SFU NatLangLab

CMPT 413/825: Natural Language Processing

Semantic Parsing

Fall 2020

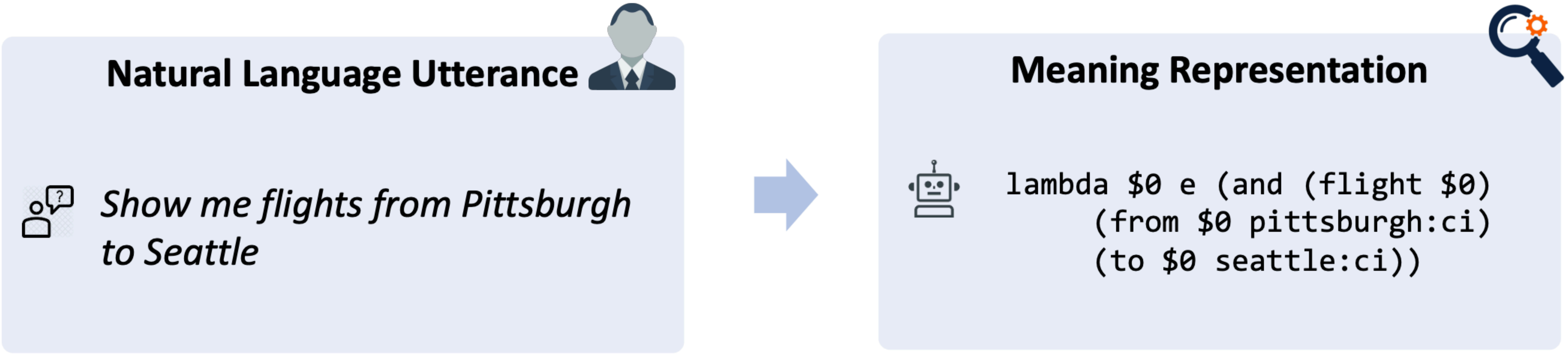
2020-11-13

Adapted from slides from Pengcheng Yin
(with some content from ACL 2018 tutorial on Neural Semantic Parsing by Pradeep Dasigi, Srini Iyer, Alane Suhr, Matt Gardner, Luke Zettlemoyer)

What is semantic parsing?

Logical form

Formal representation

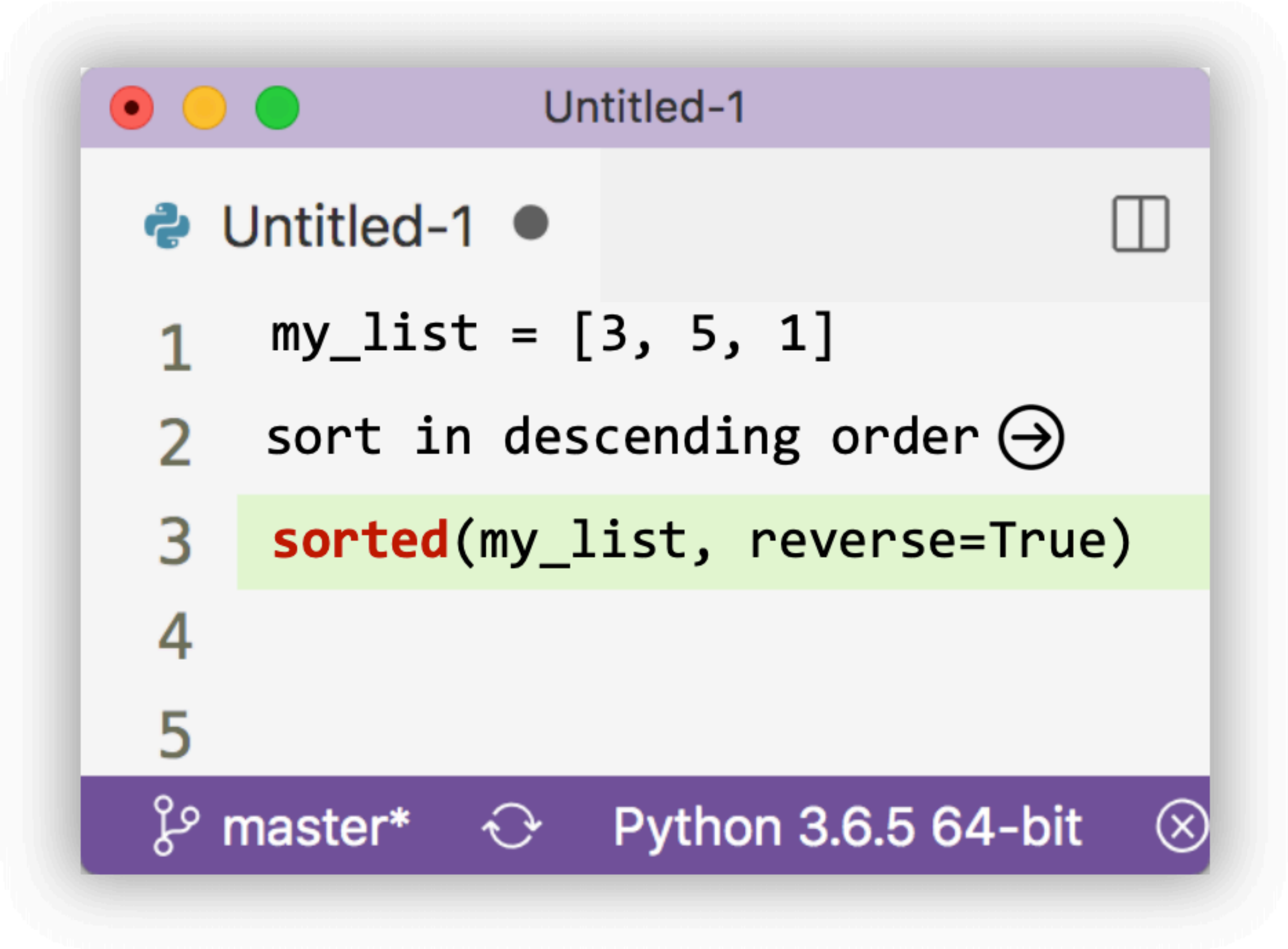


Interpretable by
a machine!

(figure credit: CMU CS 11-747, Pengcheng Yin)

What is semantic parsing good for?

- NLP Tasks
  - Question Answering
- Applications
  - Natural language interfaces
  - Dialogue agents
  - Robots



Virtual Assistants

Set an alarm at 7 AM

Remind me for the meeting at 5pm

Play Jay Chou's latest album

Natural Language Programming

Sort my_list in descending order

Copy my_file to home folder

Dump my_dict as a csv file output.csv

(figure credit: CMU CS 11-747, Pengcheng Yin)

Meaning representations

- **Machine-executable representations**: executable programs to accomplish a task
- **Meaning representation for semantic annotation**: captures the semantics of the natural language sentence

Machine-executable
Meaning Representations

Show me flights from Pittsburgh to Seattle

```
lambda $0 e (and (flight $0)
          (from $0 pittsburgh:ci)
          (to $0 seattle:ci))
```

Lambda Calculus Logical Form

Lambda Calculus

Python, SQL, …

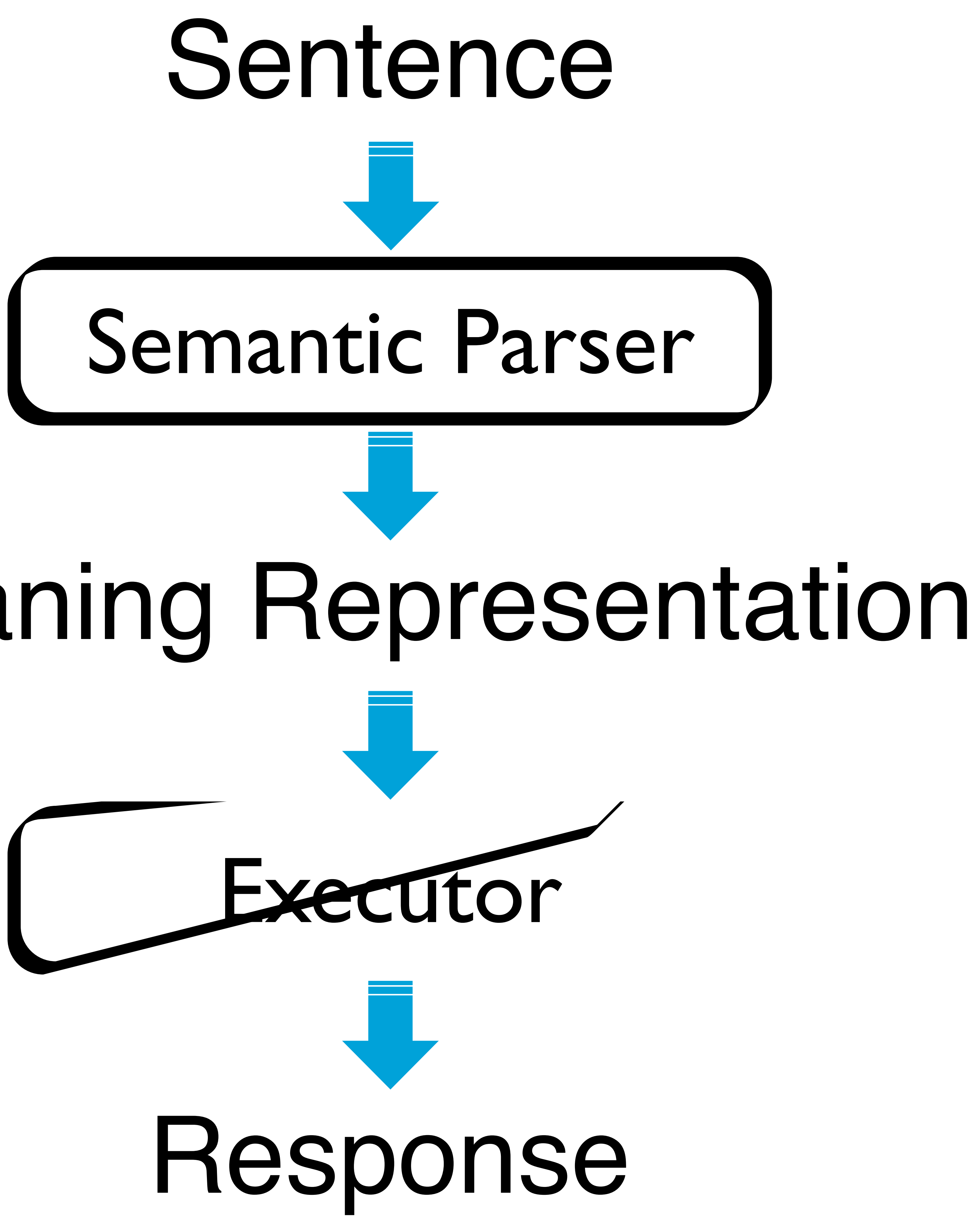
- Arithmetic expressions
- Lambda calculus
- Computer Programs:
  - SQL / Python / DSLs

Abstract Meaning Representation (AMR),

Combinatory Categorical Grammar (CCG)

(slide credit: CMU CS 11-747, Pengcheng Yin)

Semantic Parsing



(slide credit: ACL 2018 tutorial on semantic parsing, Pradeep Dasigi et al)

Semantic Parsing: QA

How many people live in Seattle?

