CSC2542
Introduction to Planning

Sheila McIlraith
Department of Computer Science
University of Toronto
Winter 2009

Administrative Announcements

- There will be *no tutorial* on Friday.
- csc2542w09@cs email list created (though I prefer communicating via the course announcements web page)
- Suggested readings for next week:
  - Skim Chapter 3, Chapter 4
  - *Skip* Chapter 5
  - Read Chapter 6 & 7 (we'll cover this in class)

Class & Tutorial Schedule

Thursdays Jan 22 – Feb 12 (inclusive): **Lectures***

Fri Jan 29 – Tutorial to go over paper selection

Fri Feb 6 – Tutorial to go over assignment

Fri Feb 13 – Tutorial related to assignment and course material

---

Week of February 16 – Reading Week: no lectures or tutorials

Thursdays Feb 26 – Mar 19 (inclusive): 2 student **paper presentations** (per class)

Fridays Feb 27 – Mar 20 (inclusive): overflow lectures/presentations, project help

---

Thursdays Mar 26 – Apr 2 (inclusive): 3 or 4 student **project presentations** (per class)

Fridays Mar 27 – Apr 3 (inclusive): 1 or 2 student project presentations (per class)

Thursday Apr 9 – Lecture: Wrap up

Friday Apr 10 – No tutorial (university closed for Good Friday)

** I'll post a more detailed lecture and readings schedule*

Important Dates for You

Last Week Jan/First Week Feb – Meet with Sheila to discuss project selection*

Early February – Make your paper selection (for presentation)

Thurs Feb 5 – Assignment handed out

Week of Feb 16 – Reading week (no classes)

Thurs Feb 26 – Assignments due

-- Student paper presentations commence

Tues Mar 3 – Marked assignments returned

Fri Mar 6 – Deadline for dropping course

Thurs Mar 26 – Student project presentations commence

Thurs Apr 9 – Last lecture

Tues May 5 – Projects** due

Fri May 14 (or before) – Marked projects returned

* *Email me next week to set up a time to meet*

** *A suggested schedule for project milestones is forthcoming*

END of
Administrative Announcements

Acknowledgements

Some of the slides used in this course are modifications of Dana Nau's lecture slides for the textbook *Automated Planning,* licensed under the Creative Commons Attribution-NonCommercial-ShareAlike License: http://creativecommons.org/licenses/by-nc-sa/2.0/

Other slides are modifications of slides developed by Malte Helmert, Bernhard Nebel, and Jussi Rintanen.

I have also used some material prepared by P@trick Haslum.

I would like to gratefully acknowledge the contributions of these researchers, and thank them for generously permitting me to use aspects of their presentation material.

Dictionary.com/plan

Address: http://dictionary.reference.com/search?q=plan

Dictionary.com | plan | Search

Dictionary | Thesaurus | Web

Home | Premium: Sign up | Login

plan *n.*

1. A scheme, program, or method worked out beforehand for the accomplishment of an objective: *a plan of attack.*
2. A proposed or tentative project or course of action: *had no plans for the evening.*
3. A systematic arrangement of elements or important parts; a configuration or outline: *a seating plan; the plan of a story.*
4. A drawing or diagram made to scale showing the structure or arrangement of something.
5. In perspective rendering, one of several imaginary planes perpendicular to the line of vision between the viewer and the object being depicted.
6. A program or policy stipulating a service or benefit: *a pension plan.*

Synonyms: *blueprint, design, project, scheme, strategy*

Dictionary.com/plan

Address: http://dictionary.reference.com/search?q=plan

Dictionary.com | plan | Search

Dictionary | Thesaurus | Web

Home | Premium: Sign up | Login

plan *n.*

1. A scheme, program, or method worked out beforehand for the accomplishment of an objective: *a plan of attack.*
2. A proposed or tentative project or course of action: *had no plans for the evening.*
3. A systematic arrangement of elements or important parts; a configuration or outline: *a seating plan; the plan of a story.*
4. A drawing or diagram made to scale showing the structure or arrangement of something.
5. In perspective rendering, one of several imaginary planes perpendicular to the line of vision between the viewer and the object being depicted.
6. A program or policy stipulating a service or benefit: *a pension plan.*

Synonyms: *blueprint, design, project, scheme, strategy*

```
03  Establish datum point at bullseye (0.25, 1.00)
side-milling tool
 at (-0.25, 1.25)
0, depth 0.50
t at (-0.25, 1.25)
0, depth 0.50
 at (-0.25, 3.00)
0, depth 0.50
t at (-0.25, 3.00)
0, depth 0.50
end-milling tool
004 T  VMC1  2.50    4.87  01  Total time on VMC1

005 A   EC1  0.00   32.29  01  Pre-clean board (scrub and wash)
                           02  Dry board in oven at 85 deg. F
005 B   EC1 30.00    0.48  01  Setup
                           02  Spread photoresist from 18000 RPM spinner
005 C   EC1 30.00    2.00  01  Setup
                           02  Photolithography of photoresist
                               using phototool in "real.iges"
005 D   EC1 30.00   20.00  01  Setup
                           02  Etching of copper
005 T   EC1 90.00   54.77  01  Total time on EC1

006 A   MC1 30.00    4.57  01  Setup
                           02  Prepare board for soldering
006 B   M
```

[a representation] of future behavior … usually a set of actions, with temporal and other constraints on them, for execution by some agent or agents.

