Colloquium at WUSTL CSE 10/5/2007

Real World Planning:

Soft Constraints & Incomplete Models

Subbarao Kambhampati

Arizona State University

Audio available here

Funding from NSF, ONR, DARPA

Yochan Research Group

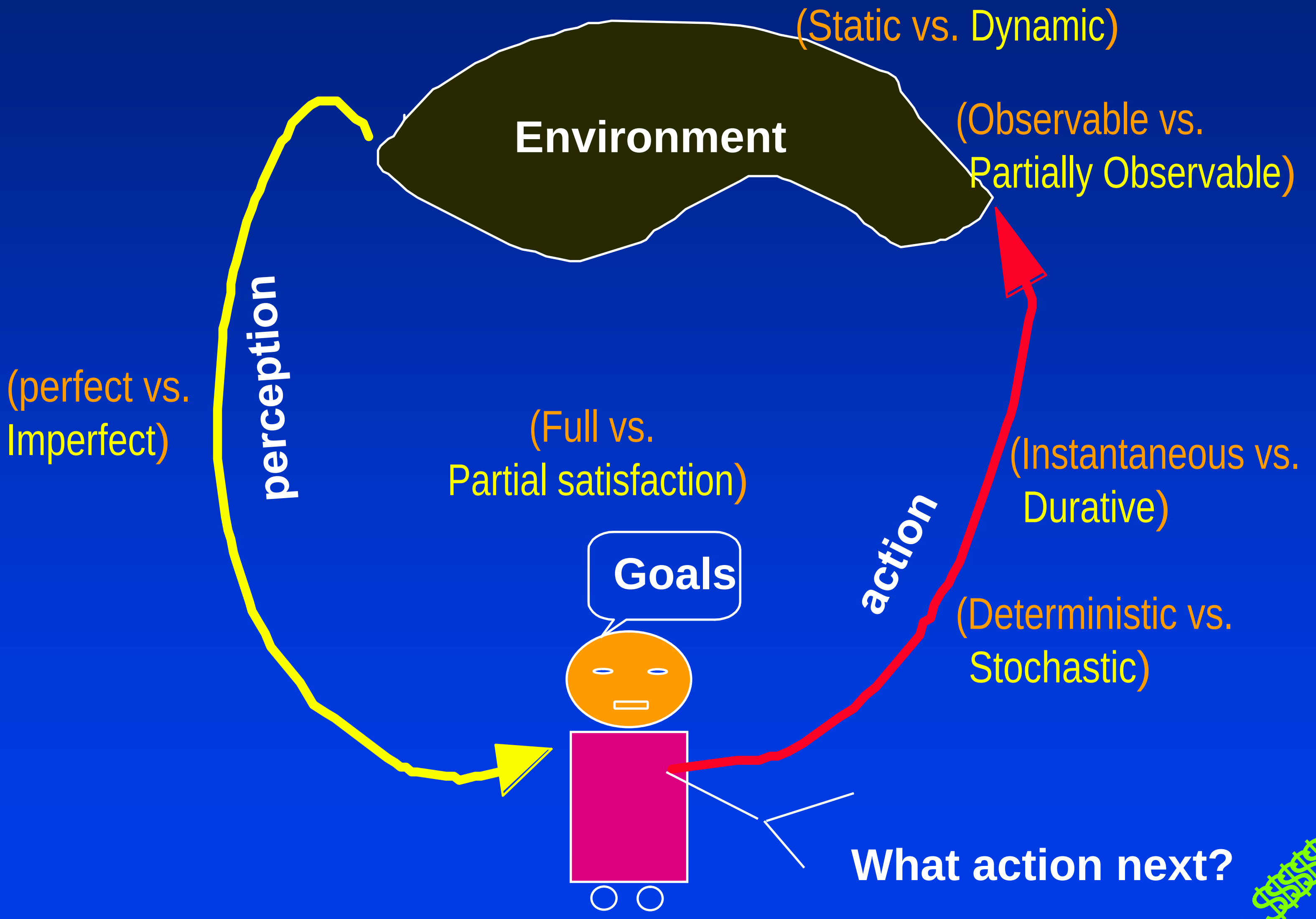
Plan-Yochan

- Automated Planning
  - Foundations of automated planning
  - Heuristics for scaling up a wide spectrum of plan synthesis problems
  - Applications to manufacturing, biological pathway discovery, web services, autonomic computing

Db-Yochan

- Information Integration
  - Mediator frameworks that are adaptive to the sources and users.
  - Applications to Bio-informatics, Archaelogical informatics
  - Systems: QUIC, QPIAD, AIMQ, BibFinder
    - VLDB 07; CIDR 07; ICDE 06…

Planning Involves Deciding a Course of Action to achieve a desired state of affairs



(Static vs. Dynamic)

Environment

(Observable vs. Partially Observable)

perception

(perfect vs. Imperfect)

(Full vs. Partial satisfaction)

action

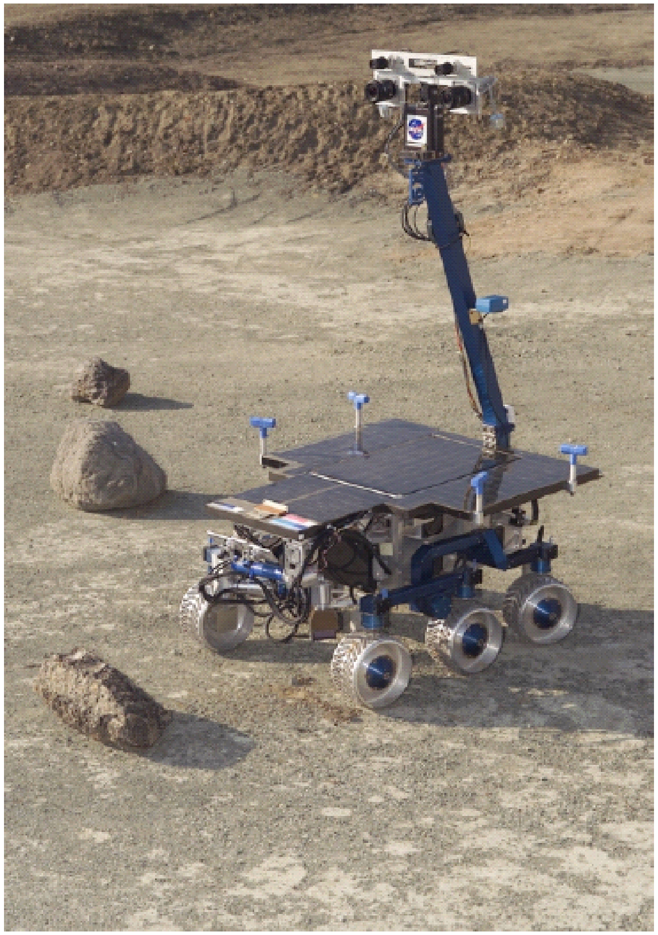
(Instantaneous vs. Durative)

(Deterministic vs. Stochastic)

Goals

What action next?

The $$$$$$ Question



Applications—sublime and mundane

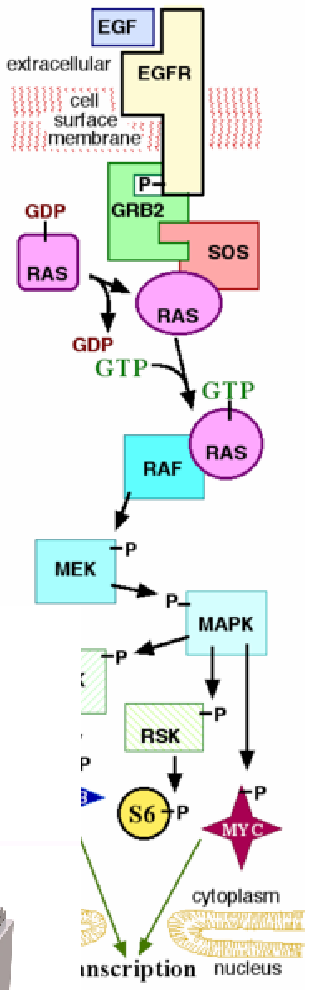
Mission planning (for rovers, telescopes)

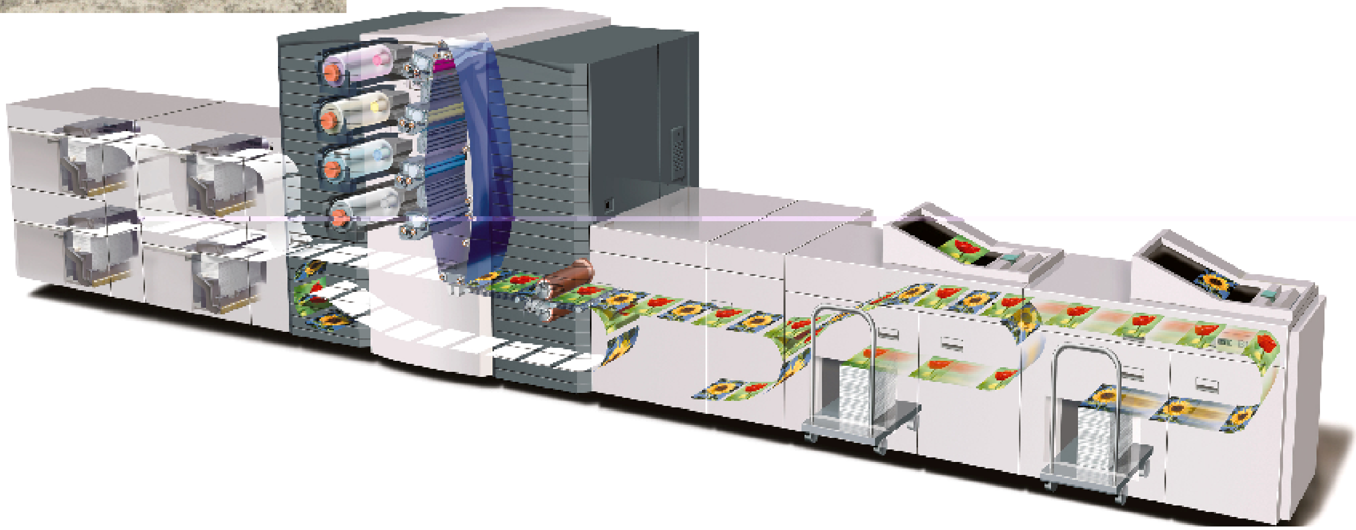
Military planning/scheduling

Web-service/Work-flow composition

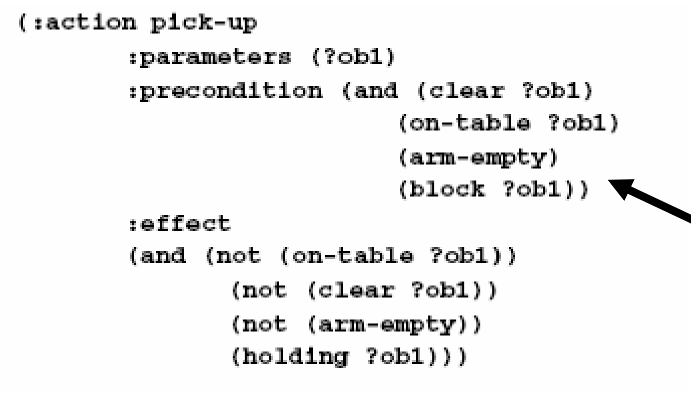
Paper-routing in copiers

Gene regulatory network intervention





Domain-Independent Planning



```
(:action pick-up
        :parameters (?ob1)
        :precondition (and (clear ?ob1)
                           (on-table ?ob1)
                           (arm-empty)
                           (block ?ob1))
        :effect
        (and (not (on-table ?ob1))
                (not (clear ?ob1))
                (not (arm-empty))
                (holding ?ob1)))
```

Blocks world

State variables:
Ontable(x) On(x,y) Clear(x) hand-empty holding(x)

Initial state:
Complete specification of T/F values to state variables
--By convention, variables with F values are omitted

Goal state:
A partial specification of the desired state variable/value combinations
--desired values can be both positive and negative

Init:
Ontable(A),Ontable(B),
Clear(A), Clear(B), hand-empty

Goal:
~clear(B), hand-empty

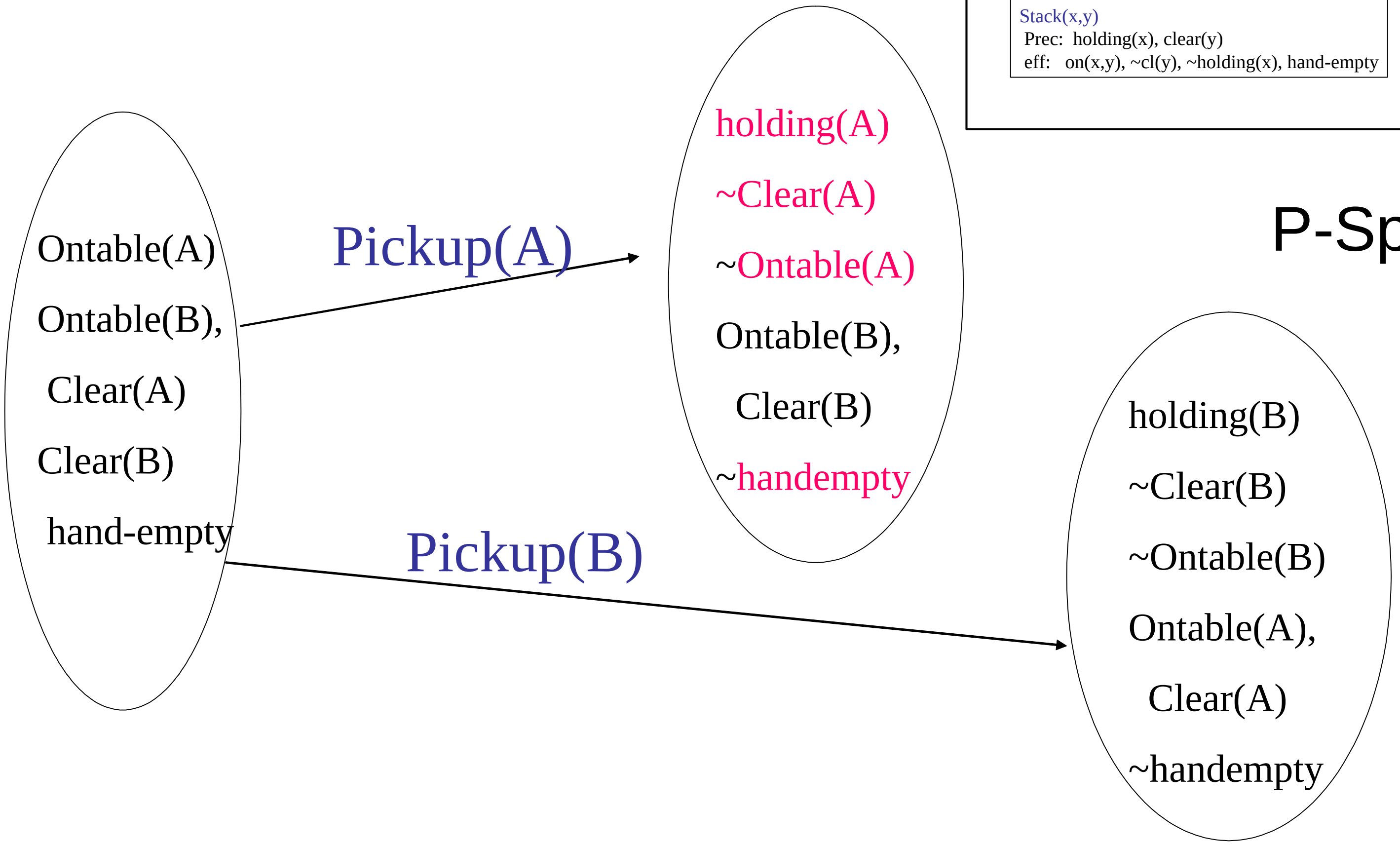
Pickup(x)
Prec: hand-empty,clear(x),ontable(x)
eff: holding(x),~ontable(x),~hand-empty,~Clear(x)