Semantic Parser

```
SELECT Population FROM CityData
where City=="Seattle”;
```

Executor

620,778

[Wong & Mooney 2007],
[Zettlemoyer & Collins 2005, 2007],
[Kwiatkowski et.al 2010, 2011],
[Liang et.al. 2011],[Berant et.al. 2013,2014],[Reddy et.al, 2014,2016],
[Dong and Lapata, 2016] .....

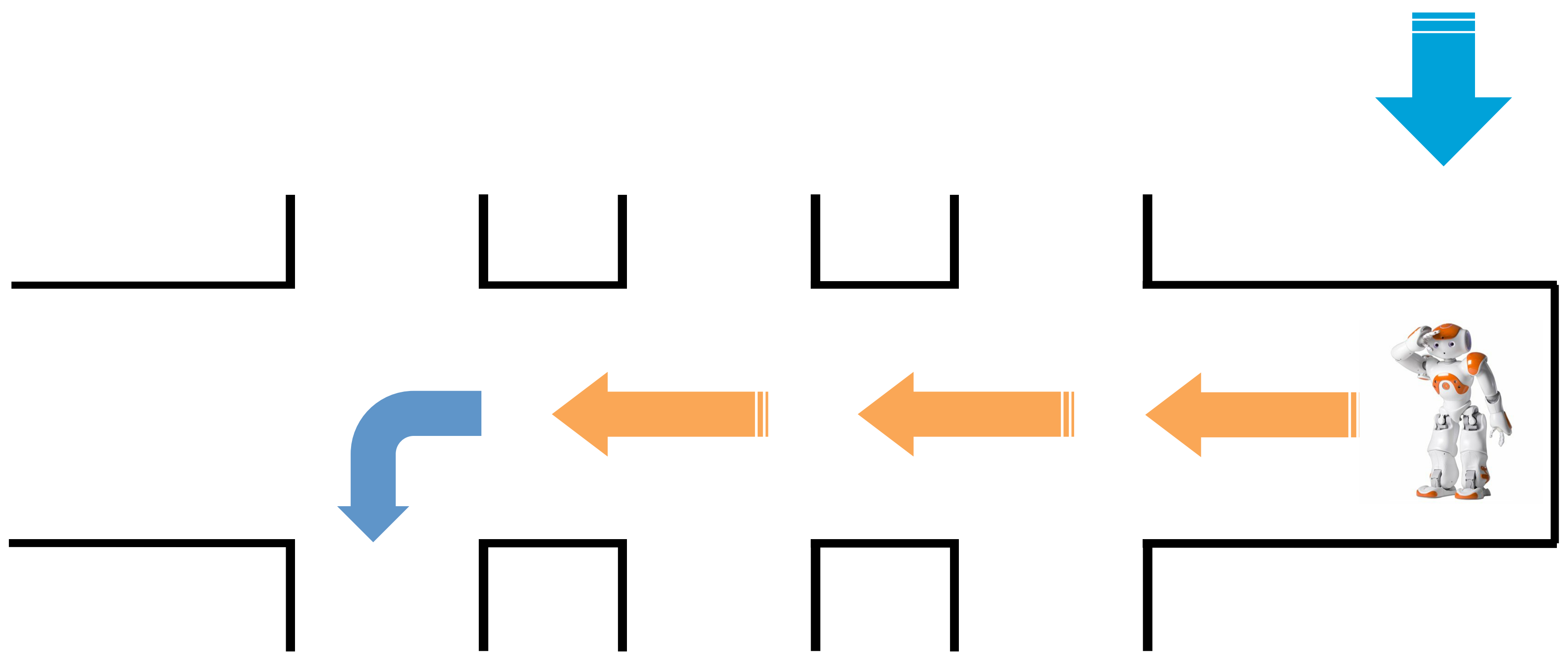
(slide credit: ACL 2018 tutorial on semantic parsing, Pradeep Dasigi et al)

Semantic Parsing: Instructions

Go to the third junction and take a left

Semantic Parser

```
(do-seq(do-n-times 3
 (move-to forward-loc
  (do-until
   (junction current-loc
    (move-to forward-loc))))
 (turn-right))
```



[Chen & Mooney 2011]
[Matuszek et al 2012]
[Artzi & Zettlemoyer 2013]
[Mei et.al. 2015][Andreas et al, 2015]
[Fried at al, 2018] ....

(slide credit: ACL 2018 tutorial on semantic parsing, Pradeep Dasigi et al)

Language to Meaning

More informative

(slide credit: ACL 2018 tutorial on semantic parsing, Pradeep Dasigi et al)

Language to Meaning

Information Extraction

Recover information about pre-specified relations and entities

More informative

Example Task

Relation Extraction

$$is\_a(OBAMA, PRESIDENT)$$

(slide credit: ACL 2018 tutorial on semantic parsing, Pradeep Dasigi et al)

Language to Meaning

Broad-coverage
Semantics

Focus on specific
phenomena

More informative

Example Task

Summarization

Obama wins election. Big party in Chicago. Romney a bit down, asks for some tea.

(slide credit: ACL 2018 tutorial on semantic parsing, Pradeep Dasigi et al)

Language to Meaning

Semantic
Parsing

Recover **complete**
meaning
representation

More informative

Example Task

Database Query

What states
border Texas?

Oklahoma
New Mexico
Arkansas
Louisiana

(slide credit: ACL 2018 tutorial on semantic parsing, Pradeep Dasigi et al)

Semantic Parsing workflow

User's Natural Language Query

Show me flights from Pittsburgh to Seattle

Parsing to Meaning Representation

```
lambda $0 e (and (flight $0)
     (from $0 pittsburgh:ci)
     (to $0 seattle:ci))
```

Execute Programs against KBs

Freebase

WIKIDATA

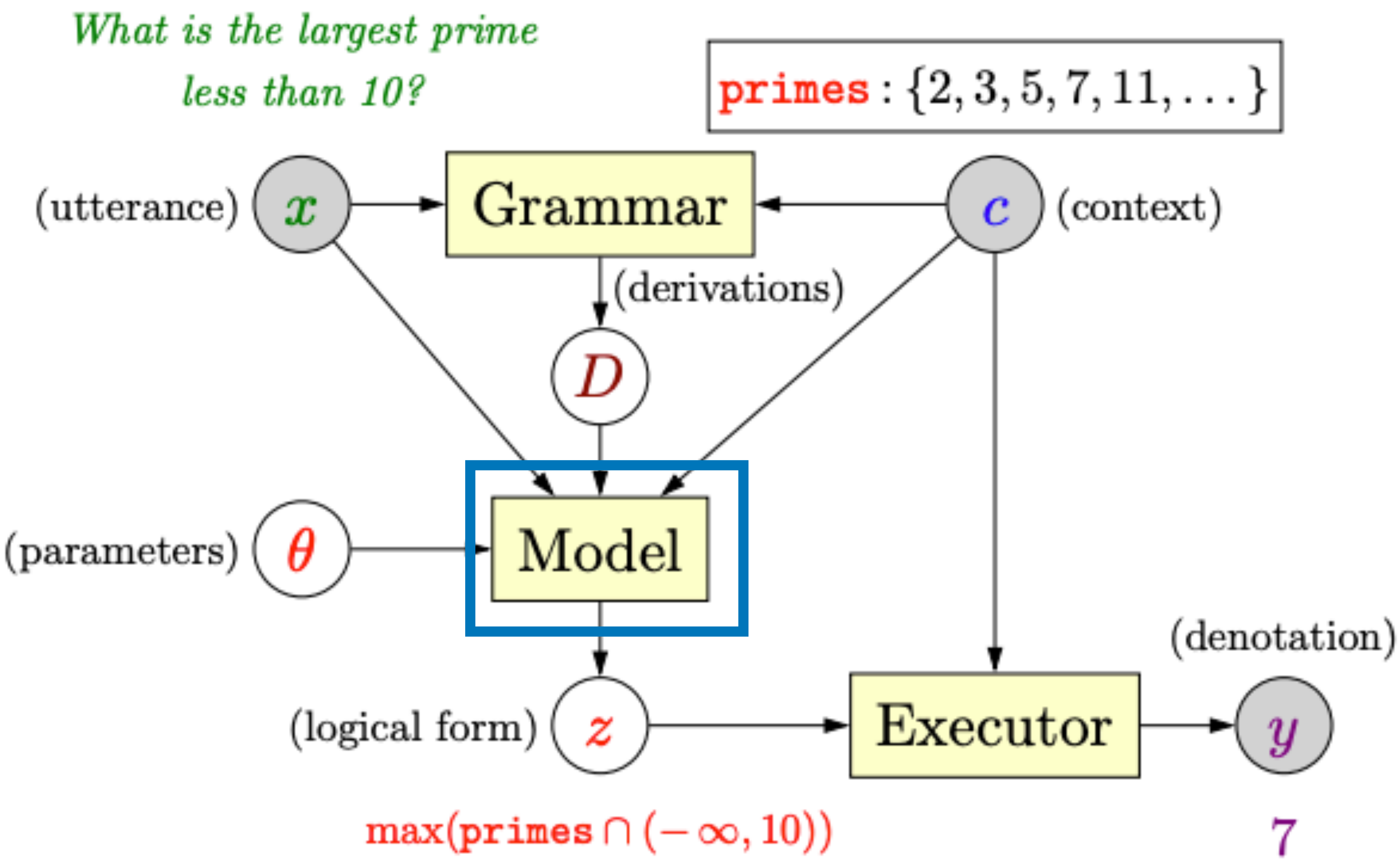
Execution Results (Answer)

```
1. Alaska Air 119
2. American 3544 -> Alaska 1101
3. …
```

