Chapter 10
Control Rules in Planning

Dana S. Nau
University of Maryland

5:01 PM    April 4, 2012
Motivation

- Often, planning can be done much more efficiently if we have domain-specific information

- Example:
  - classical planning is EXPSPACE-complete
  - block-stacking can be done in time $O(n^3)$

- But we don’t want to have to write a new domain-specific planning system for each problem!

- **Domain-configurable** planning algorithm
  - Domain-independent search engine (usually a forward state-space search)
  - Input includes domain-specific information that allows us to avoid a brute-force search
    » Prevent the planner from visiting unpromising states
Motivation (Continued)

- If we’re at some state $s$ in a state space, sometimes a domain-specific test can tell us that
  - $s$ doesn’t lead to a solution, or
  - for any solution below $s$, there’s a better solution along some other path
- In such cases we can to prune $s$ immediately
- Rather than writing the domain-dependent test as low-level computer code, we’d prefer to talk directly about the planning domain
- One approach:
  - Write logical formulas giving conditions that states must satisfy; prune states that don’t satisfy the formulas
- Presentation similar to the chapter, but not identical
  - Based partly on TLPlan [Bacchus & Kabanza 2000]

Abstract-search($u$)

if Terminal($u$) then return($u$)

$u \leftarrow$ Refine($u$) ;; refinement step

$B \leftarrow$ Branch($u$) ;; branching step

$B' \leftarrow$ Prune($B$) ;; pruning step

if $B' = \emptyset$ then return(failure)
nondeterministically choose $v \in B'$
return(Abstract-search($v$))
end
Quick Review of First Order Logic

- First Order Logic (FOL):
  - constant symbols, function symbols, predicate symbols
  - logical connectives ($\lor$, $\land$, $\neg$, $\Rightarrow$, $\Leftrightarrow$), quantifiers ($\forall$, $\exists$), punctuation
  - Syntax for formulas and sentences:
    - $on(A,B) \land on(B,C)$
    - $\exists x \ on(x,A)$
    - $\forall x \ (ontable(x) \Rightarrow clear(x))$

- First Order Theory $T$:
  - “Logical” axioms and inference rules – encode logical reasoning in general
  - Additional “nonlogical” axioms – talk about a particular domain
  - Theorems: produced by applying the axioms and rules of inference

- Model: set of objects, functions, relations that the symbols refer to
  - For our purposes, a model is some state of the world $s$
  - In order for $s$ to be a model, all theorems of $T$ must be true in $s$
  - $s \models on(A,B)$ read “$s$ satisfies $on(A,B)$” or “$s$ entails $on(A,B)$”
    » means that $on(A,B)$ is true in the state $s$
Linear Temporal Logic

● Modal logic: FOL plus *modal operators*
  to express concepts that would be difficult to express within FOL

● Linear Temporal Logic (LTL):
  
  ◆ Purpose: to express a limited notion of time
    » An infinite sequence \( \langle 0, 1, 2, \ldots \rangle \) of time instants
    » An infinite sequence \( M = \langle s_0, s_1, \ldots \rangle \) of states of the world
  
  ◆ Modal operators to refer to the states in which formulas are true:
    - \( \Diamond f \) - *next f* - \( f \) holds in the next state, e.g., \( \Diamond on(A,B) \)
    - \( \lozenge f \) - *eventually f* - \( f \) either holds now or in some future state
    - \( \Box f \) - *always f* - \( f \) holds now and in all future states
    - \( f_1 \cup f_2 \) - *f_1 until f_2* - \( f_2 \) either holds now or in some future state,
      and \( f_1 \) holds until then
  
  ◆ Propositional constant symbols TRUE and FALSE
Linear Temporal Logic (continued)

- Quantifiers cause problems with computability
  - Suppose \( f(x) \) is true for infinitely many values of \( x \)
  - Problem evaluating truth of \( \forall x \ f(x) \) and \( \exists x \ f(x) \)

