Logic: Resolution Proofs; Datalog

CPSC 322 – Logic 5

Textbook §5.2; 12.2
Lecture Overview

1 Recap

2 Resolution Proofs
Proofs

- A proof is a mechanically derivable demonstration that a formula logically follows from a knowledge base.
- Given a proof procedure, $KB \vdash g$ means $g$ can be derived from knowledge base $KB$.
- Recall $KB \models g$ means $g$ is true in all models of $KB$.

**Definition (soundness)**
A proof procedure is **sound** if $KB \vdash g$ implies $KB \models g$.

**Definition (completeness)**
A proof procedure is **complete** if $KB \models g$ implies $KB \vdash g$. 
Bottom-up proof procedure

$KB \vdash g$ if $g \subseteq C$ at the end of this procedure:

\[ C := \{\}; \]

repeat

\begin{align*}
\textbf{select} & \quad \text{clause } \textquoteright\textit{h }\leftarrow b_1 \land \ldots \land b_m\text{\"} \text{in } KB \text{ such that} \\
& \quad b_i \in C \text{ for all } i, \text{ and } h \notin C; \\
C & := C \cup \{h\}
\end{align*}

until no more clauses can be selected.
Soundness of bottom-up proof procedure

If $KB \vdash g$ then $KB \models g$.

- Suppose there is a $g$ such that $KB \vdash g$ and $KB \not\models g$.
- Let $h$ be the first atom added to $C$ that's not true in every model of $KB$.
- Suppose $h$ isn't true in model $I$ of $KB$.
- There must be a clause in $KB$ of form

$$h \leftarrow b_1 \land \ldots \land b_m$$

Each $b_i$ is true in $I$. $h$ is false in $I$. So this clause is false in $I$.

Therefore $I$ isn't a model of $KB$. Contradiction: thus no such $g$ exists.
Minimal Model

We can use proof procedure to find a model of $KB$.

- First, observe that the $C$ generated at the end of the bottom-up algorithm is a fixed point.
- Further applications of our rule of derivation will not change $C$.

**Definition (minimal model)**

Let the minimal model $I$ be the interpretation in which every element of the fixed point $C$ is true and every other atom is false.

**Claim:** $I$ is a model of $KB$. **Proof:**

- Assume that $I$ is not a model of $KB$. Then there must exist some clause $h \leftarrow b_1 \land \ldots \land b_m$ in $KB$ (having zero or more $b_i$’s) which is false in $I$.
- This can only occur when $h$ is false and each $b_i$ is true in $I$.
- If each $b_i$ belonged to $C$, we would have added $h$ to $C$ as well.
- Since $C$ is a fixed point, no such $I$ can exist.
If $KB \models g$ then $KB \vdash g$.

- Suppose $KB \models g$. Then $g$ is true in all models of $KB$.
- Thus $g$ is true in the minimal model.
- Thus $g$ is generated by the bottom up algorithm.
- Thus $KB \vdash g$. 

Completenss
Lecture Overview

1 Recap

2 Resolution Proofs
Top-down Ground Proof Procedure

Idea: search backward from a query to determine if it is a logical consequence of $KB$.

An answer clause is of the form:

$$yes \leftarrow a_1 \land a_2 \land \ldots \land a_m$$

The SLD Resolution of this answer clause on atom $a_i$ with the clause:

$$a_i \leftarrow b_1 \land \ldots \land b_p$$

is the answer clause

$$yes \leftarrow a_1 \land \ldots \land a_{i-1} \land b_1 \land \ldots \land b_p \land a_{i+1} \land \ldots \land a_m.$$
An answer is an answer clause with $m = 0$. That is, it is the answer clause $yes \leftarrow$.

A derivation of query “$q_1 \land \ldots \land q_k$” from $KB$ is a sequence of answer clauses $\gamma_0, \gamma_1, \ldots, \gamma_n$ such that

- $\gamma_0$ is the answer clause $yes \leftarrow q_1 \land \ldots \land q_k$,
- $\gamma_i$ is obtained by resolving $\gamma_{i-1}$ with a clause in $KB$, and
- $\gamma_n$ is an answer.
To solve the query \(?q_1 \land \ldots \land q_k\): 

\[ ac := \text{"yes } \leftarrow q_1 \land \ldots \land q_k\text{"} \]

**repeat**

- **select** atom \(a_i\) from the body of \(ac\);
- **choose** clause \(C\) from \(KB\) with \(a_i\) as head;
- replace \(a_i\) in the body of \(ac\) by the body of \(C\)

**until** \(ac\) is an answer.

Recall:

- **Don’t-care nondeterminism** If one selection doesn’t lead to a solution, there is no point trying other alternatives. **select**
- **Don’t-know nondeterminism** If one choice doesn’t lead to a solution, other choices may. **choose**
Example: successful derivation

\[ a \leftarrow b \land c. \quad a \leftarrow e \land f. \quad b \leftarrow f \land k. \]
\[ c \leftarrow e. \quad d \leftarrow k. \quad e. \]
\[ f \leftarrow j \land e. \quad f \leftarrow c. \quad j \leftarrow c. \]

Query: \(?a\)

\[ \gamma_0 : \quad yes \leftarrow a \]
\[ \gamma_1 : \quad yes \leftarrow e \land f \]
\[ \gamma_2 : \quad yes \leftarrow f \]
\[ \gamma_3 : \quad yes \leftarrow c \]
\[ \gamma_4 : \quad yes \leftarrow e \]
\[ \gamma_5 : \quad yes \leftarrow \]
Example: failing derivation

\[ a \leftarrow b \land c. \quad a \leftarrow e \land f. \quad b \leftarrow f \land k. \]
\[ c \leftarrow e. \quad d \leftarrow k. \quad e. \]
\[ f \leftarrow j \land e. \quad f \leftarrow c. \quad j \leftarrow c. \]

Query: \(?a\)

\[ \gamma_0 : \, yes \leftarrow a \]
\[ \gamma_1 : \, yes \leftarrow b \land c \]
\[ \gamma_2 : \, yes \leftarrow f \land k \land c \]
\[ \gamma_3 : \, yes \leftarrow c \land k \land c \]
\[ \gamma_4 : \, yes \leftarrow e \land k \land c \]
\[ \gamma_5 : \, yes \leftarrow k \land c \]
Search Graph

\begin{align*}
a & \leftarrow b \land c. \\
a & \leftarrow g. \\
b & \leftarrow j. \\
b & \leftarrow k. \\
d & \leftarrow m. \\
d & \leftarrow p. \\
f & \leftarrow m. \\
g & \leftarrow f. \\
h & \leftarrow m. \\
p & \leftarrow m. \\
?a \land d
\end{align*}

Logic: Resolution Proofs; Datalog