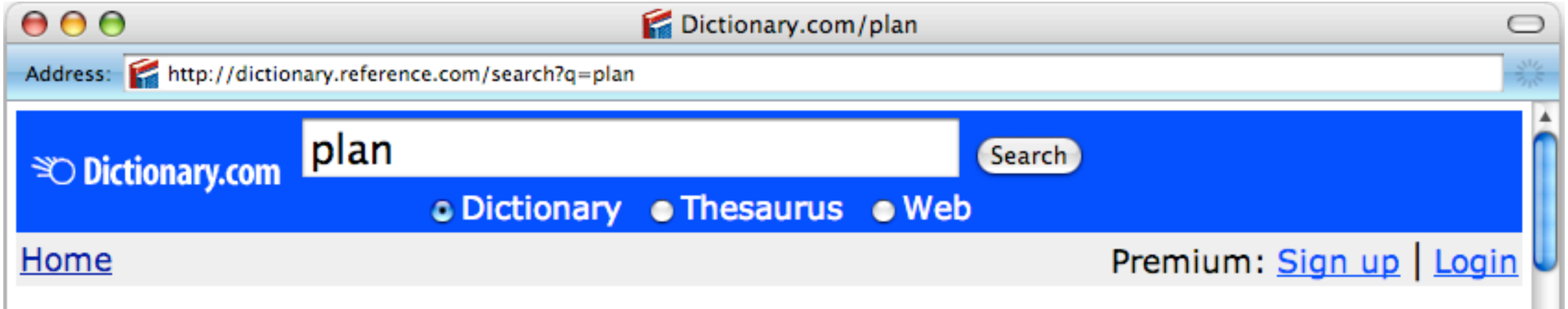
					03	Establish datum point at bullseye (0.25, 1.00)
004	B	VMC1	0.10	0.34	01	Install 0.15-diameter side-milling tool
					02	Rough side-mill pocket at (-0.25, 1.25) length 0.40, width 0.30, depth 0.50
					03	Finish side-mill pocket at (-0.25, 1.25) length 0.40, width 0.30, depth 0.50

May All Your Plans Succeed! (or have a high expected utility)

Dana S. Nau

UNIVERSITY OF MARYLAND

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	MC1	30.00	0.29	01	Setup
					02	Screenprint solder stop on board
006	C	MC1	30.00	7.50	01	Setup



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.*

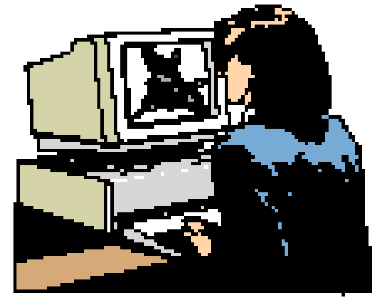
Synonyms: *blueprint, design, project, scheme, strategy* [a representation] of future behavior … usually a set of actions, with temporal and other constraints on them, for execution by some agent or agents. - Austin Tate

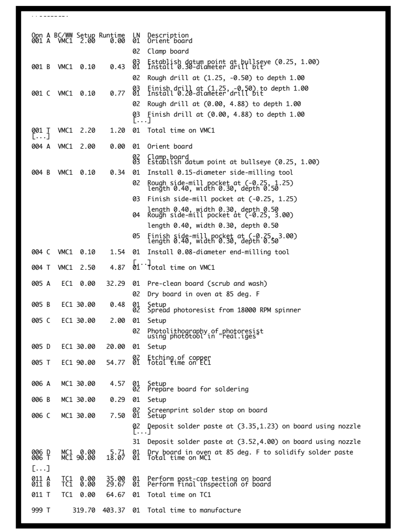
[*MIT Encyclopedia of the Cognitive Sciences*, 1999]

```
                                03  Establish datum point at bullseye (0.25, 1.00)
                                01  Install 0.15-diameter side-milling tool
                                02  Rough side-mill pocket at (-0.25, 1.25)
                                    length 0.40, width 0.30, depth 0.50
                                03  Finish side-mill pocket at (-0.25, 1.25)
                                    length 0.40, width 0.30, depth 0.50
                                04  Rough side-mill pocket at (-0.25, 3.00)
                                    length 0.40, width 0.30, depth 0.50
                                05  Finish side-mill pocket at (-0.25, 3.00)
                                    length 0.40, width 0.30, depth 0.50
                                01  Install 0.08-diameter end-milling tool
                                [...]
                                01  Total time on VMC1
                                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   MC1 30.00     0.29      01  Setup
                                02  Screenprint solder stop on board
006 C   MC1 30.00     7.50      01  Setup
```

A portion of a manufacturing process plan

Generating Plans of Action



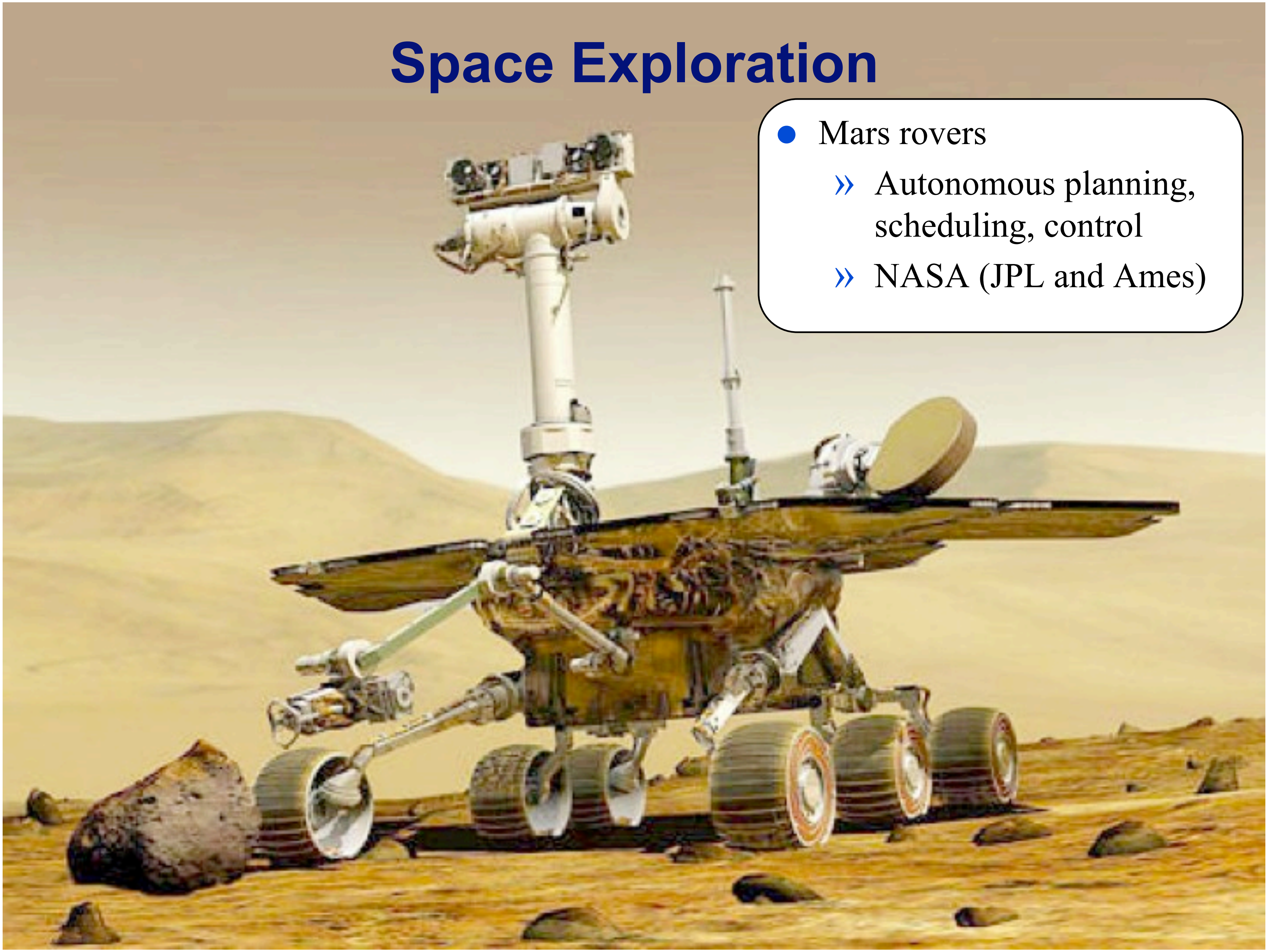


Opn	A	BC/WW	Setup	Runtime	LN	Description
001	A	VMC1	2.00	0.00	01	Orient board
					02	Clamp board
					03	Establish datum point at bullseye (0.25, 1.00)
001	B	VMC1	0.10	0.43	01	Install 0.30-diameter drill bit
					02	Rough drill at (1.25, -0.50) to depth 1.00
					03	Finish drill at (1.25, -0.50) to depth 1.00
001	C	VMC1	0.10	0.77	01	Install 0.20-diameter drill bit
					02	Rough drill at (0.00, 4.88) to depth 1.00
					03	Finish drill at (0.00, 4.88) to depth 1.00
					[...]	
001	T	VMC1	2.20	1.20	01	Total time on VMC1
[...]						
004	A	VMC1	2.00	0.00	01	Orient board
					02	Clamp board
					03	Establish datum point at bullseye (0.25, 1.00)
004	B	VMC1	0.10	0.34	01	Install 0.15-diameter side-milling tool
					02	Rough side-mill pocket at (-0.25, 1.25) length 0.40, width 0.30, depth 0.50
					03	Finish side-mill pocket at (-0.25, 1.25) length 0.40, width 0.30, depth 0.50
					04	Rough side-mill pocket at (-0.25, 3.00) length 0.40, width 0.30, depth 0.50
					05	Finish side-mill pocket at (-0.25, 3.00) length 0.40, width 0.30, depth 0.50
004	C	VMC1	0.10	1.54	01	Install 0.08-diameter 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	MC1	30.00	0.29	01	Setup
					02	Screenprint solder stop on board
006	C	MC1	30.00	7.50	01	Setup
					02	Deposit solder paste at (3.35,1.23) on board using nozzle
					[...]	
					31	Deposit solder paste at (3.52,4.00) on board using nozzle
006	D	MC1	0.00	5.71	01	Dry board in oven at 85 deg. F to solidify solder paste
006	T	MC1	90.00	18.07	01	Total time on MC1
[...]						
011	A	TC1	0.00	35.00	01	Perform post-cap testing on board
011	B	TC1	0.00	29.67	01	Perform final inspection of board
011	T	TC1	0.00	64.67	01	Total time on TC1
999	T		319.70	403.37	01	Total time to manufacture

- Computer programs to aid human planners
  - Project management (consumer software)
  - Plan storage and retrieval
    - e.g., ***variant process planning*** in manufacturing
  - Automatic schedule generation
    - various OR and AI techniques
- For some problems, we would like generate plans (or pieces of plans) automatically
  - Much more difficult
  - Automated-planning research is starting to pay off
- Here are some examples …

Space Exploration

- Mars rovers
  - Autonomous planning, scheduling, control
  - NASA (JPL and Ames)

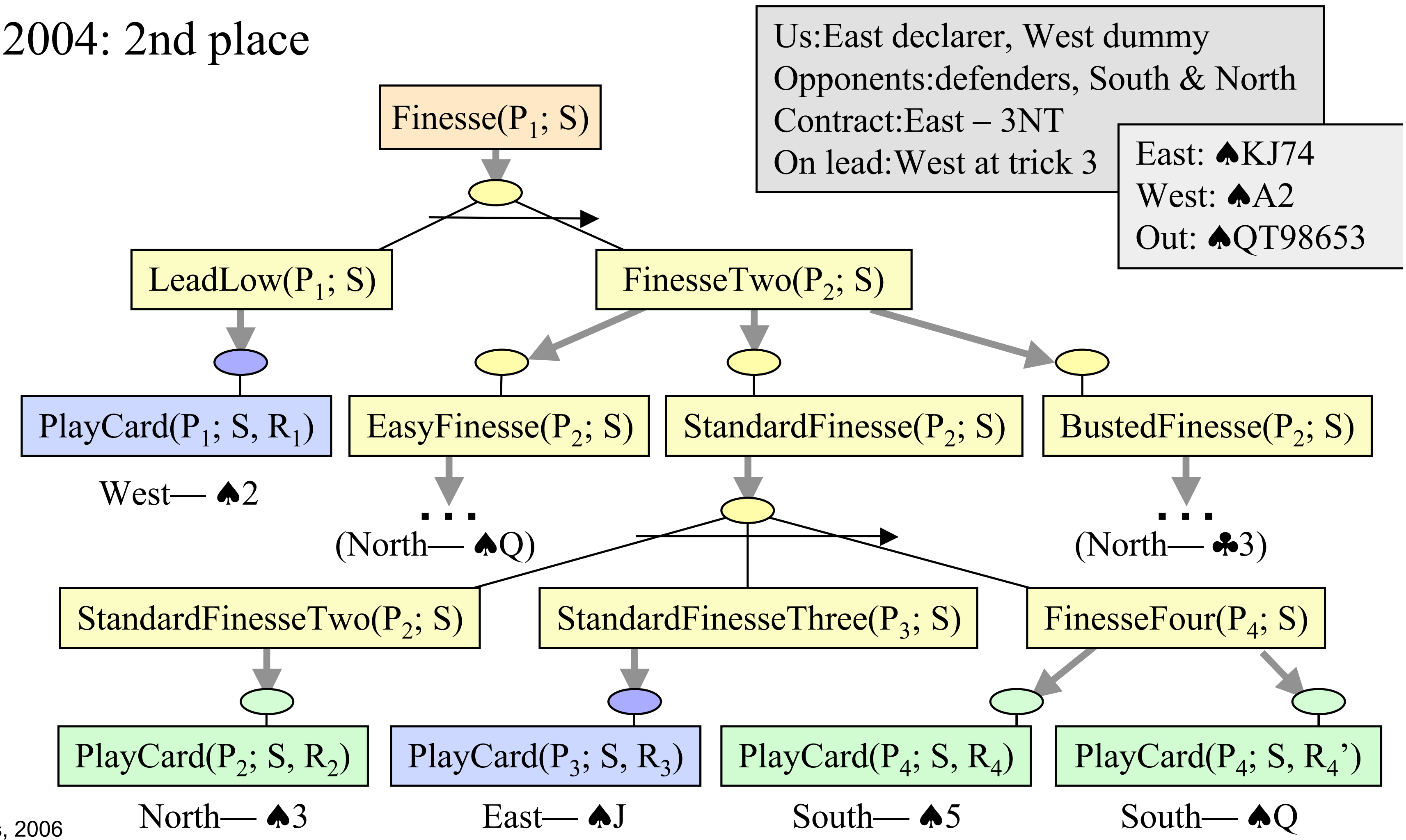


Manufacturing

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

Games

- *Bridge Baron* - Great Game Products
  - » 1997 world champion of computer bridge [Smith, Nau, and Throop, *AI Magazine*, 1998]
  - » 2004: 2nd place

