edu)


8th January 2021

Penn Engineering

# Question Answering

Which is the largest city in California?

Which is the largest city in a state bordering California?

Knowledge Base

| City | Population |
|---|---|
| LA | 3.9m |
| SF | 800k |
| … | |

| Border |
|---|
| (CA, OR) |
| (CA, NV) |
| … |

| State |
|---|
| (LA, CA) |
| (SF, CA) |
| … |

**Simple enough? ☹**

for Google using webpages! ☺

# Semantic Parsing

Which is the largest city in the states bordering California?

**Answer:**

**Logic Program for given KB:**

Phoenix

argmax({*c*: city(*c*) ^ *s*.state(*s*) ^ loc(*c*, *s*) ^ border(*s*, CA)}, population)

Simple annotation ☺

Expert annotation ☹

Executable for any computer!

Knowledge Base

| City | Population |
|---|---|
| LA | 3.9m |
| SF | 800k |
| … | |

| Border |
|---|
| (CA, OR) |
| (CA, NV) |
| … |

| State |
|---|
| (LA, CA) |
| (SF, CA) |
| … |



* example from Liang et al. 2011

# E2E Semantic Parsing

Which is the largest city in the states bordering California?

Latent logic program:

$\texttt{argmax}(\{c: \texttt{city}(c) \wedge s.\texttt{state}(s) \wedge \texttt{loc}(c, s) \wedge \texttt{border}(s, \texttt{CA})\}, \texttt{population})$

$\texttt{argmax}(\{c: \texttt{city}(c) \wedge s.\texttt{state}(s) \wedge \texttt{border}(s, \texttt{CA})\}, \texttt{population})$

$\texttt{argmax}(\{c: \texttt{city}(c) \wedge s.\texttt{state}(s) \wedge \texttt{loc}(c, \texttt{CA})\}, \texttt{population})$

…

Answer:

Phoenix

From the huge space, generate program
- Executes to correct answer
- Syntactically correct (type constraints)

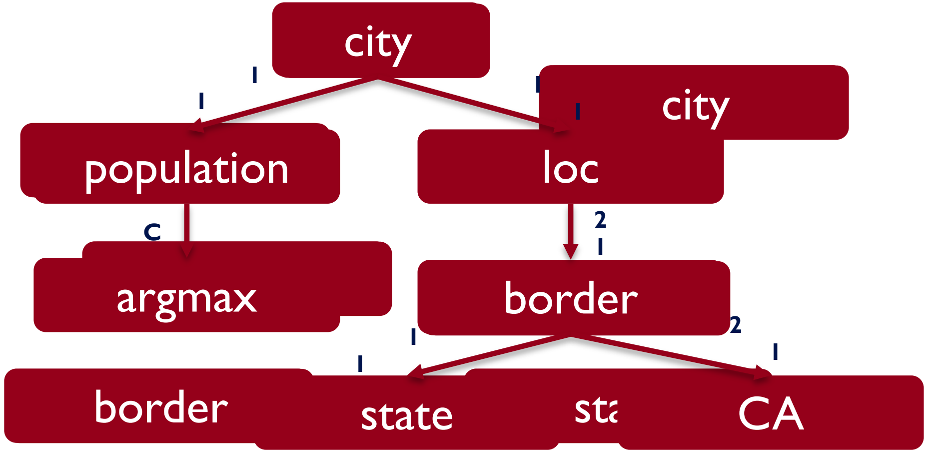
And use it to supervise

# Key idea

- Formulate a logic form with tree representation called **D**ependency based **C**ompositional **S**emantics (DCS) such that
- The DCS representation looks like syntactic dependency tree, to facilitate learning

Question: Which is the largest city in a state bordering California?

