CS 151 Project: Movie Q&A Chatbot

Lara Bagdasarian, Emma Zhong

Outline

- Introduction
- Demo
- Parsing procedure
- Why logic programming?

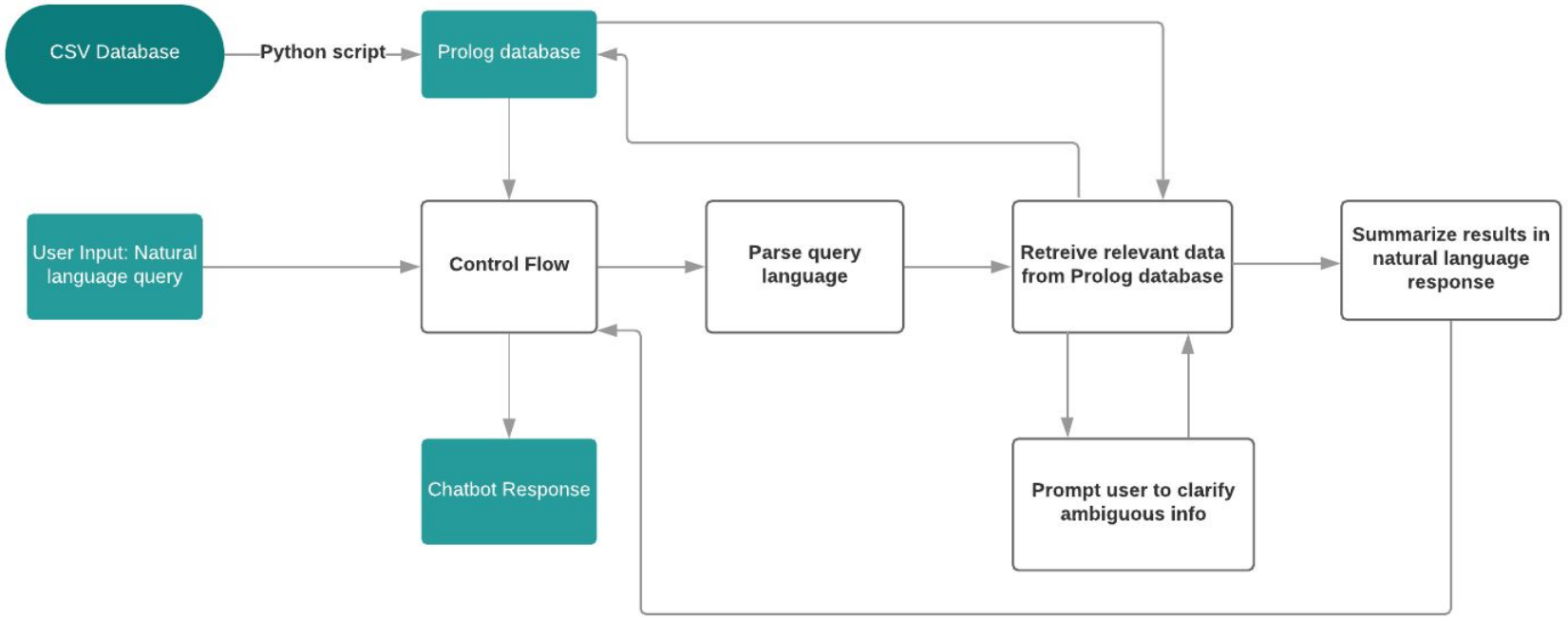
Introduction

- Movie Q&A ChatBot:
  - Parsing NL input + outputting NL answer
  - Disambiguation with user engagement
- Parsing using Logic Programming

Demo

https://swish.swi-prolog.org/p/movie_chatbot.pl

Project Architecture



Query to Response: Inside a Query

```
Question: What are some movies by Quentin Tarantino
```

1. Tokenization

["What", "are", "some", "movies", "by", "Quentin", "Tarantino"]

Casing is useful info

2. Search for named entities

Find candidate members of database ("Quentin", "Tarantino") including partial database entries e.g. "Tarantino"

Bottom-up parsing strategy

3. Pattern recognition:

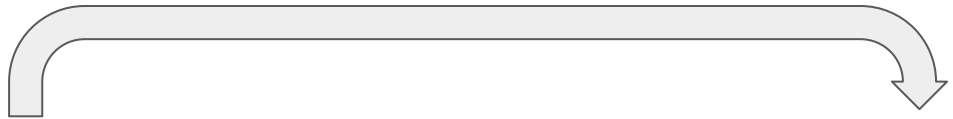
- Recognize desired attribute
- Recognize provided info

e.g. Pattern1: [**QW**] [**Q Attr**] [**P Attr**] [**∈ db**]
QW = What; **Q Attr** = movies; **P Attr** = director;
∈ db = Quentin Tarantino