Denotation

(slide credit: CMU CS 11-747, Pengcheng Yin)

Semantic Parsing Components



Goal: learn parameters θ for a function that gives a **score(x, c, d)** that judges how good a derivation d is wrt the utterance x and context c

(figure credit: Percy Liang)

Supervised learning of Semantic Parsers

User's Natural Language Query

Show me flights from Pittsburgh to Seattle

Parsing to Meaning Representation

```
lambda $0 e (and (flight $0)
     (from $0 pittsburgh:ci)
     (to $0 seattle:ci))
```


(slide credit: CMU CS 11-747, Pengcheng Yin)

Meaning Representations and Datasets

Domain-Specific, Task-Oriented Languages (DSLs)

Show me flights from Pittsburgh to Seattle

```
lambda $0 e (and (flight $0)
          (from $0 Pittsburgh:ci)
          (to $0 Seattle:ci))
```

lambda-calculus logical form

GeoQuery / ATIS / JOBS
WikiSQL / Spider
IFTTT

General-Purpose Programming Languages

Sort my_list in descending order

```
sorted(my_list, reverse=True)
```

Python code generation

Django, HeartStone,
CONCODE, CoNaLa, JuICe

(slide credit: CMU CS 11-747, Pengcheng Yin)

GEO Query, ATIS, JOBS

- **GEO Query** 880 queries about US geographical information
- **ATIS** 5410 queries about flight booking and airport transportation
- **Jobs** 640 queries to a job database

GEO Query

which state has the most rivers running through it?

```
argmax $0
 (state:t $0)
 (count $1 (and
             (river:t $1)
             (loc:t $1 $0)))
```

Lambda Calculus Logical Form

ATIS

Show me flights from Pittsburgh to Seattle

```
lambda $0 e
   (and (flight $0)
   (from $0 pittsburgh:ci)
   (to $0 seattle:ci))
```

Lambda Calculus Logical Form

JOBS

what Microsoft jobs do not require a bscs?

```
answer(
  company(J,'microsoft'),
  job(J),
  not((req deg(J,'bscs'))))
```

Prolog-style Program

(slide credit: CMU CS 11-747, Pengcheng Yin)

Text-to-SQL Tasks

Natural Language Questions with Database Schema

Input Utterance

Show me flights from Pittsburgh to Seattle

Flight	
FlightNo	UniqueId
Departure	foreign key
Arrival	foreign key

Airport	
Name	UniqueId
CityName	string
PublicTransport	boolean

SQL Query

```
SELECT Flight.FlightNo
FROM Flight
JOIN Airport as DepAirport
ON
   Flight.Departure == DepAirport.Name
JOIN Airport as ArvAirport
ON
   Flight.Arrival == ArvAirport.Name
WHERE
    DepAirport.CityName == Pittsburgh
    AND
    ArvAirport.CityName == Seattle
```

The CoNaLa Code Generation Dataset

Get a list of words `words` of a file 'myfile'

```
words = open('myfile').read().split()
```

Copy the content of file 'file.txt' to file 'file2.txt'

```
shutil.copy('file.txt’, 'file2.txt')
```

Check if all elements in list `mylist` are the same

```
len(set(mylist)) == 1
```

Create a key `key` if it does not exist in dict `dic` and append element `value` to value

```
dic.setdefault(key, []).append(value)
```

- 2,379 training and 500 test examples
- Natural Language queries collected from StackOverflow
- Manually annotated, high quality natural language queries
- Code is highly expressive and compositional

`conala-corpus.github.io` [Yin *et al.*, 2018]

(slide credit: CMU CS 11-747, Pengcheng Yin)

Supervised learning of Semantic Parsers

User's Natural Language Query	Parsing to Meaning Representation
Show me flights from Pittsburgh to Seattle	`lambda $0 e (and (flight $0) (from $0 pittsburgh:ci) (to $0 seattle:ci))`

- Train a semantic parser with source natural language utterance and target programs
- Can use general structured prediction methods (similar methods as for constituency parsing and dependency parsing)

(slide credit: CMU CS 11-747, Pengcheng Yin)

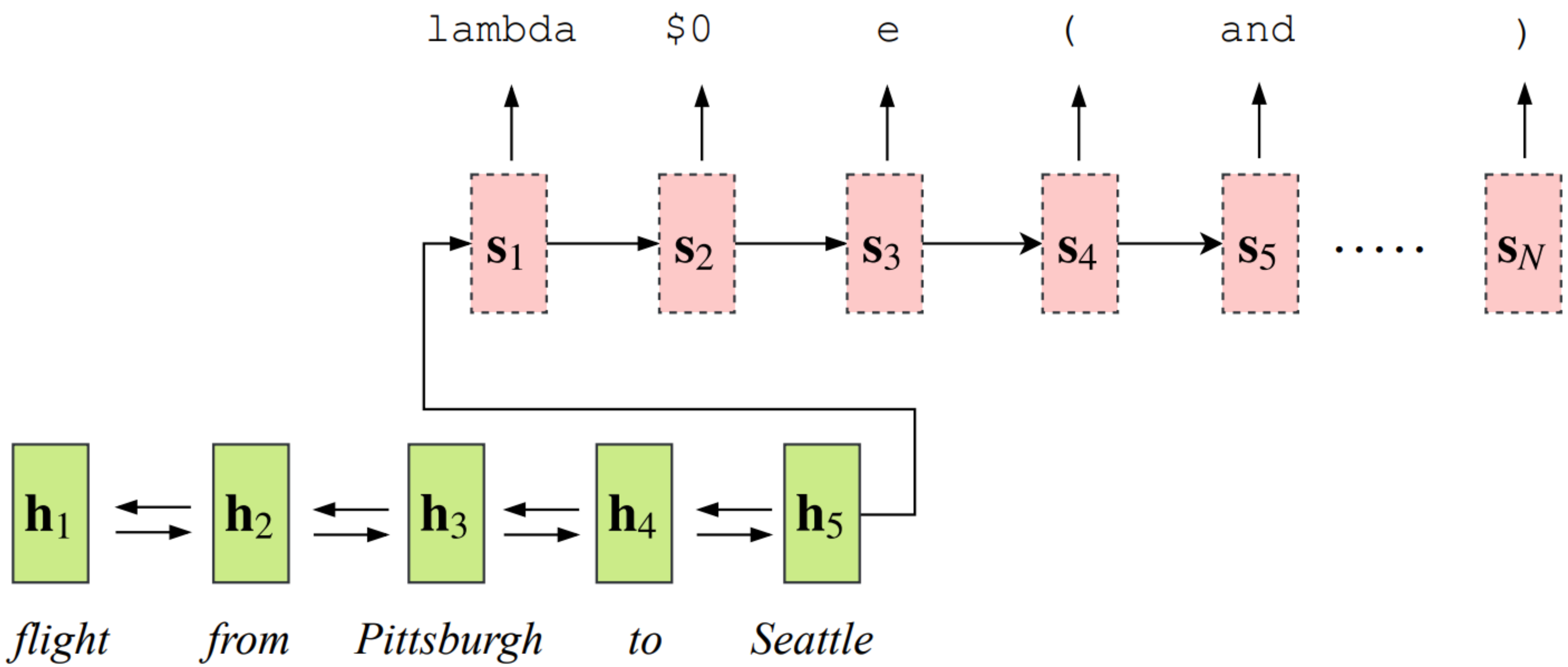
Semantic Parsing as Sequence-to-Sequence Transduction

Task-Specific Meaning Representations

Show me flights from Pittsburgh to Seattle

```
lambda $0 e (and (flight $0)
       (from $0 pittsburgh:ci)
       (to $0 seattle:ci))
```

Lambda Calculus Logical Form



- Treat the target meaning representation as a sequence of surface tokens
- Reduce the (structured prediction) task as another sequence-to-sequence learning problem