Putdown(x)
Prec: holding(x)
eff: Ontable(x), hand-empty,clear(x),~holding(x)

Stack(x,y)
Prec: holding(x), clear(y)
eff: on(x,y), ~cl(y), ~holding(x), hand-empty

Unstack(x,y)
Prec: on(x,y),hand-empty,cl(x)
eff: holding(x),~clear(x),clear(y),~hand-empty

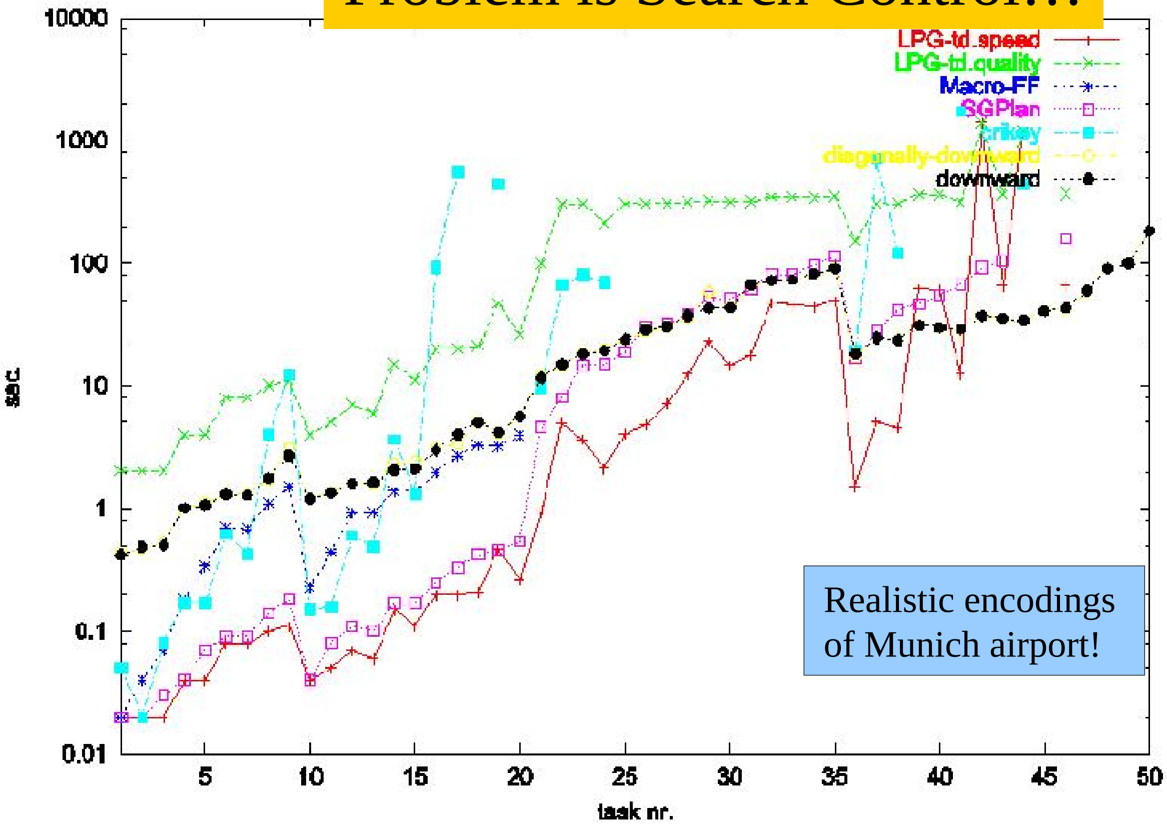
P-Space Complete



Scalability was the big bottle-neck…
We have figured out how to scale synthesis..

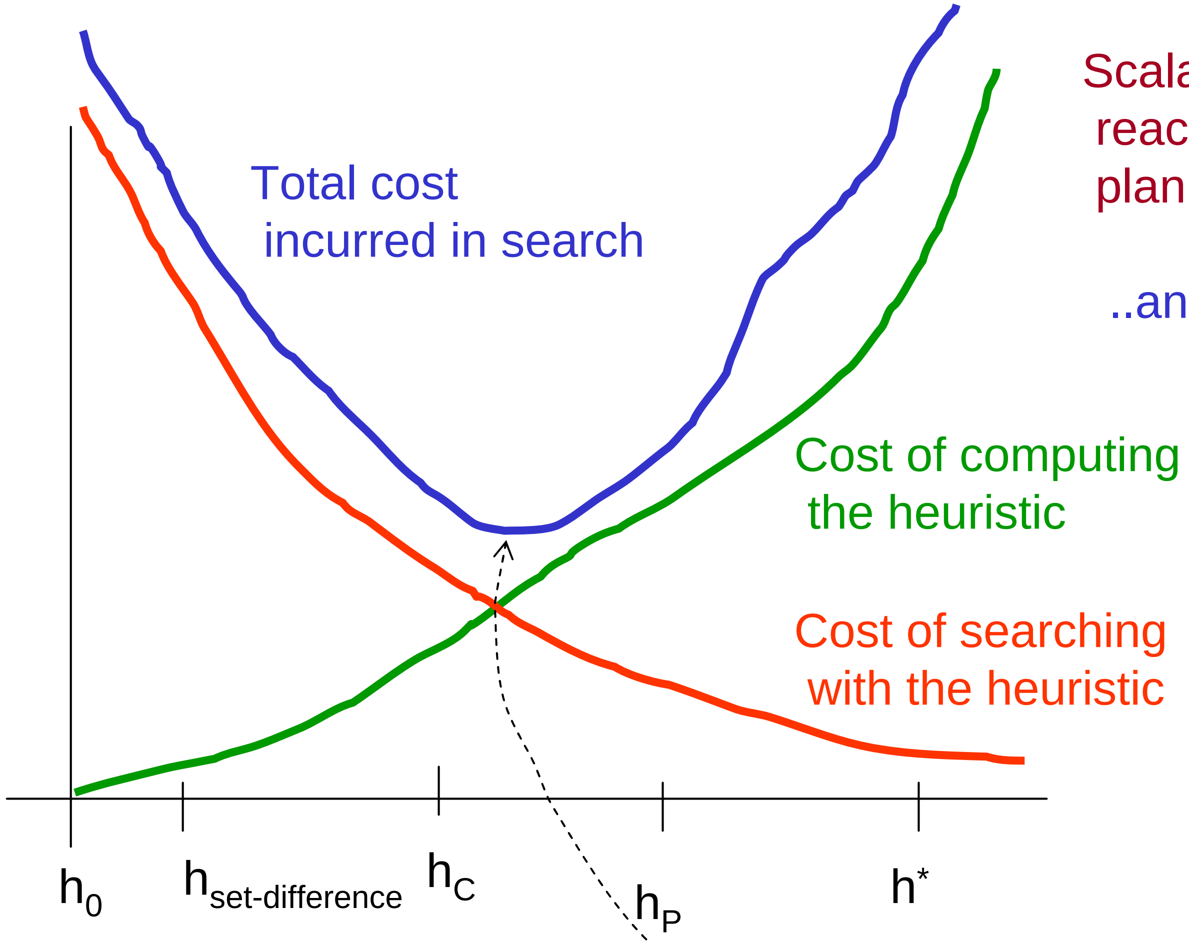
Problem is Search Control!!!

- Before, planning algorithms could synthesize about 6 – 10 action plans in minutes
- Significant scale-up in the last 6-7 years
  - Now, we can synthesize 100 action plans in seconds.



Realistic encodings of Munich airport!

The primary revolution in planning in the recent years has been **methods to** scale up plan synthesis



Scalability came from sophisticated reachability heuristics based on planning graphs..

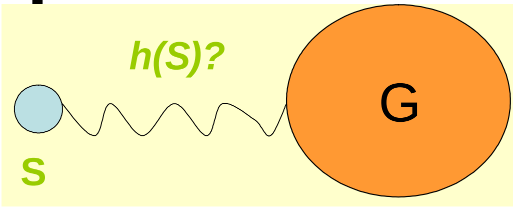
..and not from any hand-coded domain-specific control knoweldge

Not always clear where the total minimum occurs

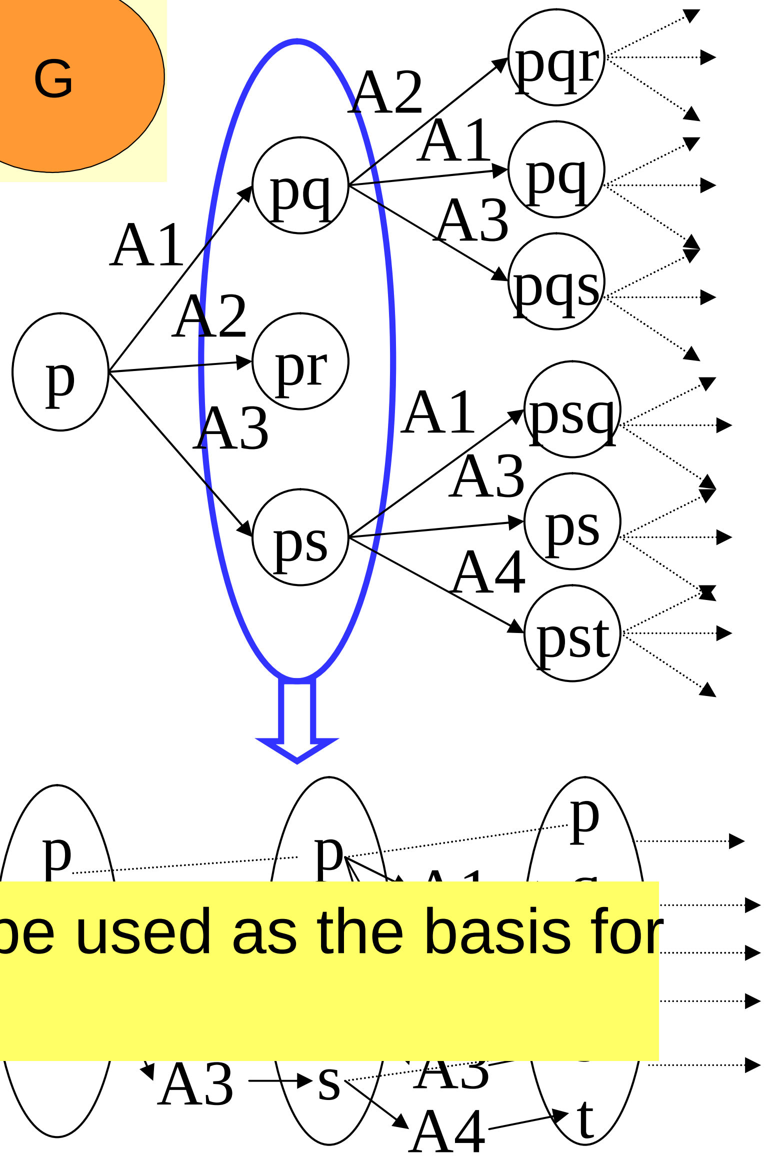
- Old wisdom was that the global min was closer to cheaper heuristics
- Current insights are that it may well be far from the cheaper heuristics for many problems
  - E.g. Pattern databases for 8-puzzle
  - Plan graph heuristics for planning

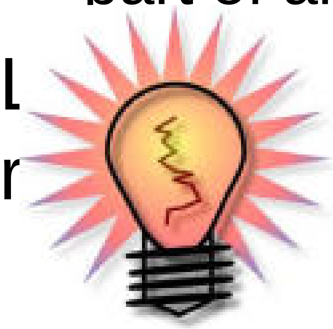
“Optimistic projection of achievability”

Planning Graph and Projection



- Envelope of Progression Tree (Relaxed Progression)
  - Proposition lists: Union of states at k^th level
  - Mutex: Subsets of literals that cannot be part of any legal state
- L nd r

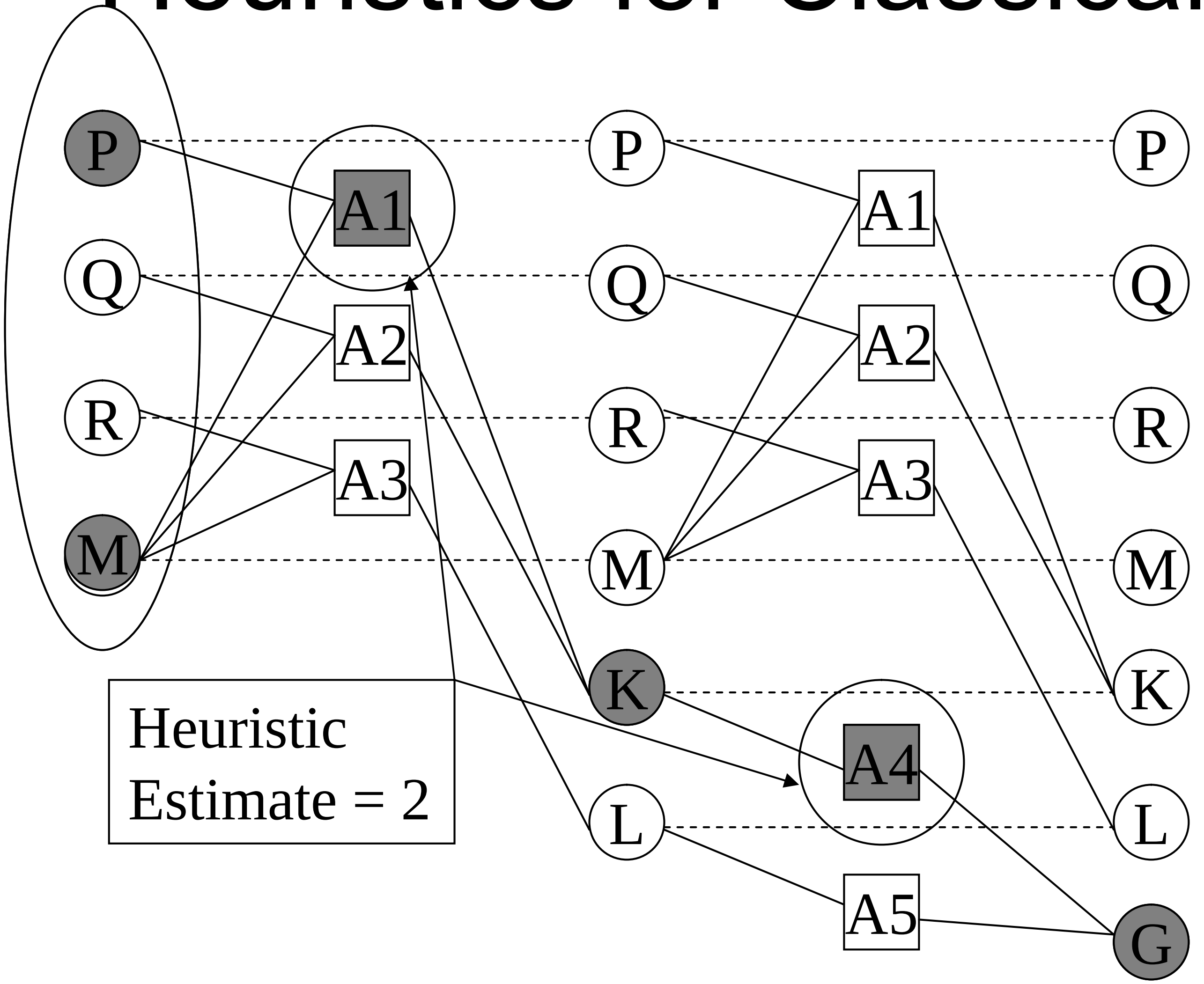




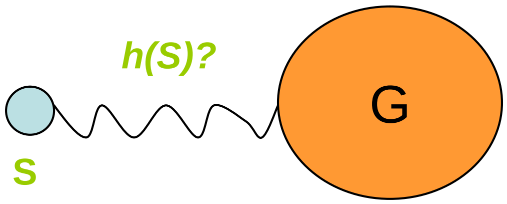
Planning Graphs can be used as the basis for heuristics!

