

# Comparison of Planners

Manuela Veloso  
Reid Simmons

PEL, Fall 2010

September 22, 2010

# Prodigy Planner

- Extension to GPS
  - Set of goals, instead of stack of goals
  - Means-ends analysis for selection of “pending goals”
  - Choice point for applying an operator when applicable and continue backward-chaining (subgoaling)

Simmons, Veloso, Carnegie Mellon

2

15-887 Fall 2010

## Prodigy4.0 (Veloso et al. 90)

1. Terminate if the goal statement is satisfied in the current state.
2. Compute the **SET** of pending goals  $\mathcal{G}$ , and the **set** of applicable operators  $\mathcal{A}$ 
  - A goal is pending if it is a precondition, not satisfied in the current state, of an operator already in the plan.
  - An operator is applicable when all its preconditions are satisfied in the state.
1. Choose a goal  $G$  in  $\mathcal{G}$  or choose an operator  $A$  in  $\mathcal{A}$

Simmons, Veloso, Carnegie Mellon

3

15-887 Fall 2010

## Prodigy4.0 Planning Algorithm

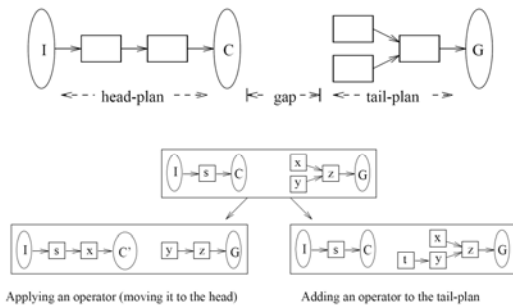
4. If  $G$  has been chosen, then
  - Expand goal  $G$ , i.e., get the set  $\mathcal{O}$  of relevant instantiated operators that could achieve  $G$ ,
  - Choose an operator  $O$  from  $\mathcal{O}$ ,
  - Go to step 1.
5. If an operator  $A$  has been selected as directly applicable, then
  - Apply  $A$ ,
  - Go to step 1.

Simmons, Veloso, Carnegie Mellon

4

15-887 Fall 2010

## Prodigy4.0 – Search Representation



Simmons, Veloso, Carnegie Mellon

5

15-887 Fall 2010

## Plan-Space Planning

- Complete, sound, and “optimal” in terms of number of steps in the plan and within the operator choices made
- Optimal handling of goal orderings

Simmons, Veloso, Carnegie Mellon

6

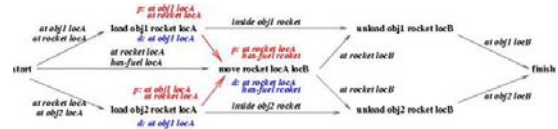
15-887 Fall 2010

## Rocket Domain - Linking



Example - LINKING

## Rocket Domain – Solving Threats



Example - THREATS

## Facts and Goals

- GOALS:
  - Identify commitments in a partial-order planner.
  - Understand the implications of such commitments.
  - Provide clear demonstration of exemplary domains where total-order planners perform better than partial-order planners.

## Comparison of Planning Algorithms

- Complete nonlinear state-space planning
- Plan-space planning
- Graphplan
- Satplan
- And more

Is there a *universally best* planning algorithm?

## State-Space and Plan-Space

- Planning is NP-hard.
- Two different planning approaches: state-space and plan-space planning

	State-space	Plan-space
Commitments in plan step orderings	Yes	No
Therefore, suffer with goal orderings	Yes	No
Therefore, handle goal interactions	Poorly	Efficiently

## Step Ordering Commitments

WHY?

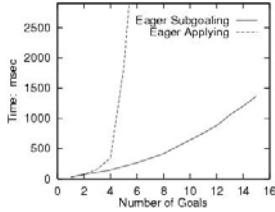
Use of the STATE of the world while planning

In Prodigy4.0 advantages include:

- Means-ends analysis - plan for goals that reduce the differences between current and goal states.
- Informed selection of operators - select operators that need less planning work than others.
- State useful for learning, generation and match of conditions supporting informed decisions.
- Helpful for generating anytime planning - provide *valid*, executable, plans at any time.

## Eagerly Subgoaling Can Be Better

Operator:  $A_i$   
 preconds:  $\{I_i\}$   
 adds:  $\{G_i\}$   
 deletes:  $\{I_j | j < i\}$



Example:

- Initial state:  $I_1, I_2, I_3$
- Goal:  $G_2, G_3, G_1$
- Plan:  $A_1, A_2, A_3$

Simmons, Veloso, Carnegie Mellon

13

15-887 Fall 2010

## Parallel between Commitments

Operator Polish  
 preconds: ()  
 adds: polished  
 deletes: ()

Operator Drill-Hole  
 preconds: ()  
 adds: has-hole  
 deletes: polished

Goal: polished and has-hole  
 Initial state: empty  
 Prodigy4.0

Goal: polished and has-hole  
 Initial state: polished  
 SNLP

- plan for goal polished  
 - select Polish  
 • order Polish as first step  
 - plan for goal has-hole  
 - select Drill-Hole  
 • order Drill-Hole  $\Rightarrow$  Polish  
 • polished deleted, backtrack  
 - Polish  $\Rightarrow$  Drill-Hole

- plan for goal polished  
 - select Initial state  
 • link Initial to polished  
 - plan for goal has-hole  
 - select Drill-Hole  
 • link Drill-Hole to has-hole  
 • threat - relink polished  
 - select Polish  
 - link Polish to polished  
 - Polish  $\Rightarrow$  Drill-Hole

Simmons, Veloso, Carnegie Mellon

14

15-887 Fall 2010

## Serializability and Linkability

- A set of subgoals is *serializable* [Korf]:
  - If there exists some ordering whereby they can be solved sequentially,
  - without ever violating a previously solved subgoal.
- Easily serializable, laboriously serializable
- A set of subgoals is *easily linkable*:
  - If, independently of the order by which the planner links these subgoals to operators,
  - it never has to undo those links.
  - Otherwise it is *laboriously linkable*.