- Austin Tate

[*MIT Encyclopedia of the Cognitive Sciences,* 1999]

A portion of a manufacturing process plan

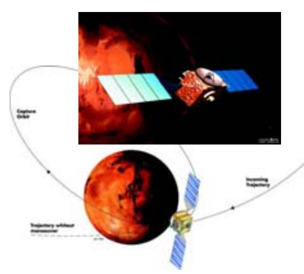
Modes of Planning

- Mixed Initiative Planning
- Automated Plan Generation

Example Planning Applications

Autonomous Agents for Space Exploration

- Autonomous planning, scheduling, control
  - NASA: JPL and Ames
- Remote Agent Experiment (RAX)
  - Deep Space 1
- Mars Exploration Rover (MER)



Other Autonomous Systems

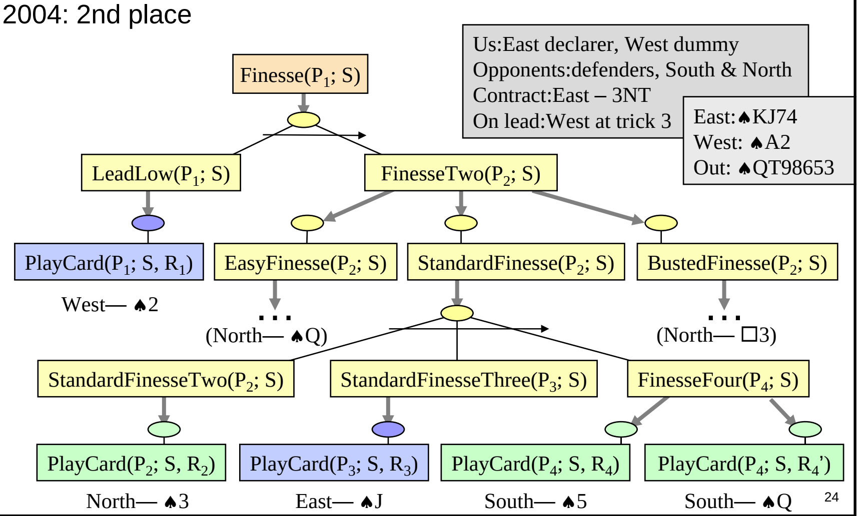
Manufacturing Automation

- Sheet-metal bending machines - Amada Corporation
  - Software to plan the sequence of bends
    [Gupta and Bourne, *J. Manufacturing Sci. and Engr.*, 1999]

Games

E.g., *Bridge Baron* - Great Game Products

- 1997 world champion of computer bridge
  [Smith, Nau, and Throop, *AI Magazine*, 1998]
- 2004: 2nd place



Other Applications

- Scheduling with Action Choices & Resource Requirements
  - Problems in supply chain management
  - HSTS (Hubble Space Telescope scheduler)
  - Workflow management
- Air Traffic Control
  - Route aircraft between runways and terminals. Crafts must be kept safely separated. Safe distance depends on craft and mode of transport. Minimize taxi and wait time.
- Character Animation
  - Generate step-by-step character behaviour from high-level spec
- Plan-based Interfaces
  - E.g. NLP to database interfaces
  - Plan recognition

Other Applications (cont.)

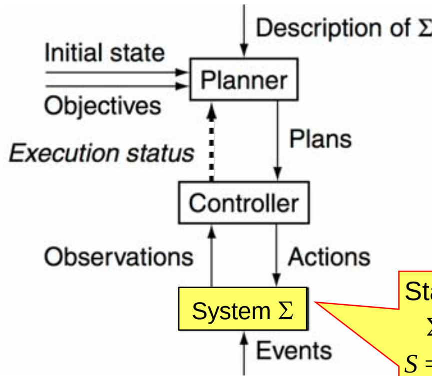
- Web Service Composition
  - Compose web services, and monitor their execution
  - Many of the web standards have a lot of connections to plan representation languages
    - BPEL; BPEL-4WS allow workflow specifications
    - DAML-S allows process specifications
- Grid Services/Scientific Workflow Management
- Genome Rearrangement
  - The relationship between different organisms can be measured by the number of "evolution events" (rearrangements) that separate their genomes
  - Find shortest (or most likely) sequence of rearrangements between a pair of genomes

Outline

- ➔ Conceptual model for planning
- Classes of planning problems
- Classes of planners and example instances
- Beyond planning
- Planning research – the big picture
- Some of what I hope you'll get from the course

Conceptual Model

1. Environment



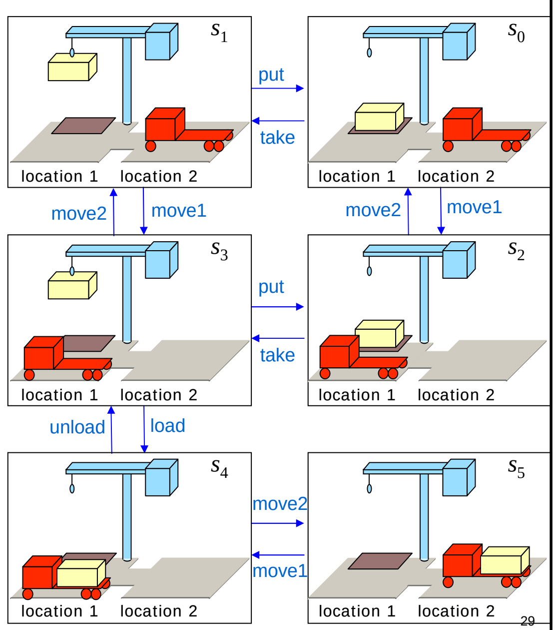
State transition system
$\Sigma = (S,A,E,\gamma)$
S = {states}
A = {actions}
E = {exogenous events}
γ = state-transition function

