CSE 571 - FALL 99. HANDOUT 3. VERSION 0.3

Chitta Baral

Department of Computer Science and Engineering
Arizona State University
Tempe, AZ 85287-5406 USA
chitta@asu.edu

http://www.public.asu.edu/~cbaral/cse571-f99/

November 9, 1999

LOGIC, KNOWLEDGE REPRESENTATION AND MODEL BASED PLANNING

First-Order Logic : Syntax

- Alphabet: object, function and predicate constants, logical connectives, punctuation signs, and variables for the objects of the language.
- Terms: built from (object) variables, object constants and function constants.
  - object constants and variables are terms.
  - if f is a function symbol of arity n and $t_1, \ldots, t_n$ are terms then $f(t_1, \ldots, t_n)$ is a term.
  - Nothing else is a term.

  Terms not containing variables are sometimes called *ground* terms.

- Atomic formulas (atoms) have the form $p(t_1, \ldots, t_n)$ (or $t_1 = t_2$).

  Atomic formulas not containing variables (that is, atomic sentences) are sometimes called ground atoms. (Equalities $t_1 = t_2$ are often not counted among ground atoms.)

- Formulas are built from atomic formulas using propositional connectives $\wedge$ (&), $\vee$, $\neg$, $\supset$, $\equiv$, and quantifiers $\forall$, $\exists$.

  – Variables in the scope of a quantifier are called *bound*.

  – Other variables are called *free*. (For instance, variable X in $\forall\, X(p(X) \wedge p(Y))$ is bound while variable Y is free.

- A *sentence* is a formula without free variables. (Formulas with free variables express properties and relations, rather than assertions.)

- By a **first-order theory** we mean a collection of sentences in a first-order language.

An example

- Story: There a two boxes – box1 and box2. A box can be in the bedroom, in the kitchen, or in the closet, but there is not enough room in the closet for both boxes.
- Its representation in first-order theory:
  - Alphabet
    - object constants: box1, box2
    - predicate constants: in_bedrm, in_kitchen, in_closet
    - ground terms: box1, box2 (since no function constants)
    - ground atoms:
  - Representation as a first-order theory: initial attempt
    - S_1: $\forall X (in\_bdrm(X) \vee in\_kitchen(X) \vee in\_closet(X))$
    - S_2: $\forall X \forall Y (in\_closet(X) \wedge in\_closet(Y) \supset X = Y)$

Semantics: Interpretation, models and logical entailment

- An interpretation (or a structure) M consists of:
  - a nonempty set $|M|$, called its universe
  - and the interpretations of all object, function and predicate constants.
    * an object constant represents an element of $|M|$,
    * an n-ary function constant represents a mapping of $|M|^n$ into $|M|$,
    * an n-ary predicate constant represents a subset of $|M|^n$.
  - For any constant K, we represent the object representing K in M by $M[[K]]$.
- An interpretation determines which sentences are true and which are false.

- Example: an interpretation M of the first-order language described before can be defined as follows: $|M| = \{b1^*, b2^*\}$, where $b1^*$ and $b2^*$ are two distinct objects; $box1$ is interpreted by $b1^*$, $box2$ by $b2^*$, in_bdrm by the empty set, $in\_kitchen$ by $\{b1^*\}$, in_closet by $\{b2^*\}$.
- Sentences S_1,S_2 are true in this interpretation.
- A **model** of a first-order theory is an interpretation M such that all axioms of the theory are true in M.
  - For instance, we can say that the interpretation defined above is a model of the theory with the axioms S_1,S_2.
  - This theory has also "unintended" models, in which the universe consists of only one element or includes more than two elements. Such models can be eliminated by adding two axioms:
    * S_3. $box1 \neq box2$ ("uniqueness of names").
    * S_4. $\forall\ X\ (X = box1 \lor X = box2)$ ("domain closure").
- A formula F is **logically entailed** by a theory T ($T \models F$) if F is true in all models of T.

Example1

- Knowledge: Mike is a professor. John is a professor.
- First order representation: $T_1 = \{prof(mike), prof(john)\}$
- Question: Does $T_1 \models prof(jack)$
- Question: Does $T_1 \models \neg prof(jack)$
- Add to our theory: Mike and John are the *only* professors in the department. (Close world assumption through Clark's completion.)
- First order representation:
  $T_2 = \{\forall X (prof(X) \Leftrightarrow (X = mike \vee X = john))\}$
- Question: Does $T_2 \models prof(jack)$
- Question: Does $T_2 \models \neg prof(jack)$
- Add **unique name axioms (UNA)** to our theory.