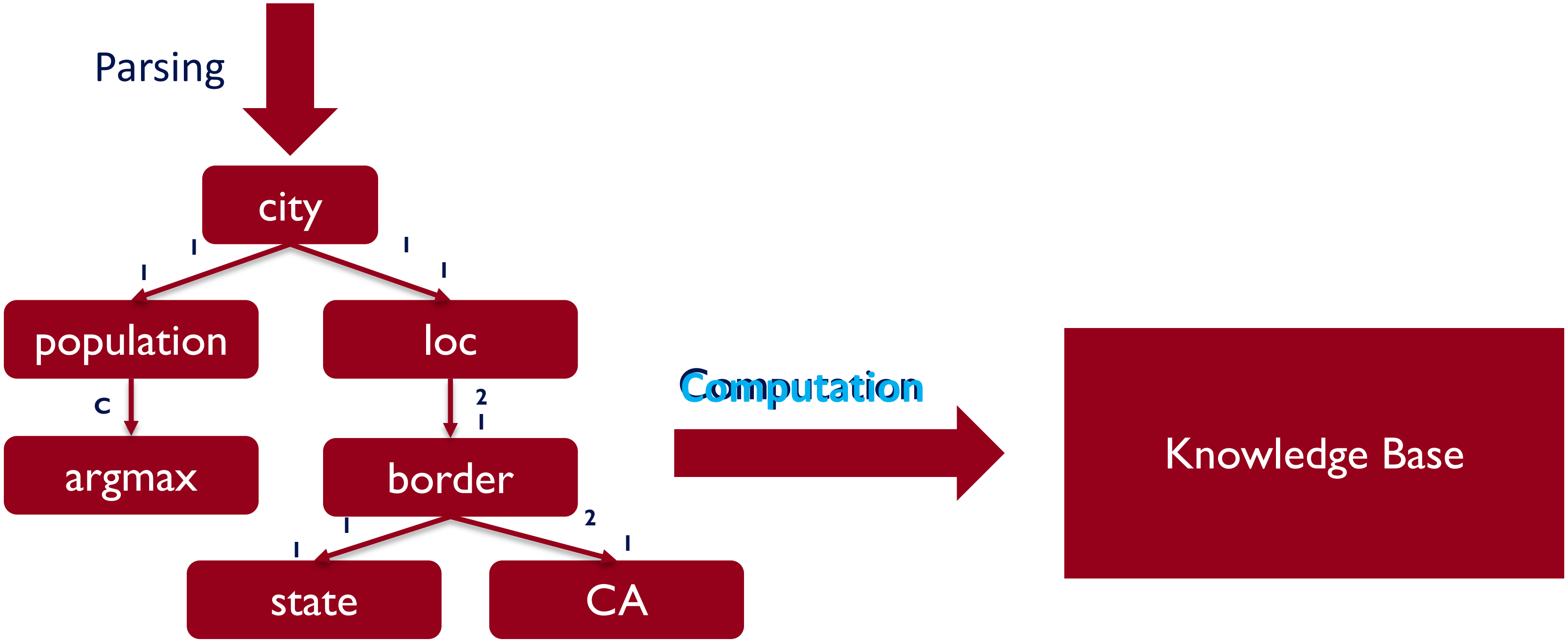
argmax({c : city(c) ∧ ∃s.state(s) ∧ loc(c, s) ∧ border(s, CA)}, population)




* example from Liang et al. 2011

# Overview

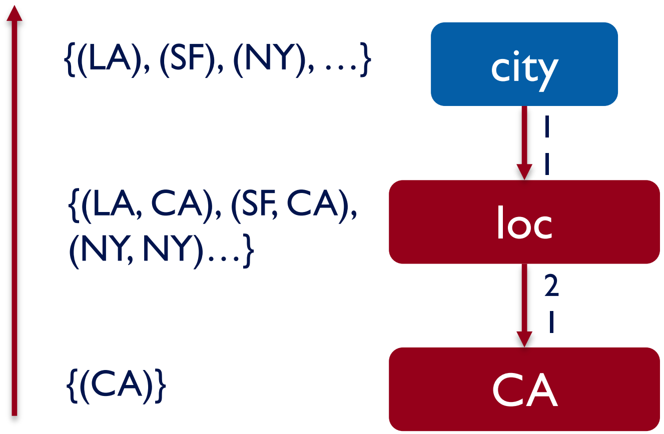
Question: Which is the largest city in a state bordering California?