State Transition System

$\Sigma = (S,A,E,\gamma)$

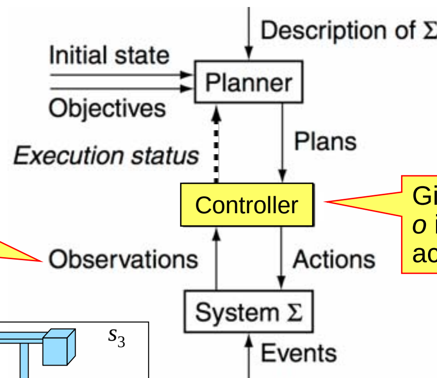
- S = {states}
- A = {actions}
- E = {exogenous events}
- State-transition function $\gamma: S \times (A \cup E) \rightarrow 2^S$
  - $S = \{s_0, \ldots, s_5\}$
  - A = {move1, move2, put, take, load, unload}
  - $E = \{\}$
  - γ: see the arrows

Dock Worker Robots (DWR):



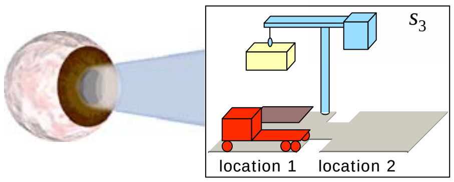
Conceptual Model

2. Controller



Given observation *o* in *O*, produces action *a* in *A*

Observation function *h*: $S \rightarrow O$

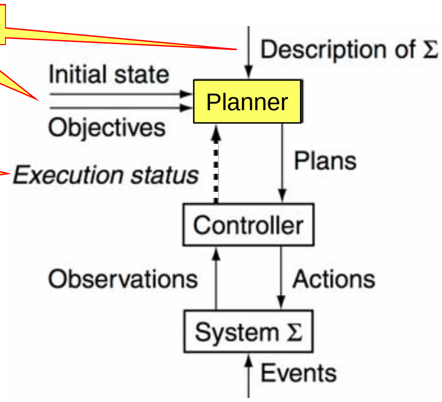


Conceptual Model

3. Planner's Input

Planning problem

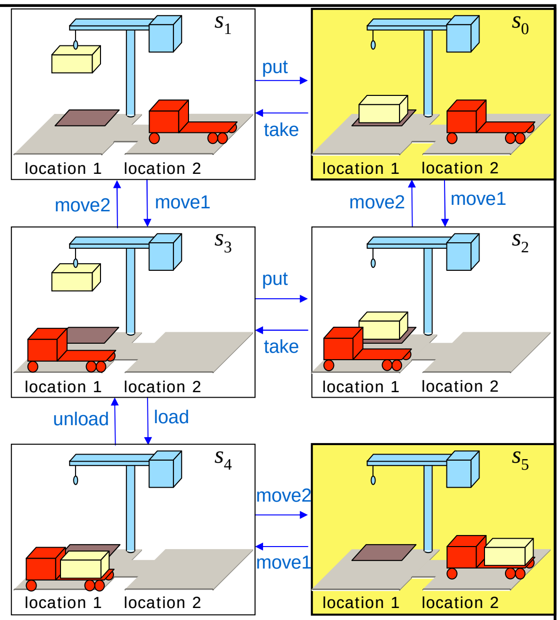
Omit unless planning is online



Planning Problem

$P = (\Sigma, s_0, G)$

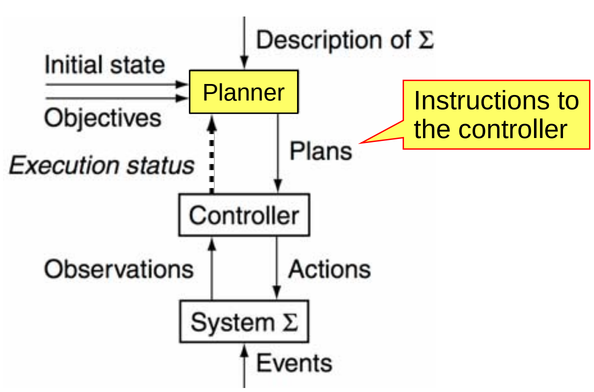
Σ: System Description
s_0: Initial state(s)
E.g., Initial state = s_0
G: Objective
Goal state,
Set of goal states,
Set of tasks,
"trajectory" of states,
Objective function, …
E.g., Goal state = s_5



