CS 520: Introduction to Artificial Intelligence

Prof. Louis Steinberg

Lecture 12:

Logic Programming and Prolog

Review

- **Inference in FOL**
  - **Inference in FOL is semi-decidable**
  - **Algorithms:**
    - **Resolution Refutation**
    - **Backchaining Generalized Modus Ponens**

Review

Normal forms:

- **Conjunctive normal form (cnf)**

  $(A_{1,1} \vee A_{1,2} \vee \ldots) \wedge (A_{2,1} \vee A_{2,2} \vee \ldots) \wedge \ldots$
  where $A_{i,j}$ is an atomic predicate or a negated atomic predicate

- **Implicative normal form (inf)**

  $((A_{1,1} \wedge A_{1,2} \wedge \ldots) \Rightarrow (B_{1,1} \vee B_{1,2} \vee \ldots))$
  $\wedge ((A_{1,1} \wedge A_{1,2} \wedge \ldots) \Rightarrow (B_{1,1} \vee B_{1,2} \vee \ldots))$
  $\wedge \ldots$
  where $A_{i,j}$, $B_{i,j}$ are atomic predicates
  – One B/clause: Horn clause

Conjunctive Normal Form

- **Eliminate implications**
- **Move ¬ inwards**
- **Standardize variables**
- **Move quantifiers left**
- **Skolemize ∃**
- **Distribute ∧**

(For INF: gather negative literals and convert to implication)

$A(x) \wedge B(x) \wedge \ldots \wedge C(x) \Rightarrow D(x)$

– **RR: conjunctive normal form (cnf)**

$A(x) \vee B(x) \vee \ldots \vee D(x)$

Initial expression

$\forall x[(\exists y\ l(x, y)) \Rightarrow \sim(\forall y\ h(x, y) \vee h(y, x))]$

Remove if-then

$\forall x[\sim(\exists y\ l(x, y)) \vee \sim(\forall y\ h(x, y) \vee h(y, x))]$

Move ~ inwards

$\forall x[(\forall y \sim l(x, y)) \vee \sim(\forall y\ h(x, y) \vee h(y, x))]$

$\forall x[(\forall y \sim l(x, y)) \vee \exists y \sim(h(x, y) \vee h(y, x))]$

$\forall x[(\forall y \sim l(x, y)) \vee \exists y\ (\sim h(x, y) \wedge \sim h(y, x))]$

Standardize variables

$\forall x[(\forall y \sim l(x, y)) \vee \exists z\ (\sim h(x, z) \wedge \sim h(z, x))]$

To cnf

$\forall x[(\forall y \sim l(x, y)) \vee \exists z\ (\sim h(x, z) \wedge \sim h(z, x))]$

Move quantifiers left

$\forall x\ \forall y\ \exists z\ [(\sim l(x, y)) \vee (\sim h(x, z) \wedge \sim h(z, x))]$

Skolemize

$\forall x\ \forall y\ [(\sim l(x, y)) \vee (\sim h(x, s(x, y)) \wedge \sim h(s(x, y), x))]$

Drop $\forall$

$(\sim l(x, y)) \vee (\sim h(x, s(x, y)) \wedge \sim h(s(x, y), x))$

Distribute

$(\sim l(x, y) \vee \sim h(x, s(x, y))) \wedge (\sim l(x, y) \vee \sim h(s(x, y), x))$

GMP

If there is a substitution θ such that

$$\text{SUBST}(\theta, P_i) = \text{SUBST}(\theta, P_i')$$

$$\text{SUBST}(\theta, Q) = \text{SUBST}(\theta, Q')$$

Then

$$\frac{P'_1, \ldots, P'_n, \quad P_1 \wedge \ldots \wedge P_n \Rightarrow Q}{Q'}$$

Summary of Unification

- **Unify-Sentences(A, B)**
  - **Unify corresponding atomic sentences**
- **Unify-Atomic Sentences (A, B)**
  - **If either A or B is a variable, add substitution var / other term to θ and apply to A and B**
  - **If predicates differ, fail**
  - **Unify corresponding terms**
- **Unify-terms**
  - **If both A and B are constants**
    if A=B succeed else fail
  - **If both A and B are function terms**
    - **If not same function, fail**
    - **Unify-terms on corresponding arguments**

Backchaining

- **Goal: prove A**
  - **Find a clause B ^ C ^ … => A'**
    - **Where A and A' unify; substitute B->B', C->C'**
  - **Subgoal: prove B'**
    - **Recur**
  - **Subgoal: prove C'**
    - **Recur**
- **If fail (can't find a clause), go back to last choice and try another.**