# Basic DCS

- Supports join and aggregate operators
- Designed for cases with correlated semantic and syntactic scopes

every city in California




Constraints

$c \in \text{city}$

$c_1 = l_1$

$l \in \text{loc}$

$l_2 = s_1$

$s \in \text{CA}$

- A DCS tree encodes a Constraint satisfaction problem

* example from Liang et al. 2011

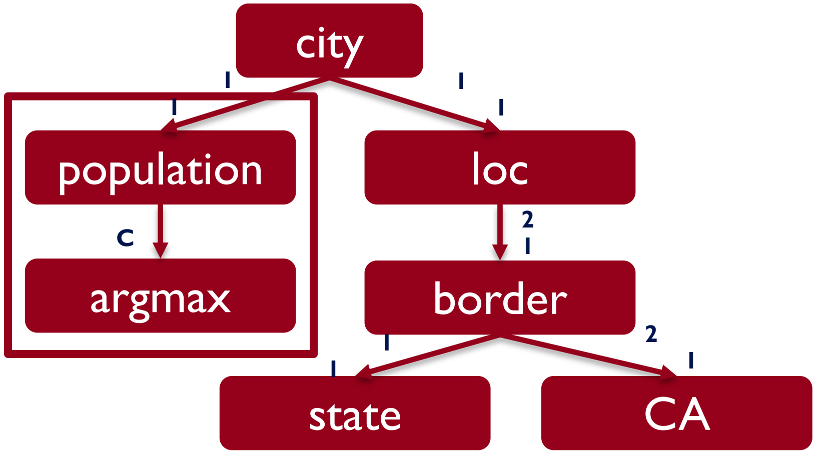
# Towards Full DCS

- Semantic and syntactic scope diverge in examples of quantification, extraction, comparison, etc.

Question: Which is the **largest** city in a state bordering California?

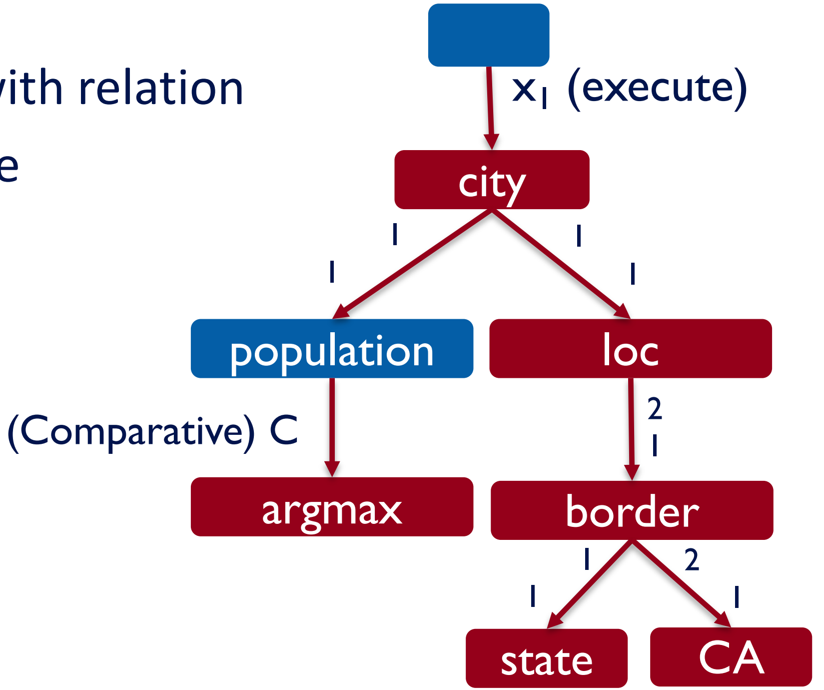
$$\mathtt{argmax}(\{c: \mathtt{city}(c) \wedge s.\mathtt{state}(s) \wedge \mathtt{loc}(c, s) \wedge \mathtt{border}(s, \mathtt{CA})\}, \mathtt{population})$$