The Dock Worker Robots (DWR) domain

Conceptual Model

4. Planner's Output



Plans

Classical plan:

a sequence of actions

E.g.,

⟨take, move1, load, move2⟩

Policy:

partial function from *S* into *A*

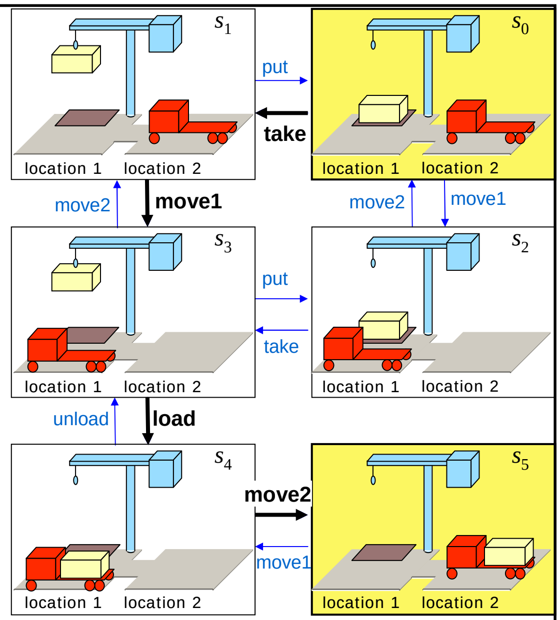
E.g.,

{(s_0, take),

(s_1, move1),

(s_3, load),

(s_4, move2)}



The Dock Worker Robots (DWR) domain

Outline

- Conceptual model for planning
- ➔ Classes of planning problems
- Classes of planners and example instances
- Beyond planning
- Planning research – the big picture
- Some of what I hope you'll get from the course

Different Classes Planning Problems

Varying components of the planning problem specification yields different classes of problems. E.g.,

dynamics: deterministic, nondeterministic, probabilistic

observability: full, partial, none

horizon: finite, infinite

objective requirement: satisfying, optimizing

…

Different Classes Planning Problems

dynamics: **deterministic**, nondeterministic, probabilistic
observability: full, partial, **none**
horizon: **finite**, infinite
objective requirement: **satisfying**, optimizing
…

- **classical planning**
- conditional planning with full observability
- conditional planning with partial observability
- conformant planning
- markov decision processes (MDP)
- partial observable MDP (POMDP)
- preference-based/over-subscription planning

Different Classes Planning Problems

dynamics: deterministic, **nondeterministic**, probabilistic
observability: **full**, partial, none
horizon: **finite**, **infinite**
objective requirement: **satisfying**, optimizing
…

- classical planning
- **conditional planning with full observability**
- conditional planning with partial observability
- conformant planning
- markov decision processes (MDP)
- partial observable MDP (POMDP)
- preference-based/over-subscription planning

Different Classes Planning Problems

dynamics: deterministic, **nondeterministic**, probabilistic
observability: full, **partial**, none
horizon: **finite**, **infinite**
objective requirement: **satisfying**, optimizing
…

- classical planning
- conditional planning with full observability
- **conditional planning with partial observability**
- conformant planning
- markov decision processes (MDP)
- partial observable MDP (POMDP)
- preference-based/over-subscription planning

Different Classes Planning Problems

dynamics: deterministic, **nondeterministic**, probabilistic
observability: full, partial, **none**
horizon: **finite**, infinite
objective requirement: **satisfying**, optimizing
…

- classical planning
- conditional planning with full observability
- conditional planning with partial observability
- **conformant planning**
- markov decision processes (MDP)
- partial observable MDP (POMDP)
- preference-based/over-subscription planning

Different Classes Planning Problems

dynamics: deterministic, nondeterministic, **probabilistic**
observability: **full**, partial, none
horizon: **finite**, **infinite**
objective requirement: satisfying, **optimizing**
…

- classical planning
- conditional planning with full observability
- conditional planning with partial observability
- conformant planning
- **markov decision processes (MDP)**
- partial observable MDP (POMDP)
- preference-based/over-subscription planning

Different Classes Planning Problems

dynamics: deterministic, nondeterministic, **probabilistic**
observability: full, **partial**, none
horizon: **finite**, **infinite**
objective requirement: satisfying, **optimizing**
…

- classical planning
- conditional planning with full observability
- conditional planning with partial observability
- conformant planning
- markov decision processes (MDP)
- **partial observable MDP (POMDP)**
- preference-based/over-subscription planning

Different Classes Planning Problems

dynamics: **deterministic,** nondeterministic, probabilistic
observability: full, partial, **none**
horizon: **finite**, **infinite**
objective requirement: satisfying, **optimizing**
…

- classical planning
- conditional planning with full observability
- conditional planning with partial observability
- conformant planning
- markov decision processes (MDP)
- partial observable MDP (POMDP)
- **preference-based/over-subscription planning**

Other Dimensions

dynamics: deterministic, nondeterministic, probabilistic
: explicit time, implicit time
: instantaneous, durative
: continuous, discrete, hybrid
perception: perfect, noisy
horizon: finite, infinite
objective requirement: satisfying, optimizing
…

- classical planning
- conditional planning with full observability
- conditional planning with partial observability
- conformant planning
- markov decision processes (MDP)
- partial observable MDP (POMDP)
- preference-based/over-subscription planning

Why is Planning Difficult?