[Blum&Furst, 1995] [ECP, 1997][AI Mag, 2007]

Heuristics for Classical Planning

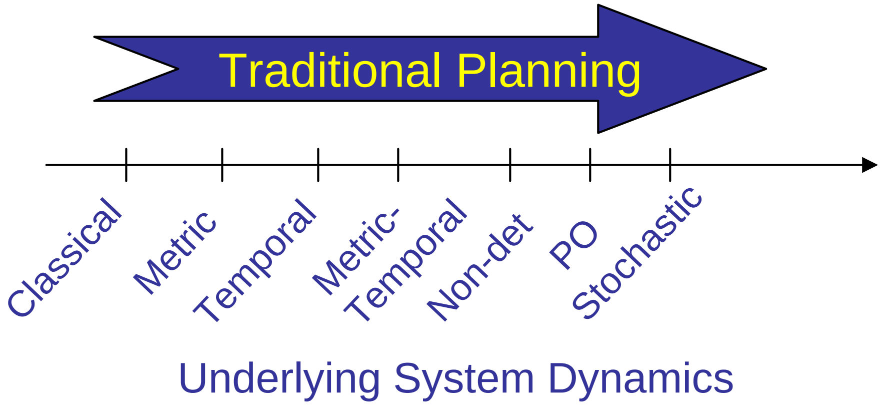


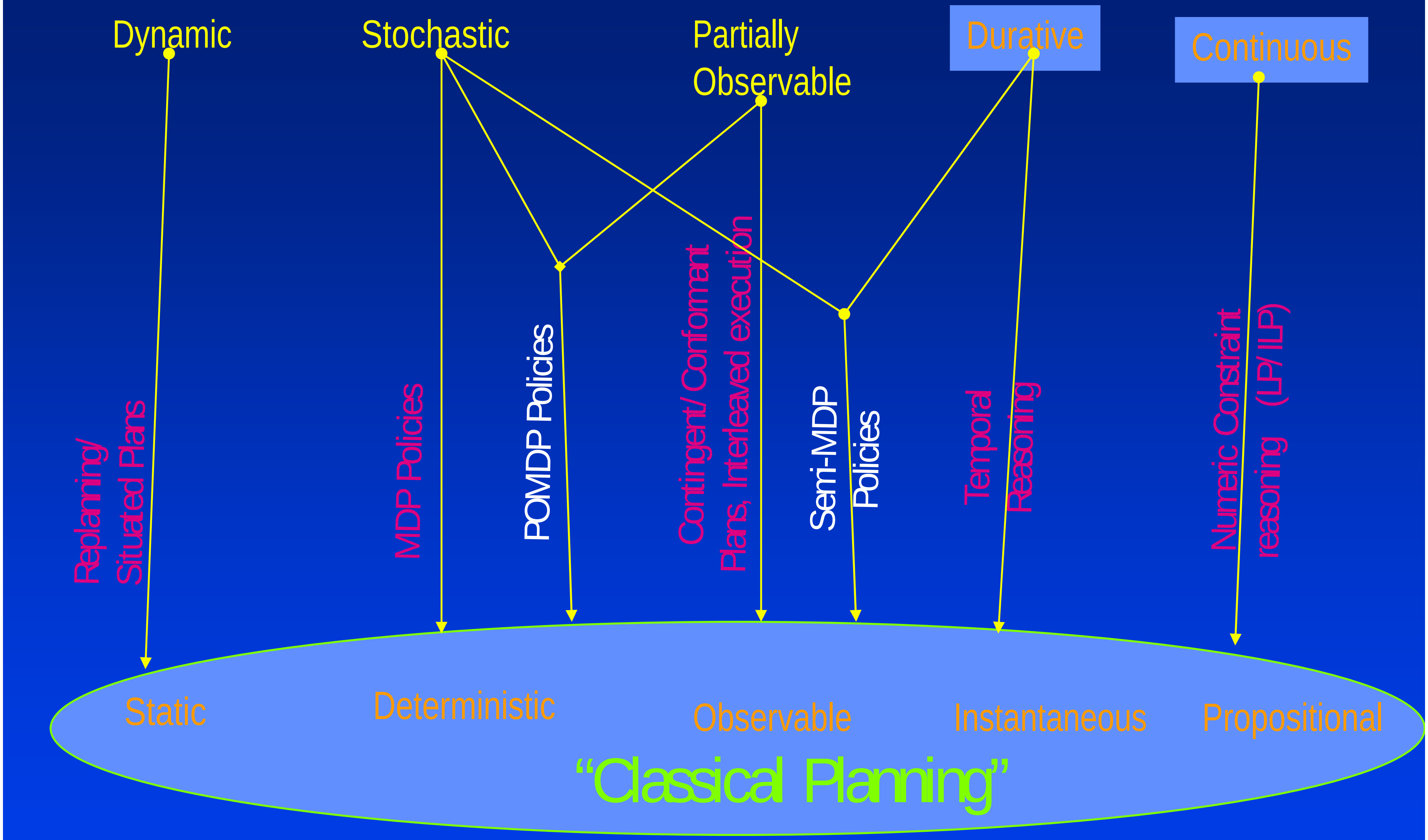
Heuristic
Estimate = 2



Relaxed plans are solutions for a relaxed problem

What are we doing next?





Dynamic

Stochastic

Partially Observable

Durative

Continuous

Replanning/ Situated Plans

MDP Policies

POMDP Policies

Contingent/Conformant Plans, Interleaved execution

Semi-MDP Policies

Temporal Reasoning

Numeric Constraint reasoning (LP/ILP)

Static

Deterministic

Observable

Instantaneous

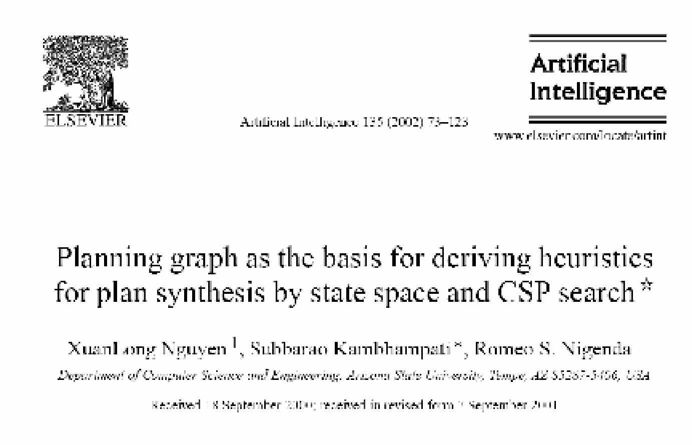
Propositional

"Classical Planning"

..and we play{ed/ing} a significant role

1001 ways to skin a planning graph for heuristic fun & profit

- Classical planning
  - AltAlt (AAAI 2000; AIJ 2002); RePOP (IJCAI 2001); AltAlt^{p} (JAIR 2003)
  - Serial vs. Parallel graphs; Level and Adjusted heuristics; Partial expansion
- Metric Temporal Planning
  - Sapa (ECP 2001; AIPS 2002; JAIR 2003); Sapa^{Mps} (IJCAI 2005)
  - Propagation of cost functions; Phased relaxation
- Nondeterministic Conformant/Conditional Planning
  - CAltAlt (ICAPS 2004); POND (AAAI 2005; JAIR 2006)
  - Multiple graphs; Labelled uncertainty graphs; State-agnostic graphs
- Stochastic planning
  - Monte Carlo Labelled uncertainty graphs [ICAPS 2006; AIJ 2007]
  - Labelled graphs capturing "particles"



ELSEVIER

Artificial Intelligence 135 (2002) 73–123

Artificial Intelligence

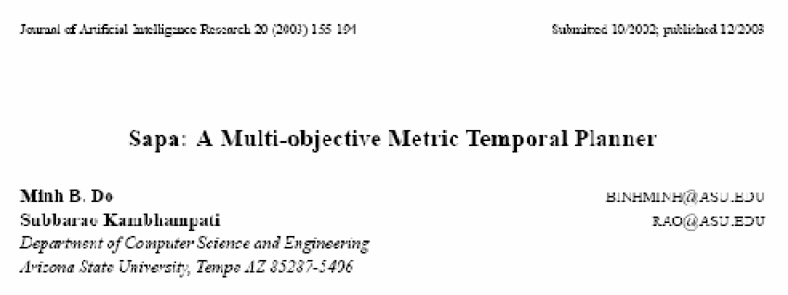
www.elsevier.com/locate/artint

Planning graph as the basis for deriving heuristics for plan synthesis by state space and CSP search *

XuanLong Nguyen [1], Subbarao Kambhampati *, Romeo S. Nigenda

Department of Computer Science and Engineering, Arizona State University, Tempe, AZ 85287-5406, USA

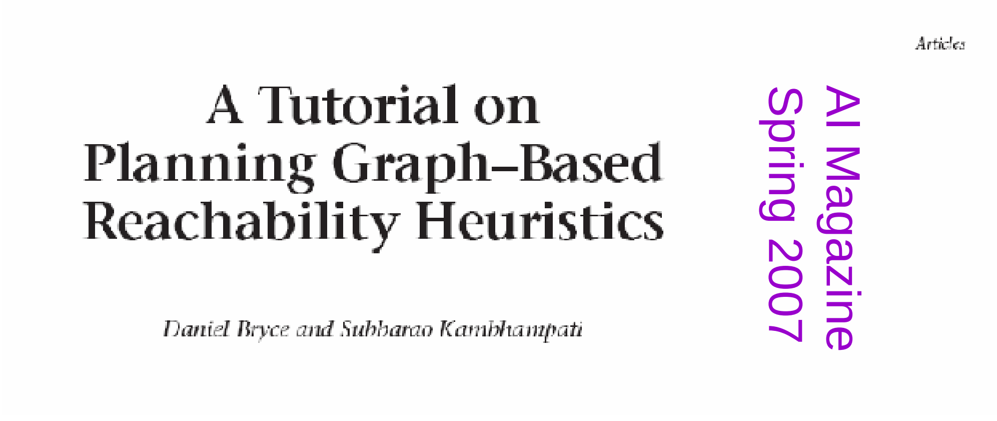
Received 18 September 2000; received in revised form 7 September 2001



Journal of Artificial Intelligence Research 20 (2003) 155-194 Submitted 10/2002; published 12/2003

Sapa: A Multi-objective Metric Temporal Planner

Minh B. Do BINHMINH@ASU.EDU
Subbarao Kambhampati RAO@ASU.EDU
Department of Computer Science and Engineering
Arizona State University, Tempe AZ 85287-5406

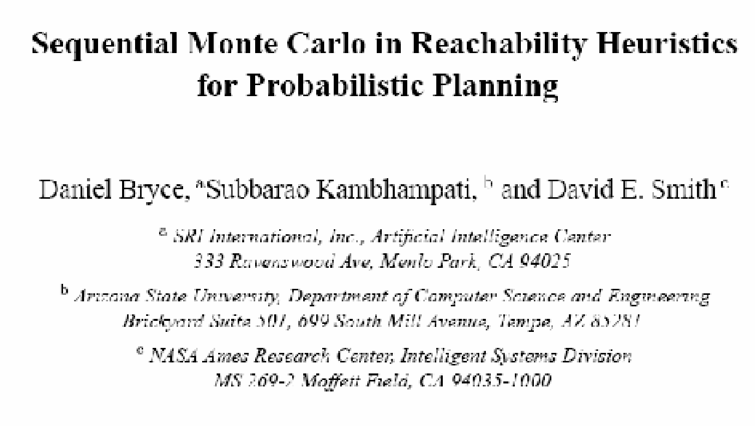


Articles

A Tutorial on Planning Graph–Based Reachability Heuristics

Daniel Bryce and Subbarao Kambhampati

AI Magazine
Spring 2007



Sequential Monte Carlo in Reachability Heuristics for Probabilistic Planning

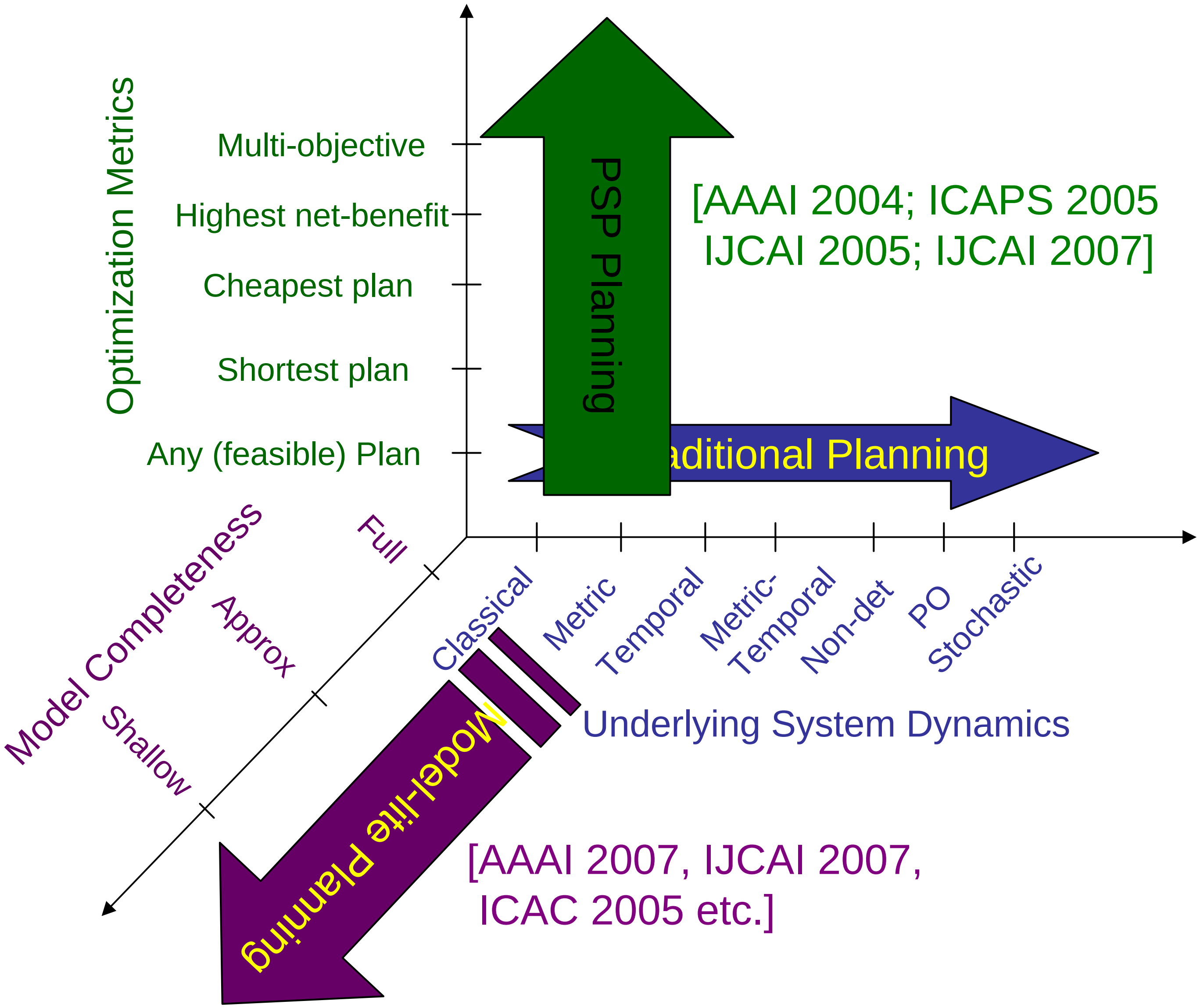
Daniel Bryce, [a]Subbarao Kambhampati, [b] and David E. Smith [c]

[a] *SRI International, Inc., Artificial Intelligence Center*
333 Ravenswood Ave, Menlo Park, CA 94025

[b] *Arizona State University, Department of Computer Science and Engineering*
Brickyard Suite 501, 699 South Mill Avenue, Tempe, AZ 85281

[c] *NASA Ames Research Center, Intelligent Systems Division*
MS 269-2 Moffett Field, CA 94035-1000

AI Journal; 2007



[AAAI 2004; ICAPS 2005
IJCAI 2005; IJCAI 2007]

[AAAI 2007, IJCAI 2007,
ICAC 2005 etc.]