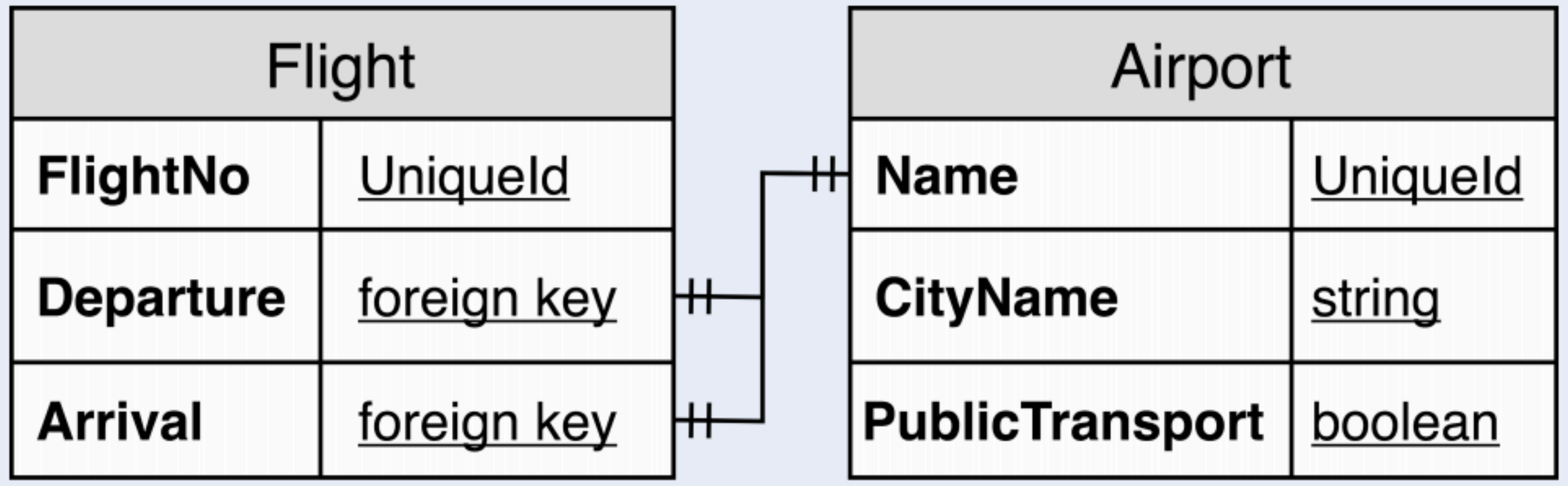
[Dong and Lapata, 2016; Jia and Liang, 2016]

(slide credit: CMU CS 11-747, Pengcheng Yin)

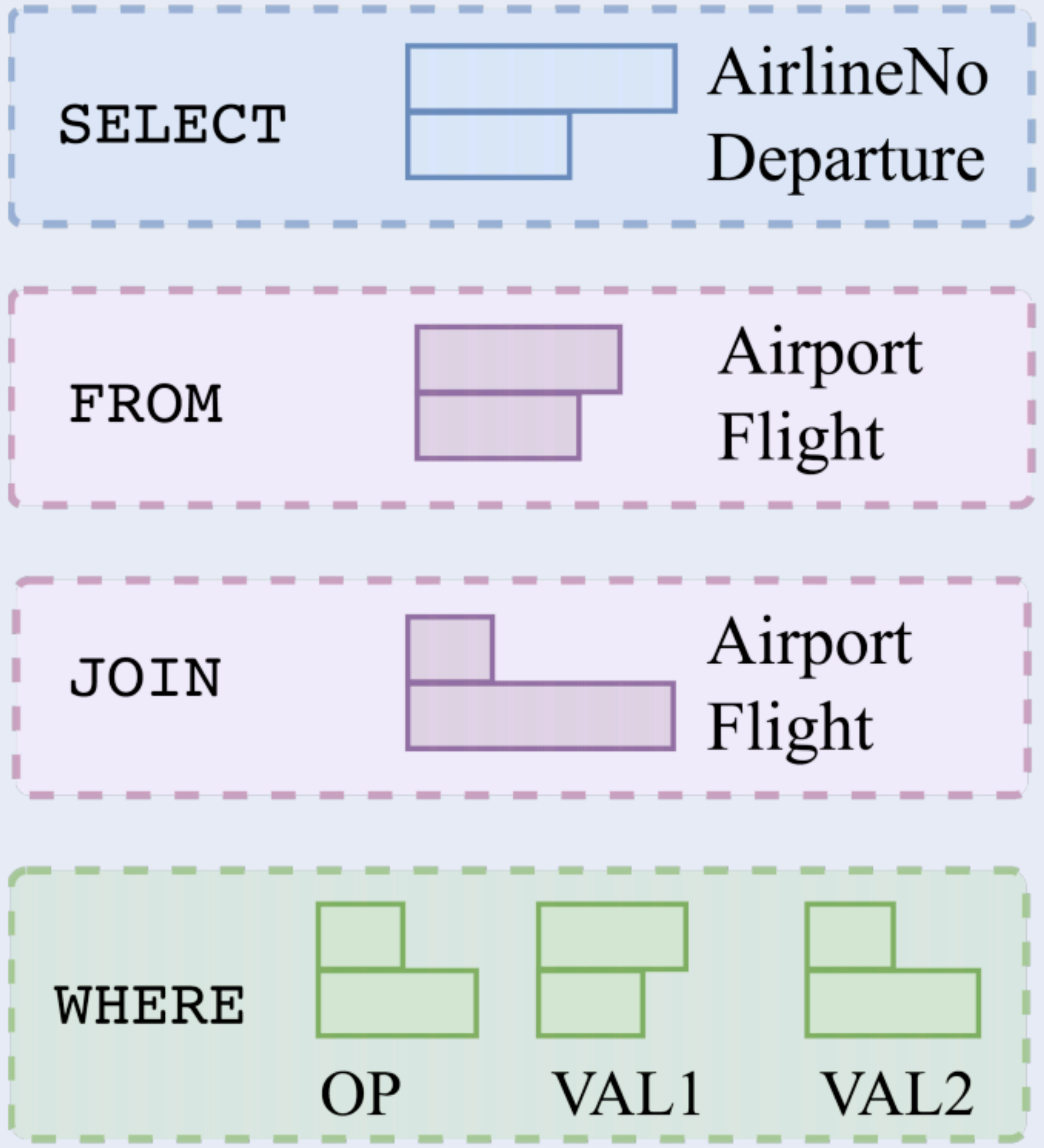
Encode Utterance and In-Domain Knowledge Schema

Input Utterance

Show me flights from Pittsburgh to Berkeley



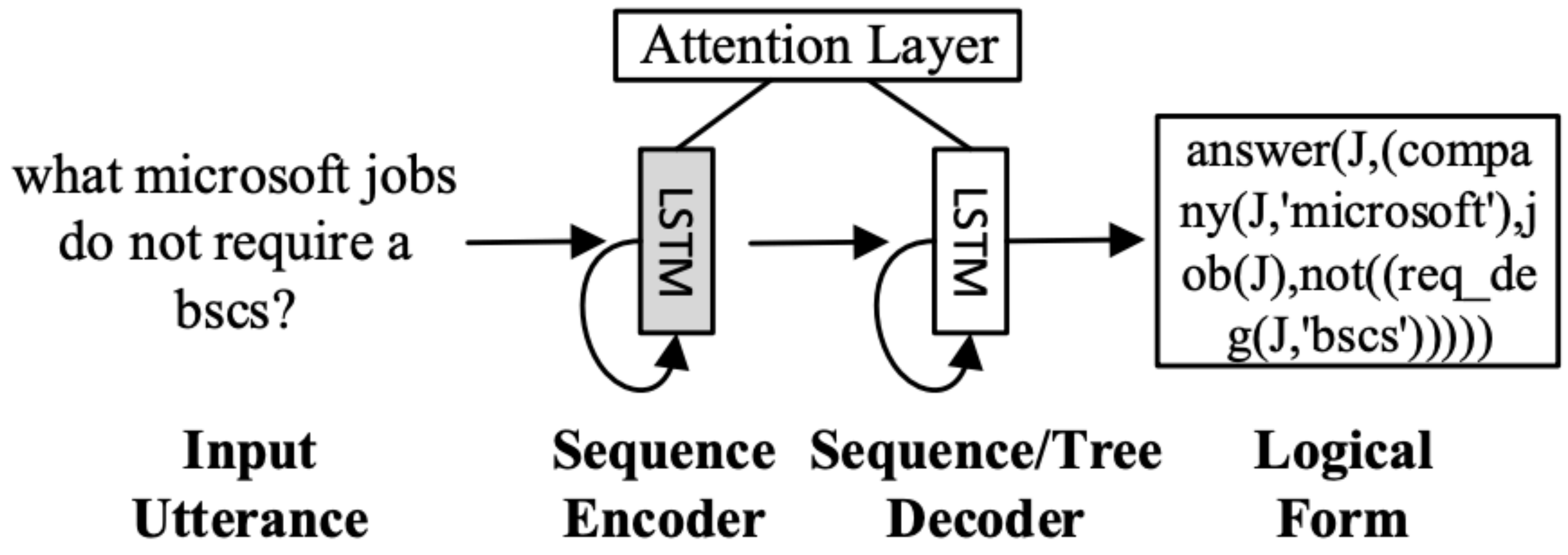
Predict Programs Following Task-Specific Program Structures