- $T_3 = T_2 \cup \{mike \neq john, john \neq jack, jack \neq mike\}$
- Question: Does $T_3 \models prof(jack)$
- Question: Does $T_3 \models \neg prof(jack)$
- Q: According to T_3 is there a person who is not a professor.
- Does $T_3 \models \exists X \neg prof(X)$?
- Does $T_3 \models \forall X prof(X)$?
- Add **Domain Closure Axiom (DCA)** to our theory.
- $T_4 = T_3 \cup \{\forall X (X = mike \vee X = john)\}$.
- Does $T_4 \models \exists X \neg prof(X)$?
- Does $T_4 \models \forall X prof(X)$?

Universal theories and Herbrand Models

- Universal theory: It is a theory consisting of universally quantified boolean formulas (boolean formula is a formula not containing quantifiers).
- Herbrand Model:

  An interpretation M of a first-order language $\mathcal{L}$ is called *Herbrand* if

  1. $|M|$ is the set of all ground terms of $\mathcal{L}$;
  2. Object constants denote themselves..
  3. A function symbol f denotes the function f^* which maps a vector $t_1, \ldots, t_n$ of ground terms into a term $f(t_1, \ldots, t_n)$.

  A Herbrand interpretation satisfying theory T is called a **Herbrand model** of T.

- Importance of Herbrand models of universal theories:

  If T is a universal theory then

  1. T has a model (i.e. is satisfiable or consistent) iff T has a Herbrand model.
  2. For any ground boolean formula Q, $T \models Q$ iff Q is true in all Herbrand models of T.

- Herbrand models are much smaller and easier to write.

  – Underlying language of a theory

  – Herbrand Universe

  – Herbrand Base

  – Herbrand Interpretations and Herbrand Models

- Why Resolution theorem proving works?

  – Result 1: For any formula F there is a set of clasues $cl(F)$ such that F has a model iff $cl(F)$ has a model.

– Note: F is not equivalent to $cl(F)$. (i.e., they do not have the same models.)

  Example: Let F be $\exists X p(X)$. $cl(F)$ would be $p(c)$.

– Result 2: Since $cl(F)$ is a universal theory, we have: $cl(F)$ has a model iff $cl(F)$ has a Herbrand model.

Limitations of First-order logic

- Transitive Closure
  - Ancestors and parents
    - $\forall X, Y, Z(anc(X,Y) \equiv par(X,Y) \vee anc(X,Z) \wedge anc(Z,Y))$
      $\forall X, Y\, par(X,Y) \equiv X = a \wedge Y = b \vee X = b \wedge Y = c$
    - $\{par(a,b), par(a,c), anc(a,b), anc(b,c), anc(a,c)\}$ is a model.
    - But,
      $\{par(a,b), par(a,c), anc(a,a), anc(a,b), anc(b,c), anc(a,c)\}$ is also a model.
  - Paths and edges
    - $\forall X, Y, Z(path(X,Y) \equiv edge(X,Y) \vee edge(X,Z) \wedge path(Z,Y))$
      $\forall X, Y\, par(X,Y) \equiv X = a \wedge Y = b \vee X = c \wedge Y = c$
    - $\{edge(a,b), edge(c,c), path(a,b), path(c,c)\}$ is a model.

    * But, $\{edge(a,b), edge(c,c), path(a,b), path(c,c), path(c,d)\}$ is also a model.

- Elaboration tolerance; Normative statements
  - Birds normally fly. Penguins are birds. Penguins do not fly. Tweety is the only Penguin we know. Koel is a bird.
  - Representation:
  - Now we want to add: Ostriches are also birds that do not fly. Skippy is the only Ostrich we know.
  - Representation:

Circumscription

- Transitive Closure
  - $par(X, Y) \supset anc(X, Y)$
  - $par(X, Z) \wedge anc(Z, Y) \supset anc(X, Y)$
  - Minimize anc
- Normative statements
  - $bird(X) \wedge \neg ab(X) \supset fly(X)$
  - $penguin(X) \supset bird(X)$
  - $penguin(X) \supset ab(X)$
  - Minimize ab
  - $ostritch(X) \supset ab(X)$
  - $ostritch(X) \supset bird(X)$