Classical vs. Partial Satisfaction Planning (PSP)

Classical Planning

- Initial state
- Set of goals
- Actions

Find a plan that achieves *all* goals

(prefer plans with fewer actions)

Partial Satisfaction Planning

- Initial state
- Goals *with differing utilities*
- Actions *with differing costs*

Find a plan with highest *net benefit*

(cumulative utility – cumulative cost)

(best plan may not achieve all the goals)

Partial Satisfaction/Over-Subscription Planning

- Traditional planning problems
  - Find the (lowest cost) plan that satisfies all the given goals
- PSP Planning
  - Find the highest utility plan given the resource constraints
    - Goals have utilities and actions have costs
- …arises naturally in many real world planning scenarios
  - MARS rovers attempting to maximize scientific return, given resource constraints
  - UAVs attempting to maximize reconnaisance returns, given fuel etc constraints
  - Logistics problems resource constraints
- … due to a variety of reasons
  - Constraints on agent's resources
  - Conflicting goals
    - With complex inter-dependencies between goal utilities
  - Soft constraints
  - Limited time

[AAAI 2004; ICAPS 2005; IJCAI 2005; IJCAI 2007; ICAPS 2007; CP 2007]

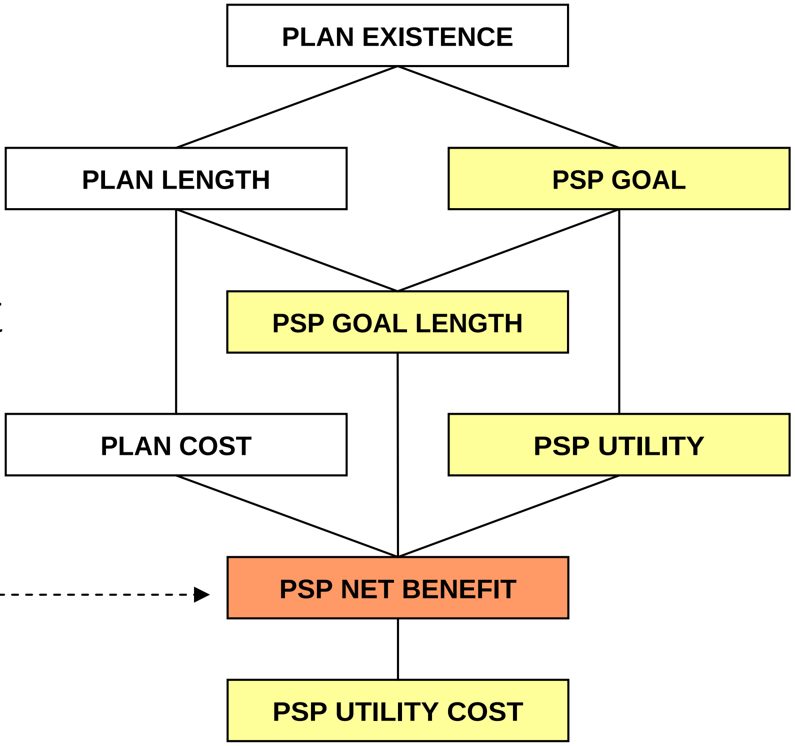
Supporting PSP planning

- PSP planning changes planning from a “satisficing” to an “optimizing” problem
    - It is trivial to find a plan; hard to find a good one!
        - Rich connections to OR(IP)/MDP
- Requires selecting “objectives” in addition to “actions”
    - Which subset of goals to achieve
    - At what degree to satisfy individual goals
        - E.g. Collect as much soil sample as possible; get done as close to 2pm as possible
- Currently, the objective selection is left to humans
    - Leads to highly suboptimal plans since objective selection cannot be done independent of planning
- Need for scalable methods for synthesizing plans in such over-subscribed scenarios

Formulation

- PSP Net benefit:
  - Given a planning problem $P = (F, A, I, G)$, and for each action a "cost" $c_a \geq 0$, and for each goal fluent $f \in G$ a "utility" $u_f \geq 0$, and a positive number k. Is there a finite sequence of actions $\Delta = (a_1, a_2, \ldots, a_n)$ that starting from I leads to a state S that has net benefit $\sum_{f\in(S\cap G)} u_f - \sum_{a\in\Delta} c_a \geq k$.

Maximize the Net Benefit



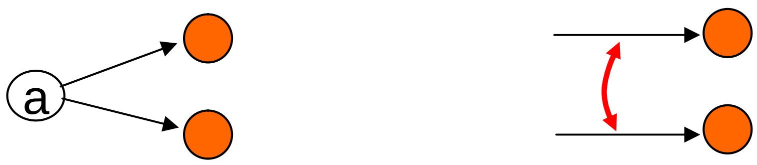
Actions have execution costs, goals have utilities, and the objective is to find the plan that has the highest net benefit.
→ easy enough to extend to mixture of soft and hard goals

Challenge: Goal Dependencies

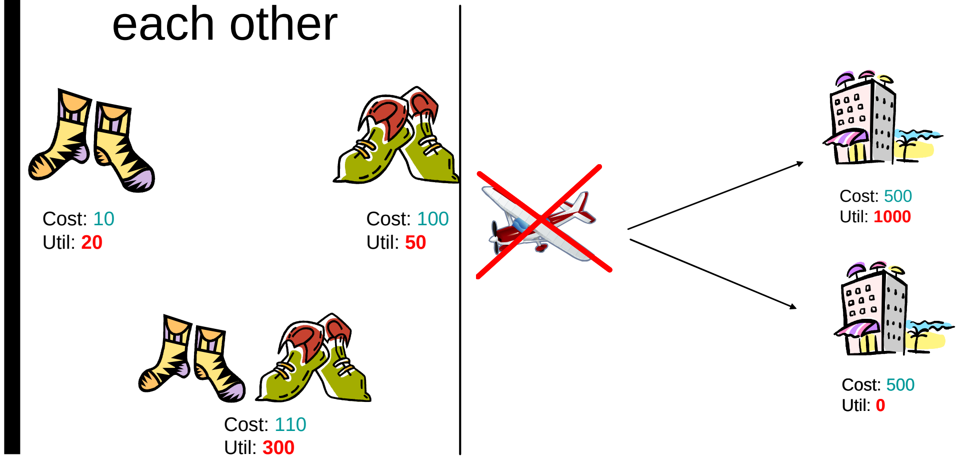
goal interactions exist as two distinct types

cost dependencies ⟷ *utility dependencies*

Actions achieving different goals interact *positively* or *negatively*



Goals may *complement* or *substitute* each other



- Modeling goal cost/utility dependencies
- Doing planning in the presence of utility (and cost) dependencies

PSP^{UD}

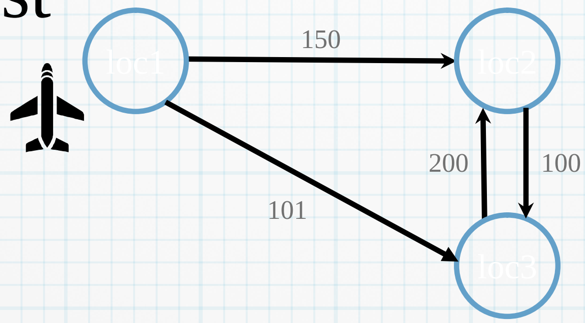
Partial Satisfaction Planning with Utility Dependency

(Smith, ICAPS 2004; van den Briel, et al., AAAI 2004)

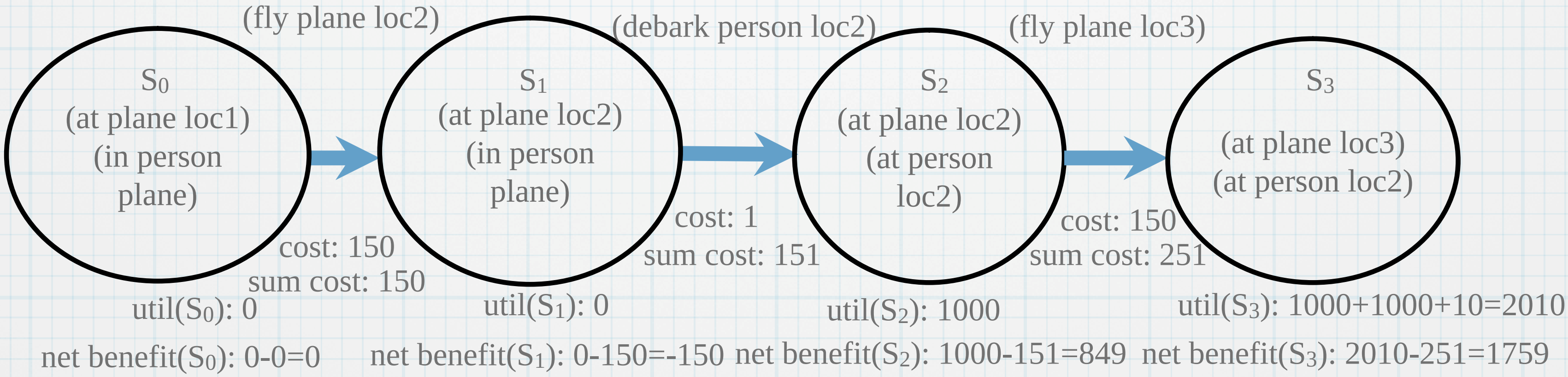
(Do, et al., IJCAI 2007)

Actions have cost

Goal sets have utility



Maximize Net Benefit (utility - cost)



(fly plane loc2)

(debark person loc2)

(fly plane loc3)

S_0
(at plane loc1)
(in person plane)

S_1
(at plane loc2)
(in person plane)

S_2
(at plane loc2)
(at person loc2)

S_3
(at plane loc3)
(at person loc2)

cost: 150
sum cost: 150

cost: 1
sum cost: 151

cost: 150
sum cost: 251

util(S_0): 0

util(S_1): 0

util(S_2): 1000

util(S_3): 1000+1000+10=2010

net benefit(S_0): 0-0=0

net benefit(S_1): 0-150=-150

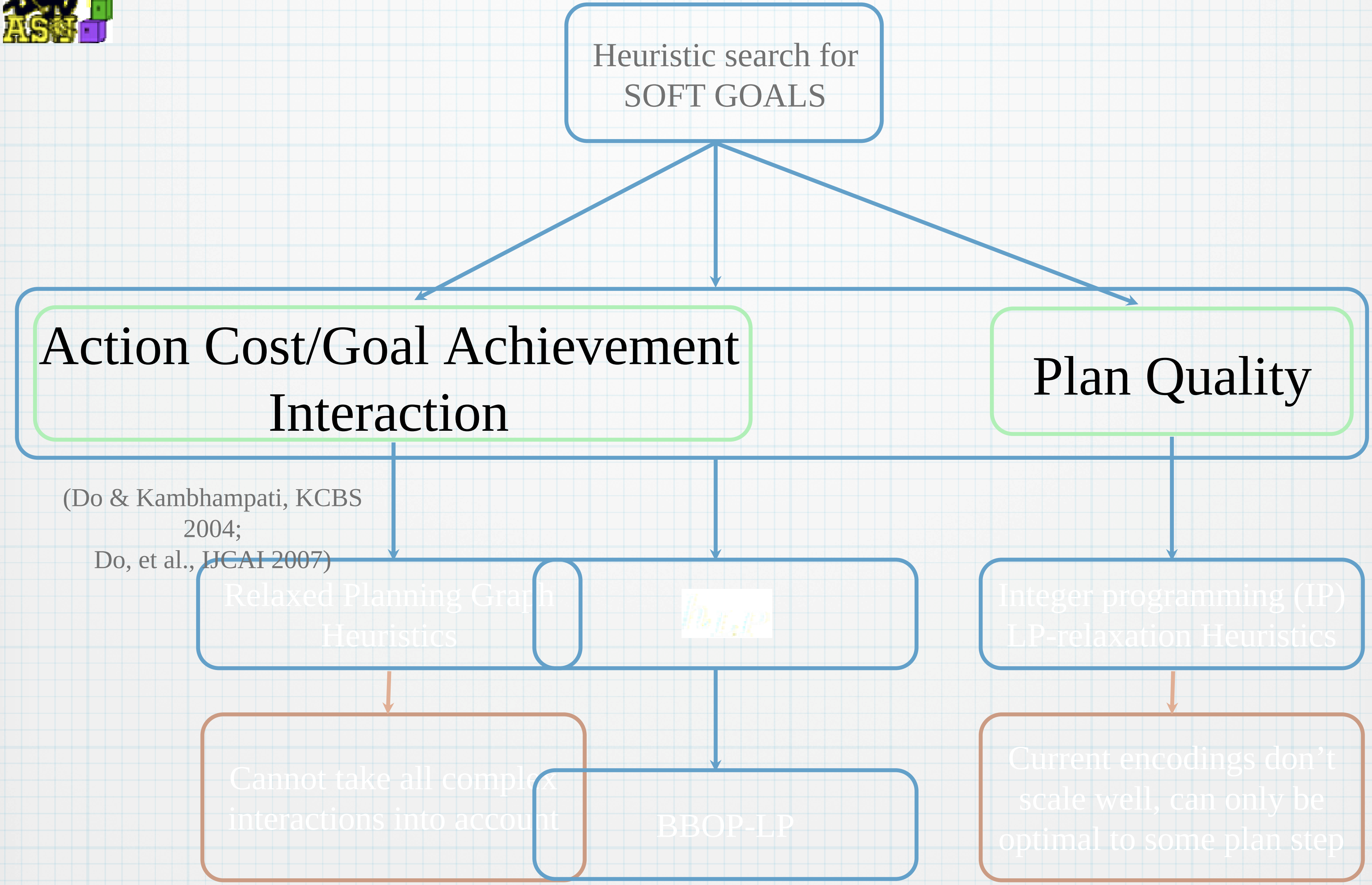
net benefit(S_2): 1000-151=849

net benefit(S_3): 2010-251=1759

utility((at plane loc3)) = 1000

utility((at person loc2)) = 1000

utility((at plane loc1) & (at person loc3)) = 10



Heuristic search for SOFT GOALS

Action Cost/Goal Achievement Interaction

Plan Quality

(Do & Kambhampati, KCBS 2004;
Do, et al., IJCAI 2007)