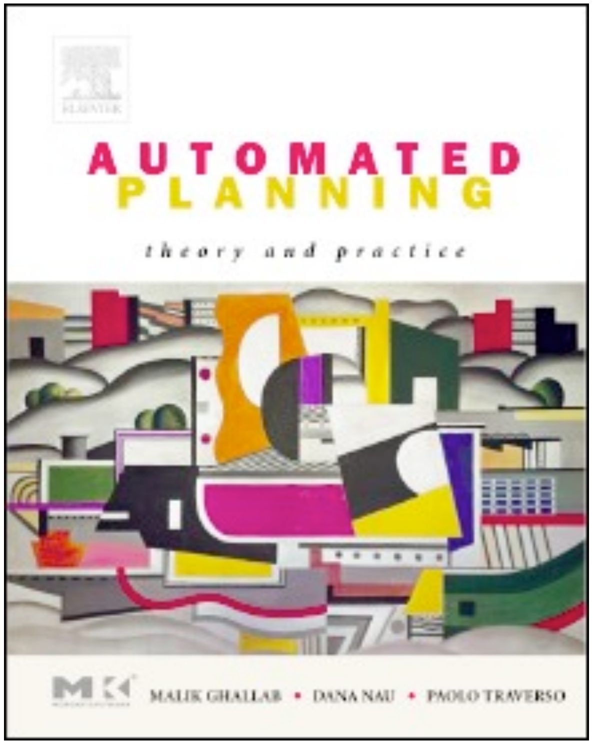


Outline

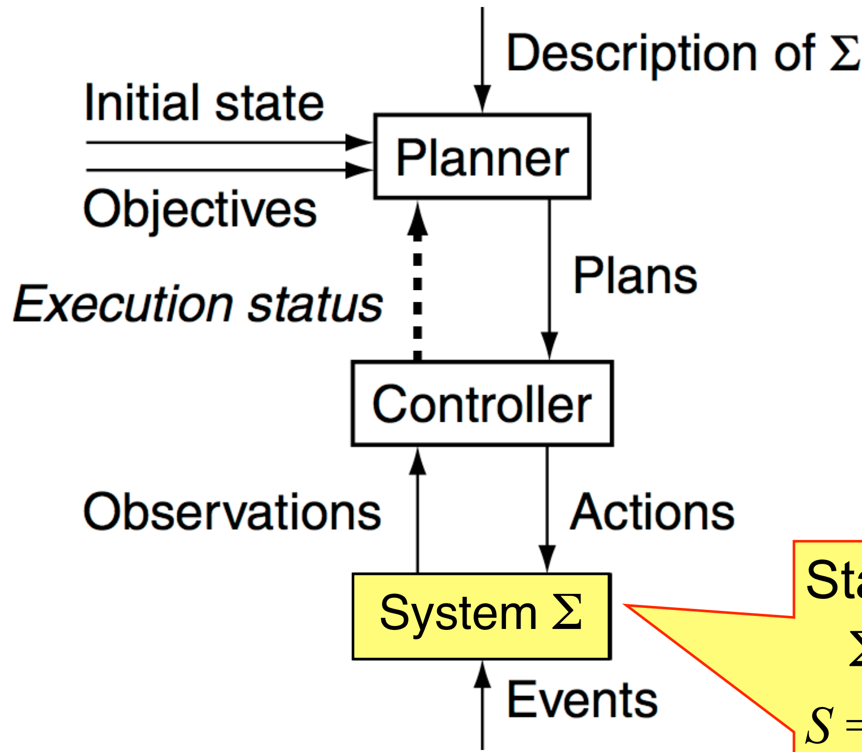
- Conceptual model for planning
- Example planning algorithms
- What's bad
- What's good
- Directions and trends

Related Reading

- My talk today is deliberately non-technical
- For technical details:
  - » Ghallab, Nau, and Traverso
    Automated Planning: Theory and Practice
    Morgan Kaufmann, May 2004
  - » First comprehensive
    textbook & reference book
    on automated planning
  - » http://www.laas.fr/planning



Conceptual Model
1. Environment



State transition system
$\Sigma = (S, A, E, \gamma)$

S = {states}

A = {actions}

E = {exogenous events}

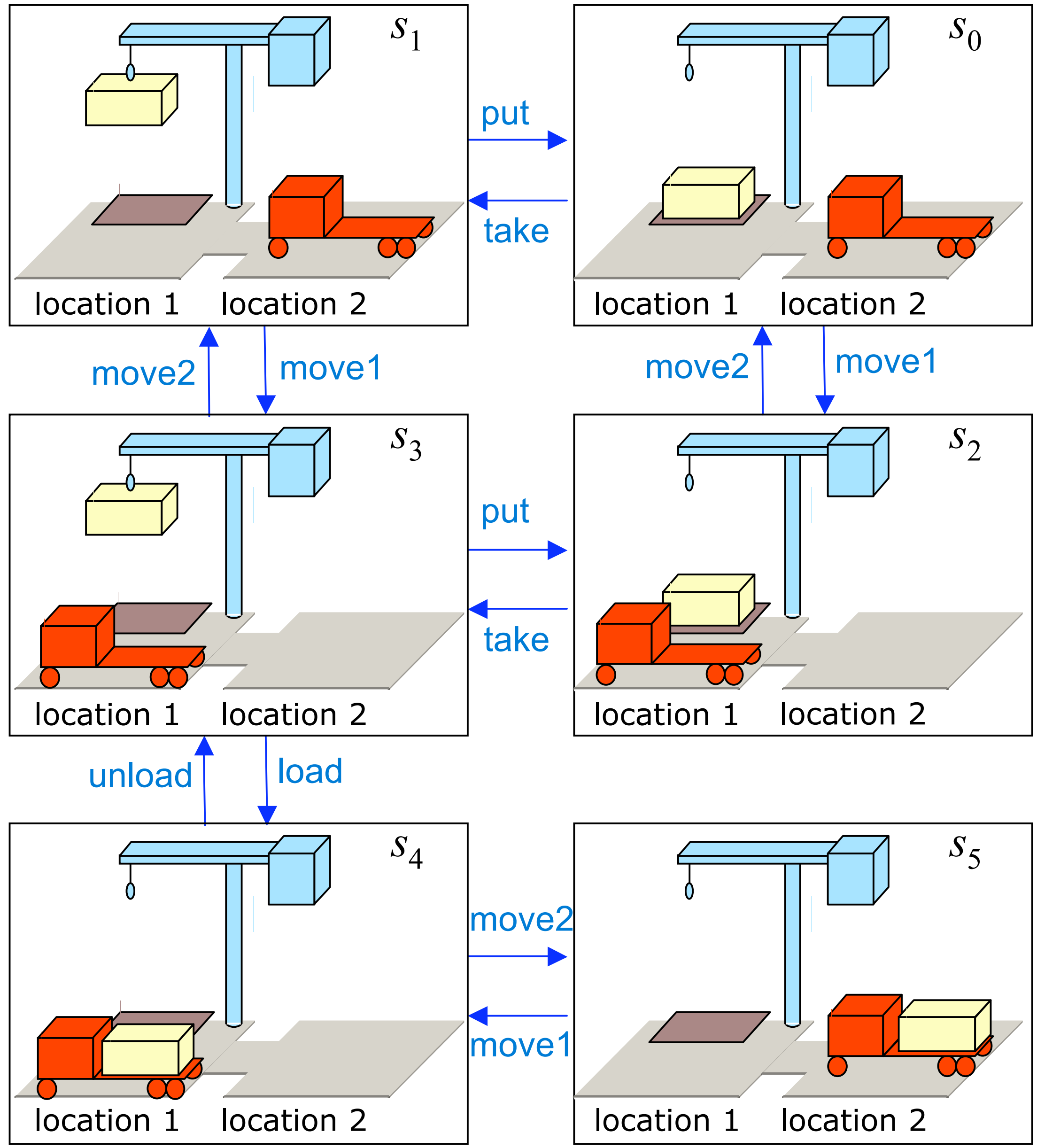
γ = state-transition function

State Transition System

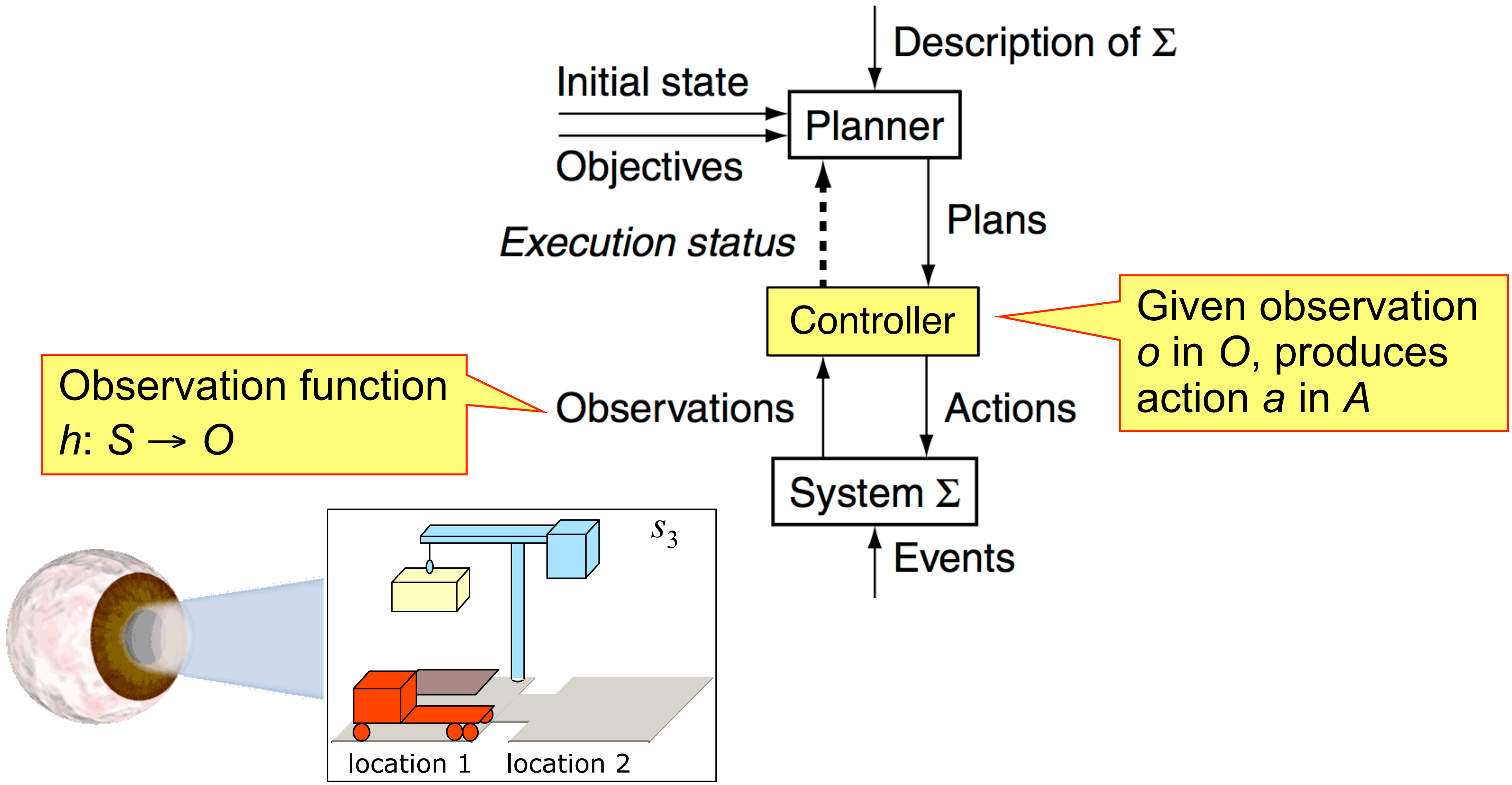
$\Sigma = (S, A, E, \gamma)$

- S = {states}
- A = {actions}
- E = {exogenous events}
- γ = state-transition function

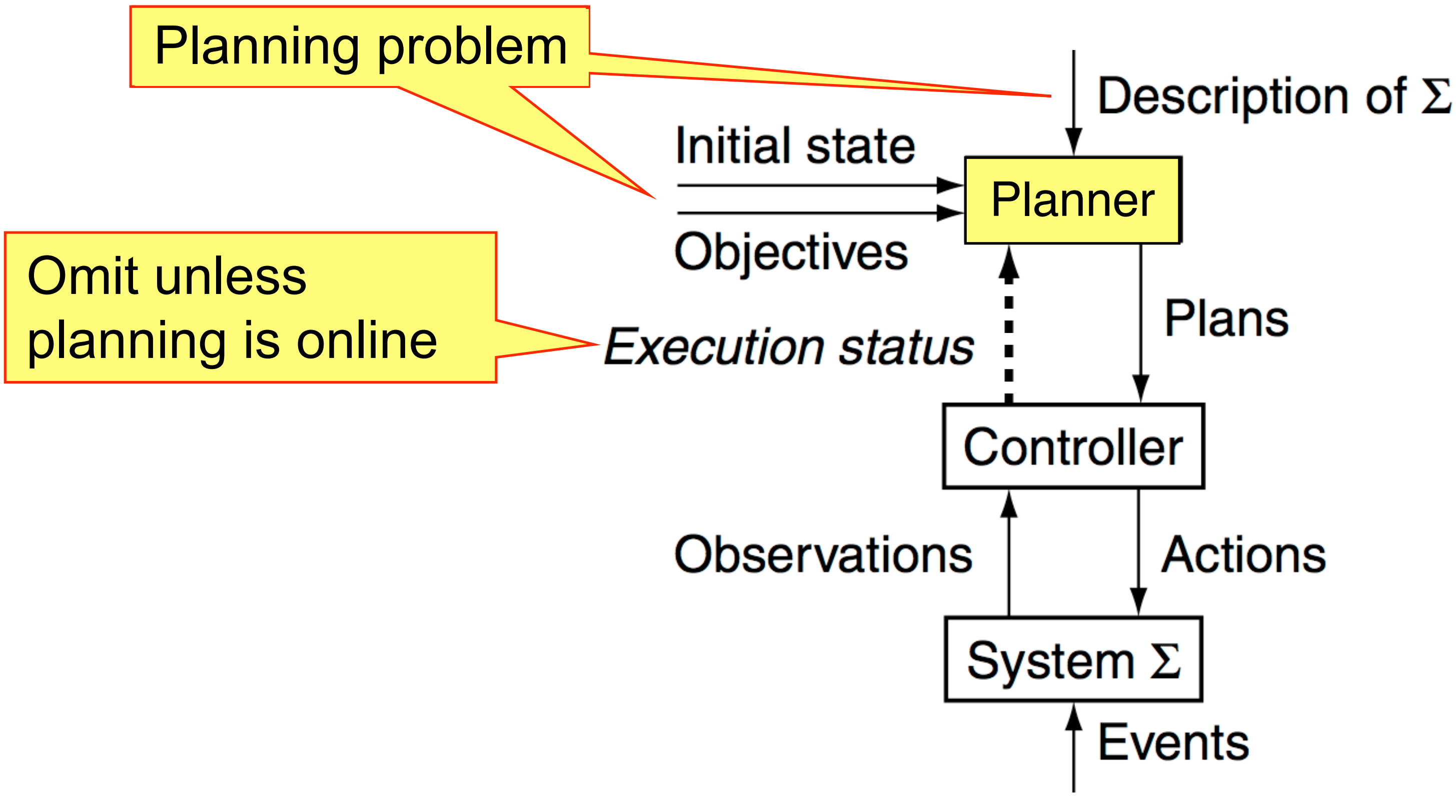
- Example:
  - » $S = \{s_0, \ldots, s_5\}$
  - » A = {put, take, load, …}
  - » $E = \varnothing$
  - » γ: see the arrows