- McCarthy's original formulation:

  The circumscription of P in $A(P)$ is the sentence schema:

  $\forall \Phi[[A(\Phi) \wedge \forall \bar{x}.(\Phi(\bar{x}) \supset P(\bar{x}))] \supset (P(\bar{x}) \supset \Phi(\bar{x}))]$

- Parallel Circumscription: Syntactic Characterization

  - P : a tuple of predicate constants
  - Z: a tuple of function and/or predicate constants disjoint with P.
  - $A(P, Z)$ – a sentence
  - Circumscription of P in $A(P, Z)$ with variable Z is the sentence
    $A(P, Z) \wedge \neg \exists pz(A(p, z) \wedge p < P)$
    This is denoted as $Circ(A(P, Z); P; Z))$ (or simply $Circ(A; P; Z)$)

- Model theoretic characterization:

  - For any two interpretations M_1 and M_2 we write $M_1 \leq^{P;Z} M_2$ if
    1. $|M_1| = |M_2|$,
    2. $M_1[[K]] = M_2[[K]]$ for every constant K not in P, Z, and

3. $M_1[[P_i]] \subset M_2[[P_i]]$ for every P_i in P.

- Proposition: An interpretation M is a model of $Circ(A; P; Z)$ iff M is minimal in the class of models of A with respect to $\leq^{P;Z}$.
- Examples:
  - $Circ(P(a); P)$ is equivalent to $\forall x(P(x) \equiv x = a)$
  - $Circ(P(a) \wedge P(b); P)$ is equivalent to $\forall x(P(x) \equiv x = a \vee x = b)$
  - $Circ(\forall x(Q(x) \supset P(x)) \wedge \forall x(R(x) \supset P(x)); P)$ is equivalent to $\forall x(P(x) \equiv Q(x) \vee R(x))$
  - Flying Example:
    * minimize: ab
    * vary: fly
    * The theory: A(ab,fly)
    * The circumscriptive theory: $Circ(A(ab, fly); ab; fly)$.

Reasoning about actions

- S_0: Initial situation
- $Res(a, s)$: Situation reached by executing action a in situation s
- $Holds(f, s)$: The fluent f holds in situation s.
- An initial attempt at formulating plan verification:
  - Notation: $H(F, S)$ denotes $Holds(P, S)$ and $H(\neg F, S)$ denotes $\neg Holds(F, S)$.)
  - $Holds(F, S_0)$, for each **initially** $F \in D$
  - $\neg Holds(F, S_0)$, for each **initially** $\neg F \in D$
  - $H(P_1, s) \wedge \ldots \wedge H(P_n, s) \supset H(F, Res(A, s))$, for each A **causes** F **if** $P_1, \ldots, P_n$.
  - Minimize $Holds$.
  - What about inertia?

– Revised formulation

* $Holds(F, S_0)$, for each **initially** $F \in D$
* $\neg Holds(F, S_0)$, for each **initially** $\neg F \in D$
* $H(f, s) \wedge \neg ab(f, a, s) \supset H(f, Res(a, s))$
* $H(P_1, s) \wedge \ldots \wedge H(P_n, s) \supset H(F, Res(A, s))$, for each A **causes** F **if** $P_1, \ldots, P_n$.
* $H(P_1, s) \wedge \ldots \wedge H(P_n, s) \supset ab(F, A, S)$, for each A **causes** F **if** $P_1, \ldots, P_n$.
* Minimize: ab
* Q: Minimize $Holds$? Vary $Holds$? Keep it constant?

- The Yale Shooting problem
  - $Holds(Alive, S_0)$
  - $Holds(Loaded, Res(Load, s))$
  - $Holds(Loaded, S) \supset$
    $Ab(Alive, Shoot, S) \wedge \neg Holds(Alive, Res(Shoot, S))$
  - $Holds(f, s) \wedge \neg Ab(f, a, s) \supset Holds(f, Res(a, s))$
  - Minimize Ab
  - Q: Is $Holds(Alive, Res(Shoot, Res(Wait, Res(Load, S_0))))$? or Is $\neg Holds(Alive, Res(Shoot, Res(Wait, Res(Load, S_0))))$?

