Logical Inference

## 1 Inference rules

- How to find what follows from a logic program ($M_P$=?)
  - Find all models of $P$ (intractable)
  - $M_P = \{A | A\ is\ a\ ground\ atom, P \models A\}$
- Use inference rules: procedures $I$ for transforming one formula (program, clause) $P$ into another one $Q$, denoted $P \vdash_I Q$.
- $I$ is *correct and complete*, if $P \vdash_I P \;\Leftrightarrow\; P_1 \models P_2$.

## 2 Inference rules – examples

- And-introduction: $\frac{A\ B}{A \wedge B}$
- And-elimination: $\frac{A \wedge B}{A}\ \frac{A \wedge B}{B}$
- Modus ponens: $\frac{A\ A \rightarrow B}{B}$
- Clause subsumption:
  - $C \succeq D$, if there exists a substitution $\theta$, such that $C\theta \subseteq D$.
  - Correctness: If $C \succeq D$, then $C \models D$.
  - Incompleteness: $C \models D \not\Rightarrow C \succeq D$ (see counter example)
- Resolution: correct and complete inference rule

## 3 Clause subsumption – examples

**Prolog:**

$C$=parent(X,Y) :- son(Y,X)
(or as a set: $C = \{parent(X, Y), \neg son(Y, X)\}$)

$D$= parent(john,bob) :- son(bob,john),male(john)
(or $D = \{parent(john, bob), \neg son(bob, john), \neg male(john)\}$)

$C$ subsumes $D$ ($\theta = \{X/john, Y/bob\}$), because $C\theta \subseteq D$.

**Incompleteness of clause subsumption (counter example):**

$C = p(f(X)) \leftarrow p(X)$

$D = p(f(f(X))) \leftarrow p(X)$

$C \models D$, but $C \not\succeq D$.

## 4 Resolution rule

- $C_1$ and $C_2$ are clauses *standartized apart* (not sharing variables).
- There exist $L_1 \in C_1$ and $L_2 \in C_2$ that can be made complementary by applying an *mgu*, i.e. $L_1\mu = \neg L_2\mu$.
- Then $C = (C_1 \backslash \{L_1\} \cup C_2 \backslash \{L_2\})\mu$ is called *resolvent* of $C_1$ and $C_2$.
- Most importantly, $C$ follows from $C_1$ and $C_2$, i.e. $C_1 \wedge C_2 \models C$.

**Example 1 (Prolog):**

$$C_1 = grandfather(X, Y) : -parent(X, Z), father(Z, Y)$$
$$C_2 = parent(A, B) : -father(A, B)$$

$$\mu = \{A/X, B/Z\}, \;\; parent(A, B)\mu = \neg parent(X, Z)$$

Then, the resolvent of $C_1$ and $C_2$ is:

$$C = grandfather(X, Y) : -father(X, Z), father(Z, Y)$$

**Example 2 (self-resolution, recursion):**

$$C_1 = p(f(X)) \leftarrow p(X)$$
$$C_2 = p(f(Y)) \leftarrow p(Y)$$

$$\mu = \{Y/f(X)\}, \;\; \neg p(Y)\mu = p(f(X))$$

Then, the resolvent of $C_1$ and $C_2$ is: $C = p(f(f(X))) \leftarrow p(X)$

## 5 Resolution procedure (Robinson, 65)

- Refutation procedure (proving unsatisfiability). If $S$ is a set of clauses $R^n(S)$ is defined as follows:
  - $R^0(S) = S$
  - $R^i(S) = R^{i-1}(S) \cup \{C | C \text{ is a resolvent of clauses from } R^{i-1}(S)\}$.
- *Refutation completeness*: $S$ is unsatisfiable if and only if there exists $n$, such that $\Box \in R^n(S)$.
- *Resolution strategies*: How to pick the clauses to resolve? Differ in efficiency and completeness.
- *Linear resolution with selection function (SLD-resolution).*
  - Prolog inference: checks if an atom $A$ logically follows from a program $P$, i.e. if $P \wedge \neg A$ is unsatisfiable. $A$ is first resolved with a clause from $P$, then at each step the obtained resolvent is resolved with a clause from $P$.
  - *Completeness of the SLD-resolution.* If $P \models A$ then the SLD-refutation tree of $P \wedge \neg A$ has a path leading to the empty clause $\Box$.
- *Incompleteness of Prolog* (depth-first search strategy). Example:

$p(a, b)$
$p(c, b)$
$p(X, Y) \leftarrow p(X, Z), p(Z, Y)$
$p(X, Y) \leftarrow p(Y, X)$
$\leftarrow p(a, c)$

**Prolog**

Question: Given logic program $P$ and atom $A$, find if $A$ logically follows from $P$.

```
grandfather(X,Y) :- parent(X,Z), father(Z,Y).
parent(A,B) :- father(A,B).
father(john,bill).
father(bill,ann).
father(bill,mary).
```

Is John a grandfather of Ann?

```
?- grandfather(john,ann).
yes
?-
```

Who are the grandchildren of John?

```
?- grandfather(john,X).
X=ann;
X=mary;
no
?-
```

## 6 Theorem proving (automated reasoning)

- Extending Prolog to FOL
- Ignoring control (the Prolog procedural semantics)