Conceptual Model
2. Controller

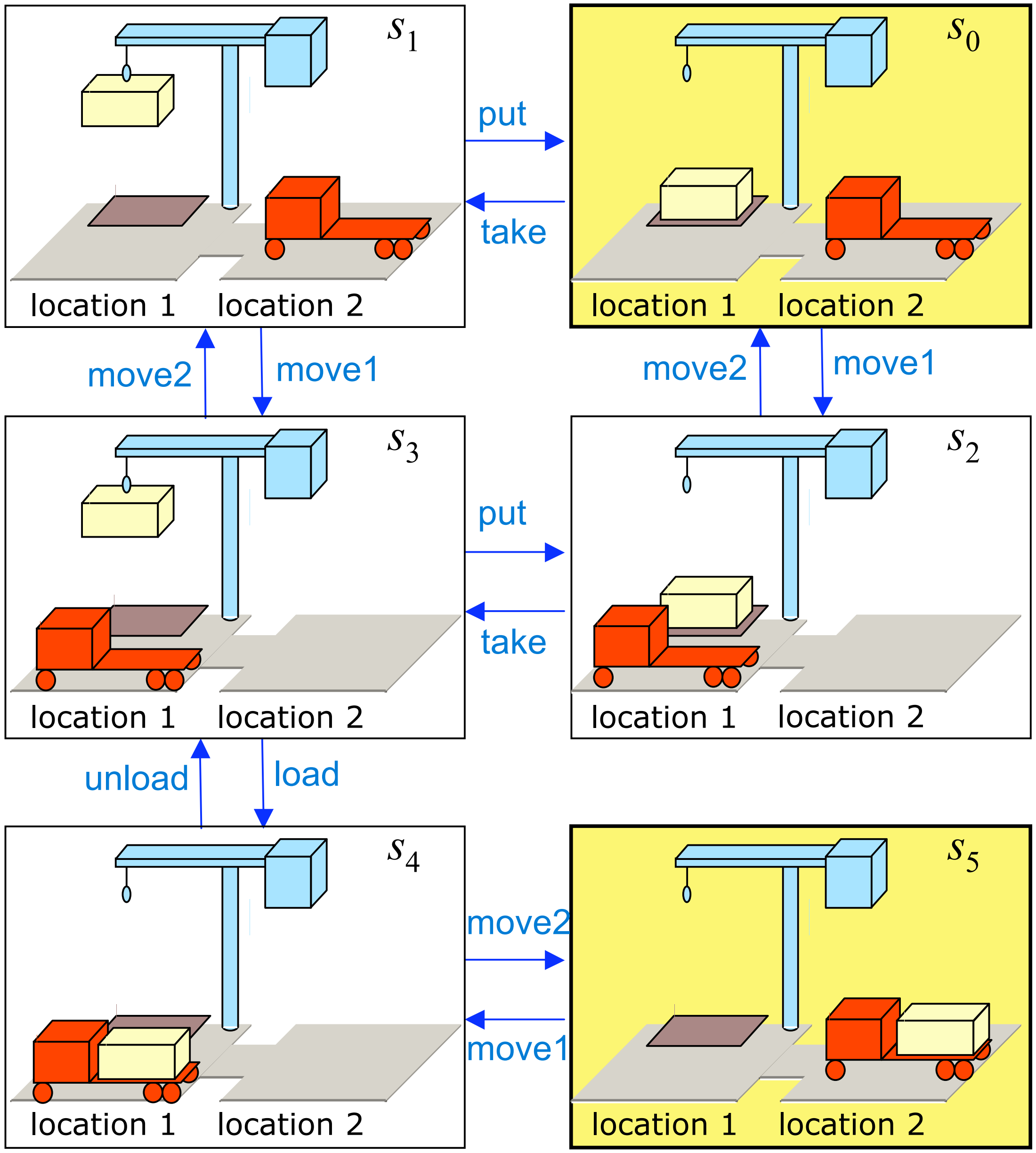


Conceptual Model
3. Planner's Input

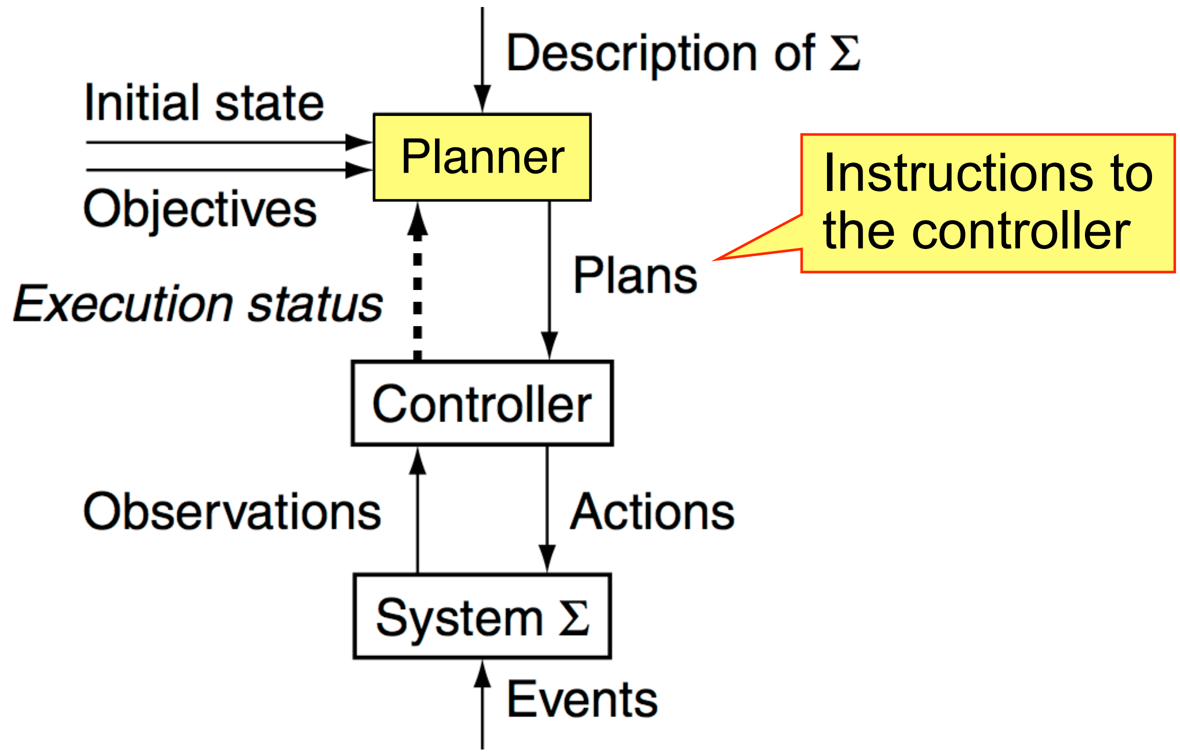


Planning Problem

- Description of Σ
- Initial state or set of states
  - » Initial state = s_0
- Objective
  - » Goal state, set of goal states, set of tasks, "trajectory" of states, objective function, …
  - » Goal state = s_5



Conceptual Model
4. Planner's Output



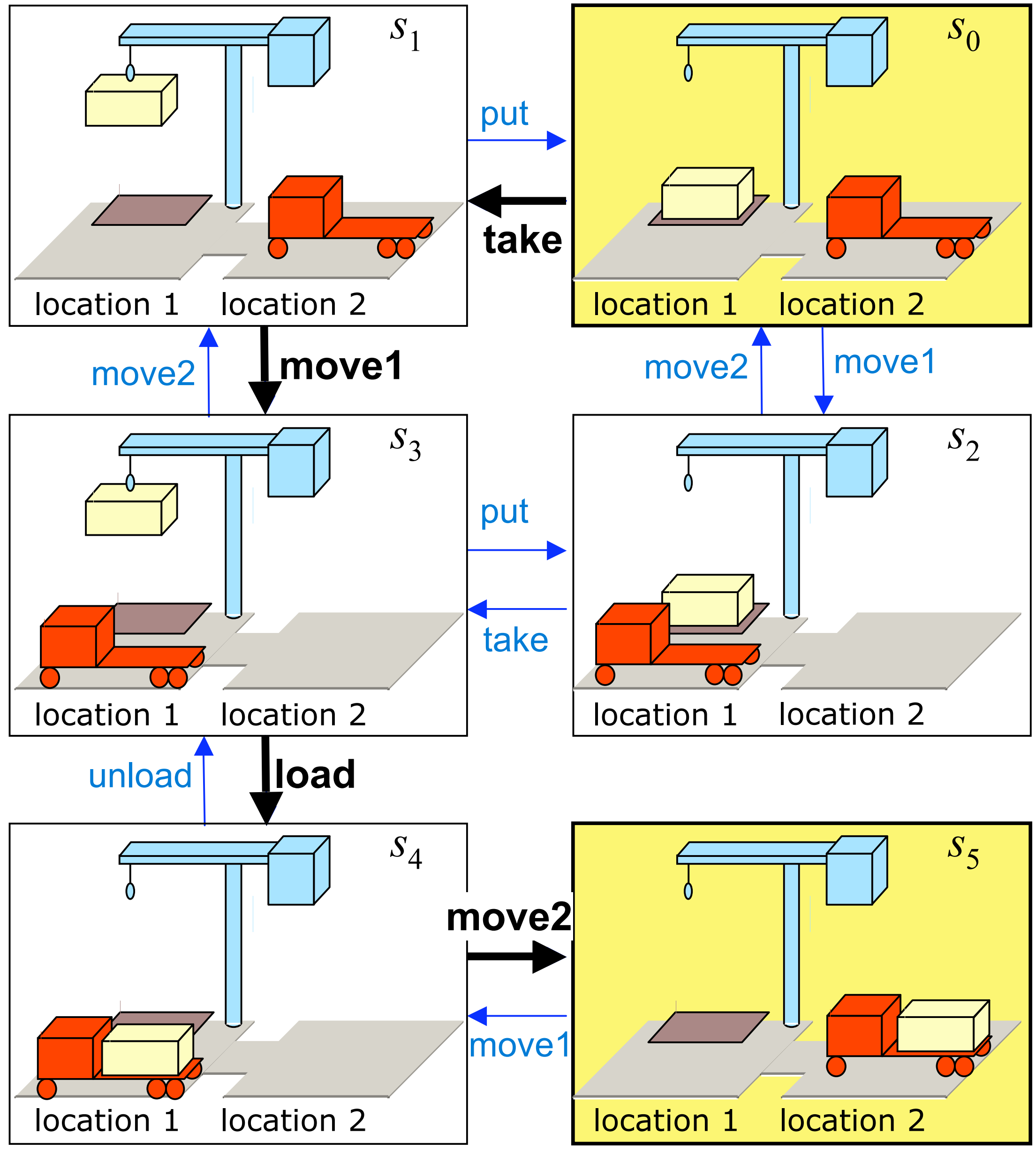
Plans

- **Classical plan**: a sequence of actions

⟨take, move1, load, move2⟩

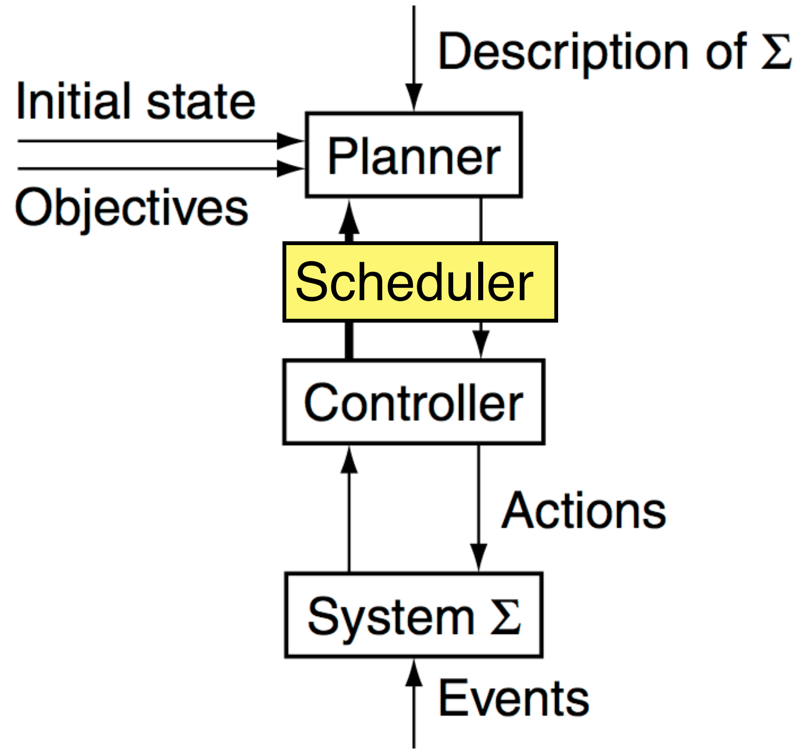
- **Policy**: partial function from S into A

{(s_0, take),
(s_1, move1),
(s_3, load),
(s_4, move2)}



Planning Versus Scheduling

- Scheduling
  - » When and how to perform a given set of actions
    - Time constraints
    - Resource constraints
    - Objective functions
  - » Typically NP-complete



- Planning
  - » Decide what actions to use to achieve some set of objectives
  - » Can be much worse than NP-complete; worst case is undecidable

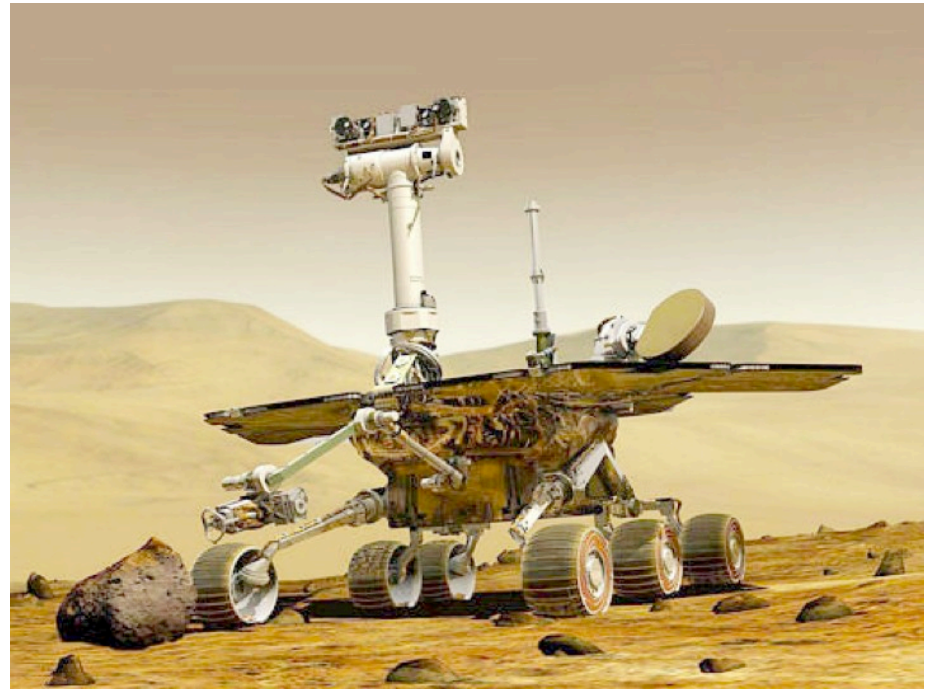
Three Main Types of Planners

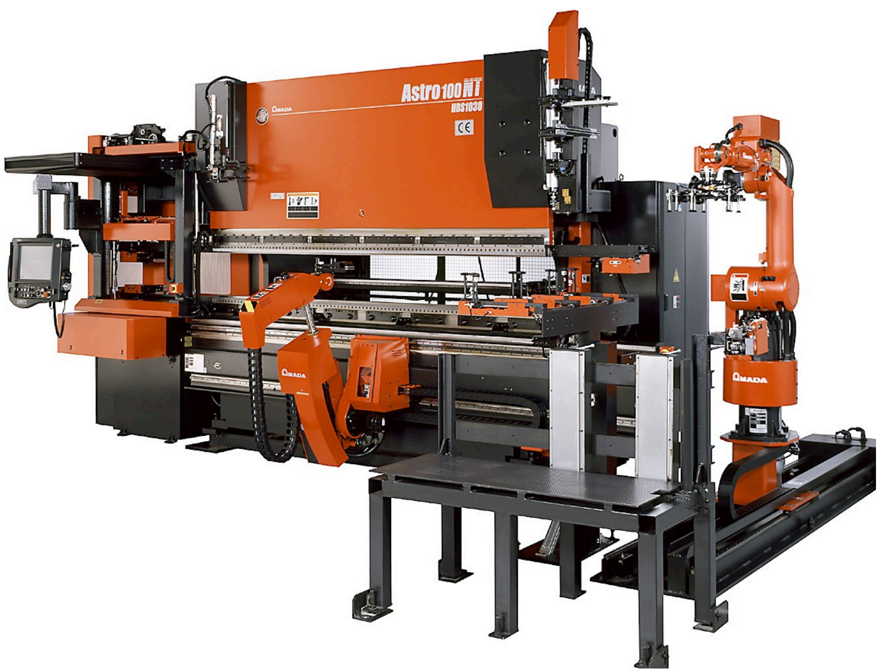
1. Domain-specific
2. Domain-independent
3. Configurable