- Bounded quantifiers
  - Let \( g(x) \) be such that \( \{ x : g(x) \} \) is finite and easily computed
    \[
    \forall [x : g(x)] \ f(x)
    \]
    - means \( \forall x (g(x) \Rightarrow f(x)) \)
    - expands into \( f(x_1) \land f(x_2) \land \ldots \land f(x_n) \)

    \[
    \exists [x : g(x)] \ f(x)
    \]
    - means \( \exists x (g(x) \land f(x)) \)
    - expands into \( f(x_1) \lor f(x_2) \lor \ldots \lor f(x_n) \)
Models for LTL

- A model is a triple \((M, s_i, v)\)
  - \(M = \langle s_0, s_1, \ldots \rangle\) is a sequence of states
  - \(s_i\) is the \(i\)'th state in \(M\),
  - \(v\) is a variable assignment function
    - a substitution that maps all variables into constants

- To say that \(v(f)\) is true in \(s_i\), write \((M, s_i, v) \models f\)

- Always require that
  - \((M, s_i, v) \models TRUE\)
  - \((M, s_i, v) \models \neg FALSE\)

- For planning, need to augment LTL to refer to goal states
  - Include a GOAL operator such that \(\text{GOAL}(f)\) means \(f\) is true in every goal state
  - \(((M, s_i, V), g) \models \text{GOAL}(f)\) iff \((M, s_i, V) \models f\) for every \(s_i \in g\)
Examples

- Suppose $M=\langle s_0, s_1, \ldots \rangle$

  $$(M,s_0,v) \models \Box\Box on(A,B)$$ means $A$ is on $B$ in $s_2$

- Abbreviations:

  $$(M,s_0) \models \Box\Box on(A,B)$$ no free variables, so $v$ is irrelevant:

  $$M \models \Box\Box on(A,B)$$ if we omit the state, it defaults to $s_0$

- Equivalently,

  $$(M,s_2,v) \models on(A,B)$$ same meaning with no modal operators

  $$s_2 \models on(A,B)$$ same thing in ordinary FOL

- $M \models \Box \neg\mathit{holding}(C)$
  - in every state in $M$, we aren’t holding $C$

- $M \models \Box (on(B,C) \Rightarrow (on(B,C) \cup on(A,B)))$
  - whenever we enter a state in which $B$ is on $C$, $B$ remains on $C$ until $A$ is on $B$. 
TLPlan

- Basic idea: forward search, using LTL for pruning tests
- Let $s_0$ be the initial state, and $f_0$ be the initial LTL control formula
- Current recursive call includes current state $s$, and current control formula $f$
- Let $P$ be the path that TLPlan followed to get to $s$
  - The proposed model $M$ is $P$ plus some (not yet determined) states after $s$
  - If $f$ evaluates to FALSE in $s$, no $M$ that starts with $P$ can satisfy $f_0 \Rightarrow$ backtrack
- Otherwise, consider the applicable actions, to see if one of them can produce an acceptable “next state” for $M$
  - Compute a formula $f^+$ that must be true in the next state
    - $f^+$ is called the progression of $f$ through $s$
  - If $f^+ = \text{FALSE}$, then there are no acceptable successors of $s$ => backtrack
  - Otherwise, produce $s^+$ by applying an action to $s$, and call TLPlan recursively

Procedure TLPlan ($s$, $f$, $g$, $\pi$)

```plaintext
if $f = \text{FALSE}$ then return failure
if $s$ satisfies $g$ then return $\pi$
$f^+ \leftarrow \text{Progress} (f, s)$
if $f^+ = \text{FALSE}$ then return failure
$A \leftarrow \{\text{actions applicable to } s\}$
if $A$ is empty then return failure
nondeterministically choose $a \in A$
$s^+ \leftarrow \gamma(s, a)$
return TLPlan ($s^+$, $f^+$, $g$, $\pi . a$)
```
Classical Operators

unstack\((x,y)\)
- Precond: on\((x,y)\), clear\((x)\), handempty
- Effects: \(\neg on(x,y)\), \(\neg clear(x)\), \(\neg handempty\), holding\((x)\), clear\((y)\)

stack\((x,y)\)
- Precond: holding\((x)\), clear\((y)\)
- Effects: \(\neg holding(x)\), \(\neg clear(y)\), on\((x,y)\), clear\((x)\), handempty

pickup\((x)\)
- Precond: ontable\((x)\), clear\((x)\), handempty
- Effects: \(\neg ontable(x)\), \(\neg clear(x)\), \(\neg handempty\), holding\((x)\)

putdown\((x)\)
- Precond: holding\((x)\)
- Effects: \(\neg holding(x)\), ontable\((x)\), clear\((x)\), handempty
Supporting Axioms