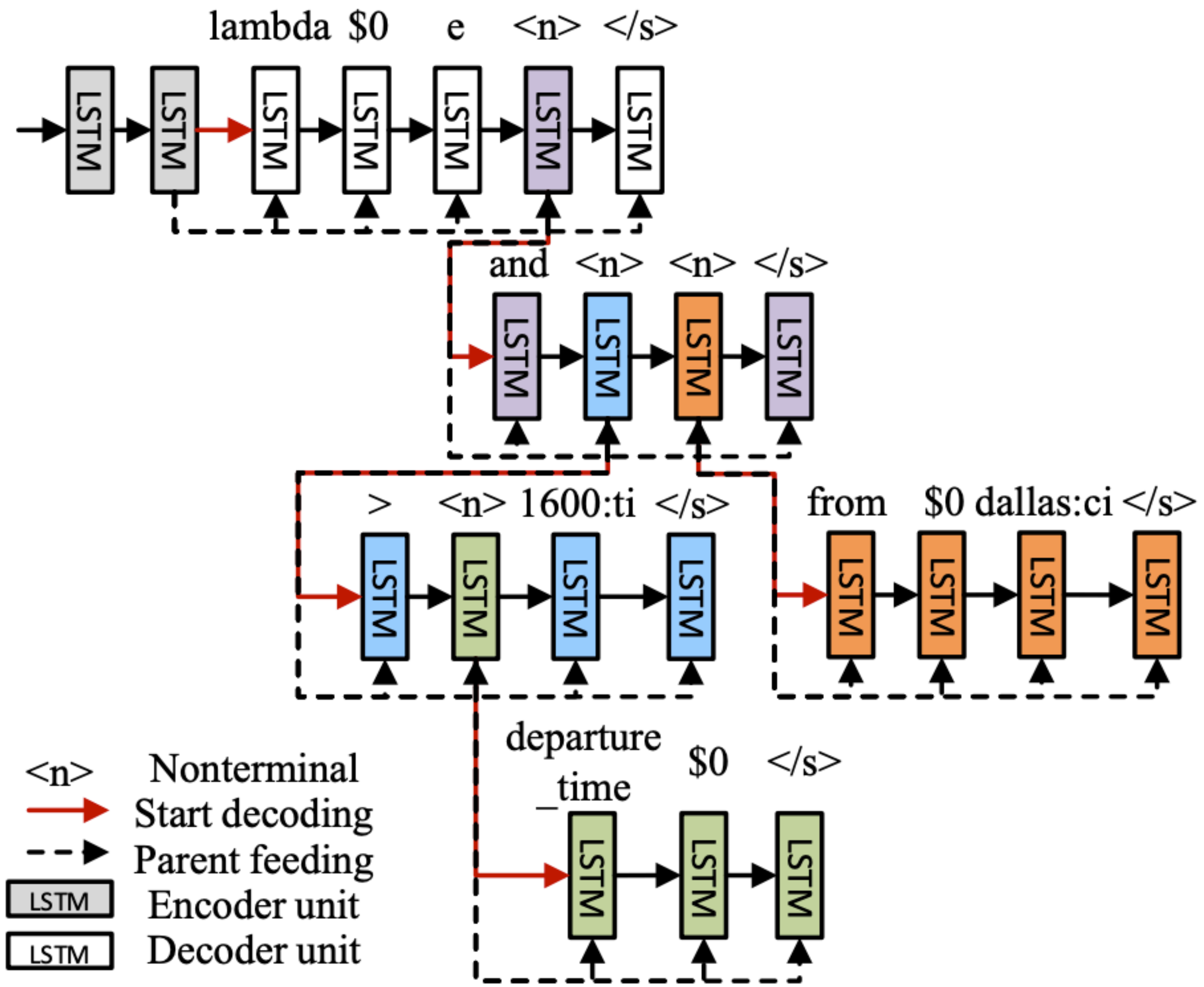
[Xu et al., 2017; Yu et al., 2018]

(slide credit: CMU CS 11-747, Pengcheng Yin)

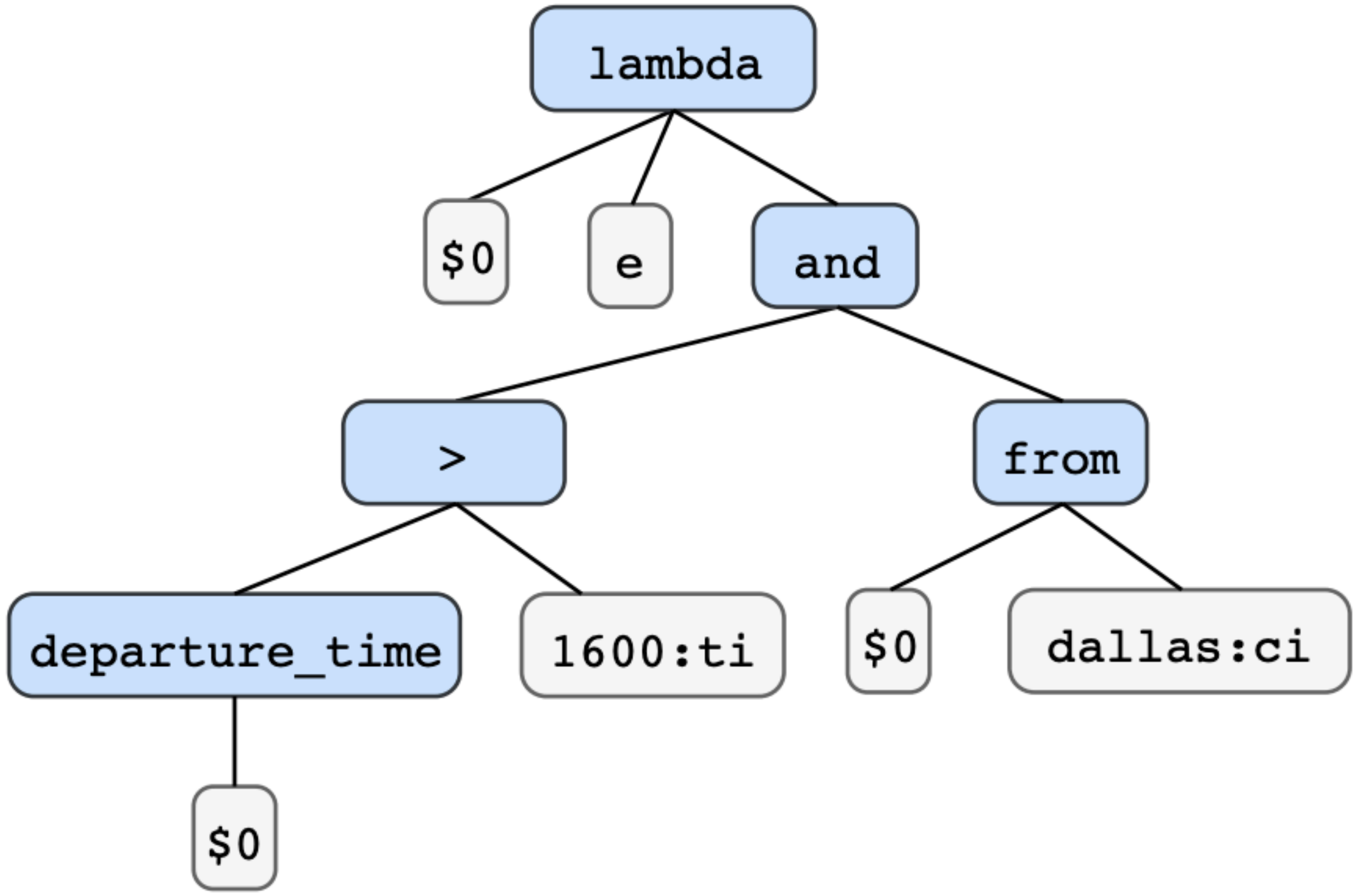
Structure-aware Decoding for Semantic Parsing (Dong and Lapata, 2016)



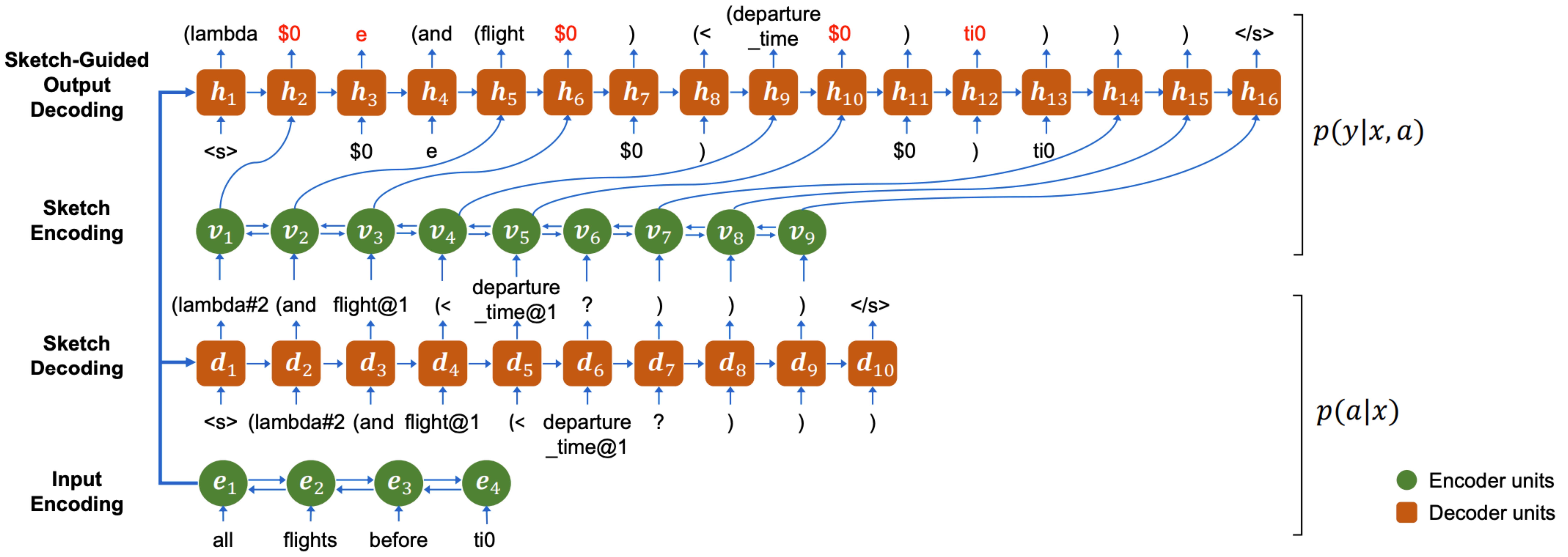
Structure-aware Decoding for Semantic Parsing (Dong and Lapata, 2016)



Show me flight from Dallas departing after 16:00



Coarse-to-Fine Decoding (Dong and Lapata, 2018)