Simmons, Veloso, Carnegie Mellon

15

15-887 Fall 2010

## Easily Linkable Goals

operator  $A_i$                       operator  $A_*$   
 preconds ()                      preconds ()  
 adds  $g_i$                           adds  $g_*$   
 deletes ()                        deletes  $g_i, \forall i$

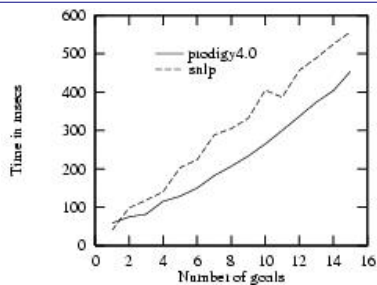
Initial state:  $g_1, g_2, g_3, g_4, g_5$   
 Goal statement:  $g_2, g_5, g_4, g_*, g_3, g_1$   
 Plan:  $A_*, A_2, A_5, A_4, A_3, A_1$

Simmons, Veloso, Carnegie Mellon

16

15-887 Fall 2010

## Easily Linkable Goals



Simmons, Veloso, Carnegie Mellon

17

15-887 Fall 2010

## Laboriously Linkable Goals

operator  $A_i$                       operator  $A_*$   
 preconds  $g_*, g_{i-1}$               preconds ()  
 adds  $g_i$                           adds  $g_*$   
 deletes  $g_*$                         deletes ()

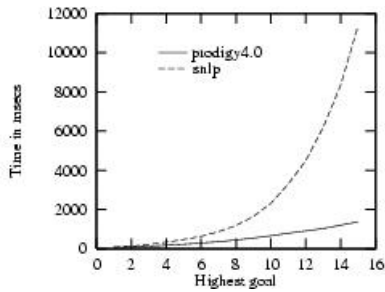
Initial state:  $g_*$   
 Goal statement:  $g_*, g_5$   
 Plan:  $A_1, A_*, A_2, A_*, A_3, A_*, A_4, A_*, A_5, A_*$

Simmons, Veloso, Carnegie Mellon

18

15-887 Fall 2010

## Laboriously Linkable Goals



Simmons, Veloso, Carnegie Mellon

19

15-887 Fall 2010

## Multiple Linking Alternatives

```

operator Ai
  preconds gj, ∀j < i
  adds gi, gj, ∀j < i-1
  deletes gi-1

operator A5      operator A4      operator A3
pre g1, g3, g2, g1  pre g3, g2, g1  pre g2, g1
add g5, g3, g2, g1  add g4, g2, g1  add g3, g1
del g4                del g3                del g2

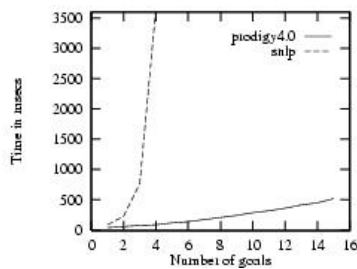
Initial state: g1, g2, g3, g4
Goal statement: g2, g5, g4, g3, g1
Plan: A5, A4, A3, A2, A1
    
```

Simmons, Veloso, Carnegie Mellon

20

15-887 Fall 2010

## Empirical Results – Multiple Linking



Simmons, Veloso, Carnegie Mellon

21

15-887 Fall 2010

## Summary – Comparison of Planners

- Similar empirical comparison results for other planning algorithms (we'll see later).
- **There is not a planning strategy that is universally better than the others.**
- Even for a particular planning algorithm: **There is no single domain-independent search heuristic that performs more efficiently than others for all problems or in all domains.**

**Learning** is challenging and appropriate for **ANY** planner.

Simmons, Veloso, Carnegie Mellon

22

15-887 Fall 2010

## Generating a Solution Plan

- Linear planning – Planning with a goal **stack**.
- Nonlinear planning – Interleaving of goals
  - State-space search
  - Plan-space search
  - Graph-based search
  - Sat-based search
  - OBDD-based search
- Hierarchical planning
  - Emphasis on action decomposition/refinement
  - Knowledge engineering/acquisition
  - Very little search

Simmons, Veloso, Carnegie Mellon

23

15-887 Fall 2010

## Summary

- **Planning:** selecting one sequence of actions (operators) that transform (apply to) an initial state to a final state where the goal statement is true.
- **Means-ends analysis:** identify and reduce, as soon as possible, *differences* between state and goals.
- **Linear planning:** backward chaining with means-ends analysis using a stack of goals - potentially efficient, possibly unoptimal, incomplete; GPS, STRIPS.
- **Nonlinear planning with means-ends analysis:** backward chaining using a set of goals; reason about *when* "to reduce the differences;" Prodigy4.0.
- **Planning as search:** control rules to capture heuristics for efficient search; learning opportunities.

Simmons, Veloso, Carnegie Mellon

24

15-887 Fall 2010