- Want to define conditions under which a stack of blocks will never need to be moved.
- If $x$ is the top of a stack of blocks, then we want $goodtower(x)$ to hold if:
  - $x$ doesn’t need to be anywhere else.
  - None of the blocks below $x$ need to be anywhere else.
- Axioms to support this:
  - $goodtower(x) \iff clear(x) \land \neg GOAL(holding(x)) \land goodtowerbelow(x)$
  - $goodtowerbelow(x) \iff$
    
    $\begin{array}{l}
    ontalle(x) \land \neg \exists[y:GOAL(on(x,y))] \\
    \lor \exists[y:on(x,y)] \{ \neg GOAL(ontalle(x)) \land \neg GOAL(holding(y)) \\
    \land \neg GOAL(clear(y)) \land \forall[z:GOAL(on(z,y))](z = y) \\
    \land \forall[z:GOAL(on(z,y))](z = x) \land goodtowerbelow(y) \}
    \end{array}$
  - $badtower(x) \iff clear(x) \land \neg goodtower(x)$
Blocks World Example (continued)

Three different control formulas:

(1) Every goodtower must always remain a goodtower:

$$\Box \left( \forall [x: \text{clear}(x)] \ \text{goodtower}(x) \Rightarrow \Diamond (\text{clear}(x) \lor \exists [y: \text{on}(y, x)] \ \text{goodtower}(y)) \right)$$

(2) Like (1), but also says never to put anything onto a badtower:

$$\Box \left( \forall [x: \text{clear}(x)] \ \text{goodtower}(x) \Rightarrow \Diamond (\text{clear}(x) \lor \exists [y: \text{on}(y, x)] \ \text{goodtower}(y) \land \text{badtower}(x) \Rightarrow \Diamond (\neg \exists [y: \text{on}(y, x)]) \right)$$

(3) Like (2), but also says never to pick up a block from the table unless you can put it onto a goodtower:

$$\Box \left( \forall [x: \text{clear}(x)] \ \text{goodtower}(x) \Rightarrow \Diamond (\text{clear}(x) \lor \exists [y: \text{on}(y, x)] \ \text{goodtower}(y)) \land \text{badtower}(x) \Rightarrow \Diamond (\neg \exists [y: \text{on}(y, x)]) \land (\text{ontable}(x) \land \exists [y: \text{GOAL(on}(x, y))] \neg \text{goodtower}(y) \Rightarrow \Diamond (\neg \text{holding}(x))) \right)$$
Outline of How TLPlan Works

- Recall that TLPlan’s input includes a current state $s$, and a control formula $f$ written in LTL
  - How can TLPlan determine whether there exists a sequence of states $M$ beginning with $s$, such that $M$ satisfies $f$?

- We can compute a formula $f^+$ such that for every sequence $M = \langle s, s^+, s^{++}, \ldots \rangle$,
  - $M$ satisfies $f$ iff $M^+ = \langle s^+, s^{++}, \ldots \rangle$ satisfies $f^+$
- $f^+$ is called the progression of $f$ through $s$

- If $f^+ = \text{FALSE}$ then there is no $M^+$ that satisfies $f^+$
  - Thus there’s no $M$ that begins with $s$ and satisfies $f$, so TLPlan can backtrack
- Otherwise, need to determine whether there is an $M^+$ that satisfies $f^+$
  - For every action $a$ applicable to $s$,
    » Let $s^+ = \gamma(s, a)$, and call TLPlan recursively on $f^+$ and $s^+$

- Next: how to compute $f^+$
Procedure Progress\((f,s)\)