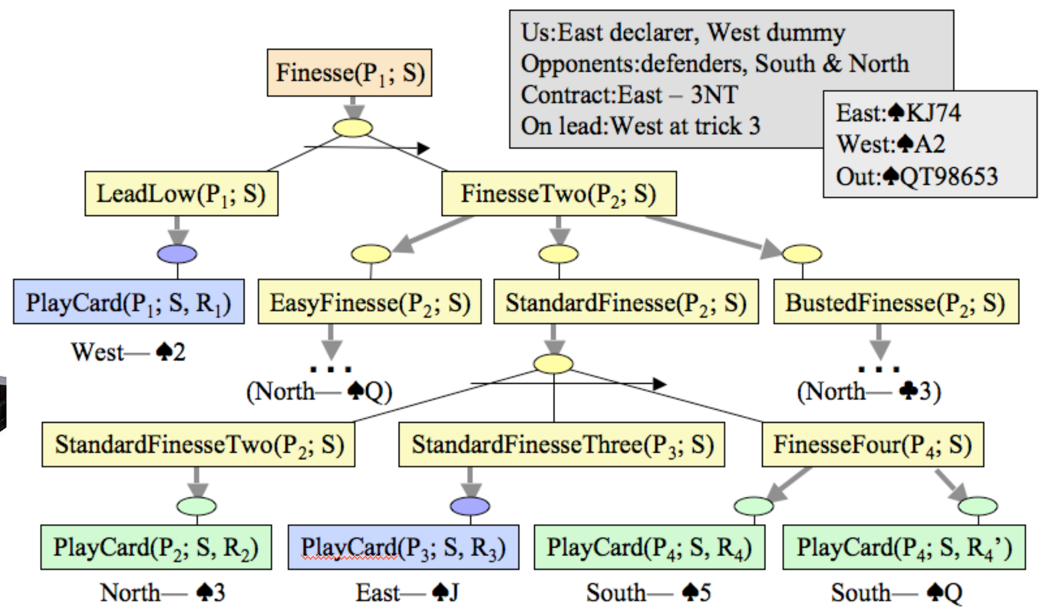
- I'll briefly discuss each

Types of Planners: 1. Domain-Specific

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

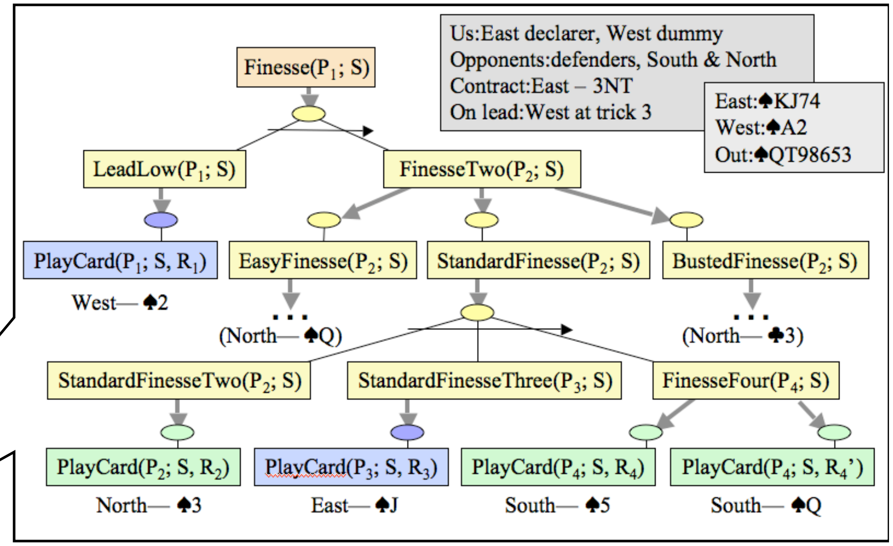


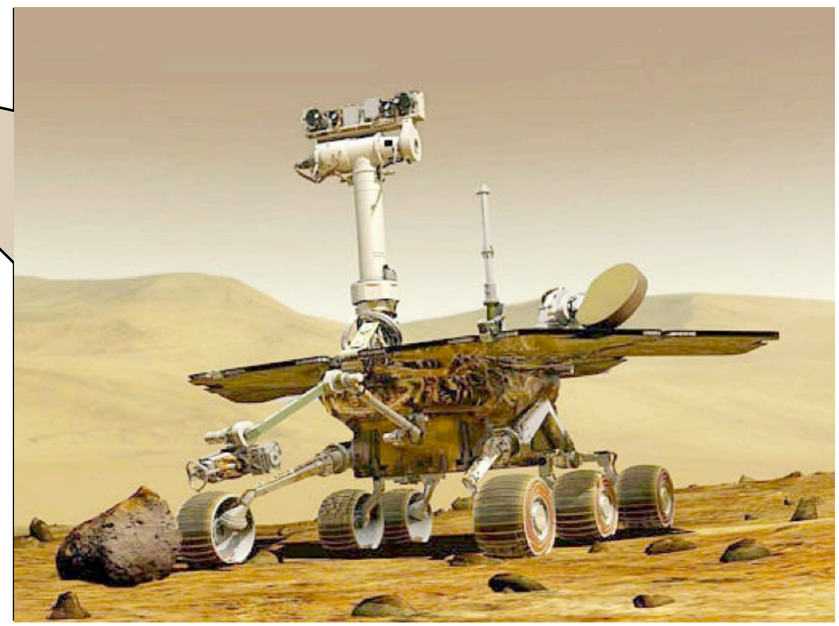




Types of Planners: 2. Domain-Independent

- In principle:
  - Works in any planning domain
  - Only domain-specific knowledge is the definitions of the basic actions
- In practice:
  - Not feasible to develop domain-independent planners that work in *every* possible domain
    - Could you to use a bridge program to explore Mars?

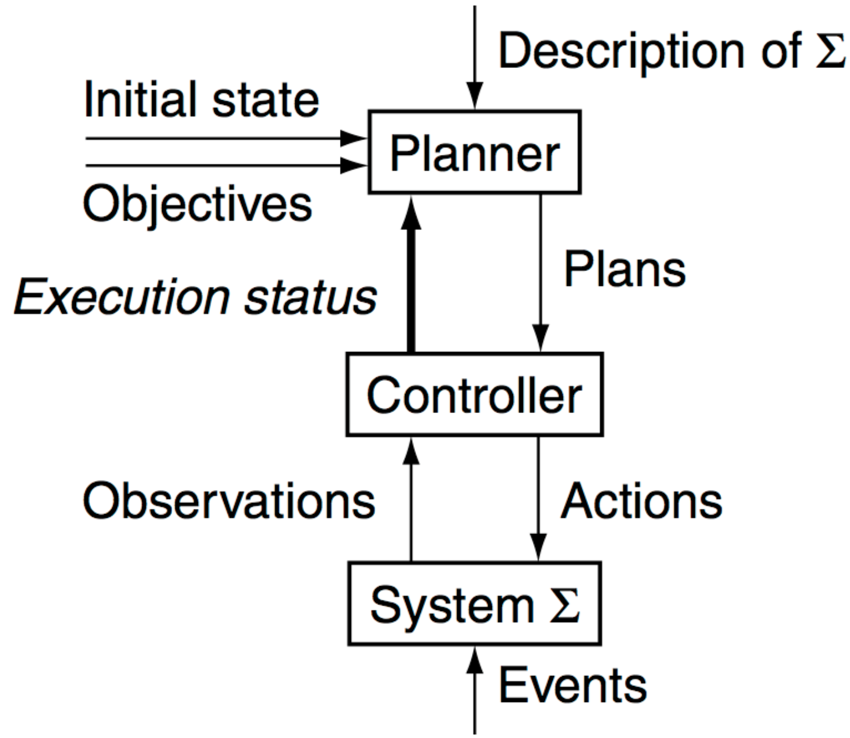




  - Restrictive assumptions to simplify the set of domains
    - *Classical planning*
    - Historical focus of most research on automated planning

Restrictive Assumptions

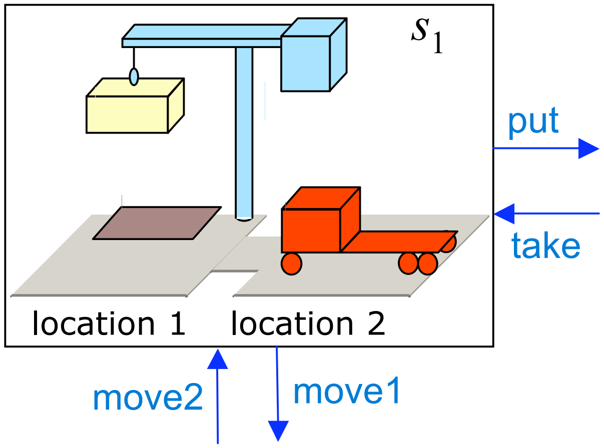
- **A0: Finite system:**
  - finitely many states, actions, events
- **A1: Fully observable:**
  - the controller always Σ's current state
- **A2: Deterministic:**
  - each action has only one outcome
- **A3: Static** (no exogenous events):
  - no changes but the controller's actions
- **A4: Attainment 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



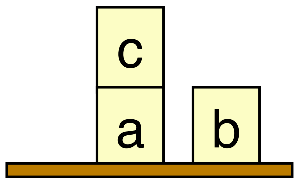
Classical Planning

- 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, S_g)
  - Find a sequence of actions $\langle a_1, a_2, \ldots a_n \rangle$ that produces a sequence of state transitions $\langle s_1, s_2, \ldots, s_n \rangle$ such that s_n is in S_g.
- This is just path-searching in a graph
  - Nodes = states
  - Edges = actions
- ***Is this trivial?***

Classical Planning

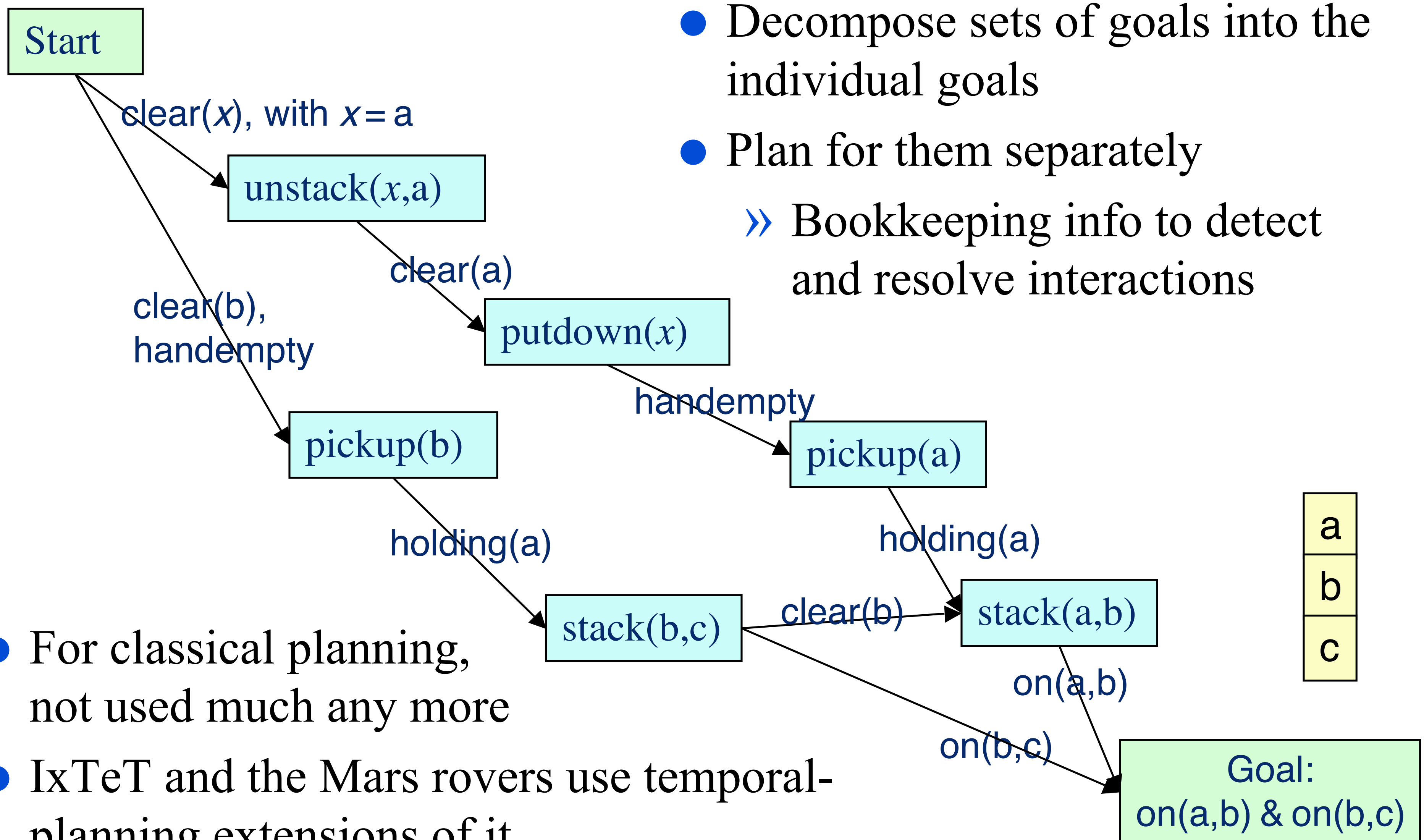


- Generalize the earlier example:
  - Five locations, three robot carts, 100 containers, three piles
    - Then there are 10^{277} states
- Number of particles in the universe is only about 10^{87}
  - The example is more than 10^{190} times as large!
- Automated-planning research has been heavily dominated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

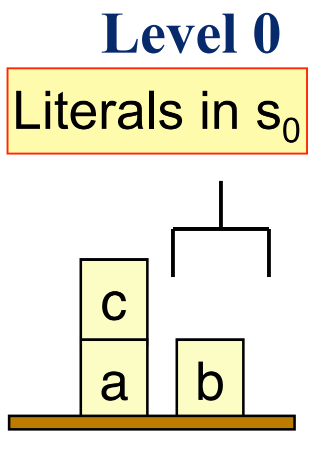


Partial-Order Planning

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

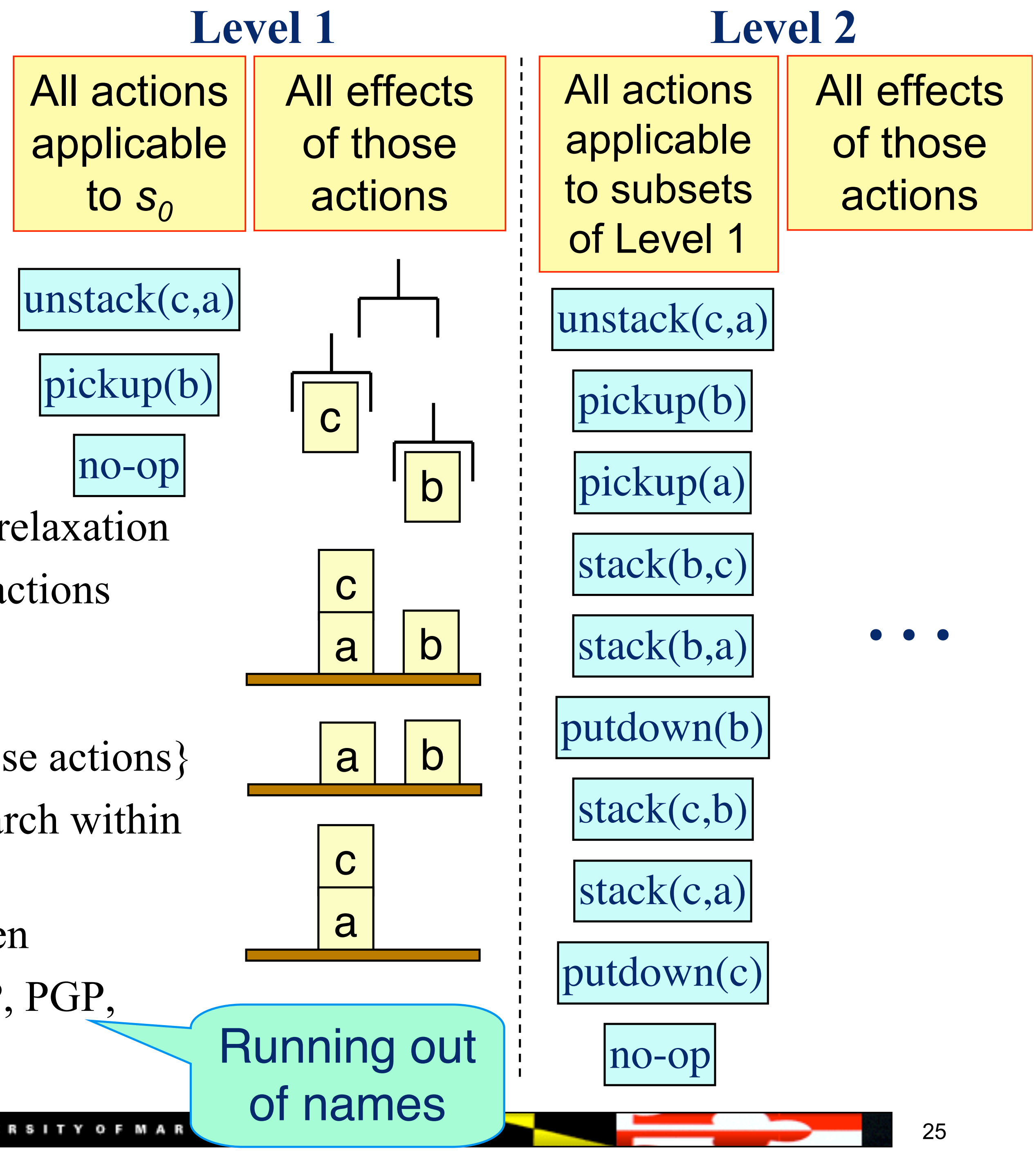


- For classical planning, not used much any more
- IxTeT and the Mars rovers use temporal-planning extensions of it