Relaxed Planning Graph Heuristics

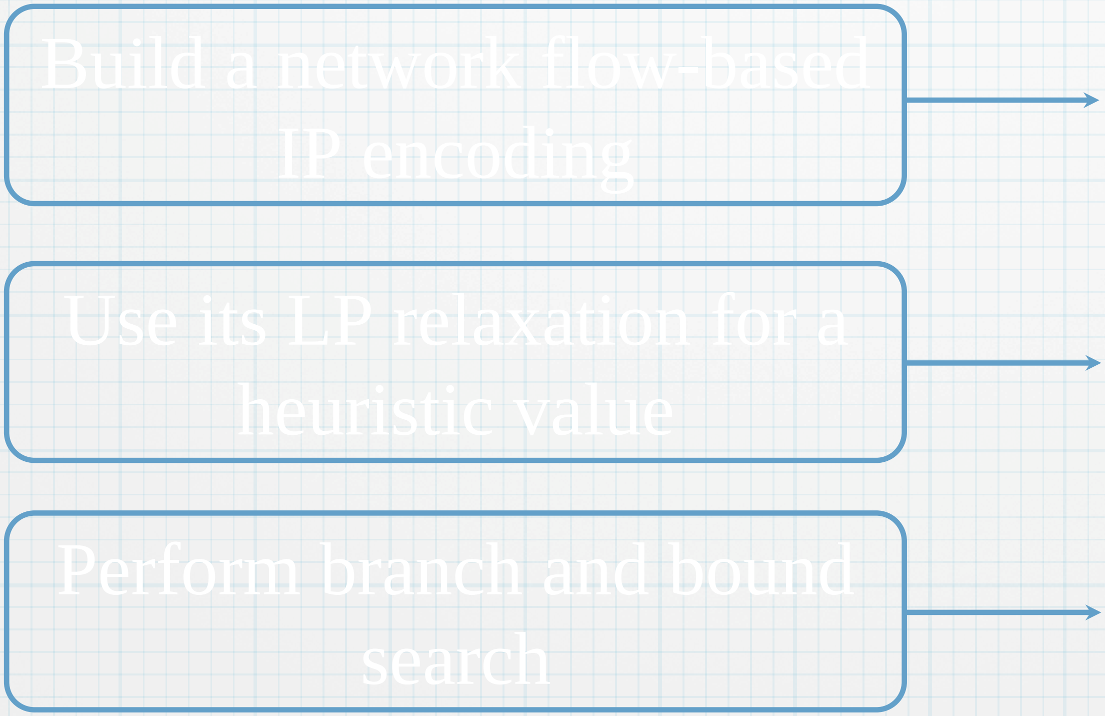
Integer programming (IP) LP-relaxation Heuristics

Cannot take all complex interactions into account

BBOP-LP

Current encodings don’t scale well, can only be optimal to some plan step

Approach



Build a network flow-based IP encoding

No time indices
Uses multi-valued variables

Use its LP relaxation for a heuristic value

Gives a second relaxation on the heuristic

Perform branch and bound search

Uses the LP solution to find a relaxed plan
(similar to YAHSP, Vidal 2004)

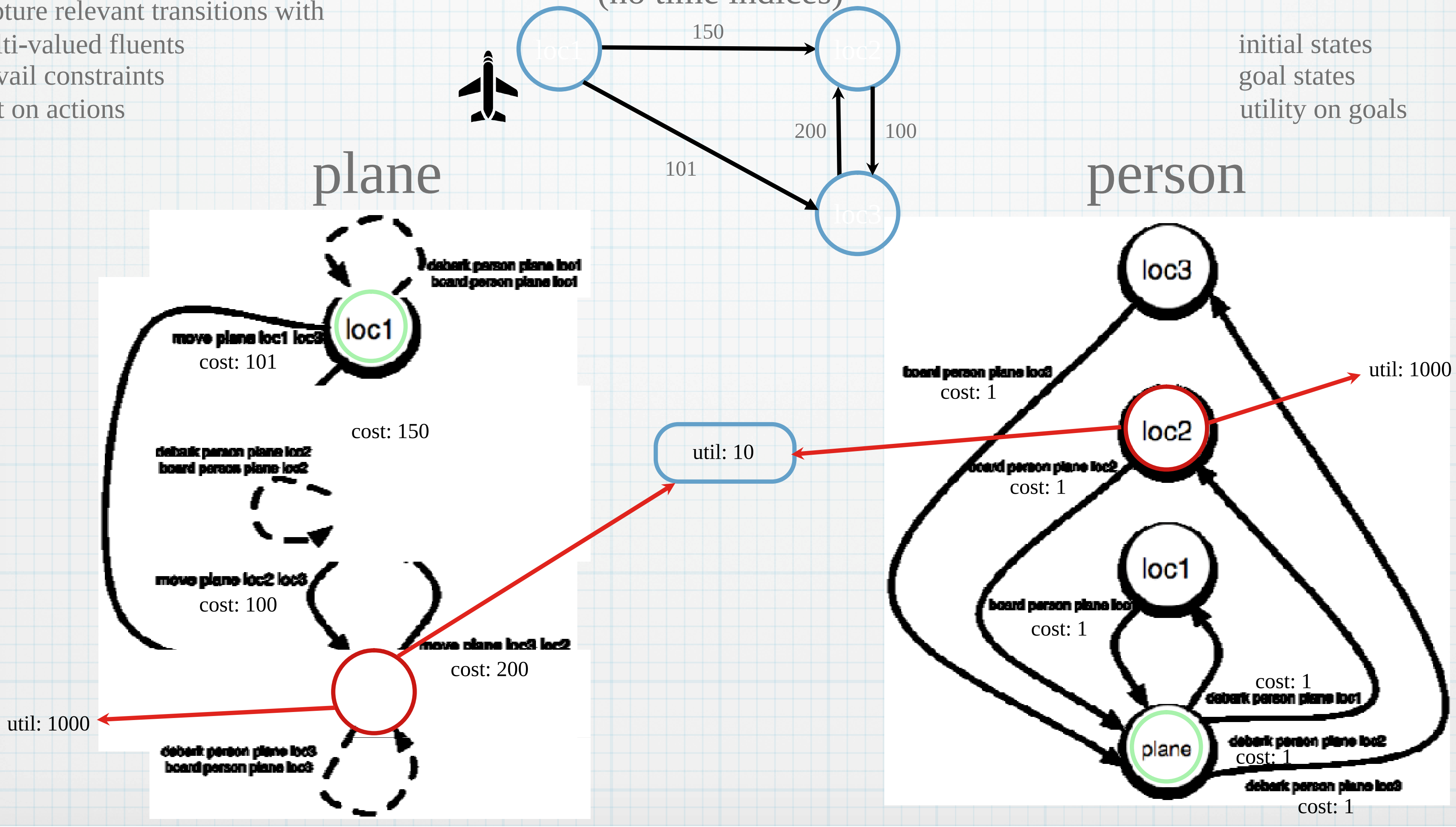
Building a Heuristic

A network flow model on variable transitions

(no time indices)

Capture relevant transitions with
multi-valued fluents
prevail constraints
cost on actions

initial states
goal states
utility on goals

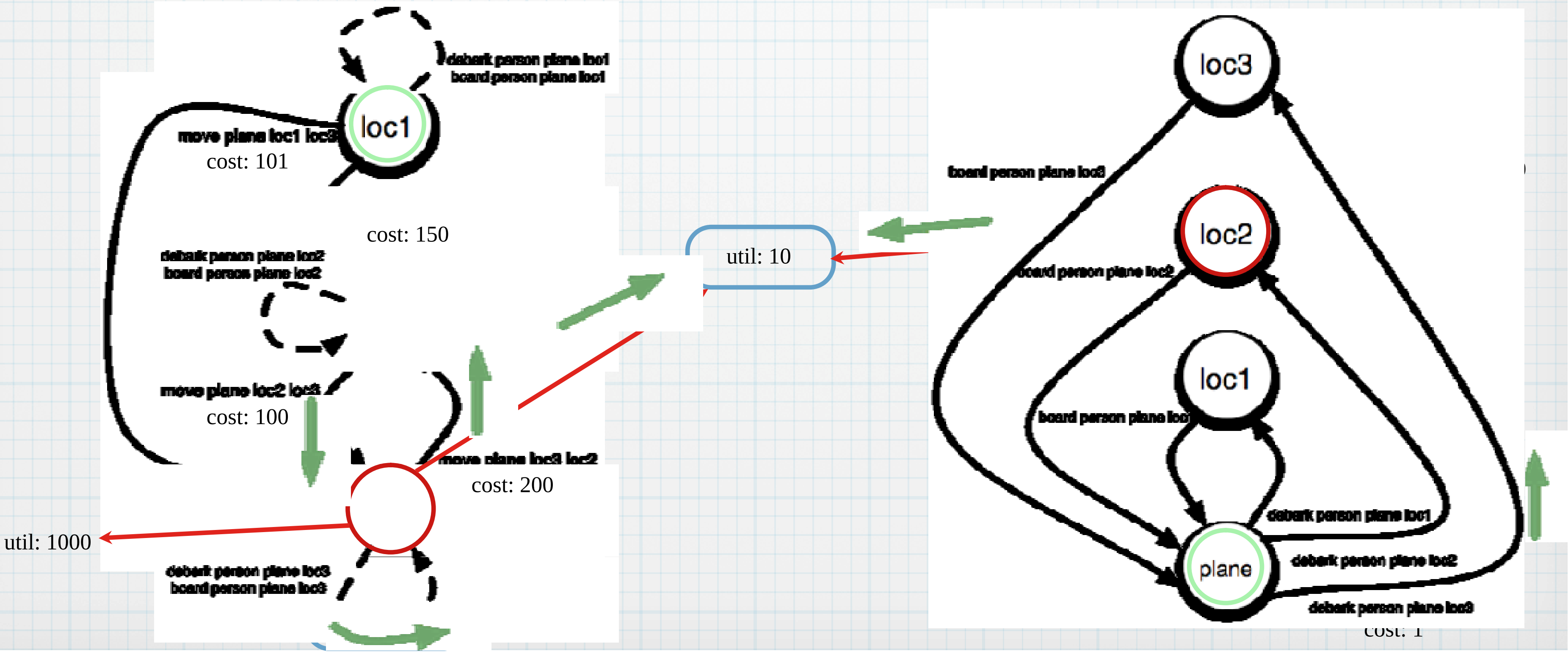


Building a Heuristic

Constraints of this model

1. If an action executes, then all of its effects and prevail conditions must also.
2. If a fact is deleted, then it must be added to re-achieve a value.
3. If a prevail condition is required, then it must be achieved.
4. A goal utility dependency is achieved iff its goals are achieved.

plane person



Building a Heuristic

Constraints of this model

1. If an action executes, then all of its effects and prevail conditions must also.

$action(a) \ = \ S_{\text{effects of a in v}}\ effect(a,v,e) + S_{\text{prevails of a in v}}\ prevail(a,v,f)$

2. If a fact is deleted, then it must be added to re-achieve a value.

$1\{\text{if } f \ ?\ s_0[v]\} + S_{\text{effects that add f}}\ effect(a,v,e) \ = \ S_{\text{effects that delete f}}\ effect(a,v,e) + endvalue(v,f)$

3. If a prevail condition is required, then it must be achieved.

$1\{\text{if } f \ ?\ s_0[v]\} + S_{\text{effects that add f}}\ effect(a,v,e) \ = \ prevail(a,v,f) / M$

4. A goal utility dependency is achieved iff its goals are achieved.

$goaldep(k) = \ S_{\text{f in dependency k}}\ endvalue(v,f) \ - \ |G_k| \ - \ 1$

$goaldep(k) = \ endvalue(v,f) \ \ ?\ f \text{ in dependency k}$

h_{LP}

Variables

$action(a)\ ?\ Z^+$	The number of times a ? A is executed
$effect(a,v,e)\ ?\ Z^+$	The number of times a transition e in state variable v is caused by action a
$prevail(a,v,f)\ ?\ Z^+$	The number of times a prevail condition f in state variable v is required by action a
$endvalue(v,f)\ ?\ \{0,1\}$	Equal to 1 if value f is the end value in a state variable v
goaldep(k)	Equal to 1 if a goal dependency is achieved

Parameters

cost(a)	the cost of executing action a ? A
utility(v,f)	the utility of achieving value f in state variable v
utility(k)	the utility of achieving achieving goal dependency G_k

Objective Function

MAX $S_{v?V,f?Dv}$ utility(v,f) endvalue(v,f) + $S_{k?K}$ utility(k) goaldep(k) – $S_{a?A}$ cost(a) action(a)

Maximize Net Benefit

2. If a fact is deleted, then it must be added to re-achieve a value.

$1\{\text{if } f ? s_0[v]\} + S_{\text{effects that add } f}\ \text{effect}(a,v,e) = S_{\text{effects that delete } f}\ \text{effect}(a,v,e) + \text{endvalue}(v,f)$

Updated at each search node

3. If a prevail condition is required, then it must be achieved.

$1\{\text{if } f ? s_0[v]\} + S_{\text{effects that add } f}\ \text{effect}(a,v,e) = \text{prevail}(a,v,f) / M$

h_{LP}

Variables

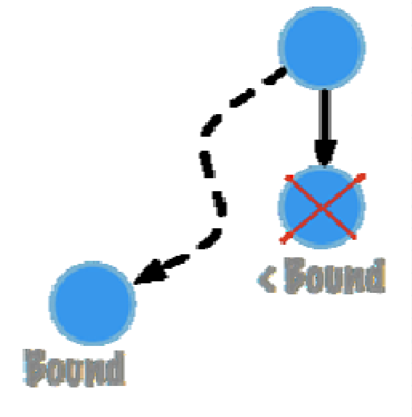
action(a) ? Z^+	The number of times a ? A is executed
effect(a,v,e) ? Z^+	The number of times a transition e in state variable v is caused by action a
prevail(a,v,f) ? Z^+	The number of times a prevail condition f in state variable v is required by action a
endvalue(v,f) ? {0,1}	Equal to 1 if value f is the end value in a state variable v
goaldep(k)	Equal to 1 if a goal dependency is achieved

Parameters

cost(a)	the cost of executing action a ? A
utility(v,f)	the utility of achieving value f in state variable v
utility(k)	the utility of achieving achieving goal dependency G_k

Search

Branch and Bound



Branch and bound with time limit

All soft goals; all states are goal states

Returns the best plan (i.e., best bound)

Greedy lookahead strategy

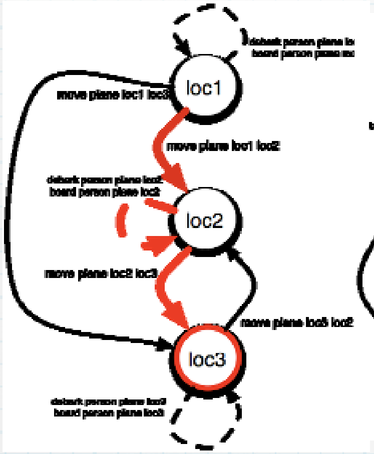
Similar to YAHSP (Vidal, 2004)