- solutions to classical planning problems are paths from an initial state to a goal state in the transition graph
  - Efficiently solvable by Dijkstra's algorithm in $O(|V| \log |V| + |E|)$ time
  - Why don't we solve all planning problems this way?
- state space may be huge: 10^9, 10^{12}, 10^{15}, …states
- constructing the transition graph is infeasible!
- planning algorithms try to avoid constructing whole graph
- planning algorithms often are – but not guaranteed to be more effiencient that obvious solution methods constructing the transition graph and using e.g., Dijkstra's algorithm

Outline

- Conceptual model for planning
- Classes of planning problems
- ➔ Classes of planners and example instances
- Beyond planning
- Planning research – the big picture
- Some of what I hope you'll get from the course

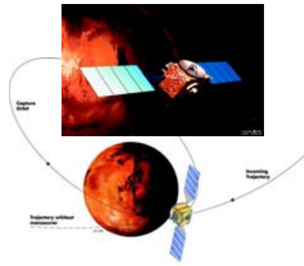
Three Main Classes of Planners

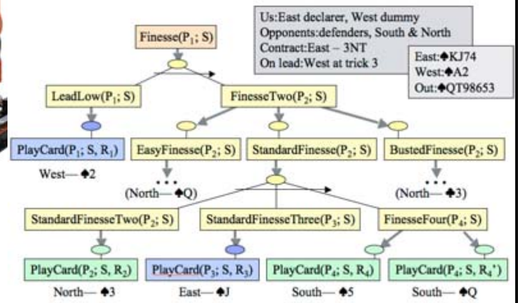
1. Domain-specific
2. Domain-independent
3. Domain-customizable

** Ghallab, Nau, and Traverso's use "configurable" (which I don't like)*
Also called "Domain-specific" or "Knowledge-Based"

1. Domain-Specific Planners

- Made or tuned for specific domain
- Won’t work well (if at all) in any other domain
- Many successful real-world planning systems work this way



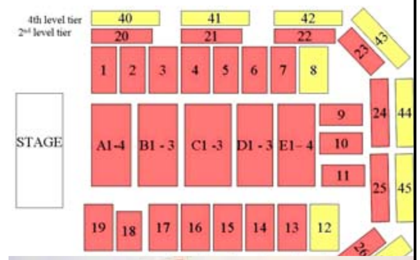


2. Domain-Independent Planners

- In principle, a domain-independent planner works in any planning domain
- Uses no domain-specific knowledge except the definitions of the basic actions

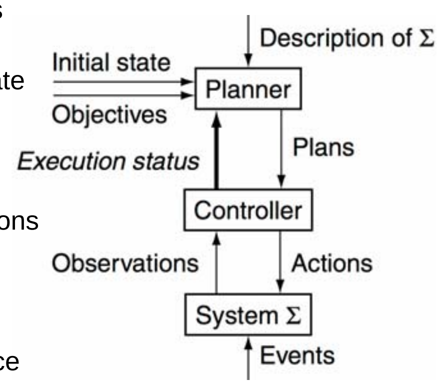
2. Domain-Independent Planners

- In practice,
  - Not feasible to develop domain-independent planners that work in *every* possible domain
- Make simplifying assumptions to restrict the set of domains
  - *Classical planning*
  - Historical focus of most automated-planning research



Restrictive Assumptions

- **A0: Finite system:**
  - finitely many states, actions, events
- **A1: Fully observable:**
  - the controller always Σ's current state
- **A2: Deterministic:**
  - each action has only one outcome
- **A3: Static** (no exogenous events):
  - no changes but the controller's actions
- **A4: Attainment goals:**
  - a set of goal states S_g
- **A5: Sequential plans:**
  - a plan is a linearly ordered sequence of actions $(a_1, a_2, \ldots a_n)$
- **A6: Implicit time:**
  - no time durations; linear sequence of instantaneous states
- **A7: Off-line planning:**
  - planner doesn't know the execution status

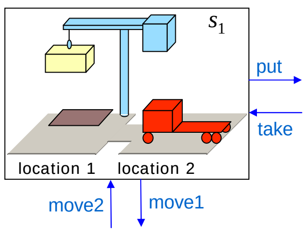


Classical Planning

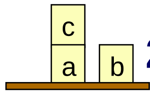
- Classical planning requires all eight restrictive assumptions
  - Offline generation of action sequences for a deterministic, static, finite system, with complete knowledge, attainment goals, and implicit time
- Reduces to the following problem:
  - Given (Σ, s_0, g)
  - Find a sequence of actions ($a_1, a_2, \ldots a_n$) that produces a sequence of state transitions (s_1, s_2, …, s_n) such that g is in s_n.
- This is just path-searching in a graph
  - Nodes = states
  - Edges = actions
- ***Is this trivial?***

Classical Planning (cont.)

It's hard because problems are huge!