Lifschitz's formulation:

$$
\left\{
\begin{array}{l}
UNA[actions], UNA[fluents], UNA[situations] \\
DCA[actions], DCA[fluents], DCA[situations] \\
(1) \neg Causes^{+}(a, f, s) \wedge \neg Causes^{-}(a, f, s) \supset [H(f, s) \equiv H(f, Res(a, s))] \\
(2) Causes^{+}(a, f, s) \supset Holds(f, Res(a, s)) \\
(3) Causes^{-}(a, f, s) \supset \neg Holds(f, Res(a, s)) \\
(4) Initially(f) \equiv Holds(f, S_0) \\
(5) Initially(F) \qquad \text{(for each } \textbf{initially} \;\; F \in D) \\
(6) \neg Initially(F) \qquad \text{(for each } \textbf{initially} \;\; \neg F \in D) \\
\;\{ \; \min Causes^{+(-)} \; : \\
\quad (7) H(P_1, s) \wedge \ldots \wedge H(P_n, s) \supset Causes^{+(-)}(A, F, s) \\
\qquad\qquad \text{(for each } A \textbf{ causes } (\neg)F \textbf{ if } P_1, \ldots, P_n \in D) \\
\;\}
\end{array}
\right.
$$

Here, $\{min\ P_1, P_2 : T\}$ means $Circ(T; P_1, P_2;)$.

Reiter's formulation

- Notations
  - $Holds(\neg F, s)$ stands for $\neg Holds(F, s)$.
  - For a conjunction of literals $\varrho = P_1 \wedge \ldots \wedge P_n$, we write $Holds(\varrho, s)$ to denote $\wedge_{i=1} Holds(P_i, s)$.
  - For each fluent name F and action A in D , let

$$A_F^+ = \{A \textbf{ causes } F \textbf{ if } \varrho \mid A \textbf{ causes } F \textbf{ if } \varrho \in D\},$$

$$A_F^- = \{A \textbf{ causes } \neg F \textbf{ if } \varrho \mid A \textbf{ causes } \neg F \textbf{ if } \varrho \in D\}, \text{ and}$$

$$S_A = \{ \textbf{executable } A \textbf{ if } \varrho \mid \textbf{ executable } A \textbf{ if } \varrho \in D\}.$$

– We define:

$$\gamma_F^+(A,s) \stackrel{def}{\equiv} \vee_{A \textbf{ causes } F \textbf{ if } \varrho \in A_F^+} Holds(\varrho, s),$$

$$\gamma_F^-(A,s) \stackrel{def}{\equiv} \vee_{A \textbf{ causes} \neg F \textbf{ if } \varrho \in A_F^-} Holds(\varrho, s), \text{ and}$$

$$Poss(A,s) \stackrel{def}{\equiv} \vee_{\textbf{executable } A \textbf{ if } \varrho \in S_A} Holds(\varrho, s).$$

- The theory

1. $Holds(F, S_0)$ if we have **initially** F
2. The successor state axiom for an ordinary fluent F is given by

$$Poss(A,s) \supset [Holds(F, do(A,s)) \equiv \quad (1)$$
$$\gamma_F^+(A,s) \vee (Holds(F,s) \wedge \neg \gamma_F^-(A,s))] \quad (2)$$

3. The domain closure assumption (DCA) for fluents:

$$\bigvee_{F \in \mathbf{F}} f = F.$$

4. The domain closure assumption (DCA) for actions:

$$\bigvee_{A \in \mathbf{A}} a = A.$$

5. The unique name assumption (UNA) for fluents:

$$\bigwedge_{F_1, F_2 \in \mathbf{F} \;\; F_1, F_2 \text{ distinct}} F_1 \neq F_2.$$

6. The unique name assumption (UNA) for actions:

$$\bigwedge_{A_1, A_2 \in \mathbf{A} \;\; A_1, A_2 \text{ distinct}} A_1 \neq A_2.$$

Planning as satisfiability: basic idea

- Simpler questions:
  - Is there a plan of length n?
  - Is there a plan of length less than n?
- In contrast Situation calculus can be used to find plan of any length, or verify plans of any length.
- Theorem proving vs Model Checking
  - In theorem proving $T \models q$ if q is true in all models of T.
  - In model checking each model of T gives us useful information.
- Planning as satisfiability: Each model of our theory gives us a plan.
- Since we have simpler questions we can use propositional logic.

- Propositional logic: finite number of propositions, no variables, no quantifiers.
- High level syntax for propositional logic: Schema variables; quantifier range over a finite domain; Can be compiled into propositional formulas.