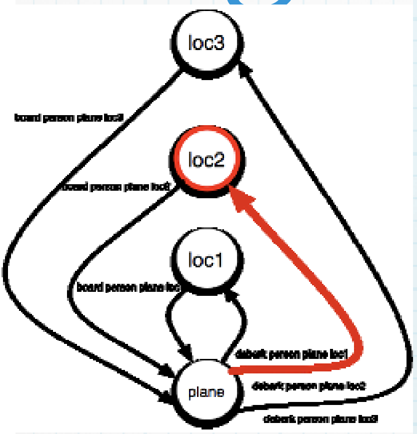
To quickly find good bounds

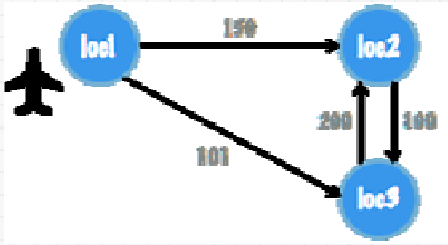
LP-solution guided relaxed plan extraction

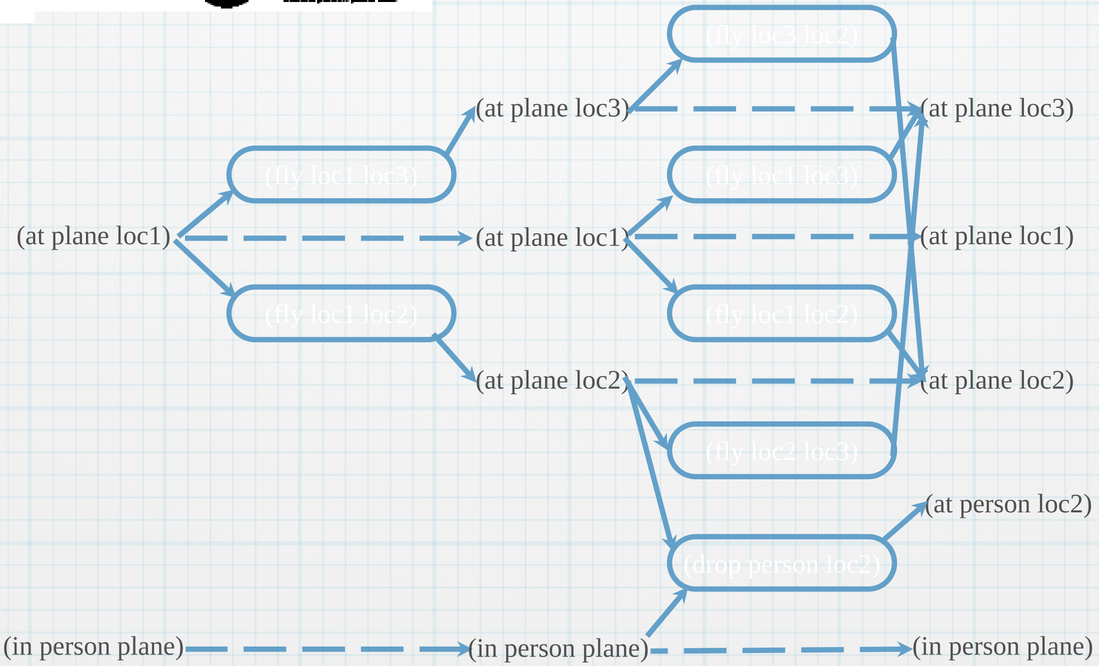
To add informedness

Getting a Relaxed Plan









(at plane loc1)

(in person plane)

(fly loc1 loc3)

(fly loc1 loc2)

(at plane loc3)

(at plane loc1)

(at plane loc2)

(in person plane)

(fly loc3 loc2)

(fly loc1 loc3)

(fly loc1 loc2)

(fly loc2 loc3)

(drop person loc2)

(at plane loc3)

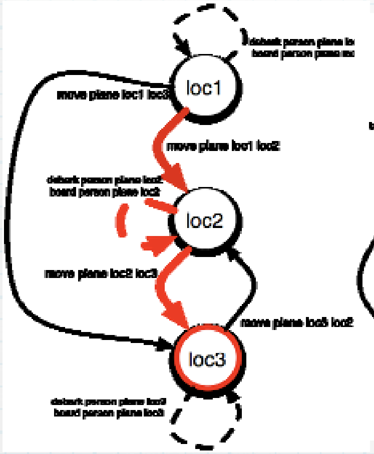
(at plane loc1)

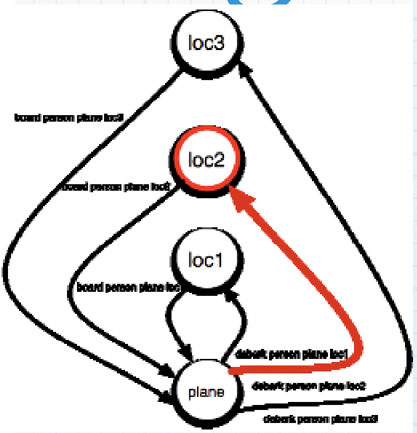
(at plane loc2)

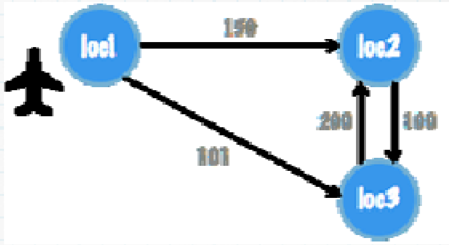
(at person loc2)

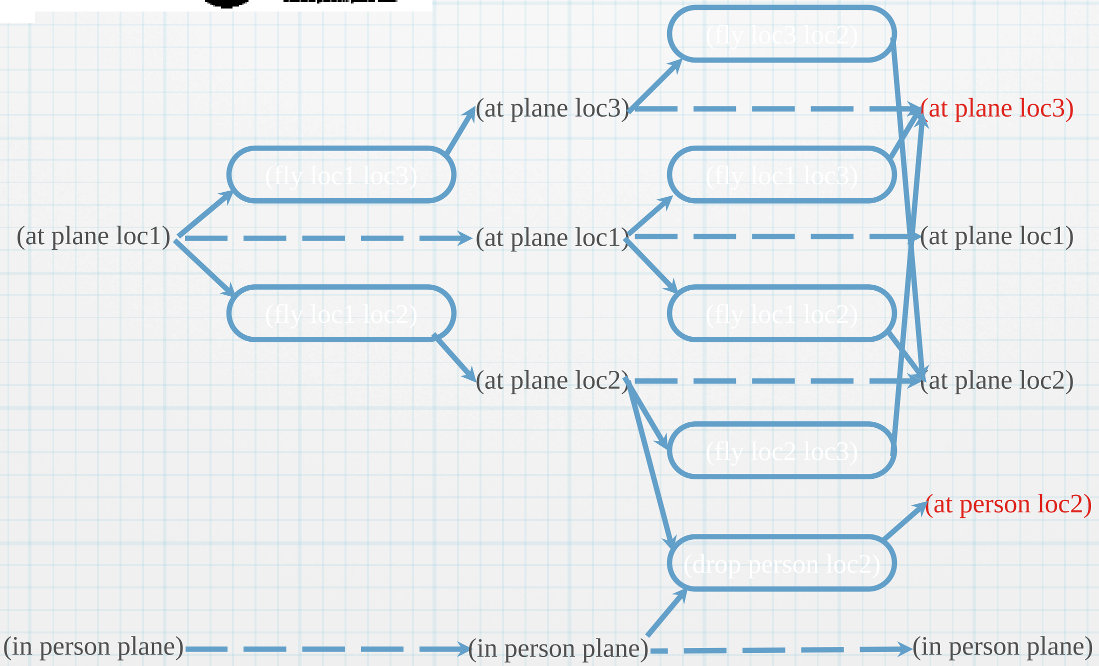
(in person plane)

Getting a Relaxed Plan








(at plane loc1)

(in person plane)

(fly loc1 loc3)

(fly loc1 loc2)

(at plane loc3)

(at plane loc1)

(at plane loc2)

(in person plane)

(fly loc3 loc2)

(fly loc1 loc3)

(fly loc1 loc2)

(fly loc2 loc3)

(drop person loc2)

(at plane loc3)

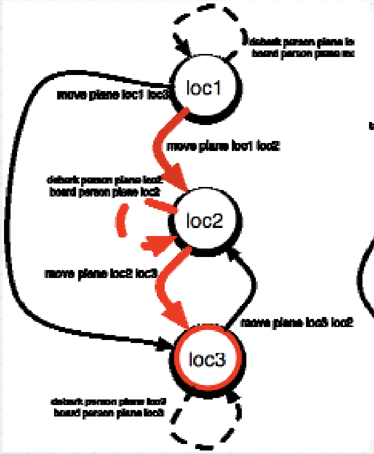
(at plane loc1)

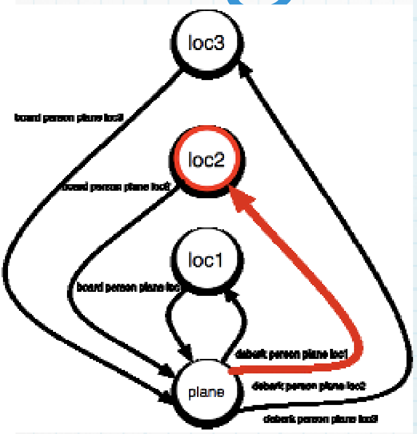
(at plane loc2)

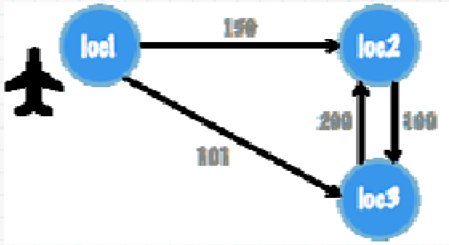
(at person loc2)

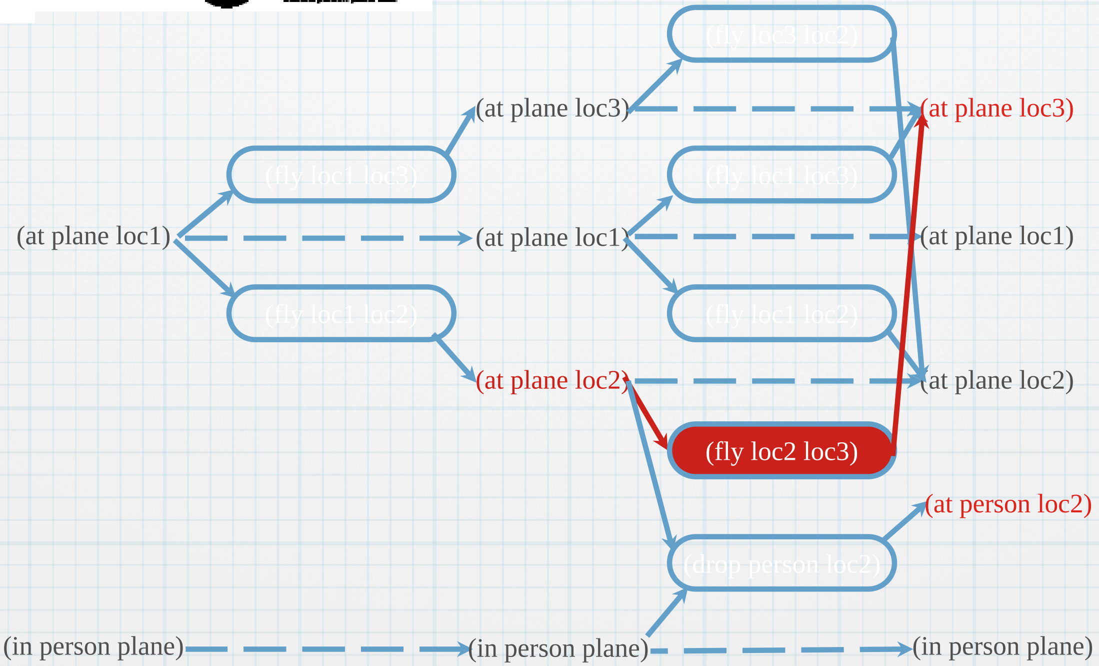
(in person plane)

Getting a Relaxed Plan









(at plane loc1)

(in person plane)

(fly loc1 loc3)

(fly loc1 loc2)

(at plane loc3)

(at plane loc1)

(at plane loc2)

(in person plane)

(fly loc3 loc2)

(fly loc1 loc3)

(fly loc1 loc2)

(fly loc2 loc3)

(drop person loc2)

(at plane loc3)

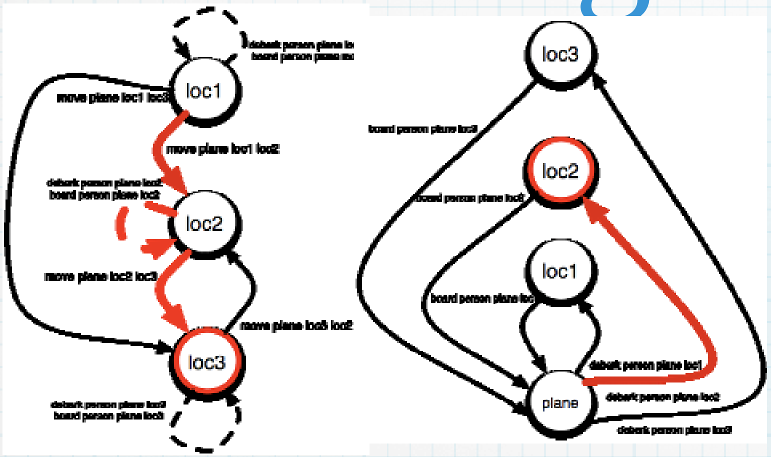
(at plane loc1)

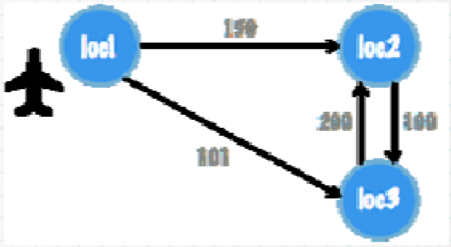
(at plane loc2)

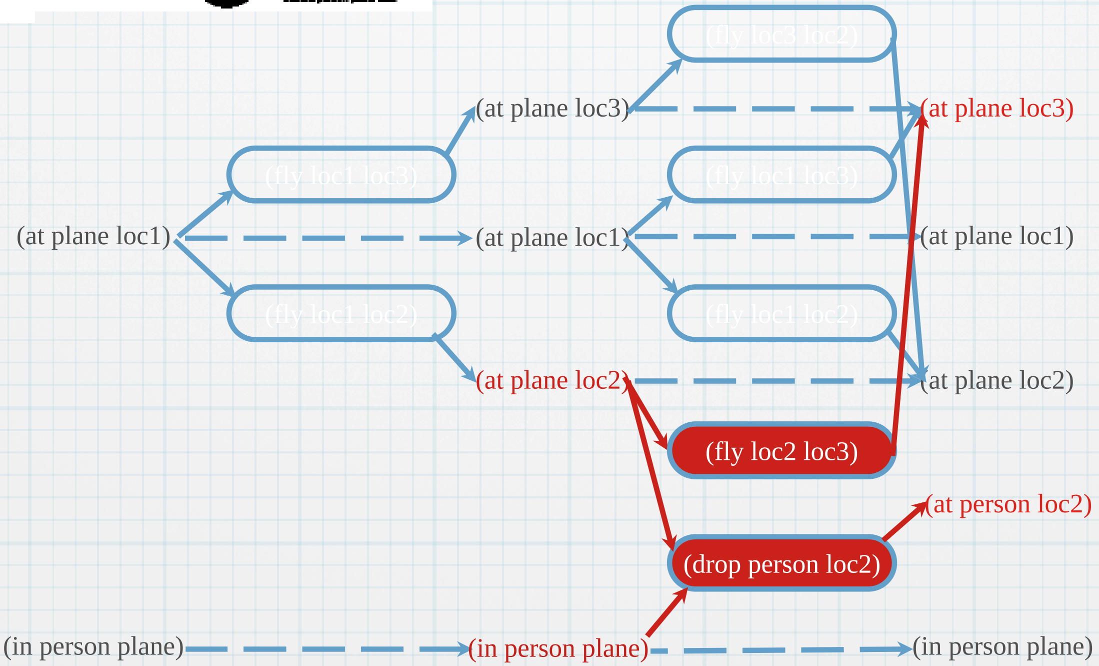
(at person loc2)