Actors can also be directors so we can't rely on a named entity to tell us the nature of the info provided

Definite clause grammars and logic programming

- What is a DCG?
- DCGs define sentence structure in terms of formal substructures
- Entire query represents *terminal rule head*
- Particular info we need from query represents *non-terminals*

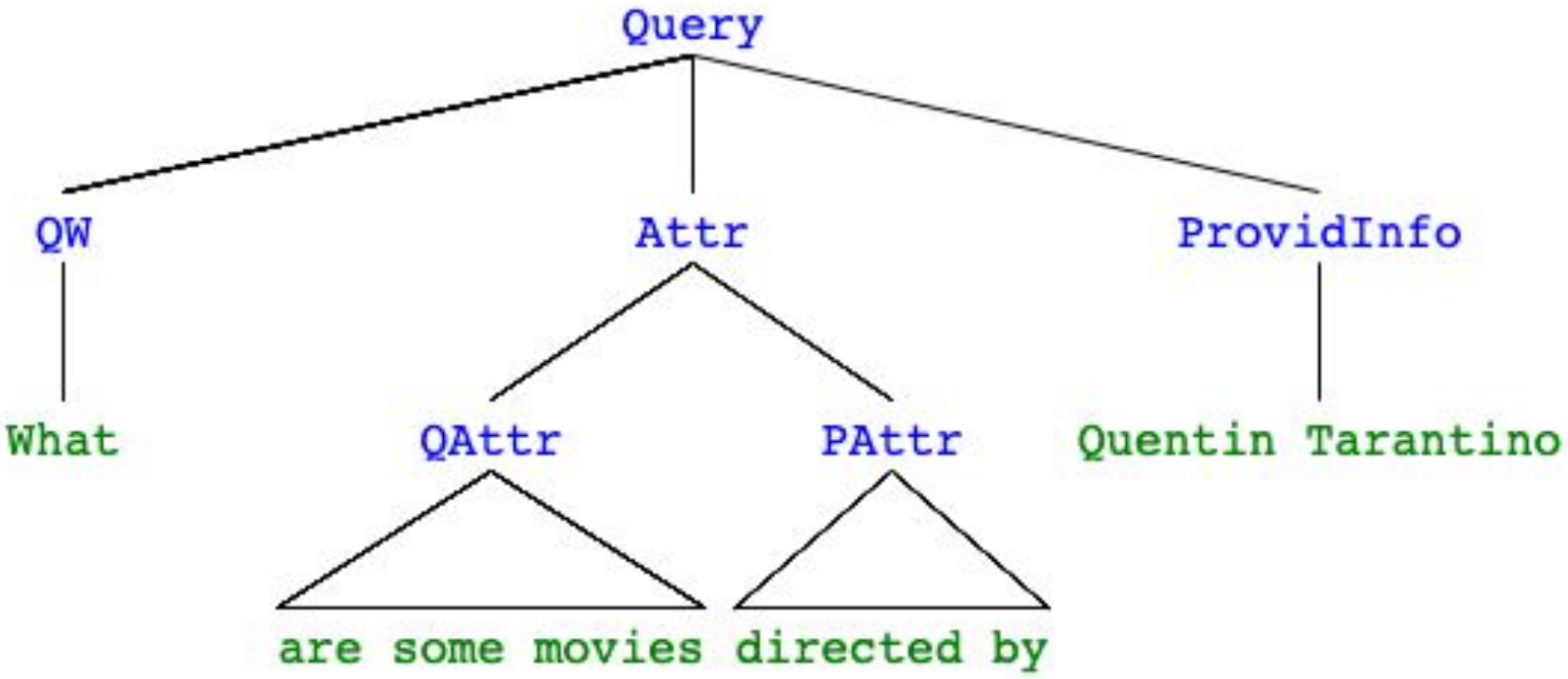


```
query -->
    question_phrase &
    attribute_phrase &
    attribute_phrase &
    word_from_db
```

```
question_phrase -->
    preposition &
    question_word
```

From which movie was the actor Kyle MacLachlan?

Definite clause grammars: an example



Definite clause grammars: an example

```
query_pattern1(QW, QAttr, FromDB, PAttr)) :-
    question_phrase(QW) & attribute_phrase(QAttr) &
    named_entity(ProvidedInfo) & attribute_phrase(PAttr)
```

What are some movies that Kyle MacLachlan has starred in?

Definite clause grammars: an example

```
query pattern1(QW, QAttr, FromDB, PAttr)) :-
    question_phrase(QW) & attribute_phrase(QAttr) &
    named_entity(ProvidedInfo) & attribute_phrase(PAttr)

attribute phrase(Phrase, Attr) :-
    evaluate(appendstring(attribute(Attr), Suffix), Phrase)
```

What are some movies that Kyle MacLachlan has starred in?