Planning as satisfiability: Blocks world 1 (ECAI 92)

- Conventions:
  - $on(x, y, i)$: $on(x, y)$ is true in time point i.
  - $clear(x, i)$: $clear(x)$ is true in time point i
  - $move(x, y, z, i)$: The action $move(x, y, z)$ occurs in between time point i and $i + 1$.
- Initial State: $on(A, B, 1)$, $on(B, Table, 1)$, $clear(A, 1)$.
- Final State: $on(B, A, N)$ (assuming we are interested to find out if there is a plan of length N.)
- Recalling the Domain Description
  - $move(x, y, z)$ **causes** $on(x, z), \neg clear(z), \neg on(x, y), clear(y)$
  - **executable** $move(x, y, z)$ **if** $on(x, y), clear(x), clear(z)$

- Action occurrence implies its effect hold

  $\forall x, y, z, i.\ move(x, y, z, i) \supset$
  $on(x, z, i+1) \wedge \neg clear(z, i+1) \wedge \neg on(x, y, i+1) \wedge clear(y, i+1)$

- Action occurrence implies its executability conditions hold

  $\forall x, y, z, i.\ move(x, y, z, i) \supset clear(x, i) \wedge clear(z, i) \wedge on(x, y, i)$

- Only one action occurs at a time

  $\forall x, x', y, y', z, z', i.\ (x \neq x' \vee y \neq y' \vee z \neq z') \supset$
  $\neg move(x, y, z, i) \vee \neg move(x', y', z', i)$

- Some action occurs at any time: $\forall i < N.\ \exists x, y, z.\ move(x, y, z, i)$

- Frame axioms

  - $\forall x, y, i.\ on(x, y, i) \wedge \neg on(x, y, i+1) \supset \exists z.\ move(x, y, z, i)$
  - $\forall x, z, i.\ \neg on(x, z, i) \wedge on(x, z, i+1) \supset \exists y.\ move(x, y, z, i)$
  - $\forall z, i.\ clear(z, i) \wedge \neg clear(z, i+1) \supset \exists x, y.\ move(x, y, z, i)$
  - $\forall y, i.\ \neg clear(y, i) \wedge clear(y, i+1) \supset \exists x, z.\ move(x, y, z, i)$

- Size of the instantiated theory
  - c = Number of elements in the largest type.
  - d = maximum depth of quantifier nesting in any schema
  - k = number of literals in the longest schema
  - Upper bound on the length of the instantiated theory = $O(kc^d)$
  - Lesson: Reduce quantifier depth

Blocks World 2: breaking predicates

- Basic idea: Replace predicates that take 3 or more arguments by several predicates that take no more than 2 arguments.
- Example: (From Kautz and Selman ECAI 92)
  - Replace $move(x, y, z, i)$ by $object(x, i)$, $source(y, i)$, $dest(z, i)$.
  - Only one actions occur a time can now be specified as:
    * $\forall i, x_1, x_2.\ x_1 \neq x_2 \supset \neg object(x_1, i) \vee \neg object(x_2, i)$
    * $\forall i, y_1, y_2.\ y_1 \neq y_2 \supset \neg source(y_1, i) \vee \neg source(y_2, i)$
    * $\forall i, z_1, z_2.\ z_1 \neq z_2 \supset \neg dest(z_1, i) \vee \neg dest(z_2, i)$

Parallelized encodings

- Main intuition:
  - So far we require that only one action occur at each time step.
  - By removing this requirement (i.e., allowing several actions at each time step) we can decrease the number of time steps. (note: i ranges from 1 to number of steps.)
  - Several actions occuring at one time means that they can be serailized in any order with the same effect.
- Graph-plan based encodings
  - Solution is a subgraph of the planning graph that contains all the facts in the initial and goal layers, and contains no two operators in the same layer that conflict.
  - the initial state holds at layer 1, and the goals hold at the highest layer

– (No rule of the kind where action occurrence implies its effect hold.)

– a fact is true in level i implies it is either true in level $i - i$ or one of the actions which has that fact in its add effect occurs in level $i - 1$.

– action occurrence implies its executability conditions hold

– (No constraint that says only one action occurs at a time point.)

– Conflicting actions are mutually exclusive. (Only one level of conflict is enough, as additional exclusiveness constraints that are obtained by propagating effects of mutually exclusive actions in Graphplan, are logically implied by the SAT encoding.)