**Case:**

1. \(f\) contains no temporal ops: \(f^+ := \text{TRUE if } s |= f, \text{ FALSE otherwise}\)
2. \(f = f_1 \land f_2\): \(f^+ := \text{Progress}(f_1, s) \land \text{Progress}(f_2, s)\)
3. \(f = f_1 \lor f_2\): \(f^+ := \text{Progress}(f_1, s) \lor \text{Progress}(f_2, s)\)
4. \(f = \neg f_1\): \(f^+ := \neg \text{Progress}(f_1, s)\)
5. \(f = \bigcirc f_1\): \(f^+ := f_1\)
6. \(f = \Diamond f_1\): \(f^+ := \text{Progress}(f_1, s) \lor f\)
7. \(f = \Box f_1\): \(f^+ := \text{Progress}(f_1, s) \land f\)
8. \(f = f_1 \cup f_2\): \(f^+ := \text{Progress}(f_2, s) \lor (\text{Progress}(f_1, s) \land f)\)
9. \(f = \forall [x:g(x)] h(x)\): \(f^+ := \text{Progress}(h_1, s) \land \ldots \land \text{Progress}(h_n, s)\)
10. \(f = \exists [x:g(x)] h(x)\): \(f^+ := \text{Progress}(h_1, s) \lor \ldots \lor \text{Progress}(h_n, s)\)

where \(h_i\) is \(h\) with \(x\) replaced by the \(i\)'th element of \(\{x : s |= g(x)\}\)

**Next, simplify** \(f^+\) **and return it**

**Boolean simplification rules:**

1. \([\text{FALSE} \land \phi] \leftrightarrow \phi \land \text{FALSE}\)
2. \([\text{TRUE} \land \phi] \leftrightarrow \phi\)
3. \(\neg \text{TRUE} \leftrightarrow \text{FALSE}\)
4. \(\neg \text{FALSE} \leftrightarrow \text{TRUE}\).
Two Examples of $\Diamond$

- Suppose $f = \Diamond on(a,b)$
  - $f^+ = on(a,b)$
  - $s^+$ is acceptable iff $on(a,b)$ is true in $s^+$

- Suppose $f = \Diamond \Diamond on(a,b)$
  - $f^+ = \Diamond on(a,b)$
  - $s^+$ is acceptable iff $\Diamond on(a,b)$ is true in $s^+$
    
    » iff $on(a,b)$ is true in $s^{++}$

**Case:**

1. $f$ contains no temporal ops: $f^+ := \text{TRUE if } s \models f, \text{ FALSE otherwise}$
2. $f = f_1 \land f_2$ : $f^+ := \text{Progress}(f_1, s) \land \text{Progress}(f_2, s)$
3. $f = f_1 \lor f_2$ : $f^+ := \text{Progress}(f_1, s) \lor \text{Progress}(f_2, s)$
4. $f = \neg f_1$ : $f^+ := \neg \text{Progress}(f_1, s)$
5. $f = \Diamond f_1$ : $f^+ := f_1$
6. $f = \Diamond f_1$ : $f^+ := \text{Progress}(f_1, s) \lor f$
7. $f = \Box f_1$ : $f^+ := \text{Progress}(f_1, s) \land f$
8. $f = f_1 \cup f_2$ : $f^+ := \text{Progress}(f_2, s) \lor (\text{Progress}(f_1, s) \land f)$
9. $f = \forall [x; g(x)] h(x)$ : $f^+ := \text{Progress}(h_1, s) \land \cdots \land \text{Progress}(h_n, s)$
10. $f = \exists [x; g(x)] h(x)$ : $f^+ := \text{Progress}(h_1, s) \lor \cdots \lor \text{Progress}(h_n, s)$
Example of $\land$

- Suppose $f = \text{on}(a,b) \land \Diamond \text{on}(b,c)$
  - $f^+ = \text{Progress}(\text{on}(a,b), s) \land \text{Progress}(\Diamond \text{on}(b,c), s)$
  - $\text{Progress}(\text{on}(a,b), s)$
    - $= \text{TRUE}$ if $\text{on}(a,b)$ is true in $s$, else $\text{FALSE}$
  - $\text{Progress}(\Diamond \text{on}(b,c), s) = \text{on}(b,c)$