Graphplan

- Planning graph: problem relaxation
  - » Apply *all* applicable actions simultaneously
  - » Next "level" = {effects of all of those actions}
- Restrict the planner to search within the planning graph
- Graphplan's many children
  - » IPP, CGP, DGP, LGP, PGP, SGP, TGP, …



Heuristic Search

- Do an A*-style heuristic search guided by a *heuristic function* that estimates the distance to a goal
  - » Can use planning graphs to compute the heuristic function

- 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

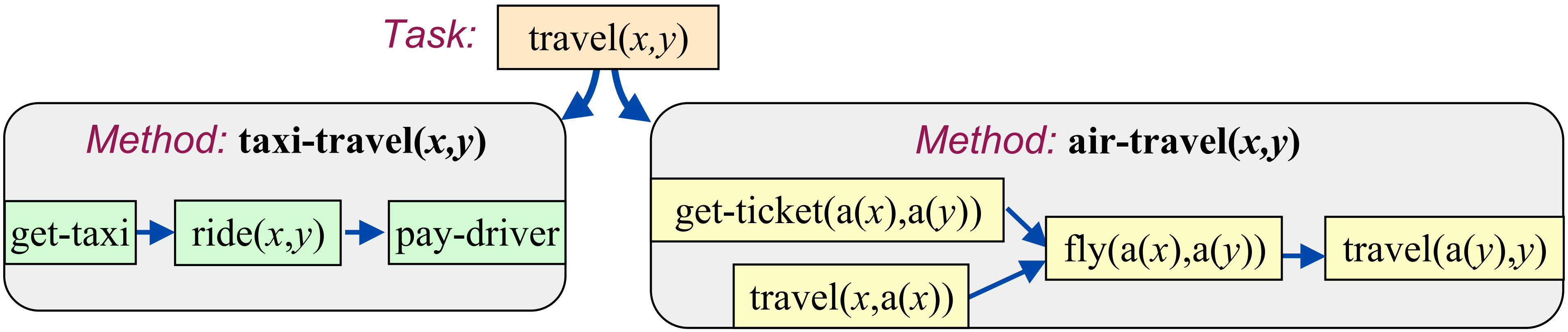
- HSP [Bonet & Geffner]
- FastForward [Hoffmann]

Translation to Other Domains

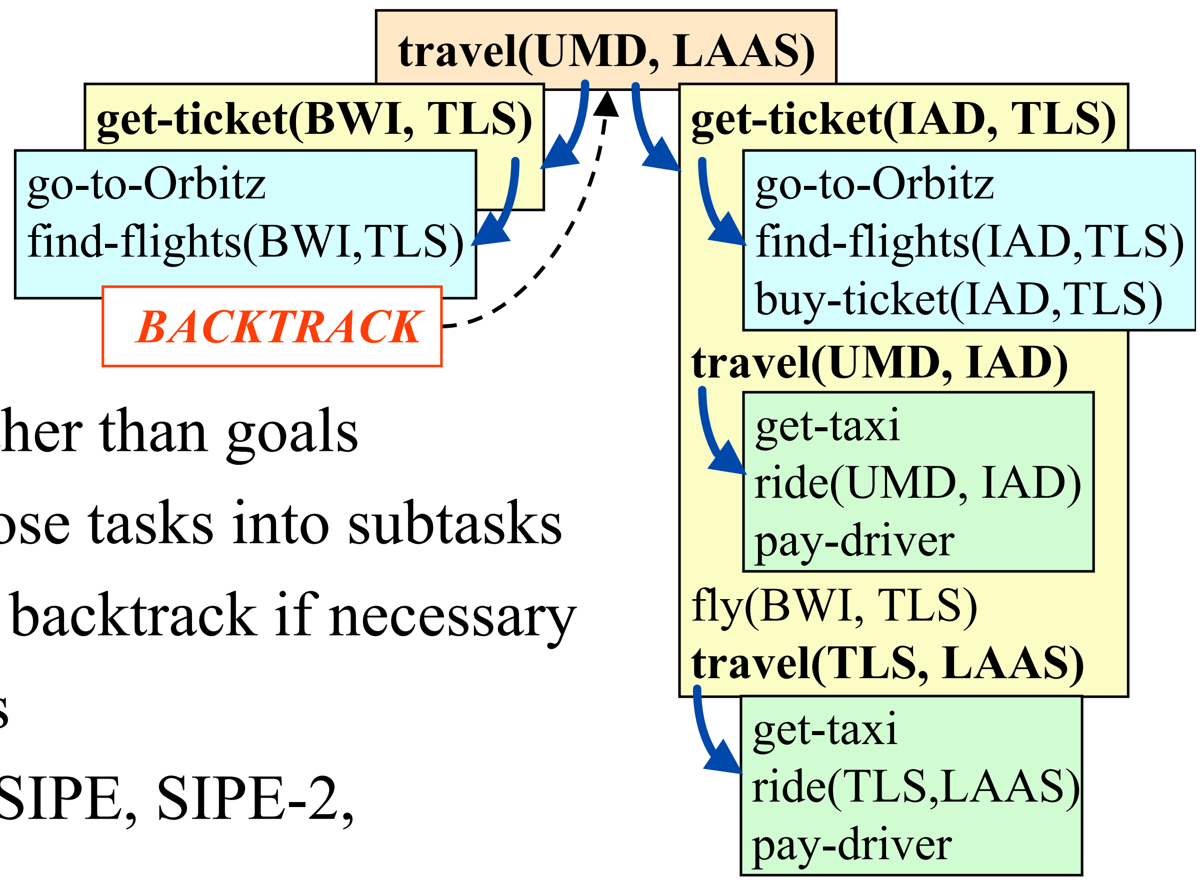
- Translate the planning problem or the planning graph into another kind of problem for which there are efficient solvers
  - » Find a solution to that problem
  - » Translate the solution back into a plan
- Satisfiability solvers, especially those that use local search
  - » Satplan and Blackbox [Kautz & Selman]
- Integer programming solvers such as Cplex
  - » [Vossen *et al.*]

Types of Planners: 3. Configurable

- 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 we don't want to write a whole new planner for every domain!
- **Configurable planners**
  - Domain-independent planning engine
  - Input includes info about how to solve problems in the domain
    - Hierarchical Task Network (HTN) planning
    - Planning with control formulas

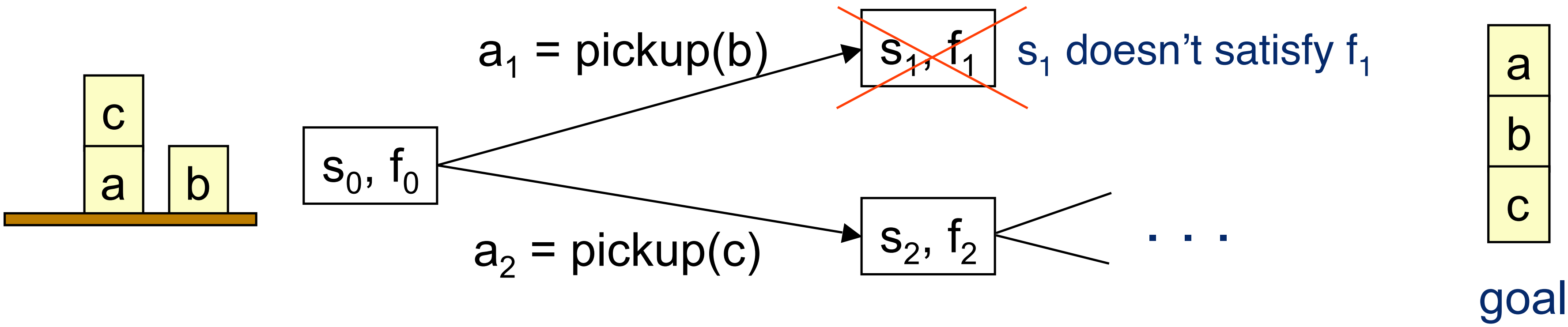


HTN Planning



- 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

Planning with Control Formulas



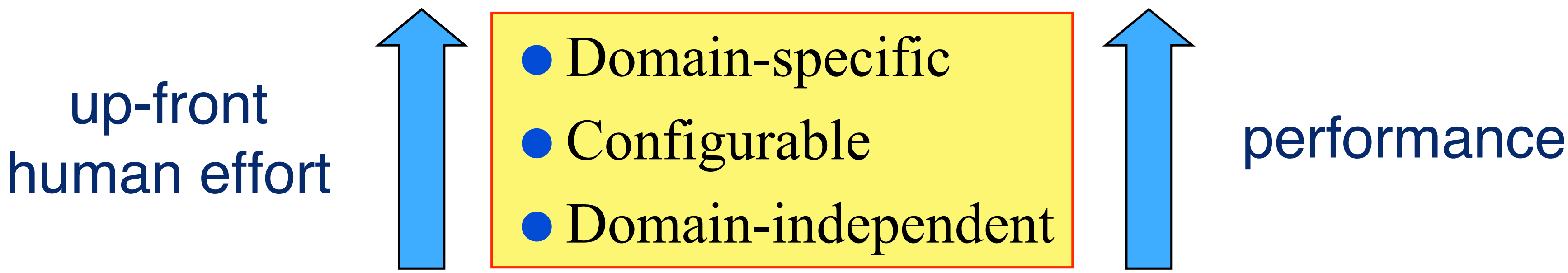
- Forward search
- At each state s_i we have a *control formula* f_i in temporal logic

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

"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]

Comparisons



- 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

Comparisons

coverage ↑

- Configurable
- Domain-independent
- Domain-specific

- A domain-specific planner only works in one domain

- **In principle**, configurable and domain-independent planners should both be able to work in any domain
- **In practice**, configurable planners work in a larger variety of domains
  - Partly due to efficiency
  - Partly due to expressive power

Example

- The planning competitions
  - » All of them included domain-independent planners
- In addition, AIPS 2000 and *IPC* 2002 included configurable planners
- The configurable planners
  - » Solved the most problems
  - » Solved them the fastest
  - » Usually found better solutions
  - » Worked in many non-classical planning domains that were beyond the scope of the domain-independent planners

AIPS 1998 Planning Competition

AIPS 2000 Planning Competition

IPC 2002

IPC 2004

IPC 2006

But Wait …

- *IPC* 2004 and *IPC* 2006 included ***no*** configurable planners.
  - Why not?

AIPS 1998 Planning Competition

AIPS 2000 Planning Competition

IPC 2002

IPC 2004

IPC 2006

But Wait …

- *IPC* 2004 and *IPC* 2006 included ***no*** configurable planners.
  - Why not?
- Hard to enter them in the competition
  - Must write all the domain knowledge yourself
  - Too much trouble except to make a point
  - The authors of TLPlan, TALplanner, and SHOP2 felt they had already made their point

AIPS 1998 Planning Competition

AIPS 2000 Planning Competition

IPC 2002

IPC 2004

IPC 2006

But Wait …

- *IPC* 2004 and *IPC* 2006 included ***no*** configurable planners.
  - » Why not?
- Hard to enter them in the competition
  - » Must write all the domain knowledge yourself
  - » Too much trouble except to make a point
  - » The authors of TLPlan, TALplanner, and SHOP2 felt they had already made their point
- Why not provide the domain knowledge?

AIPS 1998 Planning Competition

AIPS 2000 Planning Competition

IPC 2002

IPC 2004

IPC 2006

But Wait …

- *IPC* 2004 and *IPC* 2006 included ***no*** configurable planners.
  - Why not?
- Hard to enter them in the competition
  - Must write all the domain knowledge yourself
  - Too much trouble except to make a point
  - The authors of TLPlan, TALplanner, and SHOP2 felt they had already made their point
- Why not provide the domain knowledge?
  - Drew McDermott proposed this at *ICAPS-05*
  - Many people didn't like this idea
    - Cultural bias against it

AIPS 1998 Planning Competition

AIPS 2000 Planning Competition

IPC 2002

IPC 2004

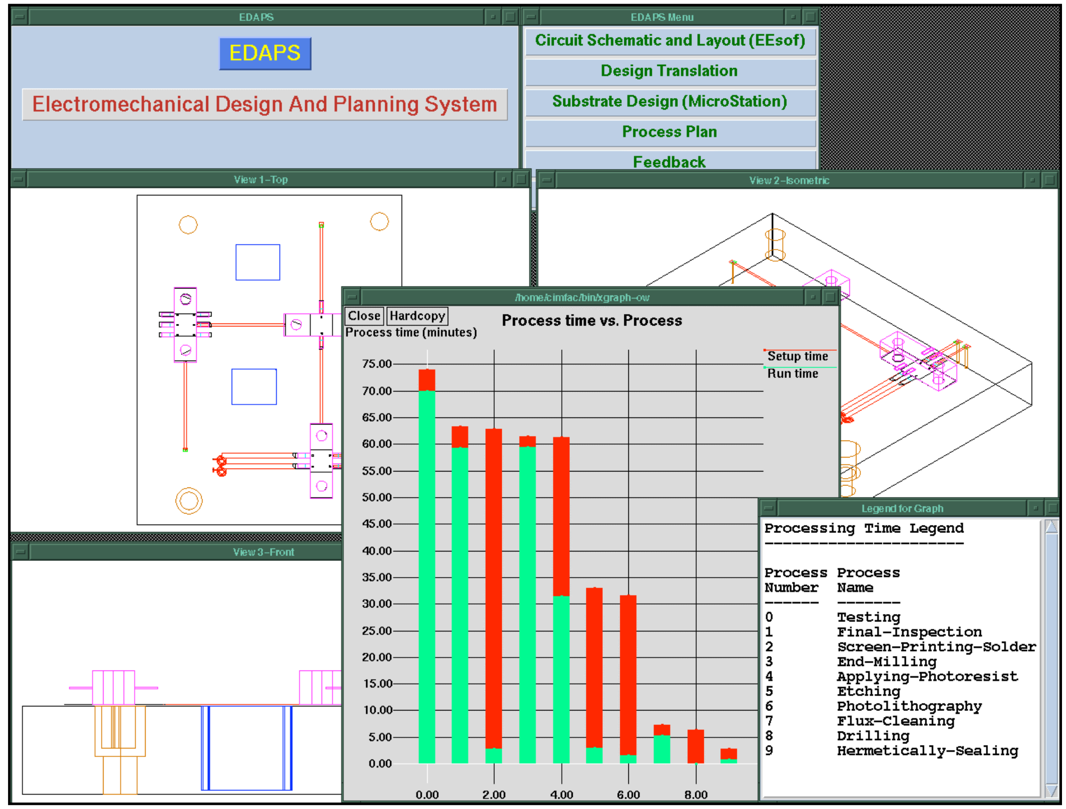
IPC 2006

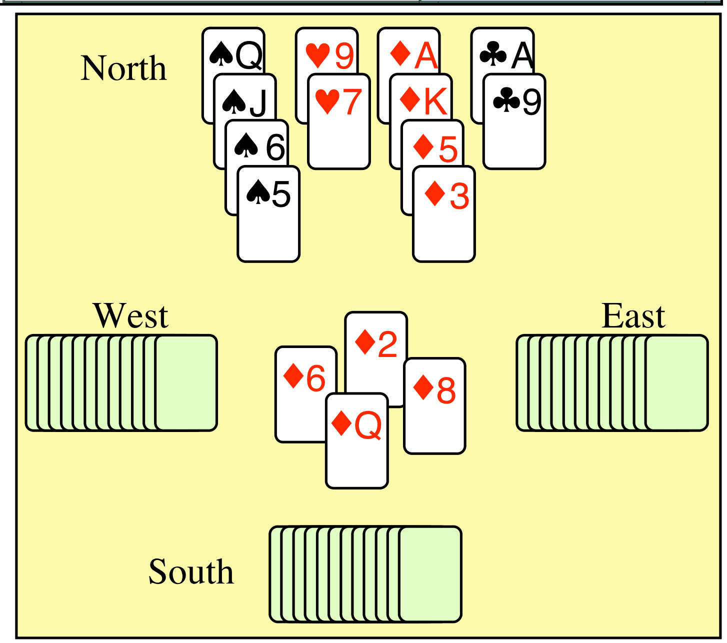
Cultural Bias

- Many (most?) automated-planning researchers feel that using domain knowledge is "cheating"
- Researchers in other fields have trouble comprehending this
  - » Operations research, control theory, engineering, …
  - » Why would anyone *not* want to use the knowledge they have about a problem they're trying to solve?
- In the past, the bias has been very useful
  - » Without it, automated planning wouldn't have grown into a separate field from its potential application areas
- But it's not useful any more
  - » The field has matured
  - » The bias is too restrictive