Semantic root (last execution) but not the syntactic root (dependency tree)



# Full DCS

- Additionally supports extract, quantify and compare operators
- Mark-Execute:
  - mark at syntactic scope with relation
  - execute at semantic scope

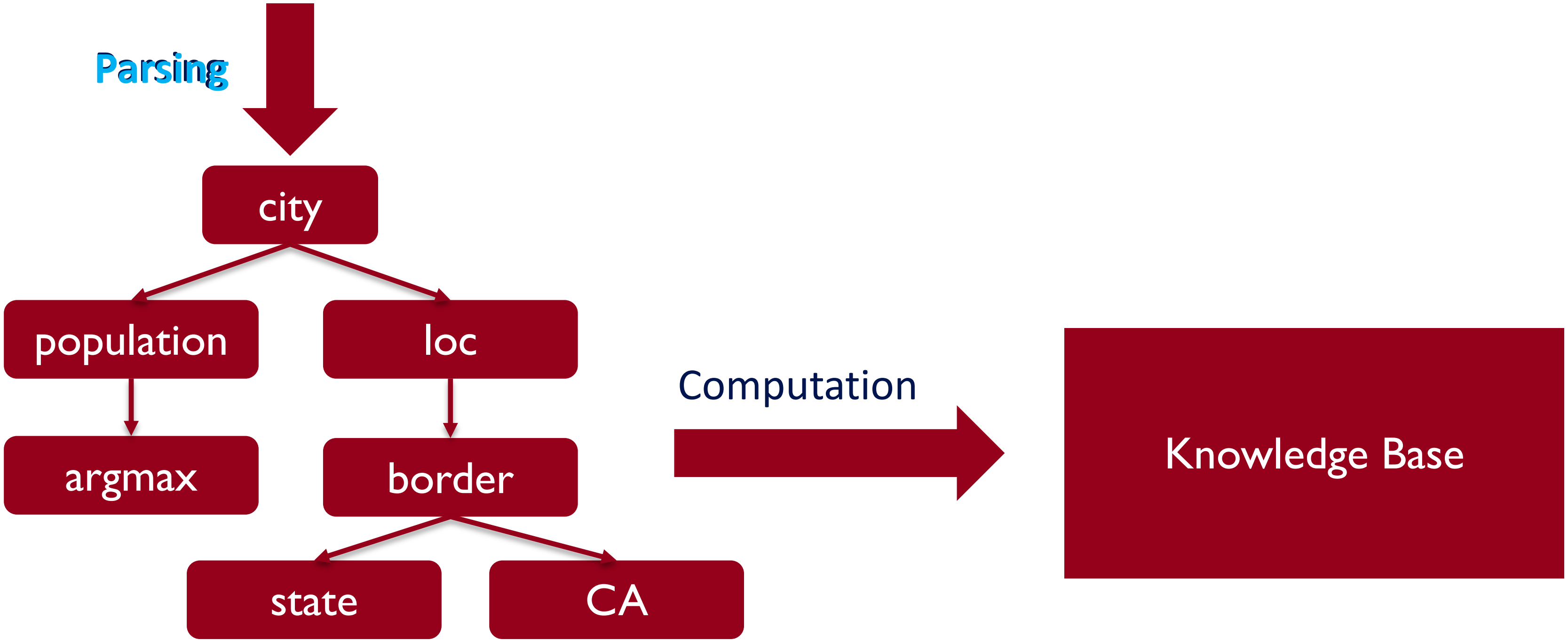




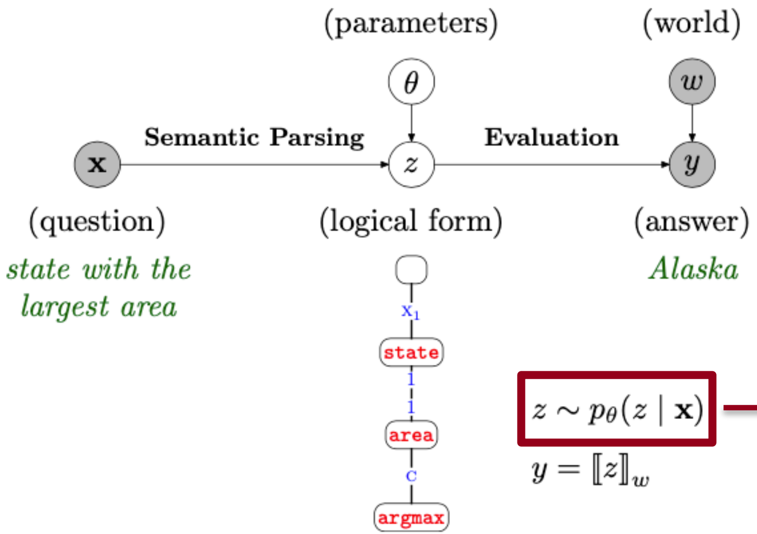

* could be a little inaccurate

# Overview

Question: Which is the largest city in a state bordering California?