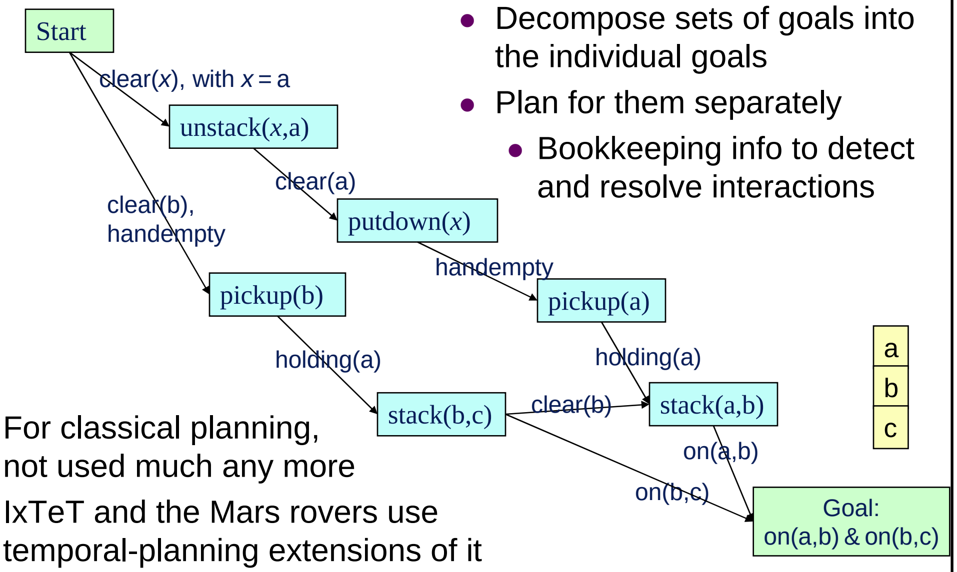


- Generalize the earlier example:
  - Five locations, three robot carts, 100 containers, three piles
    - Then there are 10^{277} states
- Number of particles in the universe is only about 10^{87}
  - The example is more than 10^{190} times as large!
- Automated planning research has been heavily dominated by classical planning
  - Dozens (hundreds?) of different algorithms
  - We'll cover the state-of-the-art in this area



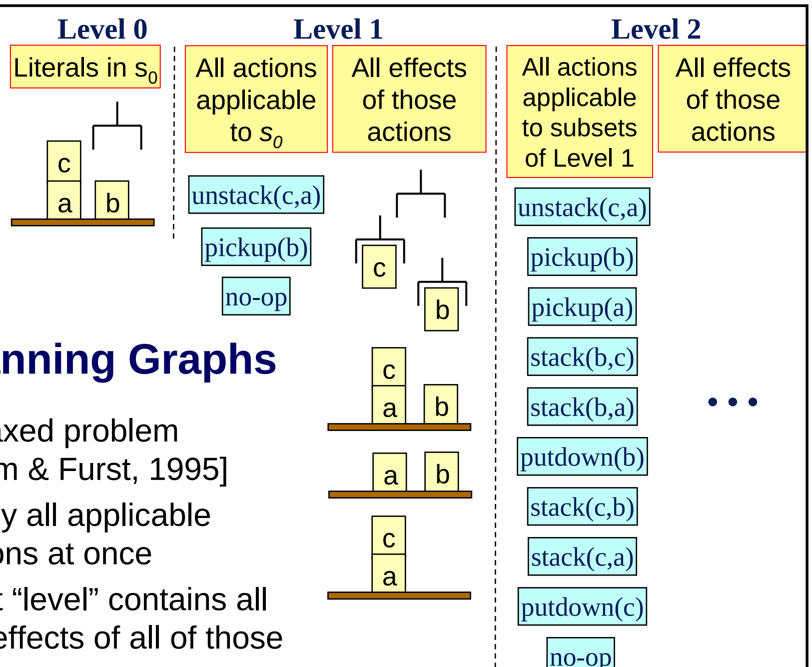
2.2 Plan-Space Planning

- Decompose sets of goals into the individual goals
- Plan for them separately
  - Bookkeeping info to detect and resolve interactions



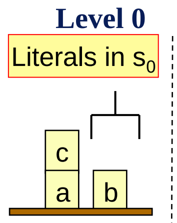
For classical planning, not used much any more

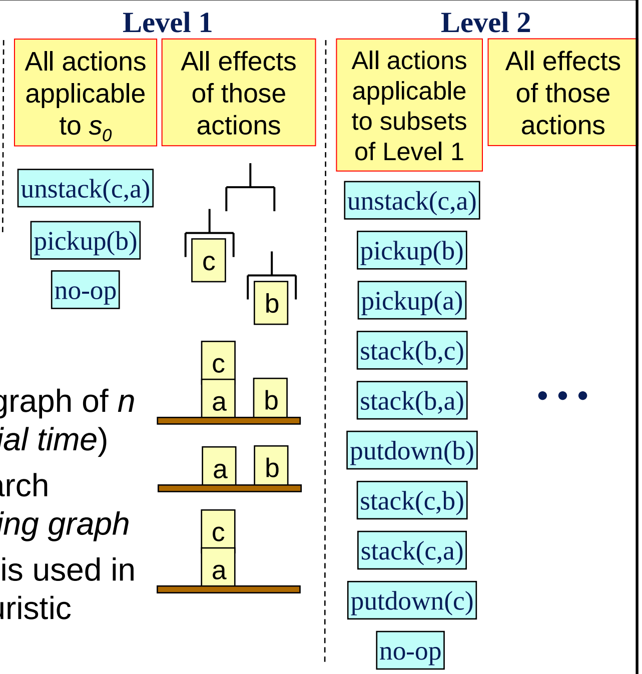
IxTeT and the Mars rovers use temporal-planning extensions of it



2.3 Planning Graphs

- Relaxed problem [Blum & Furst, 1995]
- Apply all applicable actions at once
- Next “level” contains all the effects of all of those actions





E.g., Graphplan

- For n = 1, 2, …
  - Make planning graph of n levels (*polynomial time*)
  - State-space search *within the planning graph*
- Such a plan graph is used in many SAT and heuristic search planners

