# Heuristics in Planning; Uncertainty in Planning

**Manuela M. Veloso**

Carnegie Mellon University
Computer Science Department

*15-780 Graduate AI – Spring 2013*

Readings:
- Chapter 10, Russell & Norvig

## Heuristics in Planning

- State-space planning
  - Goal selection
    - Not domain-independent
  - Action selection
    - Number of preconditions true in state
    - Total number of preconditions
- Plan-space planning
  - No need for goal selection
  - Action with less open (unlinked) conditions

## How to Generate Heuristics

- Domain-Specific
  - Program in (or ***learn***) heuristics specifically for that domain
- Domain Analysis
  - Preprocess domain to generate meta-control knowledge
- On-Line
  - Solve a ***relaxed*** form of the problem
  - Use as estimate for original problem



## Heuristics

- Assume complete *subgoal* ***independence***
- Assume ***no*** *or limited negative* interactions
- Identify *landmarks*

## Sources of Uncertainty

- Initial State
  - Unknown predicates
- Action Models
  - Non-deterministic effects
- Sensor Noise
  - Partially hidden state



## Uncertainty

- State uncertainty
  - Possible worlds
- Action uncertainty
  - Preconditions
  - Effects
    - Conditional effects

## Conditional Effects

- Action Sprinkle-grass
- Pre:
  - turned-on water
- Effects:
  - wet grass
  - If object on grass
    - Then wet object
- Result of applying action
- Use of conditional effects as goals



## Conditional Planning

- When applying an action, need “to sense” to verify conditions
- Determine state in which action will be applied
  - State-space planners
  - Plan-state planners
  - Other planners

## Conformant Planning

- Create a plan that achieves the goal *in spite of uncertainty* in state and action
- Examples:
  - Tray tilting



## Conformant Graphplan (Smith & Weld, 1998)

- Extends Graphplan
  - Conditional effects
  - Multiple possible worlds
    - Uncertainty represented using disjunction
- Basic Approach:
  - Create parallel plan graphs for each possible world
  - Search backwards through each plan graph in parallel
  - ***Confront*** undesired effects in one possible world caused by choosing action in another possible world

## BDD Representation

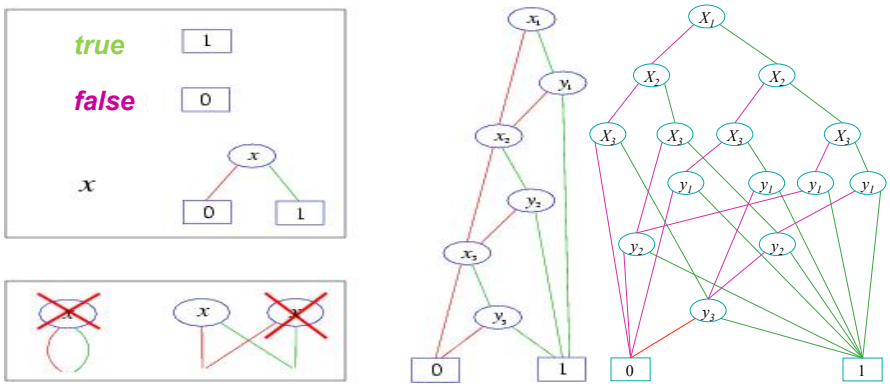
- Compact representation of Boolean formulae
  - Invented by Dean Randy Bryant (1985)
- A BDD is a DAG with:
  - Terminal nodes labeled 0 or 1
  - A set of variable nodes with two edges: *low* (false) and *high* (true)
  - A linear ordering of variables.
    - Bad variable ordering leads to exponential size
  - Reductions:
    - Uniqueness of nodes associated to the same variable
    - No redundant tests



## Summary

- Heuristics
- Uncertainty
  - Conditional planning
  - Conformant planning
  - Nondeterministic planning
- Next
  - Probabilistic representations
  - Planning under probabilistic uncertainty

## BDD Example




$(x_1 \wedge y_1) \vee (x_2 \wedge y_2) \vee (x_3 \wedge y_3)$



## Representing Actions

- Representing the Transition Relation
  - Boolean formula describing relationship between variables in current and next state: T($a$, $x$, $x'$)

Action 0

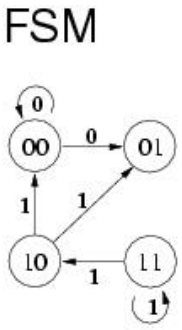
Pre: $\neg x_1 \wedge \neg x_2$

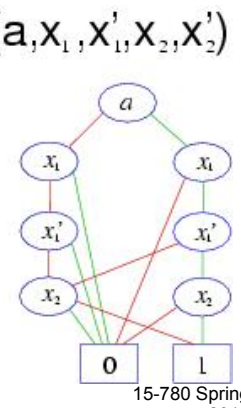
Eff: $\neg x'_1$

Action 1

Pre: $x_1$

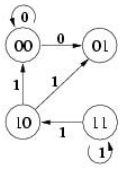
Eff: $(\neg x_2 \Rightarrow \neg x'_1)$
$(x_2 \Rightarrow x'_1)$



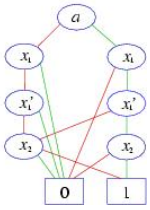



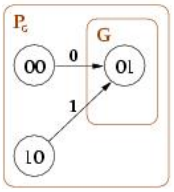
# Pre-Image Computation

FSM

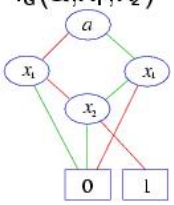



$T(a,x_1,x_1',x_2,x_2')$




$P_G(a,x_1,x_2)$




Pre-Image$(a, x)$ = $\exists x'$. $T(a, x, x') \wedge V(x')$

$P_G(a,x_1,x_2) = \exists\, x_1', x_2'.\ T(a, x_1, x_2, x_1', x_2') \wedge (\neg x_1' \wedge x_2')$

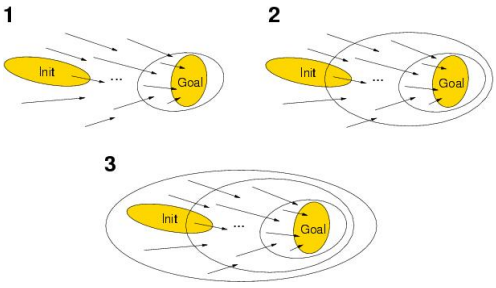
Post-Image$(a, x)$ = $\exists x$ . $T(a, x, x') \wedge V(x)$



# BDD-Based Planning

- Key Idea: Solve for ***Sets*** of States

**function** *Plan*(*T*,*I*,*G*)
  *U* :=0; *V* := *G*
  **while** *I* ⊄ *V*
    $U_c$ := *PreComp*(*T*,*V*)
    **if** $U_c$ = 0 **then**
      **return** *failure*
    **else**
      *U* := *U* ∪ $U_c$
      *V* := *V* ∪ *states*($U_c$)
  **return** *U*



# BDD-Based Planning

- Different "Guarantees" for Generated Policies
  - Depends on strictness of *PreComp* rule