(in person plane)

Getting a Relaxed Plan







(at plane loc1)

(fly loc1 loc3)

(fly loc1 loc2)

(at plane loc3)

(at plane loc1)

(at plane loc2)

(fly loc3 loc2)

(fly loc1 loc3)

(fly loc1 loc2)

(fly loc2 loc3)

(drop person loc2)

(at plane loc3)

(at plane loc1)

(at plane loc2)

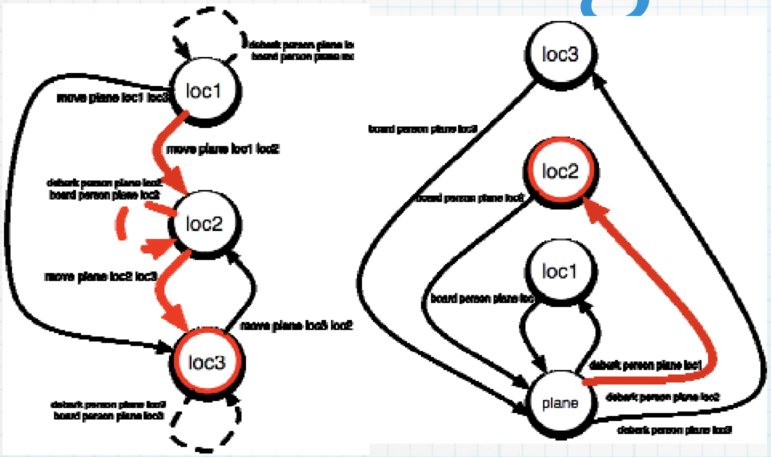
(at person loc2)

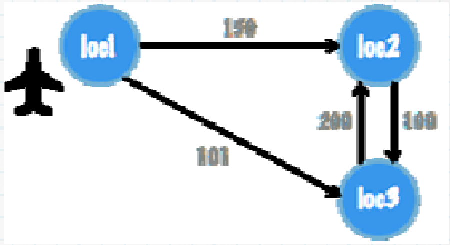
(in person plane)

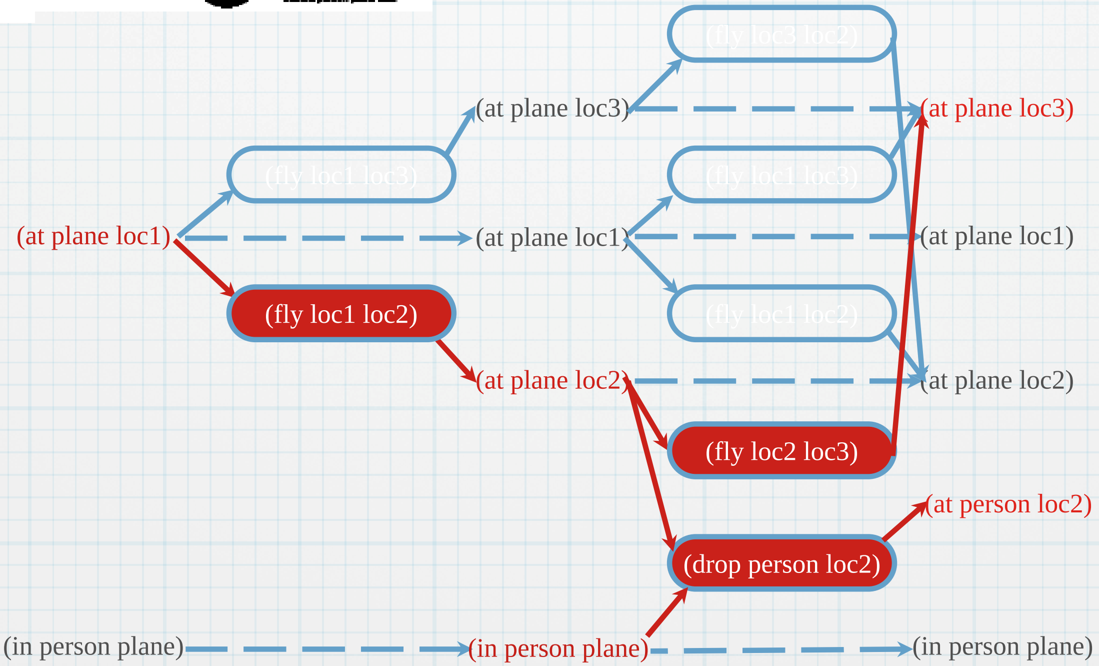
(in person plane)

(in person plane)

Getting a Relaxed Plan







(at plane loc1)

(fly loc1 loc3)

(fly loc1 loc2)

(in person plane)

(at plane loc3)

(at plane loc1)

(at plane loc2)

(in person plane)

(fly loc3 loc2)

(fly loc1 loc3)

(fly loc1 loc2)

(fly loc2 loc3)

(drop person loc2)

(at plane loc3)

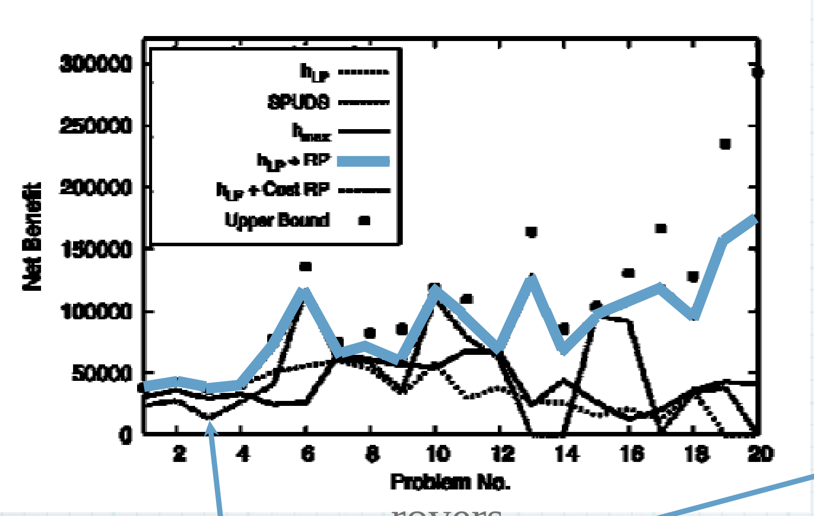
(at plane loc1)

(at plane loc2)

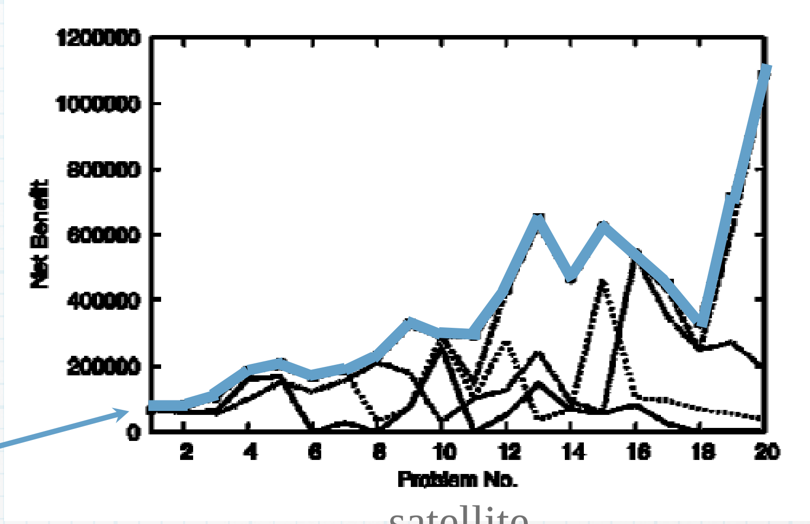
(at person loc2)

(in person plane)

Results

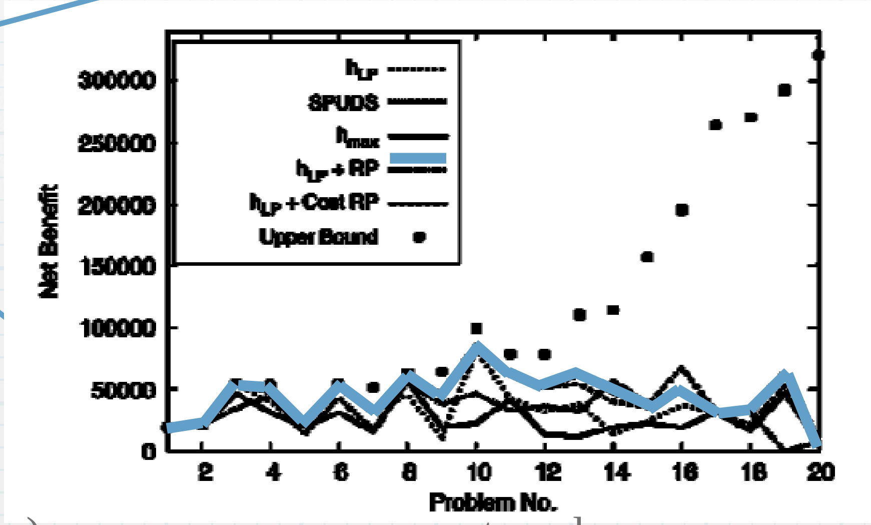


rovers



satellite

optimal solutions

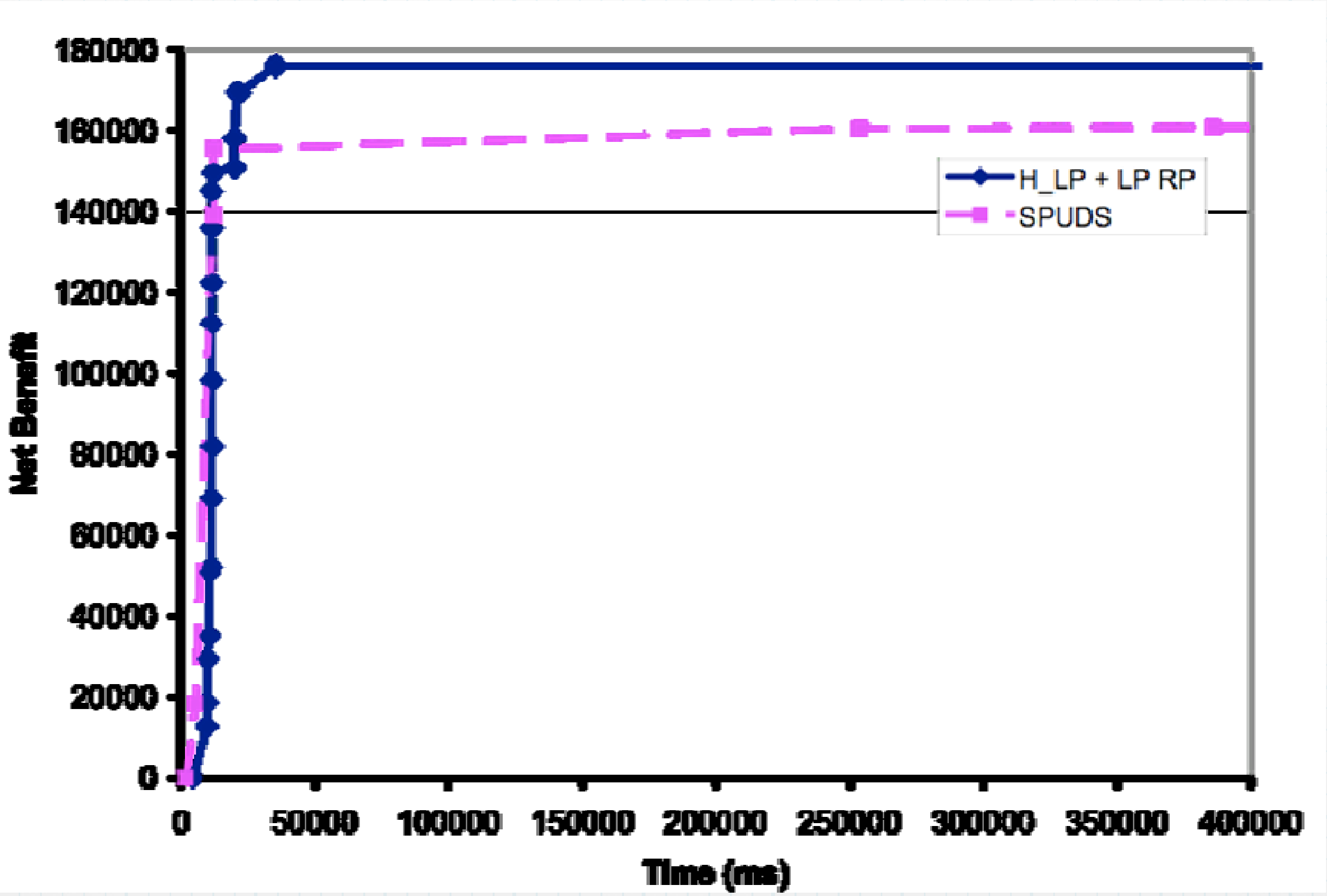


zenotravel

Found optimal solution in 15 of 60 problems

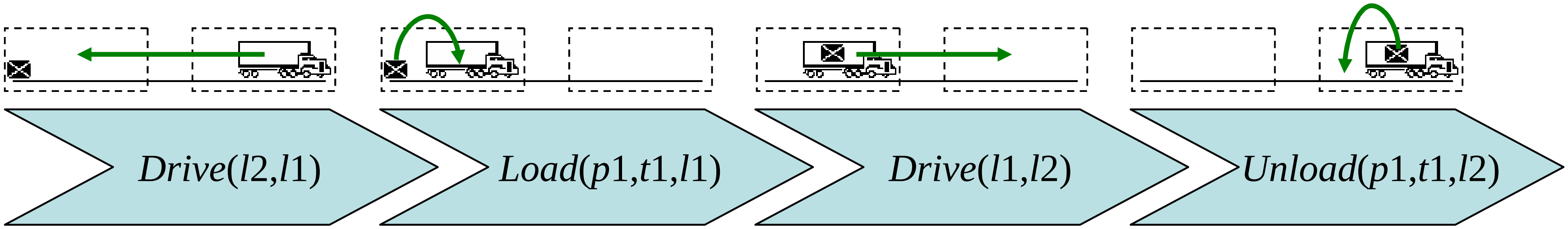
(higher net benefit is better)

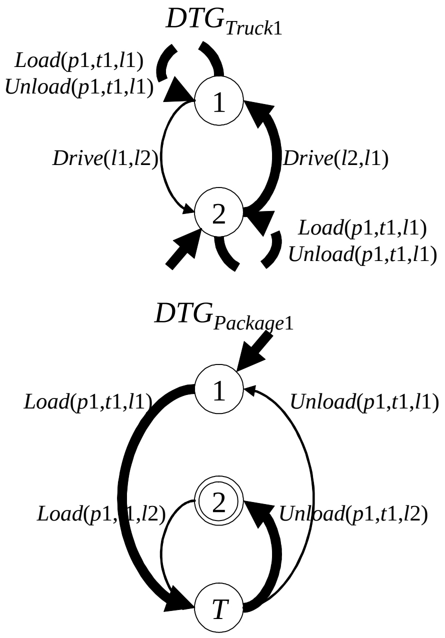
Results



180000
160000
140000
120000
100000
80000
60000
40000
20000
0
Net Benefit
0
50000
100000
150000
200000
250000
300000
350000
400000
Time (ms)
H_LP + LP RP
SPUDS