- If $\text{on}(a,b)$ is true in $s$, then $f^+ = \text{on}(b,c)$
  - i.e., $\text{on}(b,c)$ must be true in $s^+$

- Otherwise, $f^+ = \text{FALSE}$
  - i.e., there is no acceptable $s^+$

Case:

1. $f$ contains no temporal ops: $f^+ := \text{TRUE}$ if $s \models f$, $\text{FALSE}$ otherwise

2. $f = f_1 \land f_2$
   : $f^+ := \text{Progress}(f_1, s) \land \text{Progress}(f_2, s)$

3. $f = f_1 \lor f_2$
   : $f^+ := \text{Progress}(f_1, s) \lor \text{Progress}(f_2, s)$

4. $f = \neg f_1$
   : $f^+ := \neg \text{Progress}(f_1, s)$

5. $f = \Diamond f_1$
   : $f^+ := f_1$

6. $f = \Box f_1$
   : $f^+ := \text{Progress}(f_1, s) \lor f$

7. $f = f_1 \lor f_2$
   : $f^+ := \text{Progress}(f_2, s) \lor (\text{Progress}(f_1, s) \land f)$

8. $f = \forall [x:g(x)] h(x)$
   : $f^+ := \text{Progress}(h_1, s) \land \ldots \land \text{Progress}(h_n, s)$

9. $f = \exists [x:g(x)] h(x)$
   : $f^+ := \text{Progress}(h_1, s) \lor \ldots \lor \text{Progress}(h_n, s)$
Example of □

- Suppose \( f = \square \text{on}(a,b) \)
  - \( f^+ = \text{Progress}(\text{on}(a,b), s) \land \square \text{on}(a,b) \)

- If \( \text{on}(a,b) \) is true in \( s \), then
  - \( f^+ = \text{TRUE} \land \square \text{on}(a,b) = \square \text{on}(a,b) = f \)
  - i.e., \( \text{on}(a,b) \) must be true in \( s^+, s^{++}, s^{+++}, \ldots \)

- If \( \text{on}(a,b) \) is false in \( s \), then
  - \( f^+ = \text{FALSE} \land \square \text{on}(a,b) = \text{FALSE} \)
  - There is no acceptable \( s^+ \)

Case:

1. \( f \) contains no temporal ops: \( f^+ := \text{TRUE} \) if \( s \models f \), FALSE otherwise
2. \( f = f_1 \land f_2 \) : \( f^+ := \text{Progress}(f_1, s) \land \text{Progress}(f_2, s) \)
3. \( f = f_1 \lor f_2 \) : \( f^+ := \text{Progress}(f_1, s) \lor \text{Progress}(f_2, s) \)
4. \( f = \neg f_1 \) : \( f^+ := \neg \text{Progress}(f_1, s) \)
5. \( f = \bigcirc f_1 \) : \( f^+ := f_1 \)
6. \( f = \bigdiamond f_1 \) : \( f^+ := \text{Progress}(f_1, s) \lor f \)
7. \( f = \square f_1 \) : \( f^+ := \text{Progress}(f_1, s) \land f \)
8. \( f = f_1 \lor f_2 \) : \( f^+ := \text{Progress}(f_2, s) \lor (\text{Progress}(f_1, s) \land f) \)
9. \( f = \forall x : g(x) \ h(x) \) : \( f^+ := \text{Progress}(h_1, s) \land \ldots \land \text{Progress}(h_n, s) \)
10. \( f = \exists x : g(x) \ h(x) \) : \( f^+ := \text{Progress}(h_1, s) \lor \ldots \lor \text{Progress}(h_n, s) \)
Example of $\bigcup$

- Suppose $f = \text{on}(a,b) \cup \text{on}(c,d)$
  - $f^+ = \text{Progress}(\text{on}(c,d), s) \lor (\text{Progress}(\text{on}(a,b), s) \land f)$

- If $\text{on}(c,d)$ is true in $s$, then $\text{Progress}(\text{on}(c,d), s) = \text{TRUE}$
  - $f^+ = \text{TRUE}$, so any $s^+$ is acceptable