# Graphical Model




Parse scoring model
- Generate candidate parses
- Score the parses

* image from Liang et al. 2011

# Generate parse space

- Use fixed set of lexical triggers
  - ("California", CA), ("most"

Recursively construct trees for sul

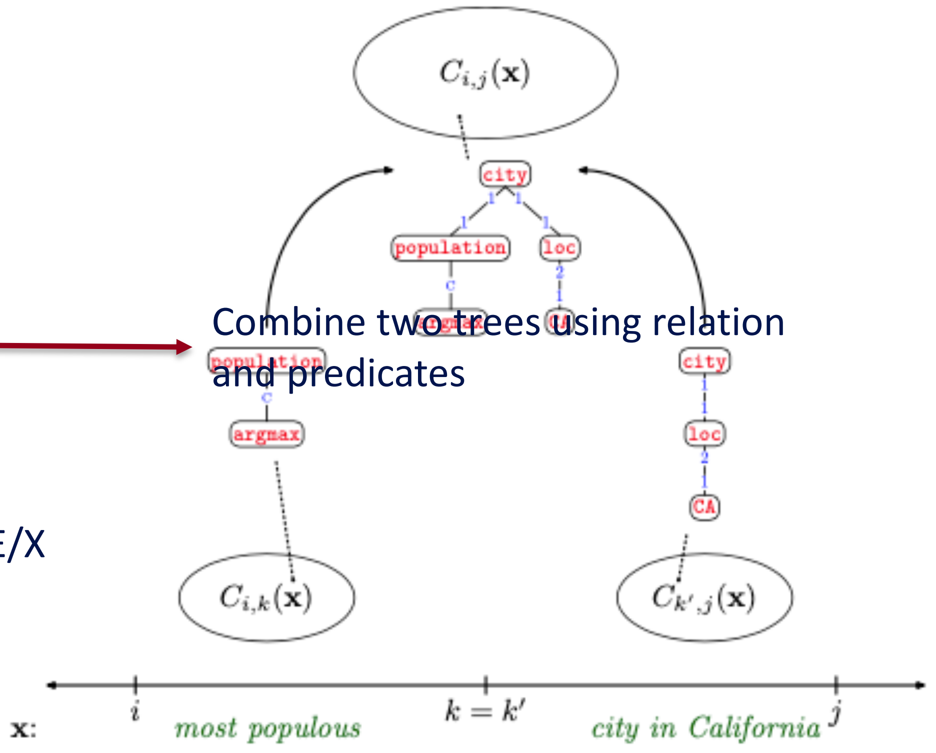
$$C_{i,j} = F\Big(A\Big(L(\mathbf{x}_{i+1..j}) \cup \bigcup_{\substack{i \le k \le k' < j \\ a \in C_{i,k} \\ b \in C_{k',j}}} T_1(a,b)\Big)\Big)$$