Grammar/Syntax-driven Semantic Parsing

Python Abstract Grammar

expr $\longmapsto$ Name | Call

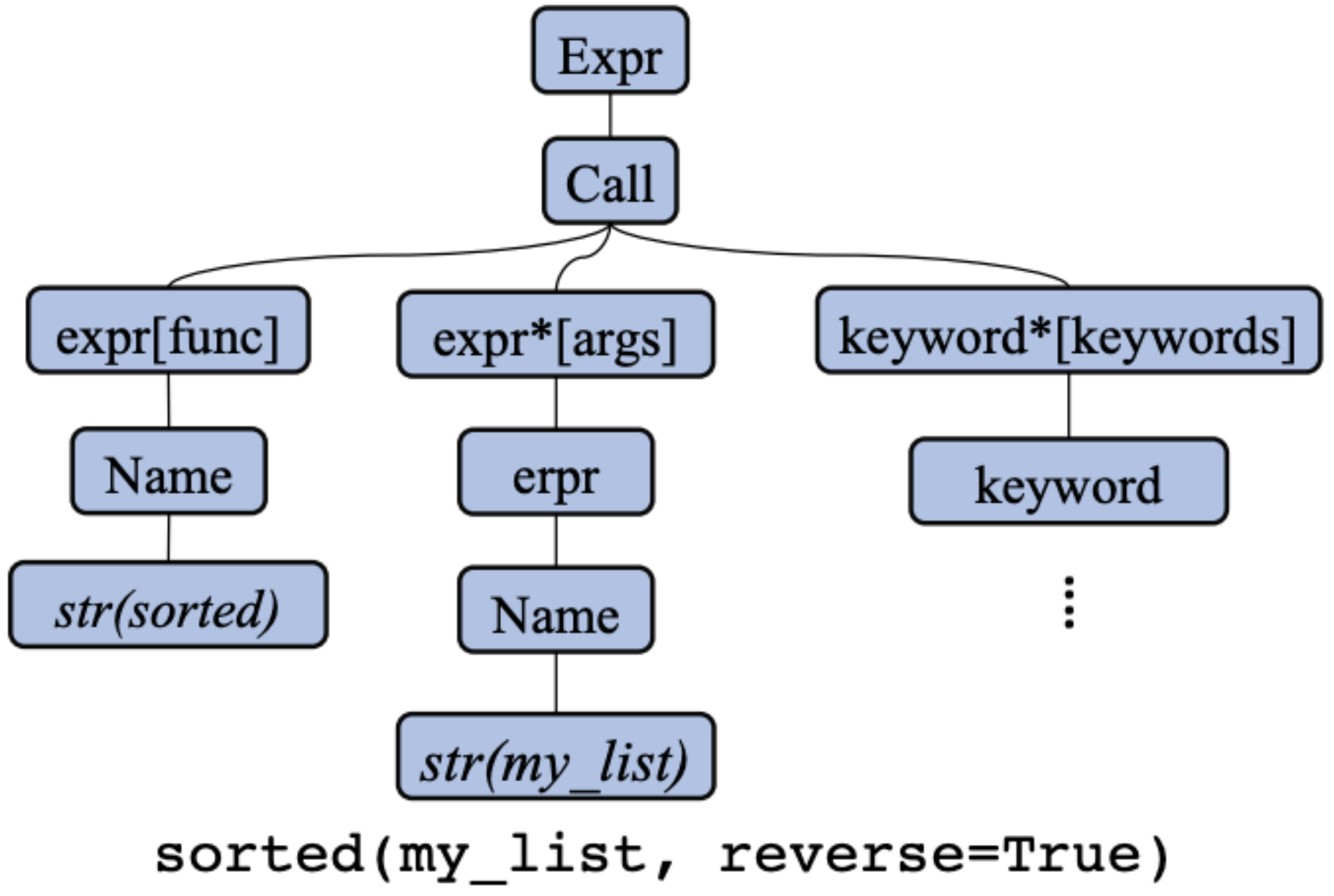
Call $\longmapsto$ expr[*func*] expr*[*args*] keyword*[*keywords*]

If $\longmapsto$ expr[*test*] stmt*[*body*] stmt*[*orelse*]

For $\longmapsto$ expr[*target*] expr*[*iter*] stmt*[*body*] stmt*[*orelse*]

FunctionDef $\longmapsto$ identifier[*name*] expr*[*iter*] stmt*[*body*] stmt*[*orelse*]

Abstract Syntax Tree

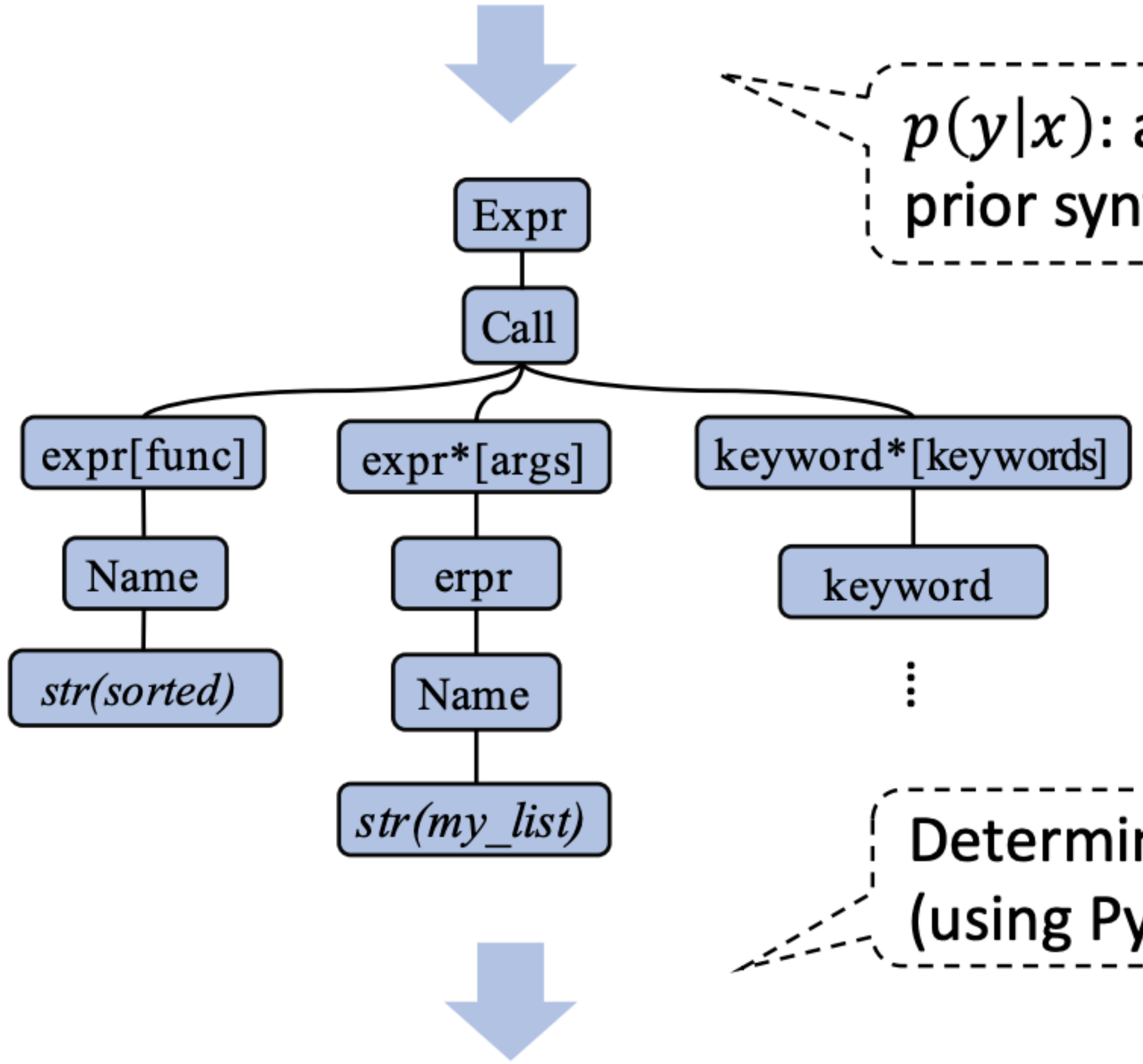


(slide credit: CMU CS 11-747, Pengcheng Yin)