Example

- Typical characteristics of application domains
  - Dynamic world
  - Multiple agents
  - Imperfect/uncertain info
  - External info sources
    - users, sensors, databases
  - Durations, time constraints, asynchronous actions
  - Numeric computations
    - geometry, probability, etc.
- Classical planning excludes all of these





In Other Words …

- We ***like*** to think classical planning is domain-independent planning
- **But it isn't!**
  - Classical planning only includes domains that satisfy some **very** specific restrictions
  - Classical planners depend heavily on those restrictions
- This is fine for "toy problems" like the **blocks world**
- *Not* so fine for the **real world**

Good News, Part 1

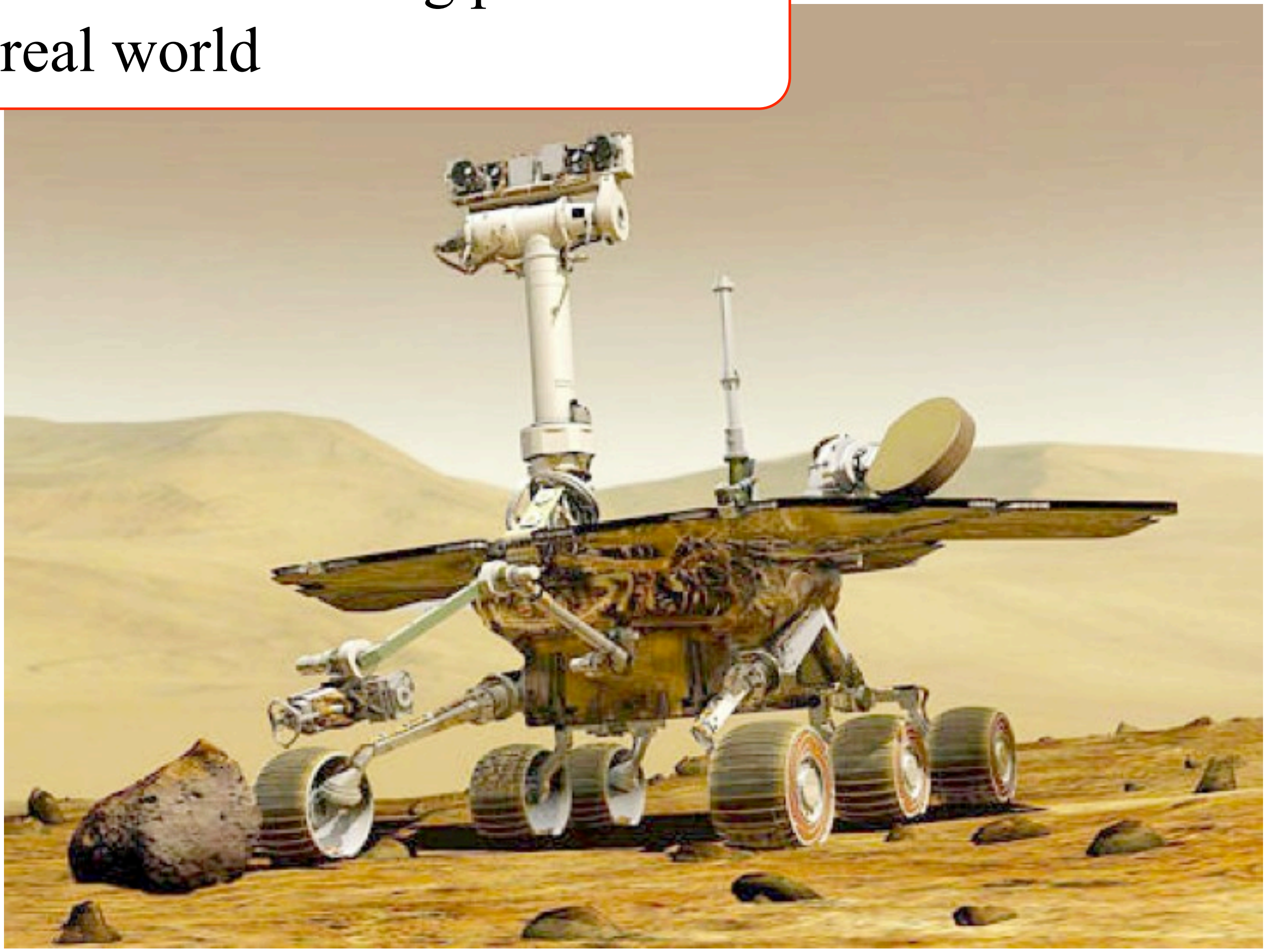
- We're already moving away from classical planning
- Example: the planning competitions
  - » AIPS 1998, AIPS 2000, *IPC* 2002, *IPC* 2004
- Increasing divergence from classical planning
  - » 1998, 2000: classical planning
  - » 2002: added elementary notions of time durations, resources
  - » 2004: added inference rules, derived effects, and a separate track for planning under uncertainty
  - » 2006: added soft goals, trajectory constraints, preferences, plan metrics

AIPS 1998 Planning Competition

AIPS 2000 Planning Competition

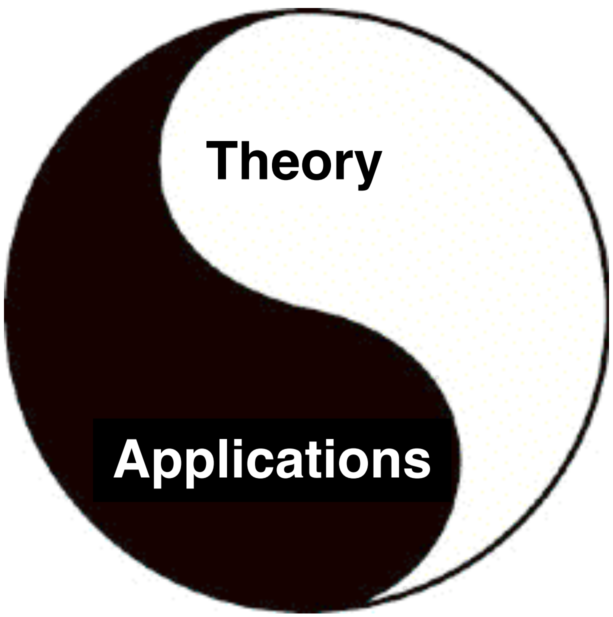
Good News, Part 2

- Success in high-profile applications
  - A success like the Mars rovers is a big deal
  - Creates excitement about building planners that work in the real world

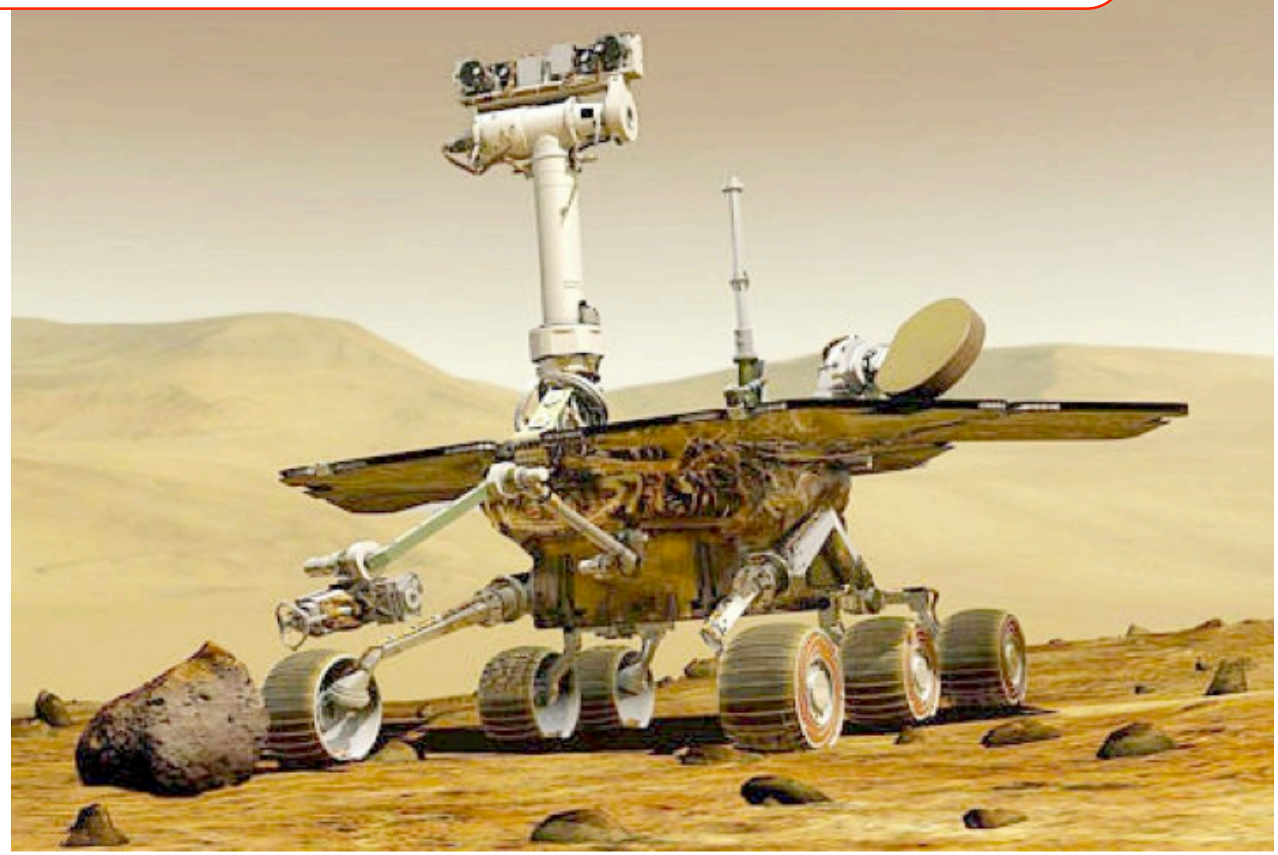


Good News, Part 3

- These successes provide opportunities for synergy between theory and practice
  - » Understanding real-world planning leads to better theories
  - » Better theories lead to better real-world planners



Nau: Plans, 2006

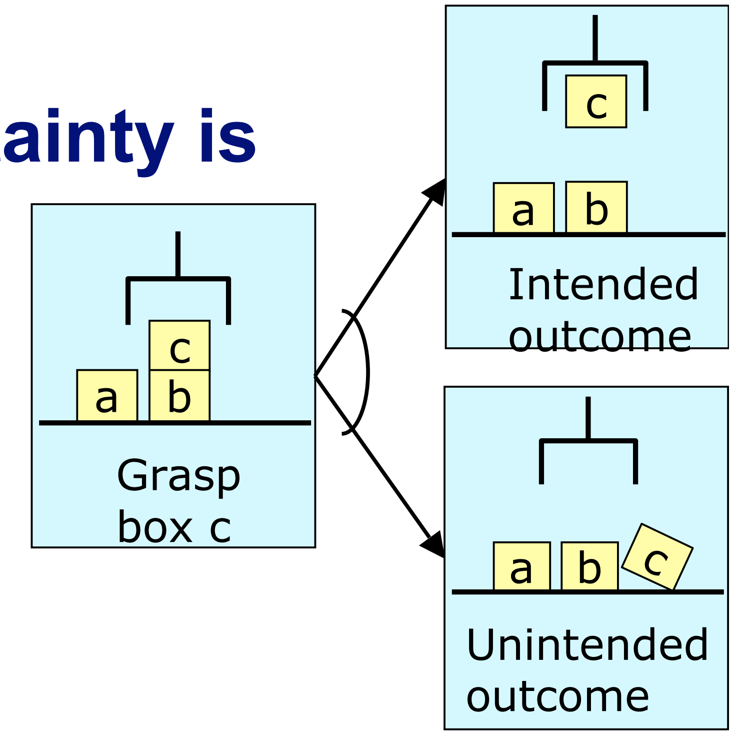


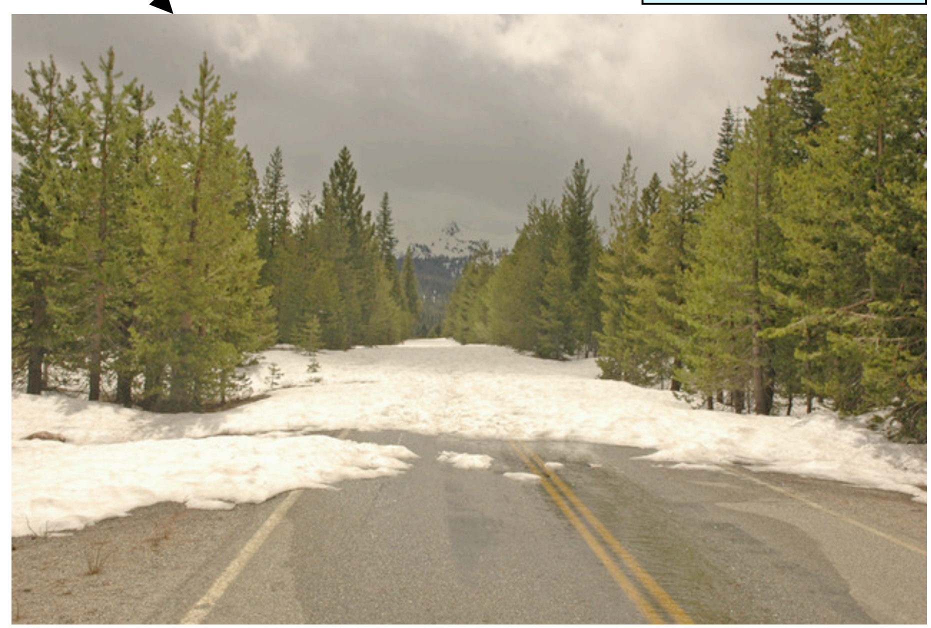
Good News, Part 4

- Classical planning research has produced some very powerful techniques for reducing the size of the search space
- We can generalize these techniques to work in non-classical domains
- Examples:
  1. Partial order planning
     - Extended to do temporal planning
       - Mars rovers
  2. HTN planning
     - Lots of applications
  3. Planning under uncertainty …

Digression:
What planning under uncertainty is

- Actions with several possible outcomes
  - Action failures, e.g., *gripper drops its load*
  - Exogenous events, e.g., *road closed*
- Primary models
  - Markov Decision Processes (MDPs)
    - Probabilities, costs, rewards, optimize expected utility
    - Dynamic programming
  - Nondeterministic planning domains
    - No numbers
    - Solutions: weak, strong, strong-cyclic, …
    - Symbolic model checking
  - Game-theoretic
    - game-tree search (e.g., minimax)