Lexical trigger tree

Augmentation function to add E/X relation on a single tree

Filtering function

Combine two trees using relation and predicates






* image from Liang et al. 2011

# Log linear scoring model

| | | | |
|---|---|---|---|
| $x$: | city | in | California |
| $z$: | `city` | `loc` | `CA` |

$$\texttt{city} \xrightarrow{1\quad 1} \texttt{loc} \xrightarrow{2\quad 1} \texttt{CA}$$

$$\mathit{features}(x, z) = (\texttt{in...loc}, \texttt{city-1-1-loc}, \ldots) \in \mathrm{R}^{d}$$

$$\mathit{score}(x, z) = \mathit{features}(x, z) \,.\, \boldsymbol{\theta}$$

$$p(z \mid x, \boldsymbol{\theta}) = e^{\mathit{score}(x, z)} / \boldsymbol{\Sigma}_{z' \in Z(x)} e^{\mathit{score}(x, z')}$$

# Learning

## Objective:

$$\max_{\boldsymbol{\theta}} \boldsymbol{\Sigma}_z \, p(y|z,w) \, . \, p(z|x, \boldsymbol{\theta})$$

## EM style learning:

$\boldsymbol{\theta}$ $\xrightarrow{\text{score trees}}$ *k-best list*

$\boldsymbol{\theta}$ $\xleftarrow[\text{update parameters}]{}$ *k-best list*

*k-best list*

tree1 ✅

tree2 ❌

tree3 ❌

…



* credit: Liang et al. 2011

# Results

| System | GEO | Specialized lexicon | Logical forms |
|---|---|---|---|
| Clarke et al. (2010) w/answers | 73.2 | ✅ | ❌ |
| Clarke et al. (2010) w/logical forms | 80.4 | ✅ | ✅ |
| Our system (DCS with $L$) | 78.9 | ❌ | ❌ |
| Our system (DCS with $L^+$) | **87.2** | ✅ | ❌ |

| System | GEO | JOBS |
|---|---|---|
| Tang and Mooney (2001) | 79.4 | 79.8 |
| Wong and Mooney (2007) | 86.6 | – |
| Zettlemoyer and Collins (2005) | 79.3 | 79.3 |
| Zettlemoyer and Collins (2007) | 81.6 | – |
| Kwiatkowski et al. (2010) | 88.2 | – |
| Kwiatkowski et al. (2010) | 88.9 | – |
| Our system (DCS with $L$) | 88.6 | 91.4 |
| Our system (DCS with $L^+$) | **91.1** | **95.0** |

**+ Logical forms**

**+ Specialized lexicon**



* credit: Liang et al. 2011

# Observations

- Supervision compared to Clark et al.:
  - Use a smaller lexicon
  - Use POS tags vs dependency trees
  - Use simple indicator features vs WordNet features
- Assumptions (impact generalization):
  - Lexicon is general purpose but needs to be exhaustive
  - Indicator features would generalize well
  - DCS space restricted by lexicon and beam search
  - DCS:
    - Expressive enough
    - Efficiently executed over any KB

✓ Baked inductive bias towards syntactic structure

# Contributions

- Present a new semantic framework DCS
  - Expressive
  - Computationally efficient
  - Well motivated to counter lambda calculus
- Show amazing results with simple, cheap e2e supervision

# Personal thoughts

Thank you!

## About the paper:

+ Beautiful idea, well executed
+ Build a nicely presented logical framework
+ Cheaply supervised, beats highly supervised methods
- Very complicated framework

• Semantic Parsing is a fundamental roadblock to NLU
• Need more developments in building more elegant and richer logical frameworks

# References


1. Highly influenced by Percy Liang's talk on YouTube at Learning Semantics Workshop at NIPS 2011.
2. Learning Dependency-Based Compositional Semantics. Liang, Percy and Jordan, Michael I. and Klein, Dan. Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies 2011.
3. Learning Dependency-Based Compositional Semantics. Liang, Percy and Jordan, Michael I. and Klein, Dan. Computational Linguistics 2013.