- If $\text{on}(c,d)$ is false in $s$, then $\text{Progress}(\text{on}(c,d), s) = \text{FALSE}$
  - $f^+ = \text{Progress}(\text{on}(a,b), s) \land f$
  - If $\text{on}(a,b)$ is false in $s$ then $f^+ = \text{FALSE}$: no $s^+$ is acceptable
  - If $\text{on}(a,b)$ is true in $s$ then $f^+ = f$

Case:

1. $f$ contains no temporal ops: $f^+ := \text{TRUE}$ if $s \models f$, FALSE otherwise
2. $f = f_1 \land f_2$ : $f^+ := \text{Progress}(f_1, s) \land \text{Progress}(f_2, s)$
3. $f = f_1 \lor f_2$ : $f^+ := \text{Progress}(f_1, s) \lor \text{Progress}(f_2, s)$
4. $f = \neg f_1$ : $f^+ := \neg \text{Progress}(f_1, s)$
5. $f = \bigcirc f_1$ : $f^+ := f_1$
6. $f = \Diamond f_1$ : $f^+ := \text{Progress}(f_1, s) \lor f$
7. $f = \square f_1$ : $f^+ := \text{Progress}(f_1, s) \land f$
8. $f = f_1 \cup f_2$ : $f^+ := \text{Progress}(f_2, s) \lor (\text{Progress}(f_1, s) \land f)$
9. $f = \forall [x : g(x)] \ h(x)$ : $f^+ := \text{Progress}(h_1, s) \land \ldots \land \text{Progress}(h_n, s)$
10. $f = \exists [x : g(x)] \ h(x)$ : $f^+ := \text{Progress}(h_1, s) \lor \ldots \lor \text{Progress}(h_n, s)$
Another Example

- Suppose \( f = \Box (on(a,b) \Rightarrow \Diamond clear(a)) \)
  - \( f^+ = \text{Progress}[on(a,b) \Rightarrow \Diamond clear(a), s] \land f \)
    \[ = (\neg \text{Progress}[on(a,b)] \lor clear(a)) \land f \]
  - If \( on(a,b) \) is false in \( s \), then \( f^+ = (\text{TRUE} \lor clear(a)) \land f = f \)
    » So \( s^+ \) must satisfy \( f \)
  - If \( on(a,b) \) is true in \( s \), then \( f^+ = clear(a) \land f \)
    » So \( s^+ \) must satisfy both \( clear(a) \) and \( f \)

### Case:

1. \( f \) contains no temporal ops: \( f^+ := \text{TRUE} \text{ if } s \models f, \text{ FALSE otherwise} \)
2. \( f = f_1 \land f_2 \) : \( f^+ := \text{Progress}(f_1, s) \land \text{Progress}(f_2, s) \)
3. \( f = f_1 \lor f_2 \) : \( f^+ := \text{Progress}(f_1, s) \lor \text{Progress}(f_2, s) \)
4. \( f = \neg f_1 \) : \( f^+ := \neg \text{Progress}(f_1, s) \)
5. \( f = \bigcirc f_1 \) : \( f^+ := f_1 \)
6. \( f = \bigtriangleup f_1 \) : \( f^+ := \text{Progress}(f_1, s) \lor f \)
7. \( f = \Box f_1 \) : \( f^+ := \text{Progress}(f_1, s) \land f \)
8. \( f = f_1 \cup f_2 \) : \( f^+ := \text{Progress}(f_2, s) \lor (\text{Progress}(f_1, s) \land f) \)
9. \( f = \forall [x:g(x)] h(x) \) : \( f^+ := \text{Progress}(h_1, s) \land \ldots \land \text{Progress}(h_n, s) \)
10. \( f = \exists [x:g(x)] h(x) \) : \( f^+ := \text{Progress}(h_1, s) \lor \ldots \lor \text{Progress}(h_n, s) \)
Pseudocode for TLPlan

- Nondeterministic forward search
  - Input includes a control formula \( f \) for the current state \( s \)
  - If \( f^+ = \) FALSE then \( s \) has no acceptable successors => backtrack
  - Otherwise the progressed formula is the control formula for \( s \)’s children