An Example With a Choice

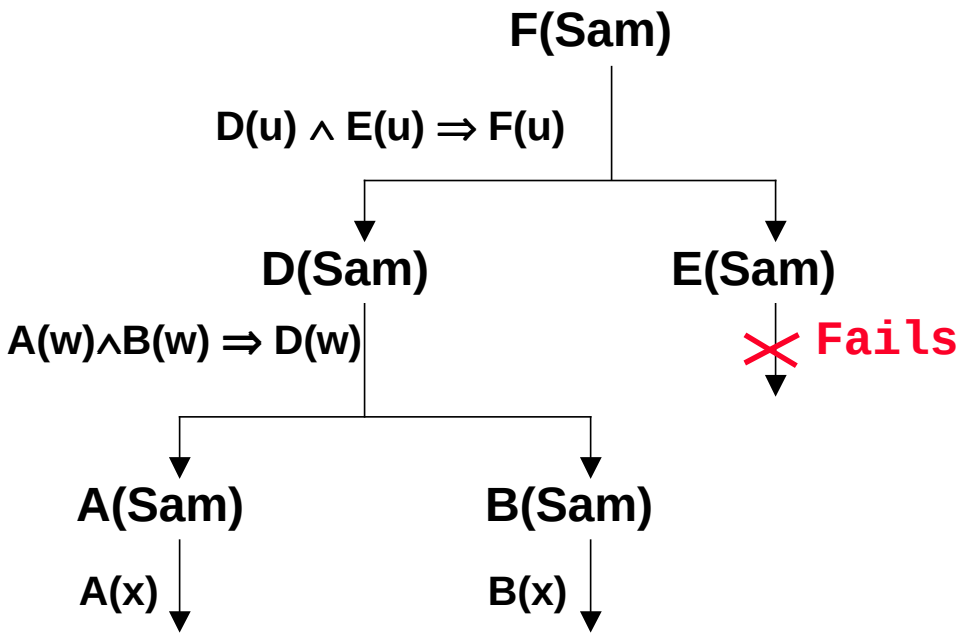
- **KB:**

  A(x)
  B(y)
  C(z)
  $A(w) \wedge B(w) \Rightarrow D(w)$
  $D(u) \wedge E(u) \Rightarrow F(u)$
  $D(v) \wedge C(v) \Rightarrow F(v)$

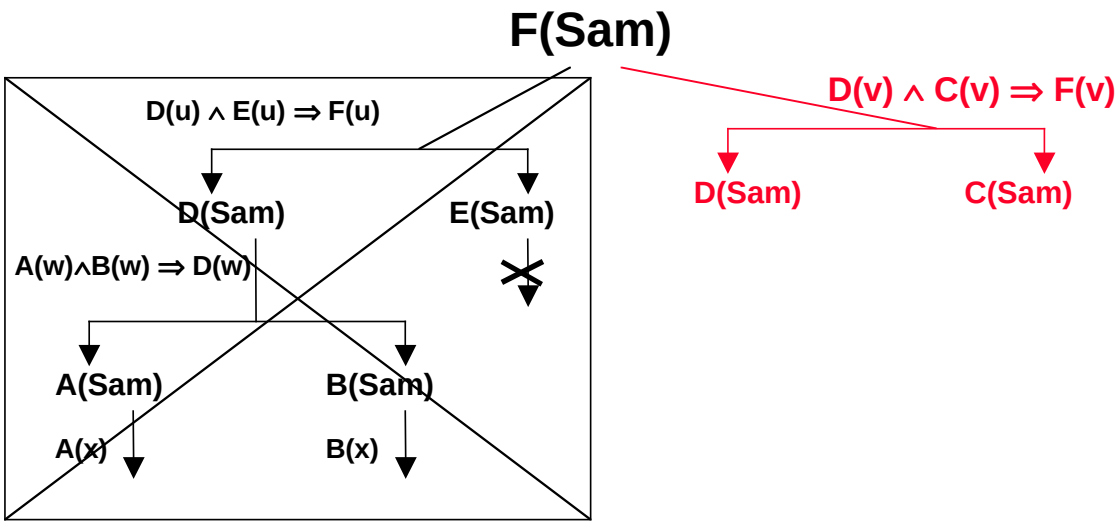
- **Query:**

  F(Sam)