2.3 Heuristic Search

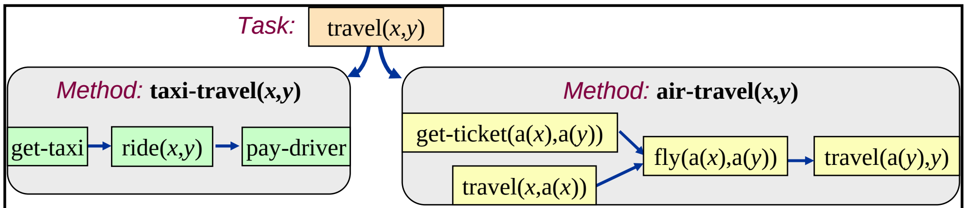
- Can we do an A*-style heuristic search?
- Historically, it was difficult to find a good *h* function
  - Planning graphs make it feasible
    - Can extract *h* from the planning graph
- Problem: A* quickly runs out of memory
  - So do a greedy search
- Greedy search can get trapped in local minima
  - Greedy search plus local search at local minima
- HSP [Bonet & Geffner]
- FastForward (FF) [Hoffmann], etc

2.4 Translation to General Problem Solver

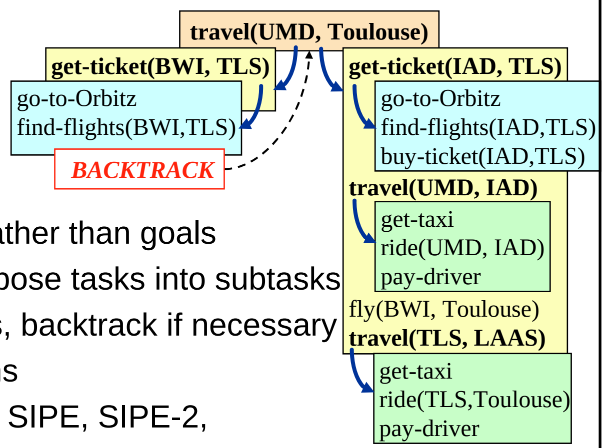
- Translate the planning problem or the planning graph into another kind of problem for which there are efficient solvers
  - Find a solution to that problem
  - Extract the plan from the solution
- SAT solvers
  - SATplan and Blackbox [Kautz & Selman]
- Answer Set Programming (ASP) solvers
  - [Son *et al.*], [Lifschitz *et al.*], etc.
- Integer programming solvers such as Cplex
  - [Vossen *et al.*]

3. Domain-customizable

- Domain-independent planners are quite slow compared with domain-specific planners
  - Blocks world in linear time [Slaney and Thiébaux, *A.I.*, 2001]
  - Can get analogous results in many other domains
- But don't want to write a new planner for every domain!
- **Domain-customizable planners**
  - Domain-independent planning engine
  - Input (the "objective") includes info about how to solve problems in the domain.
    - Hierarchical Task Network (HTN) planning
    - Planning with control formulas
    - Planning with a plan script or agent program

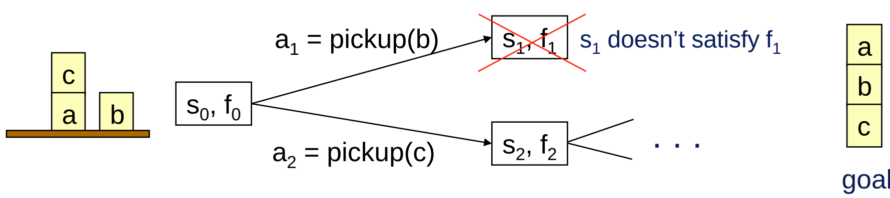


3.1 HTN Planning



- Problem reduction
  - *Tasks* (activities) rather than goals
  - *Methods* to decompose tasks into subtasks
  - Enforce constraints, backtrack if necessary
- Real-world applications
- Noah, Nonlin, O-Plan, SIPE, SIPE-2, SHOP, SHOP2

3.2 Planning with Control Formulas



- At each state s_i we have a *control formula* f_i in temporal logic

  $$ontable(x) \wedge \neg\exists[y{:}\mathrm{GOAL}(on(x,y))] \Rightarrow \bigcirc(\neg holding(x))$$

  "never pick up *x* from table unless *x* needs to be on another block"
- For each successor of *s*, derive a control formula using *logical progression*
- Prune any successor state in which the progressed formula is false
  - TLPlan [Bacchus & Kabanza]
  - TALplanner [Kvarnstrom & Doherty]

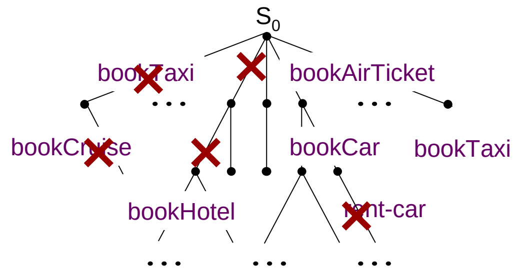
3.3 Planning w/ Agent Program or Plan Script

E.g., **Golog** [Levesque et al.]
Nondeterministic programs that act as procedural control knowledge, placing constraints on the valid action sequence/plans

E.g., *bookAirTicket(x);* if *far* then *bookCar(x)* else *bookTaxi(y)*

procedural constructs:

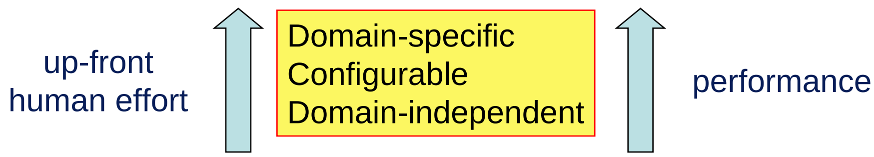
- sequence
- if-then-else
- nondeterministic choice
- while-do, etc.