Grammar/Syntax-driven Semantic Parsing

Input Intent $(\boldsymbol{x})$ *sort my_list in descending order*

Generated AST $(\boldsymbol{y})$



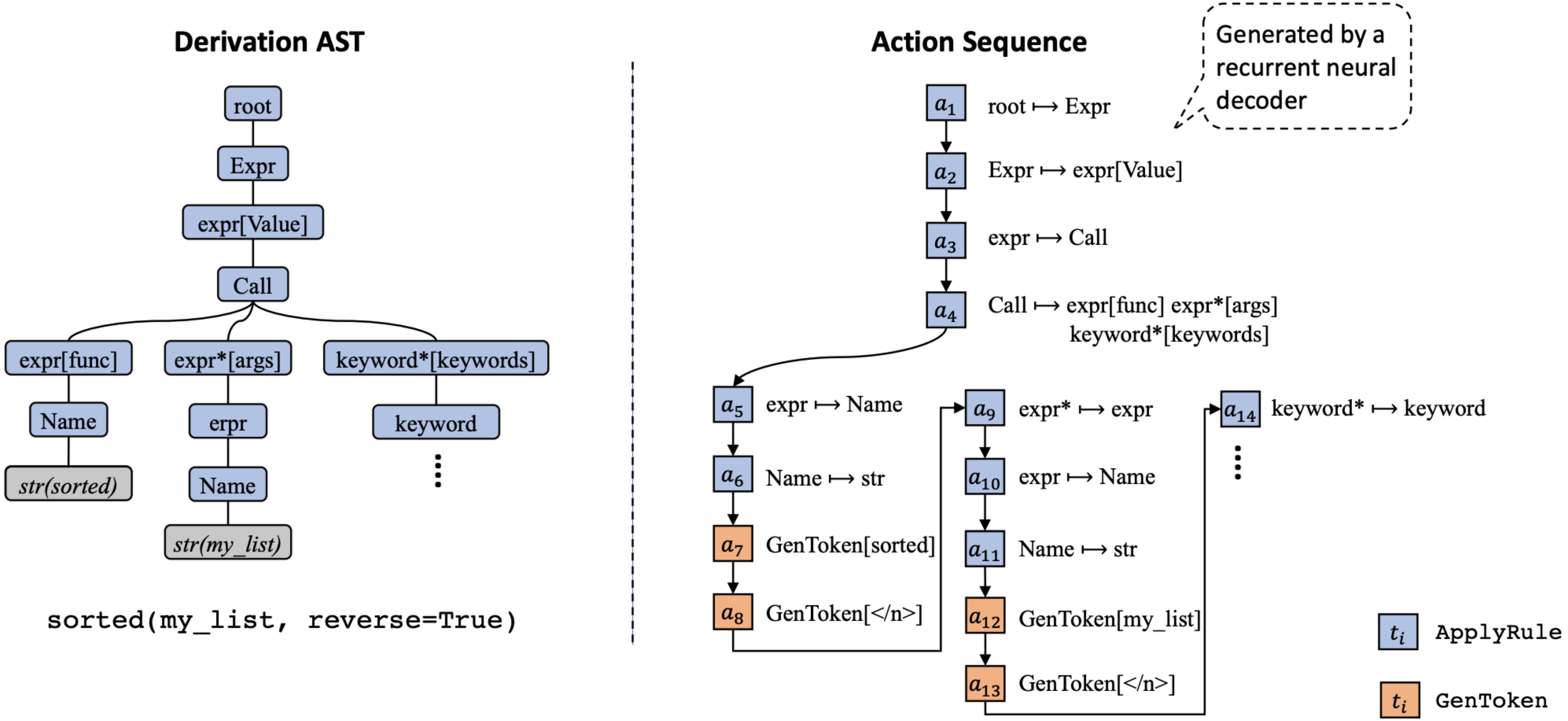
$p(y|x)$: a seq2seq model with prior syntactic information

Deterministic transformation (using Python `astor` library)

Surface Code $(\boldsymbol{c})$ `sorted(my_list, reverse=True)`

(slide credit: CMU CS 11-747, Pengcheng Yin)

Grammar/Syntax-driven Semantic Parsing



(slide credit: CMU CS 11-747, Pengcheng Yin)

Weakly Supervised Semantic Parsing

Learning from denotations

User’s Natural Language Query

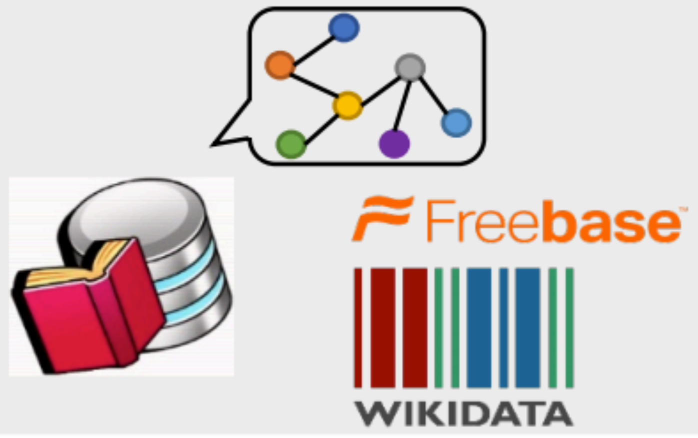
Show me flights from Pittsburgh to Seattle

Parsing to Meaning Representation

```
lambda $0 e (and (flight $0)
      (from $0 pittsburgh:ci)
      (to $0 seattle:ci))
```

As unobserved latent variable

Query Execution



Execution Results (Answer)

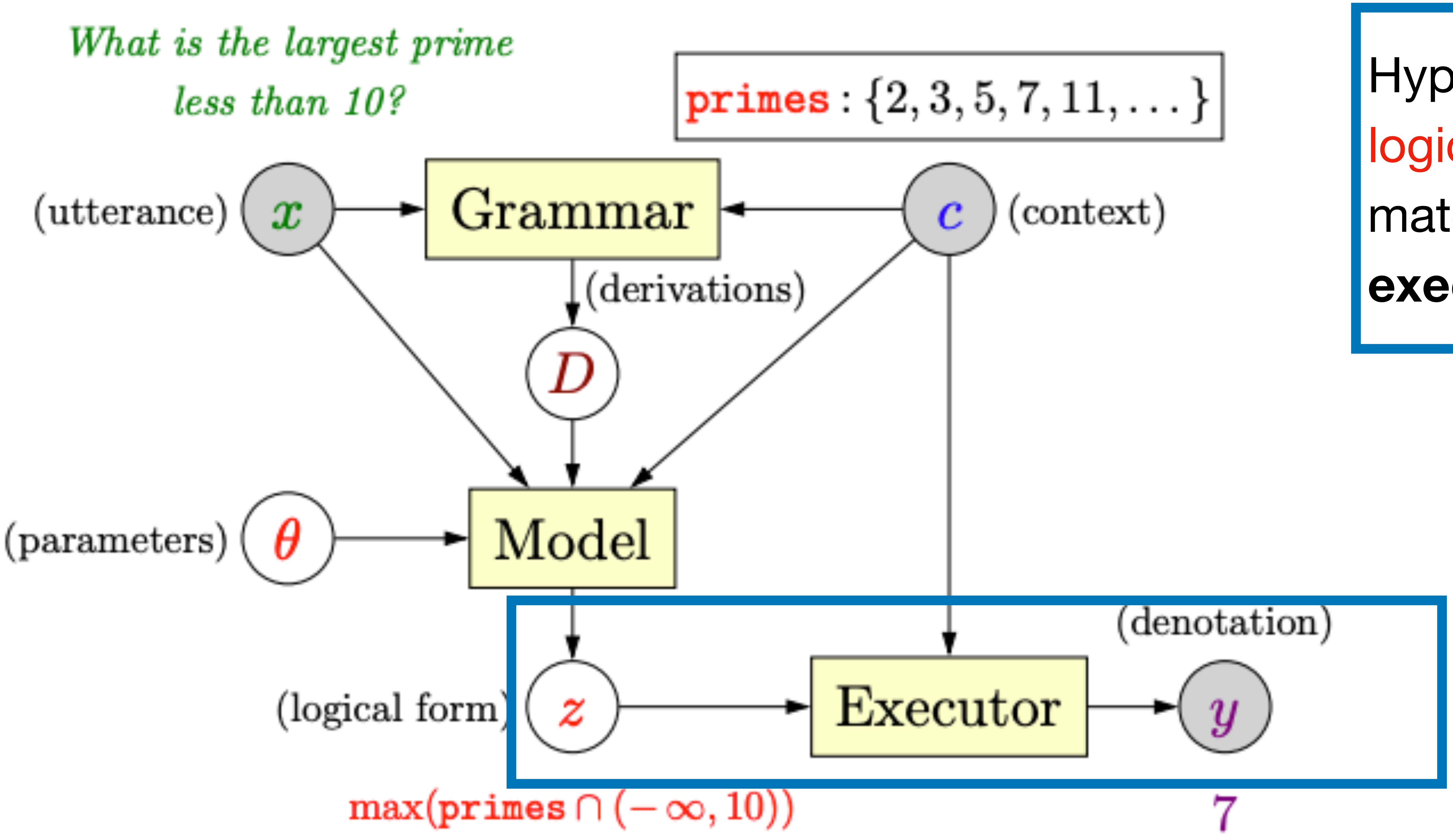
```
1. AS 119
2. AA 3544 -> AS 1101
3. …
```

Weak supervision signal

Train a semantic parser using natural language query and the execution results (a.k.a. Semantic Parsing **with Execution**)

(slide credit: CMU CS 11-747, Pengcheng Yin)

Semantic Parsing Components



Hypothesize possible logical forms that may match the utterance x and **execute** to get denotation.

(figure credit: Percy Liang)

Weakly Supervised Semantic Parsing

Weakly Supervised Semantic Parsing

What is the most populous city in United States?

City	Country	Population	GDP
New York	USA	8.62M	1275B
Hong Kong	China	7.39M	341.4B
Tokyo	Japan	9.27M	1800B
London	UK	8.78M	650B
Los Angeles	USA	4.00M	941B

`Answer: New York`

Hypothesized Programs

```
City.OrderBy(Population)
    .First() => Result: Tokyo
```

```
City.Filter(Country==‘USA’)
    .OrderBy(Population)
    .First() => Result: New York
```

```
City.Filter(Country==‘USA’)
    .OrderBy(GDP)
    .First() => Result: New York
```


(slide credit: CMU CS 11-747, Pengcheng Yin)

Weakly Supervised Semantic Parsing - Challenges

Hypothesized Programs

```
City.OrderBy(Population)
    .First() => Result: Tokyo
```

```
City.Filter(Country=='USA')
    .OrderBy(Population)
    .First() => Result: New York
```

```
City.Filter(Country=='USA')
    .OrderBy(GDP)
    .First() => Result: New York
```

Large Search Space

Exponentially large search space w.r.t. the size of programs

Very Sparse Rewards

Only very few programs are actually correct

Spurious Programs

Spurious programs could also hit the correct answer, leading to noisy reward signals.

(slide credit: CMU CS 11-747, Pengcheng Yin)

Weakly Supervised Semantic Parsing

- Maximum Marginal Likelihood
- Structured Learning Methods
- Reinforcement Learning

Maximum Marginal Likelihood

- Given $D = \{x_i, w_i, z_i\}_{i=1}^{N}$
- We want to optimize $\max_{\theta} \prod_{x_i, z_i \in D} p(z_i | x_i; \theta)$
- But the semantic parser defines a distribution over logical forms.
- So we marginalize over logical forms that yield z_i

$$\max_{\theta} \prod_{x_i, w_i, z_i \in D} \sum_{y_i \in Y | [\![y_i]\!]^{w_i} = z_i} p(y_i | x_i; \theta)$$

- Y could be the set of all valid logical forms, if we are using constrained decoding during training
- Even then, the summation could be intractable!