Definite clause grammars: an example

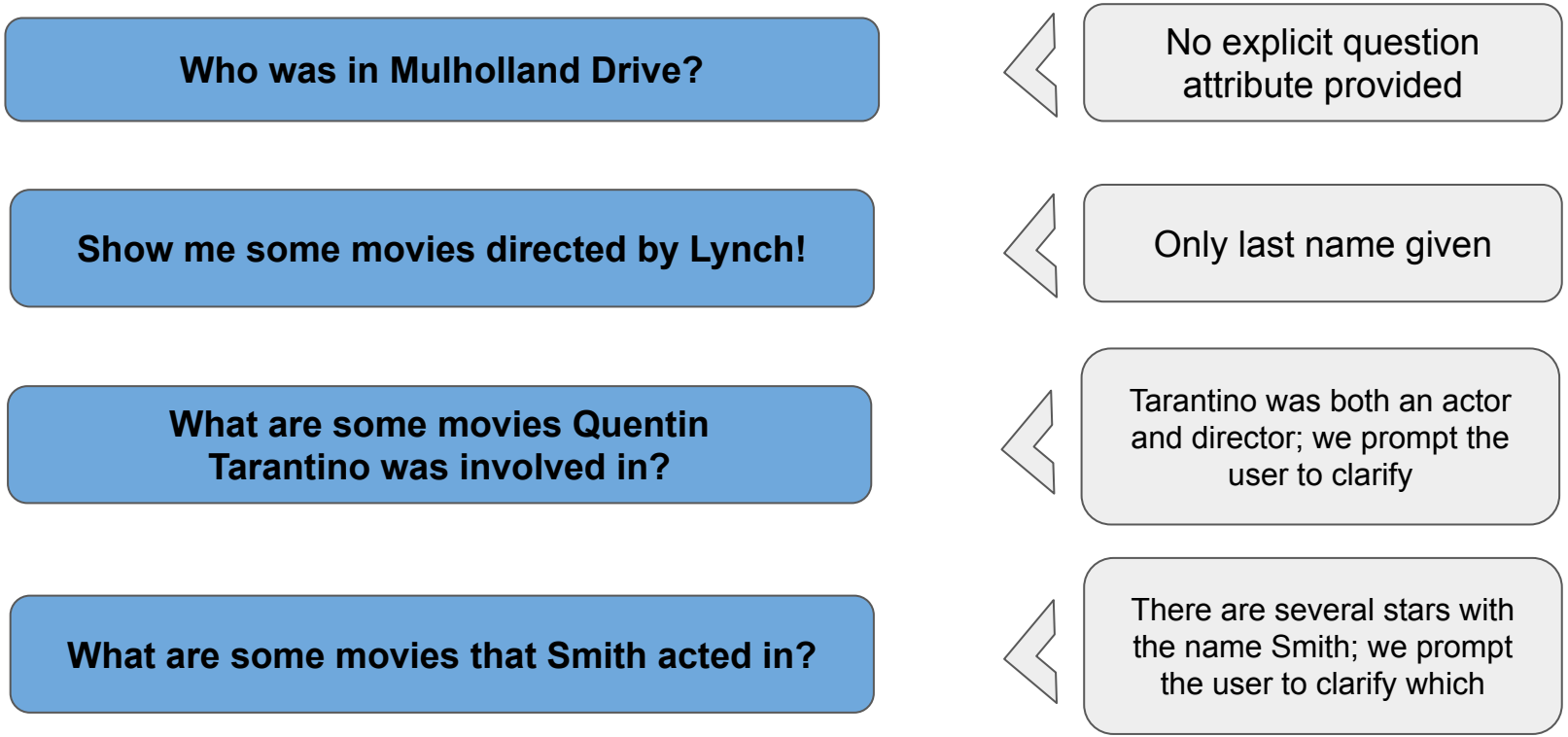
```
query_pattern1(QW, QAttr, FromDB, PAttr)) :-
    question_phrase(QW) & attribute_phrase(QAttr) &
    named_entity(ProvidedInfo) & attribute_phrase(PAttr)

attribute_phrase(Phrase, Attr) :-
    evaluate(appendstring(attribute(Attr), Suffix), Phrase)

attribute("star")
attribute("direct")
attribute("movie")
.
.
.
```

What are some movies that Kyle MacLachlan has starred in?

What sorts of queries do we support?



Conclusion: Why Logic Programming

Advantages:

- **Understandable**: reading our implementation of DCG rules structurally mirrors reading a query input itself
- **Predictive:** in our code it is visibly apparent what query formulations are supported and which are not
- **Easily generalizable**: our DCG can be applied to other scenarios with minimal revision
    - What about a music info chatbot?
    - Q&A chatbot for a website?

Conclusion: Challenges

- **Control flow** when engaging with user when non-state-based can become unwieldy with logic programming

- Naming subproblems for easy reference

Q & A