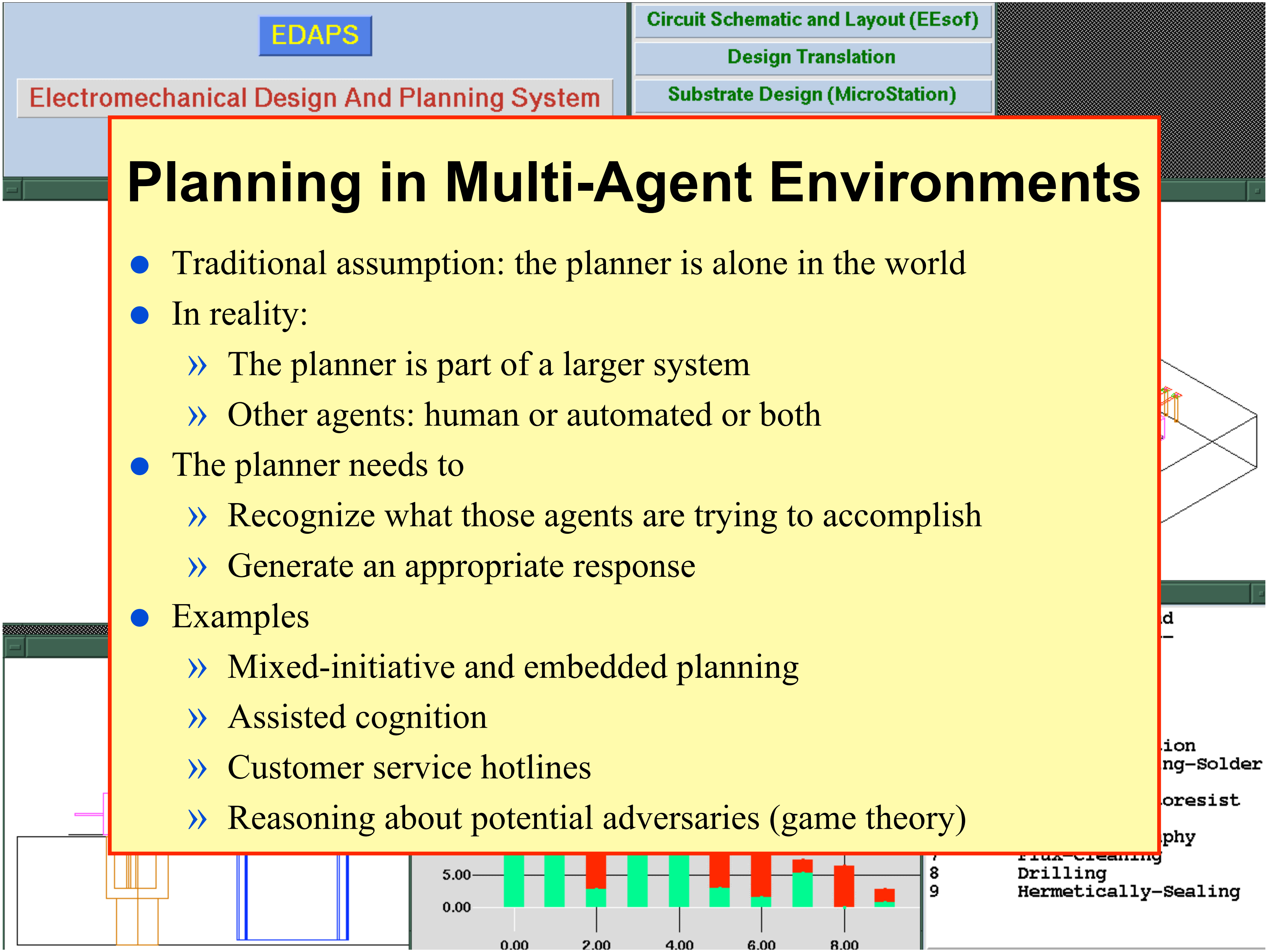


Good News, Part 4 (continued)

3. General way to *nondeterminize* forward-chaining planners
   - Rewrite them to work in nondeterministic domains
     - TLPlan → ND-TLPlan
     - TALplanner → ND-TALplanner
     - SHOP2 → ND-SHOP2
   - Big (exponential) speedups compared to previous planners for nondeterministic domains [Kuter and Nau, *AAAI*-04]
   - Even bigger speedups if we use the BDD representation used in the previous planners for nondeterministic domains
     - [Kuter, Nau, Pistore, and Traverso, *ICAPS*-05]

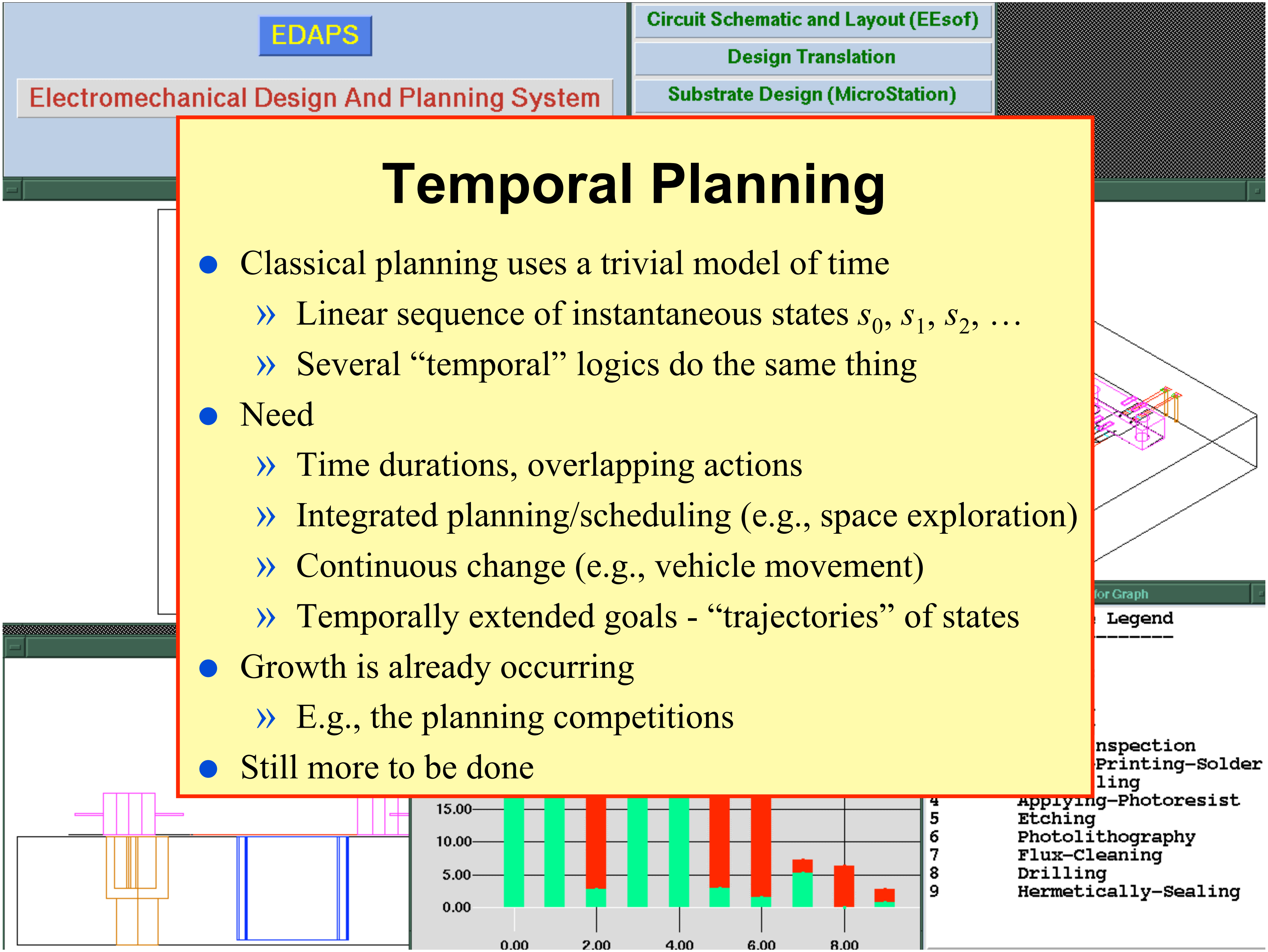
- Analogous results for MDPs [Kuter and Nau, *AAAI*-05]
- Possible extension to game-theoretic environments?

Important Trends, and Directions for Growth



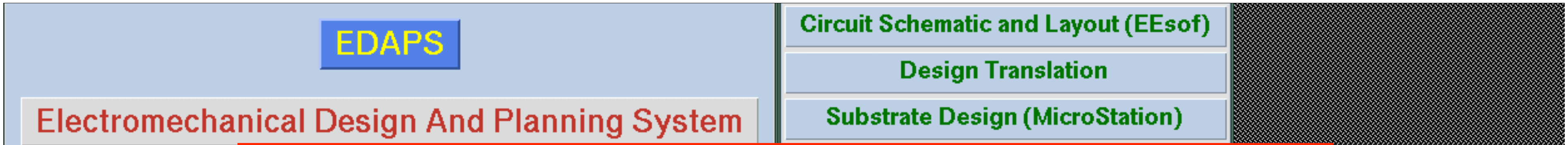
Planning in Multi-Agent Environments

- Traditional assumption: the planner is alone in the world
- In reality:
  - The planner is part of a larger system
  - Other agents: human or automated or both
- The planner needs to
  - Recognize what those agents are trying to accomplish
  - Generate an appropriate response
- Examples
  - Mixed-initiative and embedded planning
  - Assisted cognition
  - Customer service hotlines
  - Reasoning about potential adversaries (game theory)



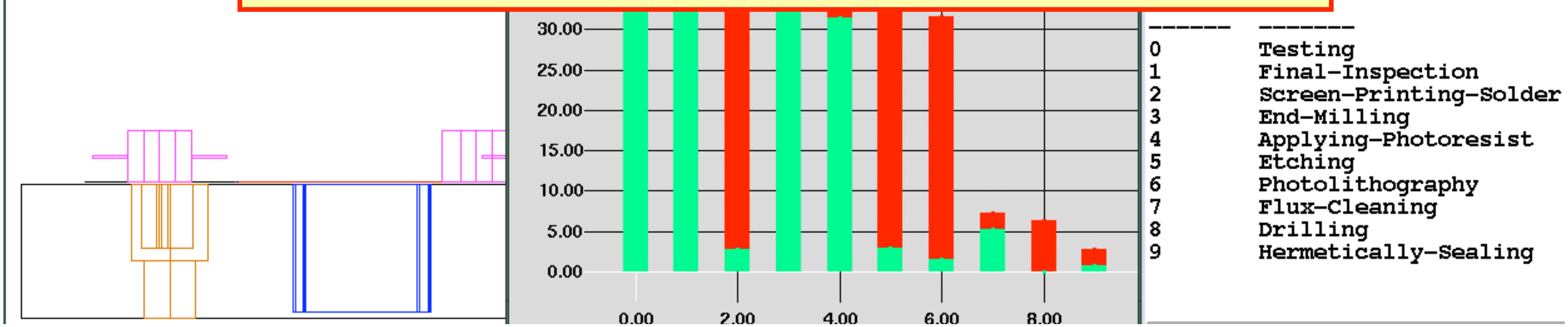
Temporal Planning

- Classical planning uses a trivial model of time
  - » Linear sequence of instantaneous states s_0, s_1, s_2, …
  - » Several “temporal” logics do the same thing
- Need
  - » Time durations, overlapping actions
  - » Integrated planning/scheduling (e.g., space exploration)
  - » Continuous change (e.g., vehicle movement)
  - » Temporally extended goals - “trajectories” of states
- Growth is already occurring
  - » E.g., the planning competitions
- Still more to be done



Dynamic External Information

- Traditional assumption
  - » Information is static; planner starts with all of it
- Real-world planning
  - » Acquire information during planning and execution
    - Applications: web services, many others
  - » What info to look for? Where to get it?
  - » How to deal with lag time and information volatility?
  - » What if the query itself causes change in the world?
- Candidate for a new IPC track?

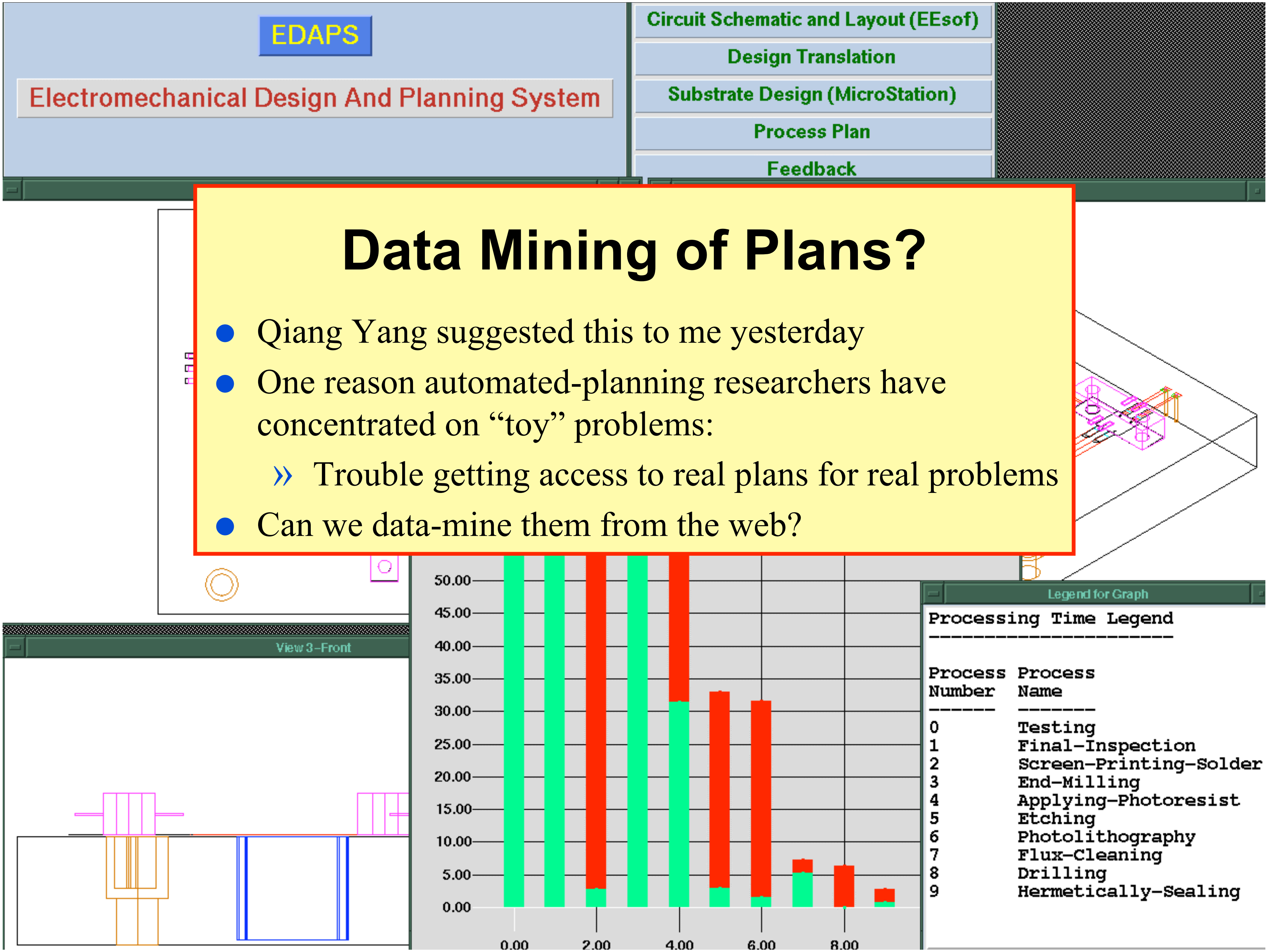


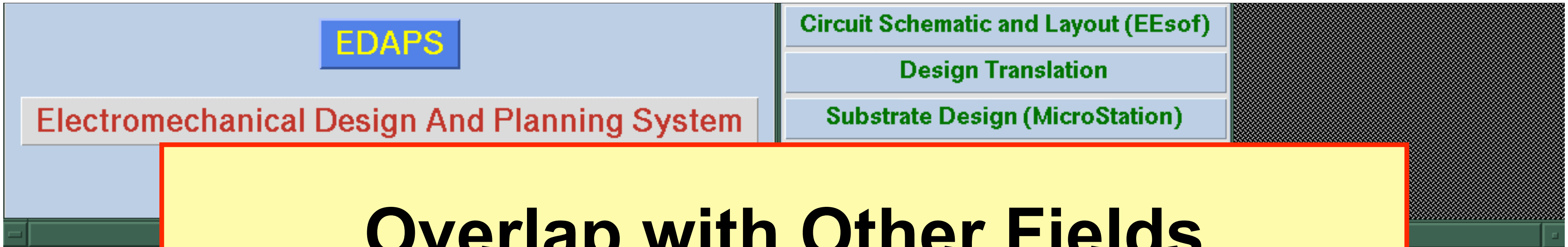
Acquiring Domain Knowledge

- How to get the domain knowledge needed to plan efficiently?
  - » One of the most neglected topics for planning research, but one of the most important
  - » If we could do this well on real-world problems, planners would be hundreds of times more useful
- Researchers are starting to realize this
  - » At *ICAPS*-05 there was an informal "Knowledge Engineering Competition"
    - GUIs for creating knowledge bases for planning
    - Ways for planners to learn domain knowledge
- Overlap with HCI, ML, and CBR

Data Mining of Plans?