Three Main Classes of Planners

1. Domain-specific
2. Domain-independent
   E.g.,
   Planning graph-based, SAT-based, heuristic search
3. Domain-customizable
   E.g.,
   HTN, domain control formula, agent programs/scripts

Comparisons (in general)



- Domain-specific planner
  - Write an entire computer program - lots of work
  - Lots of domain-specific performance improvements
- Domain-independent planner
  - Just give it the basic actions - not much effort
  - Not very efficient

Outline

- Conceptual model for planning
- Classes of planning problems
- Classes of planners and example instances
- ➔ Beyond planning
- Planning research – the big picture
- Some of what I hope you'll get from the course

Broad Application of Planning Techniques

Planning algorithms are applicable to a broad range of applications that can roughly be viewed as reachability problems. E.g.,

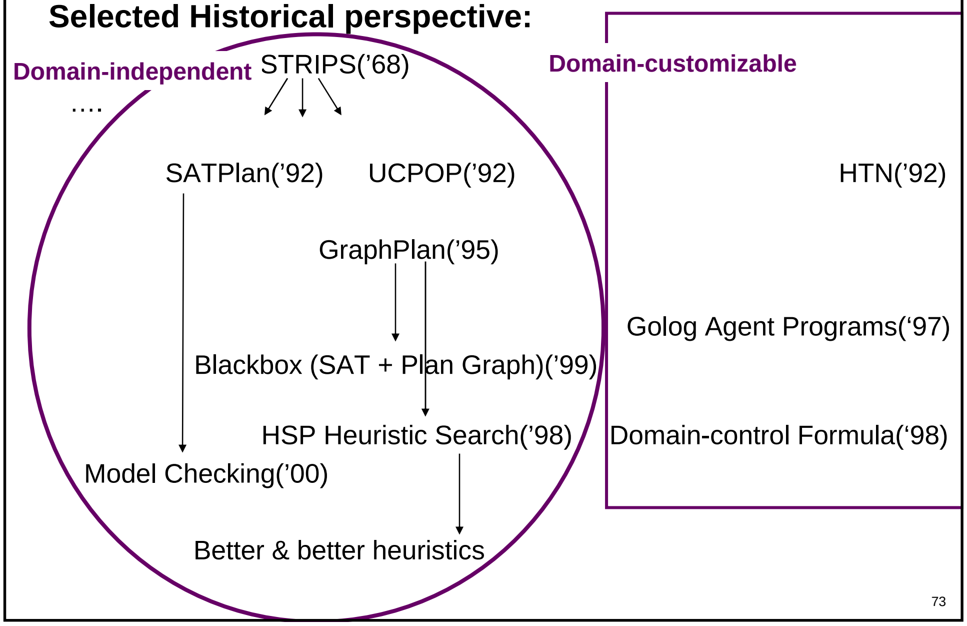
- Software verification
- Diagnosis of dynamical systems
- Story understanding
- Situation assessment/Plan recognition
- Gene sequencing

Outline

- Conceptual model for planning
- Classes of planning problems
- Classes of planners and example instances
- Beyond planning
- ➔ Planning research
- Some of what I hope you'll get from the course

Planning Research – The big picture

Selected Historical perspective:



Planning Research – The big picture

The Landscape:

CONFERENCES

ICAPS* (Int. Conf on AI Planning and Scheduling)
*merging of AIPS and ECP

AAMAS (Int. Conf. on Autonomous Agents and Multiagent Systems)

IJCAI, AAAI, ECAI

JOURNALS

JAIR, AIJ

BIENNIAL COMPETITION and BENCHMARKING DOMAINS

IPC-*n* (International Planning Competition)

PDDL (Planning Domain Definition Language)
standard input language for most benchmark problem sets

Planning Research – The big picture

Recent Advances

Very “active” field -- lots of papers in top conferences

- Tremendous strides in deterministic plan synthesis
  - Biennial Intl. Planning Competitions
- Current interest is in exploiting the insights from deterministic planning techniques to other planning scenarios

Some topics of recent focus:

- Better heuristics
- Richer domain customization (including preferences)
- From discrete to timed hybrid and/or continuous systems
- Planning and learning

Outline

- Conceptual model for planning
- Classes of planning problems
- Classes of planners and example instances
- Beyond planning
- Planning research – the big picture
- ➔ Some of what I hope you’ll get from the course

What will you get from this course?

- big picture of different kinds of planning problems
- algorithms for solving different problem classes, with an emphasis on the classical (“simplest”) setting:
  - algorithms based on heuristic search
  - algorithms based on SAT
  - algorithms that exploit rich objectives (domain control knowledge, temporally extended goals, preferences)
- many of these techniques are applicable to problems outside AI as well
- hands-on experience with a classical planner (optional)