(slide credit: ACL 2018 tutorial on semantic parsing, Pradeep Dasigi et al)

MML: Approximating $\mathbf{Y}$

- Perform heuristic search
- Search may be bounded, by length or otherwise
- $\mathbf{Y}$ is approximated as a subset of retrieved logical forms

Two options for search:

Online Search	Offline Search
Search for consistent logical forms during training, as per model scores	Search for consistent logical forms before training
Candidate set changes as training progresses	Candidate set is static
Less efficient	More efficient

(slide credit: ACL 2018 tutorial on semantic parsing, Pradeep Dasigi et al)

Structured Learning Methods

- More commonly used with traditional semantic parsers
  - Eg. Margin based models and Latent variable structured perceptron (Zettlemoyer and Collins 2007)
- Typically involve heuristic search over the state space like MML methods
- Unlike MML, can use arbitrary cost function
- Training typically maximizes margins or minimizes expected risks

(slide credit: ACL 2018 tutorial on semantic parsing, Pradeep Dasigi et al)

Reinforcement Learning Methods

- Comparison with MML:
  - Like MML $\mathbb{Y}$ is approximated
  - Unlike MML, the approximation is done using sampling techniques
- Comparison with structured learning methods
  - Like structured learning methods, the reward function can be arbitrary
  - Unlike structured learning methods, reward is directly maximized
- Training typically uses policy gradient methods

  Example from Liang et al., 2017, using REINFORCE

$$\max_{\theta} \sum_{x} \mathbb{E}_{P_\theta(a_{0:T}|x)}[R(x, a_{0:T})]$$

(slide credit: ACL 2018 tutorial on semantic parsing, Pradeep Dasigi et al)

Weakly Supervised Semantic Parsing as Reinforcement Learning

NL question

What is the most populous city in United States?

Semantic Parsing

Sampled Logical From
(Lambda DCS, Liang 2011)

z_1 **argmax(λx.city(x)∧located(x,US), λx.population(x))** ✓

z_2 argmax(λx.city(x), λx.population(x)) ✗

z_3 argmax(λx.city(x)∧loc(x,US), λx.GDP(x)) ✓

⋮

Query Execution

Answer
(with rewards)

y_1 New York ✓

y_2 Tokyo ✗

y_3 New York ✓

Optimize Objective

Probability of Gold Answer

$$p(\boldsymbol{y}^* = \text{New York}) = p(\boldsymbol{z}_1|\boldsymbol{x}) + p(\boldsymbol{z}_3|\boldsymbol{x})$$

Gradient Updates

(slide credit: CMU CS 11-747, Pengcheng Yin)

Maximum Marginal Likelihood

- Intuitively, the gradient from each candidate logical form is weighted by its normalized probability. The more likely the logical form is, the higher the weight of its gradient

What is the most populous city in United States?

Semantic Parsing

Reward

z_1 **argmax(λx.city(x)∧located(x,US), λx.population(x))** ✓

z_3 argmax(λx.city(x)∧loc(x,US), λx.GDP(x)) ✓

Marginalization over all (sampled) hypotheses

$$\nabla \log p_\theta(\boldsymbol{y}^*|\boldsymbol{x}) = \sum_{\boldsymbol{z}:\text{answer}(\boldsymbol{z})=\boldsymbol{y}^*} w(\boldsymbol{z},\boldsymbol{x}) \cdot \nabla \log p_\theta(\boldsymbol{z}|\boldsymbol{x})$$

Gold Answer

Candidate Logical Form (Latent Variable)

where $$w(\boldsymbol{z},\boldsymbol{x}) = \frac{p_\theta(\boldsymbol{z}|\boldsymbol{x})}{\sum_{\boldsymbol{z}':\text{answer}(\boldsymbol{z}')=\boldsymbol{y}^*} p_\theta(\boldsymbol{z}'|\boldsymbol{x})}$$

(slide credit: CMU CS 11-747, Pengcheng Yin)

Retrieve and Edit
(Hashimoto et al, 2018)

x — Input

Name: Spellbreaker
Stats: ATK4 DEF3 COST4 DUR-1
Type: Minion
Class: Neutral
Minion type: NIL
Rarity: Common
Description:
Battlecry:
Silence a minion

y' — Retrieved prototype

```
class DarkIronDwarf (MinionCard):
    def __init__(self):
        super().__init__("Dark Iron Dwarf",4,
                CHARACTER_CLASS.ALL,CARD_RARITY.COMMON,
                minion_type=MINION_TYPE.NONE,
                battlecry=Battlecry(Give(
                          BuffUntil(ChangeAttack(2),
                          TurnEnded(player=CurrentPlayer()))),
                          MinionSelector(players=BothPlayer(),
                          picker = UserPicker())))
    def create_minion(self, player):
        return Minion(4, 4)
```

y — Ground truth

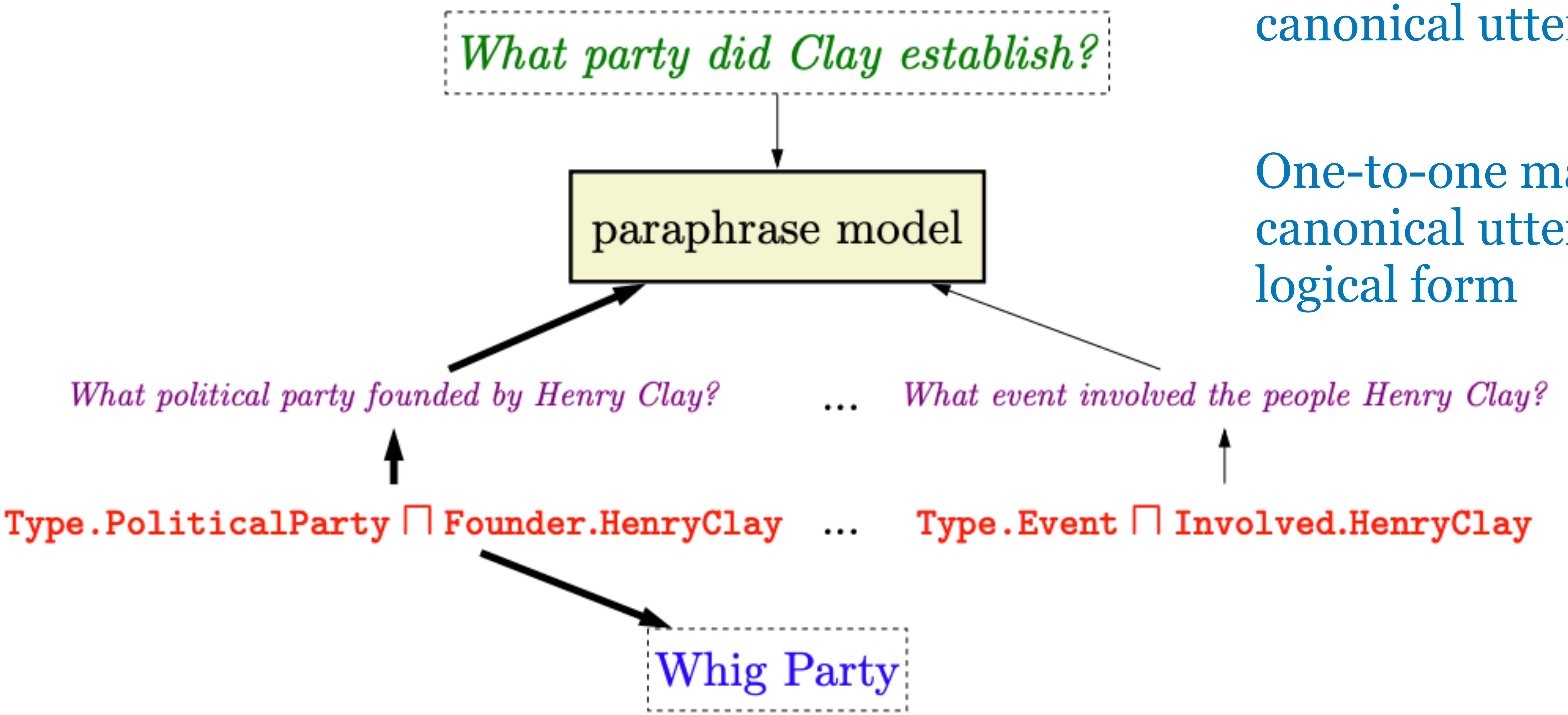
```
class Spellbreaker (MinionCard):
    def __init__(self):
        super().__init__ ("Spellbreaker",4,
                CHARACTER_CLASS.ALL,CARD_RARITY.COMMON,
                minion_type=MINION_TYPE.NONE,
                battlecry=Battlecry(Silence(),
                          MinionSelector(players=BothPlayer(),
                          picker = UserPicker())))
     def create_minion(self, player):
         return Minion(4, 3)
```

Edited output

```
class Spellbreaker (MinionCard):
    def __init__(self):
        super().__init__ ("Spellbreaker",4,
                CHARACTER_CLASS.ALL,CARD_RARITY.COMMON,
                minion_type=MINION_TYPE.NONE,
                battlecry=Battlecry(Silence(),
                          MinionSelector(players=BothPlayer(),
                          picker = UserPicker())))
     def create_minion(self, player):
         return Minion(4, 3)
```

Red text: appears in generation, but not in ground truth
Blue text: missing from generation, but appears in ground truth

Semantic Parsing via Paraphrasing (Berant and Liang, 2014)



Learn to map input to canonical utterance

One-to-one mapping between canonical utterance and logical form

Interactive Semantic Parsing (Wang et al, 2016)

$$p_\theta(z \mid x) \propto \exp(\phi(x, z) \cdot \theta)$$

x : *add a cyan block to red blocks*

z : add(hascolor(red), cyan)

y :