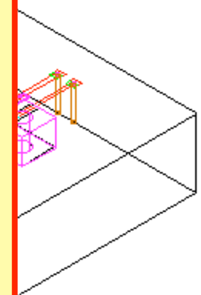
- Qiang Yang suggested this to me yesterday
- One reason automated-planning researchers have concentrated on “toy” problems:
  - » Trouble getting access to real plans for real problems
- Can we data-mine them from the web?

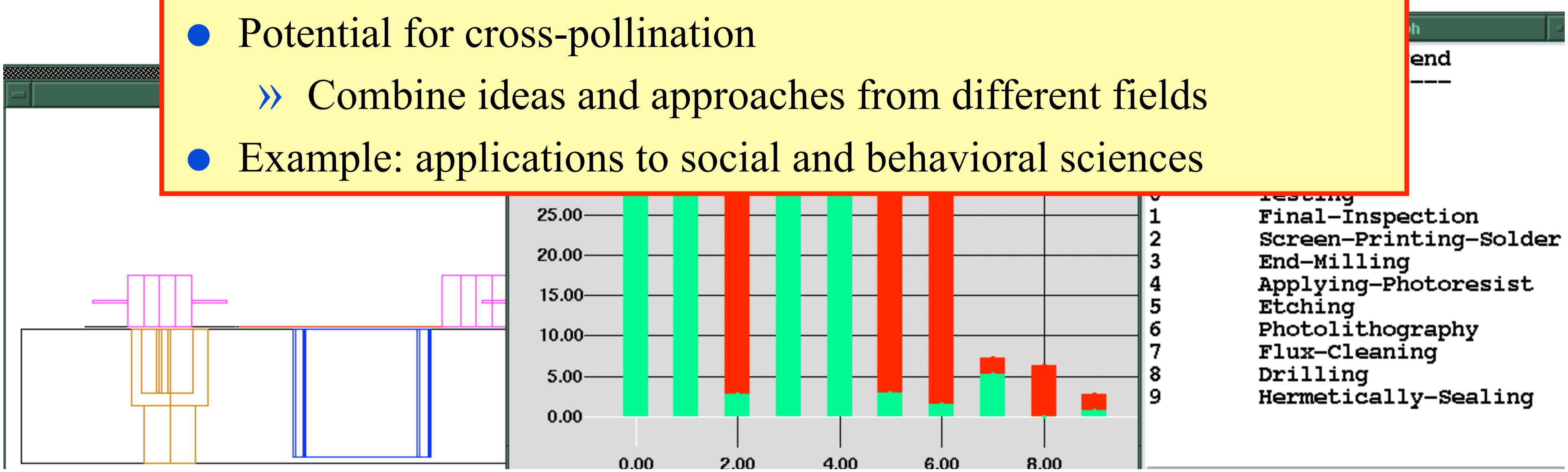




Overlap with Other Fields

- Various kinds of planning are studied in many different fields
  - » AI planning, computer games, game theory, OR, economics, psychology, sociology, political science, industrial engineering, systems science, control theory
- The research groups are often nearly disjoint
  - » Different terminology, assumptions, ideas of what’s important
  - » Hard to tell what the similarities and differences are
- Potential for cross-pollination
  - » Combine ideas and approaches from different fields
- Example: applications to social and behavioral sciences





LCCD Laboratory for **Computational** Cultural Dynamics

- Cross-disciplinary research laboratory at the University of Maryland
  - http://www.cs.umd.edu/projects/lccd
  - Faculty from CS, Business, EE, Government & Politics, International Development, Conflict Management
- Very ambitious goals
  - Develop theory and algorithms needed for tools to support decision making in cultural contexts.
  - Help understand how/why other cultures make decisions
    - More effective cross-cultural interactions
    - Better governance when different cultures are involved
    - Recovery from conflicts and disasters
    - Improve quality of life in developing countries
- **Example:** research by Tsz-Chiu Au, a graduate student at UMD

Prisoner's Dilemma

- One of the best-known examples of a non-zero-sum game
- Two players, each has two possible moves:
  - » Cooperate (C) with the other player
  - » Defect (D), i.e., take advantage of the other player
- Nash equilibrium strategy: (D, D)

- But what if you know you will meet the other player again?

Payoff matrix:

$Player_1$ \ $Player_2$	C	D
C	3, 3	0, 5
D	5, 0	1, 1

My best move is "defect," regardless of whether he cooperates or defects

Iterated Prisoner's Dilemma (IPD)

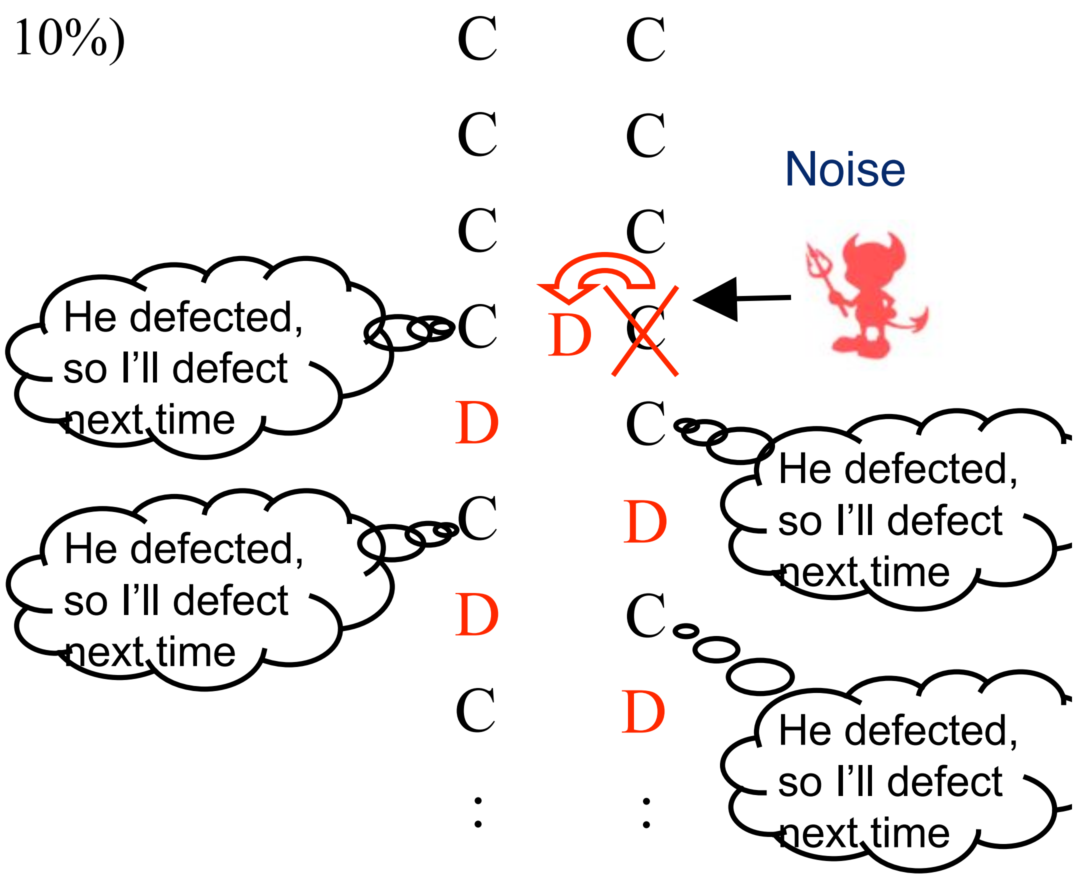
- Axelrod (1984), *The Evolution of Cooperation*
- Two players, finite number of iterations of the Prisoner's Dilemma
- Widely used to study emergence of cooperative behavior among agents
  - » No optimal strategy
  - » Performance depends on the strategies of all of the players
- The best strategy in Axelrod's tournaments:
  - » ***Tit-for-Tat (TFT)***
    - On 1st move, cooperate. On *n*th move, repeat the other player's (n–1)-th move
  - » Could establish and maintain cooperations with many other players
  - » Could prevent malicious players from taking advantage of it

Payoff matrix:

$Player_1$ \ $Player_2$	C	D
C	3, 3	0, 5
D	5, 0	1, 1

IPD with Noise

- Models accidents and misinterpretations
- There’s a nonzero probability (e.g., 10%) that a “noise gremlin” will change some of the actions
  - » *Cooperate* (C) will become *Defect* (D), and vice versa
- Tit-for-Tat and other strategies fail to maintain cooperation
- Tsz-Chiu Au’s ***DBS strategy:***
  - » Build a model of the other player’s strategy by observing his/her behavior
  - » Use this model to detect noise
  - » Use it to plan DBS’s actions
  - » Detect when the other player’s strategy changes
    - Update the model



The 20th-Anniversary Iterated Prisoner's Dilemma Competition

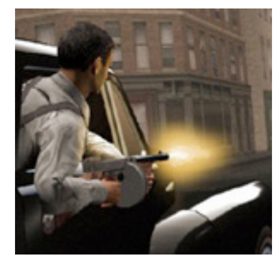
http://www.prisoners-dilemma.com

- Category 2: IPD with noise
  - » 165 programs participated
- DBS dominated the top 10 places
- Only two programs beat DBS
  - » Both used a strategy that was dangerously close to cheating

Rank	Program	Avg. score
1	BWIN	433.8
2	IMM01	414.1
3	DBSz	408.0
4	DBSy	408.0
5	DBSpl	407.5
6	DBSx	406.6
7	DBSf	402.0
8	DBStft	401.8
9	DBSd	400.9
10	lowESTFT_classic	397.2
11	TFTIm	397.0
12	Mod	396.9
13	TFTIz	395.5
14	TFTIc	393.7
15	DBSe	393.7
16	TTFT	393.4
17	TFTIa	393.3
18	TFTIb	393.1
19	TFTIx	393.0
20	mediumESTFT_classic	392.9

How BWIN and IMM01 worked

- Each participant could submit up to 20 programs
- Some participants submitted 20 programs that worked as a team
    - 1 *master* + 19 *slaves*
  - » When slaves play with master
    - they cooperate and master defects
    - master gets all the points
  - » When slaves play with anyone not in their team, they defect
- Analysis
  - » The average score of each master-and-slaves team was much lower than DBSz's average score
  - » If BWIN and IMM01 each had ≤ 10 slaves, DBS would have placed 1st
  - » If BWIN and IMM01 had no slaves, they would have done badly

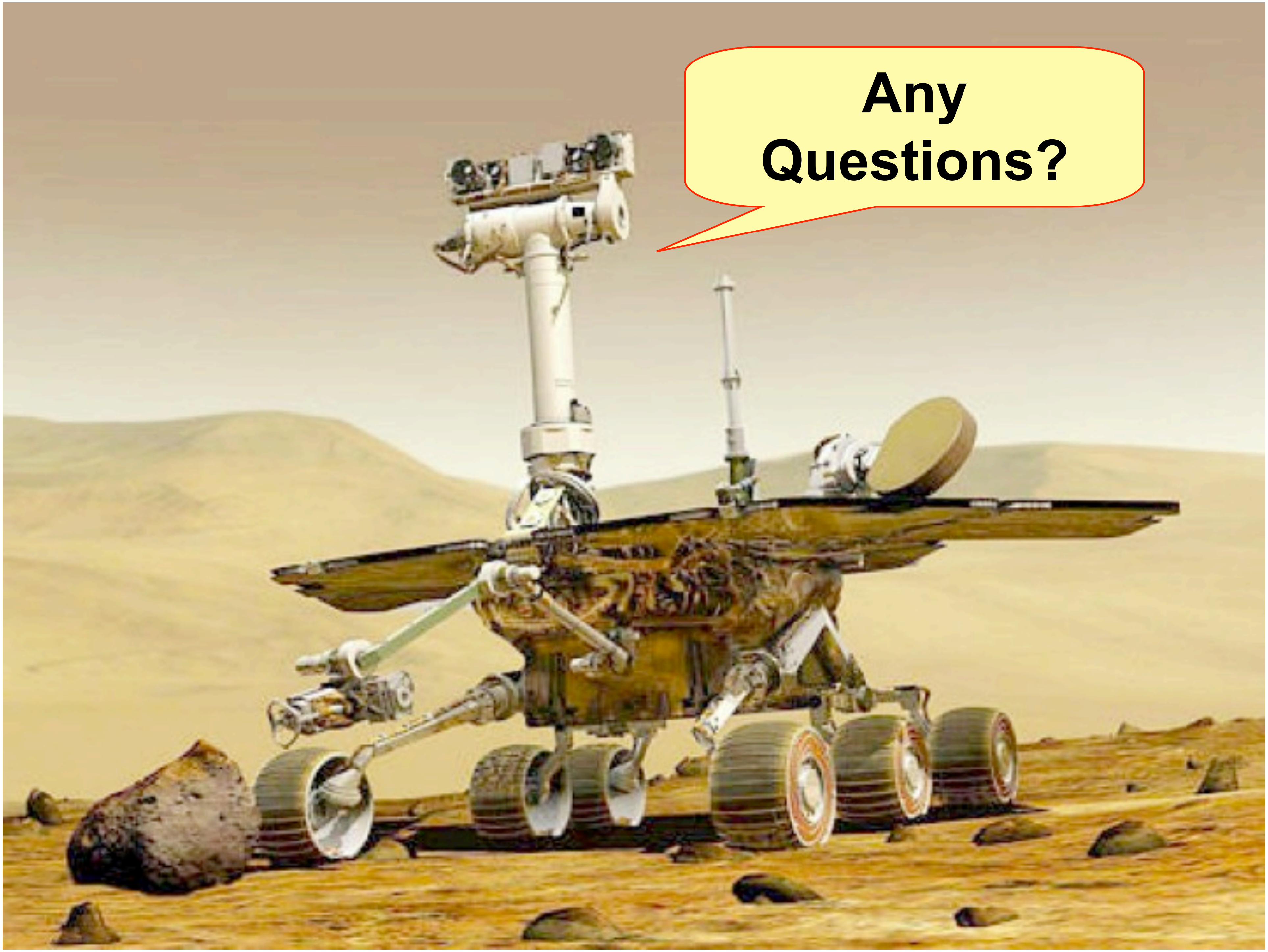


DBS cooperates, not coerces

- Unlike BWIN and IMM01, DBS had *no* slaves
  - » None of the DBS programs even knew the others were there
- DBS worked by establishing cooperation with *many* other agents
- DBS could do this *despite* the noise, because it could filter out the noise

Conclusion

- Automated planning has improved a lot in the last few years
  - Historically, limited by focus on classical planning
  - Scope is broadening to include things important for real-world planning
  - Increased use in practical settings
- Important areas for future growth
  - reasoning about other agents
  - time durations
  - information that is external to the planner
  - acquiring domain knowledge
  - cross-pollination with other fields
    - Example: social and behavioral sciences


Any Questions?