Procedure TLPlan \((s, f, g, \pi)\)

\[
\begin{align*}
\text{if } f &= \text{FALSE then return failure} \\
\text{if } s \text{ satisfies } g \text{ then return } \pi \\
\text{if } f^+ &= \text{Progress } (f, s) \\
\text{if } f^+ &= \text{FALSE then return failure} \\
A &= \{\text{actions applicable to } s\} \\
\text{if } A \text{ is empty then return failure} \\
\text{nondeterministically choose } a \in A \\
s^+ &= \gamma(s, a) \\
\text{return TLPlan } (s^+, f^+, g, \pi.a)
\end{align*}
\]
Example Planning Problem

- \( s = \{\text{on}(a), \text{on}(b), \text{clear}(a), \text{clear}(c), \text{on}(c,b)\} \)
- \( g = \{\text{on}(b, a)\} \)
- \( f = \Box \forall [x:\text{clear}(x)] \{(\text{on}(x) \land \neg \exists [y:\text{GOAL}(\text{on}(x,y))]) \Rightarrow \bigcirc \neg \text{holding}(x)\} \)
  - never pick up a block \( x \) if \( x \) is not required to be on another block \( y \)
- \( f^+ = \text{Progress}(f_1, s) \land f, \) where
  - \( f_1 = \forall [x:\text{clear}(x)] \{(\text{on}(x) \land \neg \exists [y:\text{GOAL}(\text{on}(x,y))]) \Rightarrow \bigcirc \neg \text{holding}(x)\} \)
- \( \{x: \text{clear}(x)\} = \{a, c\}, \) so
- \( \text{Progress}(f_1, s) = \text{Progress}((\text{on}(a) \land \neg \exists [y:\text{GOAL}(\text{on}(a,y))]) \Rightarrow \bigcirc \neg \text{holding}(a)), s) \)
  \( \land \text{Progress}((\text{on}(c) \land \neg \exists [y:\text{GOAL}(\text{on}(c,y))]) \Rightarrow \bigcirc \neg \text{holding}(b)), s) \)
  \( = (\text{TRUE} \Rightarrow \neg \text{holding}(a)) \land \text{TRUE} = \neg \text{holding}(a) \)
- \( f^+ = \neg \text{holding}(a) \land f \)
  \( = \neg \text{holding}(a) \land \)
  \( \Box \forall [x:\text{clear}(x)] \{(\text{on}(x) \land \neg \exists [y:\text{GOAL}(\text{on}(x,y))]) \Rightarrow \bigcirc \neg \text{holding}(x)\} \)
- Two applicable actions: pickup(a) and pickup(c)
  - Try \( s^+ = \gamma(s, \text{pickup}(a)) \): \( f^+ \) simplifies to FALSE \( \Rightarrow \) backtrack
  - Try \( s^+ = \gamma(s, \text{pickup}(c)) \): \( f^+ \) doesn’t simplify to FALSE \( \Rightarrow \) keep going
Blocks-World Results

Control 1 fails on 1 problem of size 11

No Control (breadth-first)
Control 1
Control 2
Control 3
Control 3 (breadth-first)
Blocks-World Results

SatPlan fails on 3 problems of size 10

UCPOP fails on all problems of size 6

BlackBox fails on 1 problem of size 10

IPP fails on 2 problems of size 11 and 12 exceeds 1GB RAM on problems of size 13
Logistics-Domain Results

- IPP fails on problems of size > 9
- Satplan fails on 2 problems of size 14 and 15
- BlackBox fails on problems of size > 15

Seconds CPU Time vs. Number of Packages
Discussion

- 2000 International Planning Competition
  - TALplanner: similar algorithm, different temporal logic
    » received the top award for a “hand-tailored” (i.e., domain-configurable) planner
- TLPlan won the same award in the 2002 International Planning Competition
- Both of them:
  - Ran several orders of magnitude faster than the “fully automated” (i.e., domain-independent) planners
    » especially on large problems
  - Solved problems on which the domain-independent planners ran out of time or memory