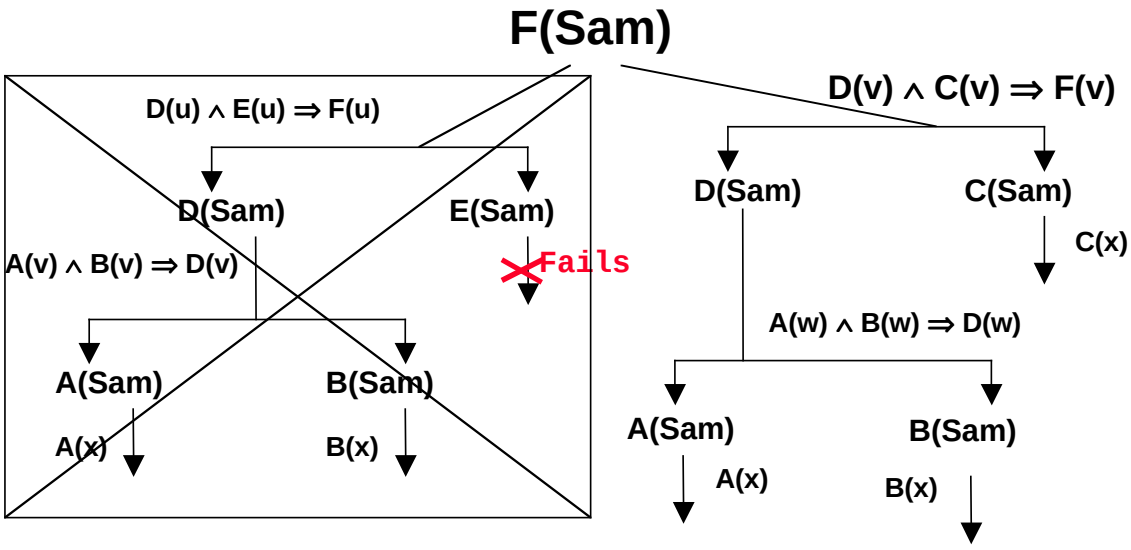
Searching for a Proof Tree



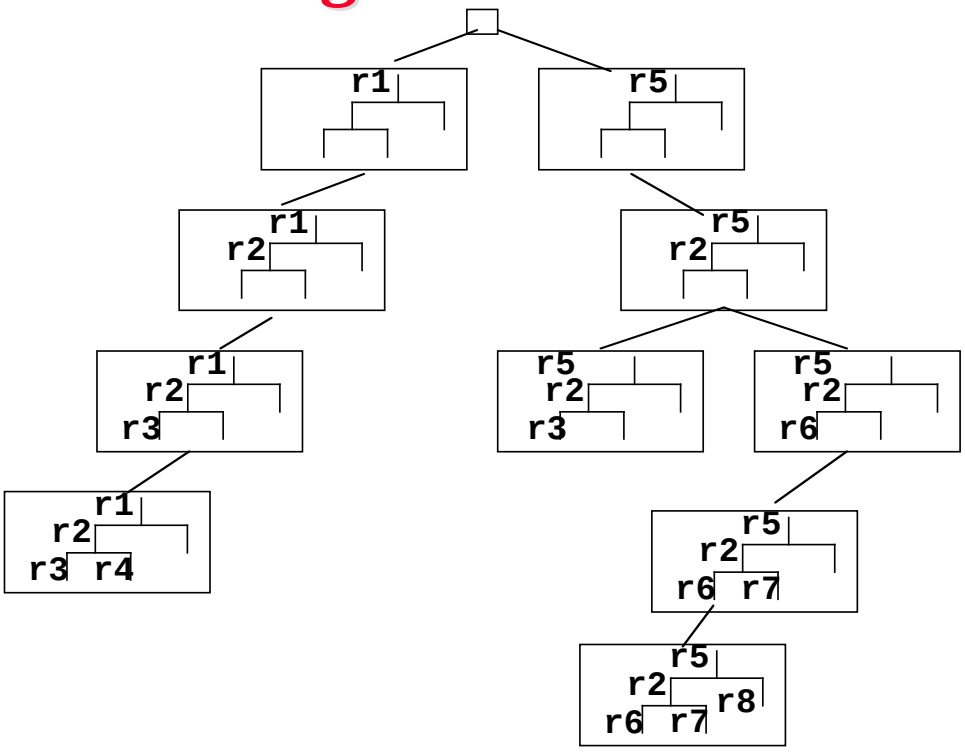
Find a Choice Point



And Continue



Searching a Tree of Search Trees



Logic Programing

- **Goal:**
  - **Tell computer the facts**
  - **Ask a question**
  - **Computer uses facts as needed to answer the question.**
  - **“logic + control = algorithm”**
- **Reality:**
  - **This is very hard**
    - **Sorted (x, y)<= permutation(x, y) ^ inorder(y)**

Prolog

- **Fallback goal**
  - **Separate logic from control**
    - **Control implicit in way logic formulated**
    - **Additional explicit notation for control**