Fluent Merging to Strengthen LP Relaxation





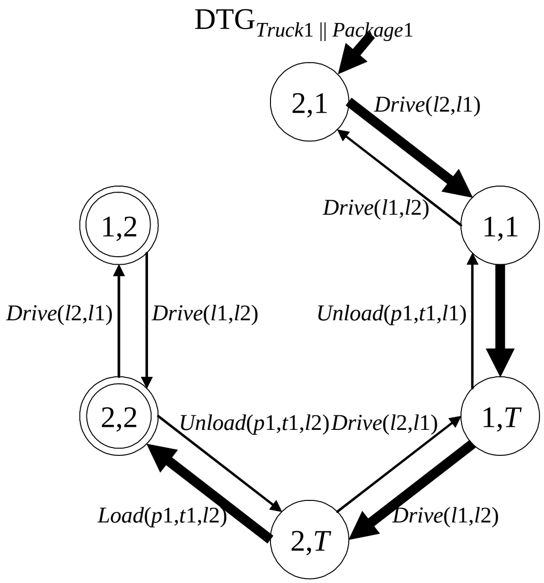
LP solution

$x_{Load(p1,t1,l1)} = 1$

$x_{Unload(p1,t1,l2)} = 1$

$x_{Drive(l2,l1)} = 1/M$

2 + 1/M



LP solution

$x_{Drive(l2,l1)} = 1$

$x_{Load(p1,t1,l1)} = 1$

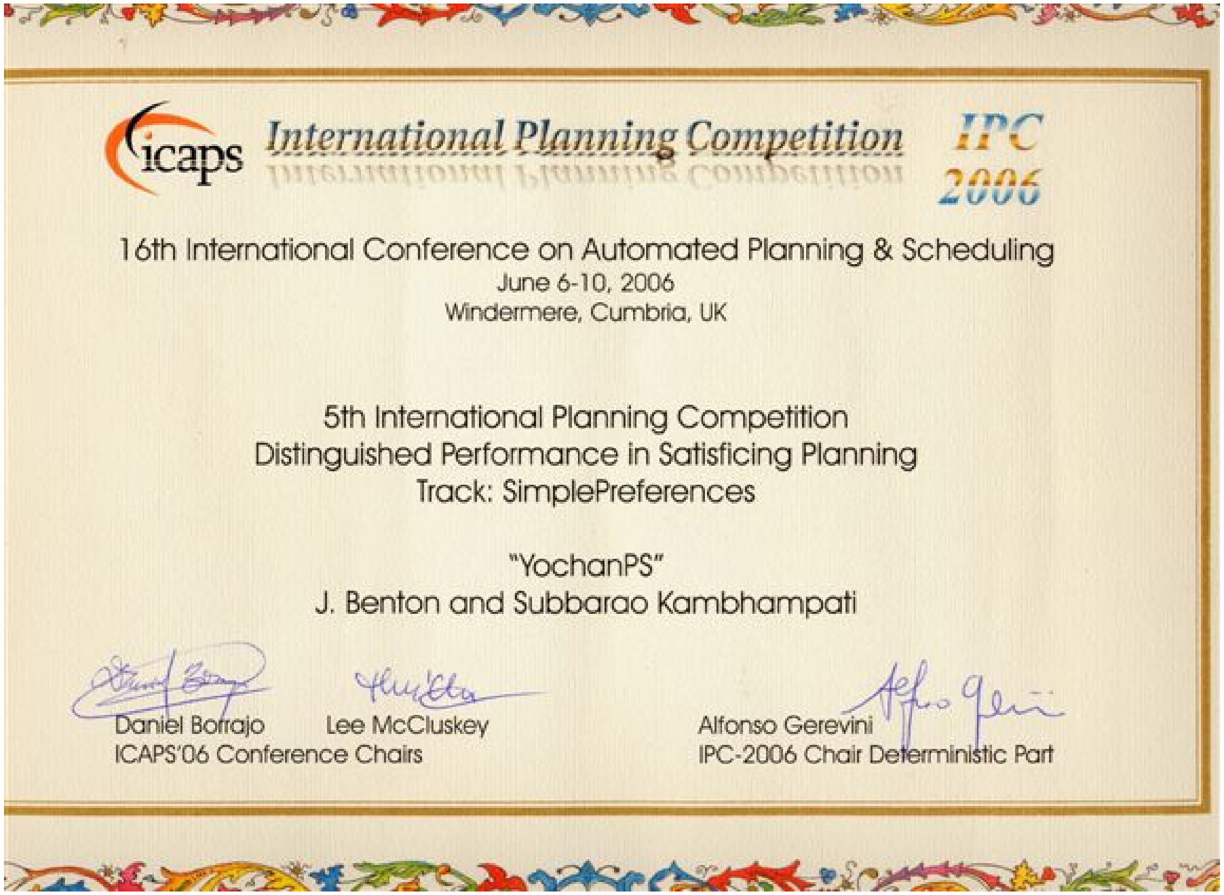
$x_{Drive(l1,l2)} = 1$

$x_{Unload(p1,t1,l2)} = 1$

4

Participation in IPC 2006

- A version of BBOP-LP -- called Yochan^{ps} took part in IPC 2006 and did quite well..



icaps International Planning Competition IPC 2006

16th International Conference on Automated Planning & Scheduling
June 6-10, 2006
Windermere, Cumbria, UK

5th International Planning Competition
Distinguished Performance in Satisficing Planning
Track: SimplePreferences

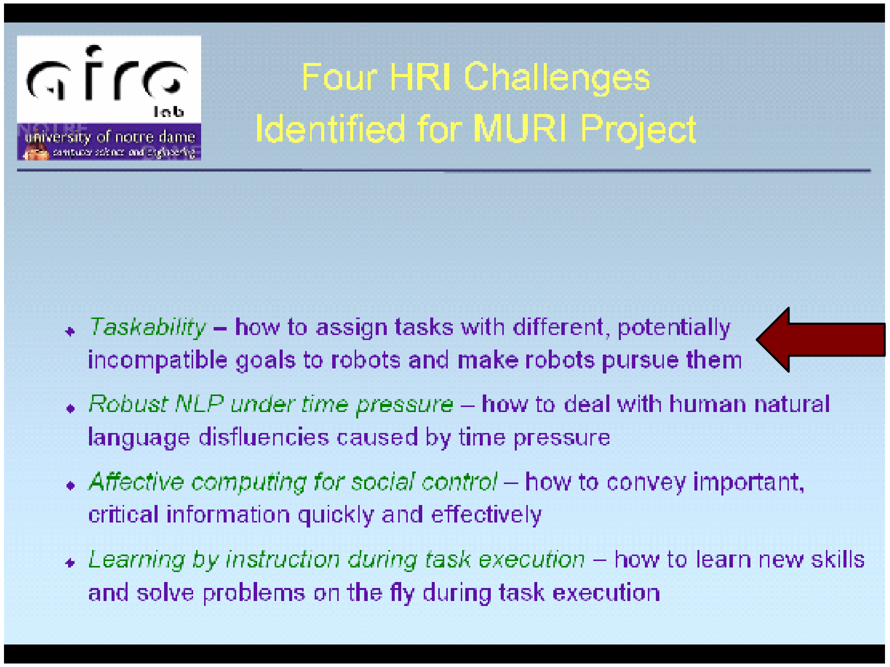
"YochanPS"
J. Benton and Subbarao Kambhampati

Daniel Borrajo Lee McCluskey
ICAPS'06 Conference Chairs

Alfonso Gerevini
IPC-2006 Chair Deterministic Part

Indana Univ; ASU
Stanford, Notre Dame

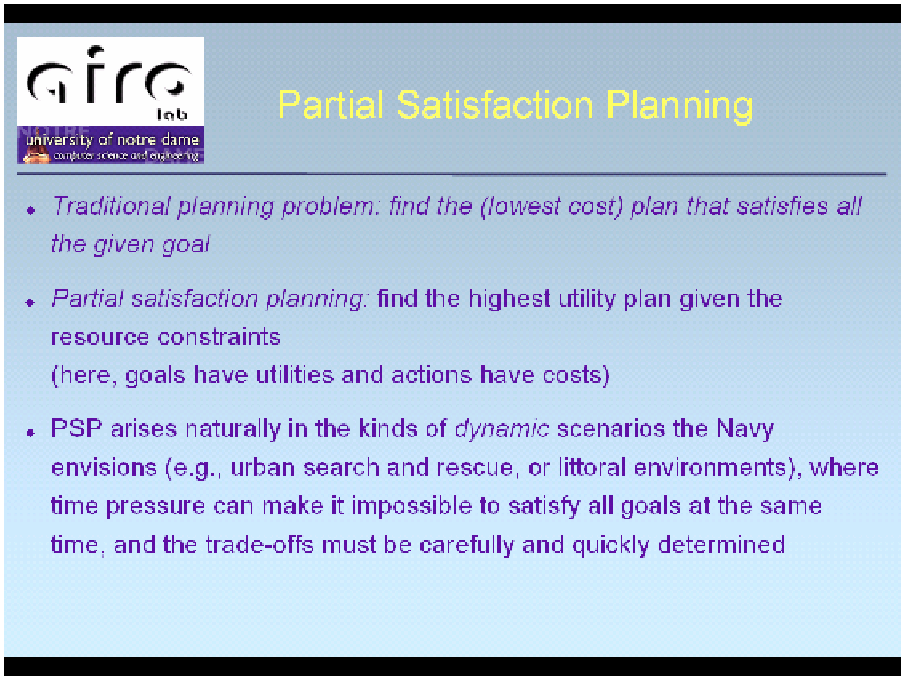
MURI 2007: Effective Human-Robot Interaction under Time Pressure



Four HRI Challenges Identified for MURI Project

- *Taskability* – how to assign tasks with different, potentially incompatible goals to robots and make robots pursue them
- *Robust NLP under time pressure* – how to deal with human natural language disfluencies caused by time pressure
- *Affective computing for social control* – how to convey important, critical information quickly and effectively
- *Learning by instruction during task execution* – how to learn new skills and solve problems on the fly during task execution

airo lab
university of notre dame
computer science and engineering

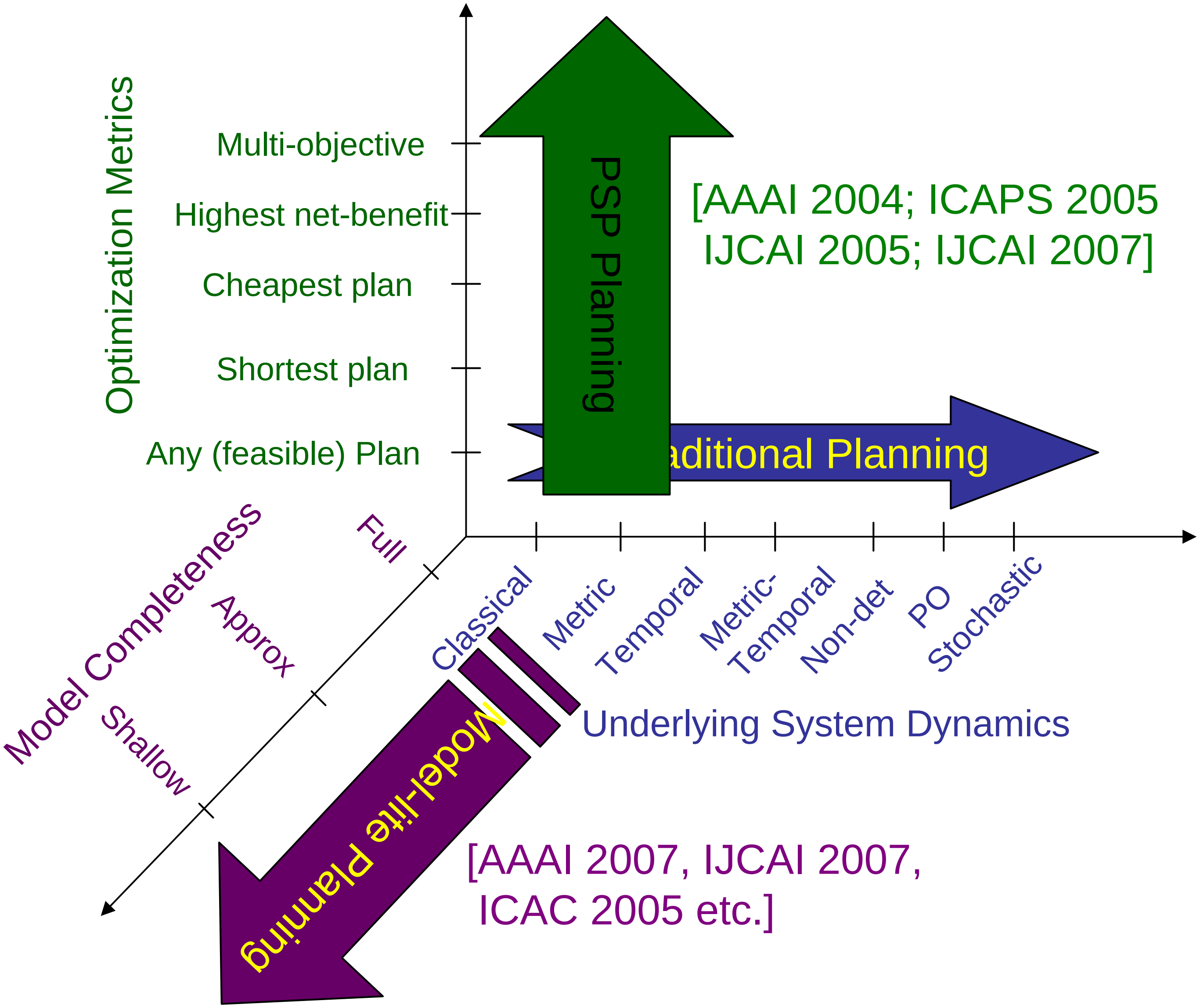


Partial Satisfaction Planning

- *Traditional planning problem: find the (lowest cost) plan that satisfies all the given goal*
- *Partial satisfaction planning:* find the highest utility plan given the resource constraints
  (here, goals have utilities and actions have costs)
- PSP arises naturally in the kinds of *dynamic* scenarios the Navy envisions (e.g., urban search and rescue, or littoral environments), where time pressure can make it impossible to satisfy all goals at the same time, and the trade-offs must be carefully and quickly determined

PSP Summary

- PSP problems are ubiquitous and foreground quality considerations
- Challenges include modeling and handling cost and utility interactions between objectives (goals)
- It is possible to combine the progress in planning graph heuristics, IP encodings and factored utility representations to attack the problem well
- Future directions
  - Strengthening the IP encodings with valid inequalities derived from fluent merging
  - Explaining why certain objectives are selected in mixed initiative scenarios..



Optimization Metrics

Multi-objective

Highest net-benefit

Cheapest plan

Shortest plan

Any (feasible) Plan

PSP Planning

[AAAI 2004; ICAPS 2005
IJCAI 2005; IJCAI 2007]

aditional Planning

Model Completeness

Full

Approx

Shallow

Classical

Metric

Temporal

Metric-Temporal

Non-det

PO

Stochastic

Underlying System Dynamics

Model-lite Planning

[AAAI 2007, IJCAI 2007,
ICAC 2005 etc.]

Motivations for Model-lite

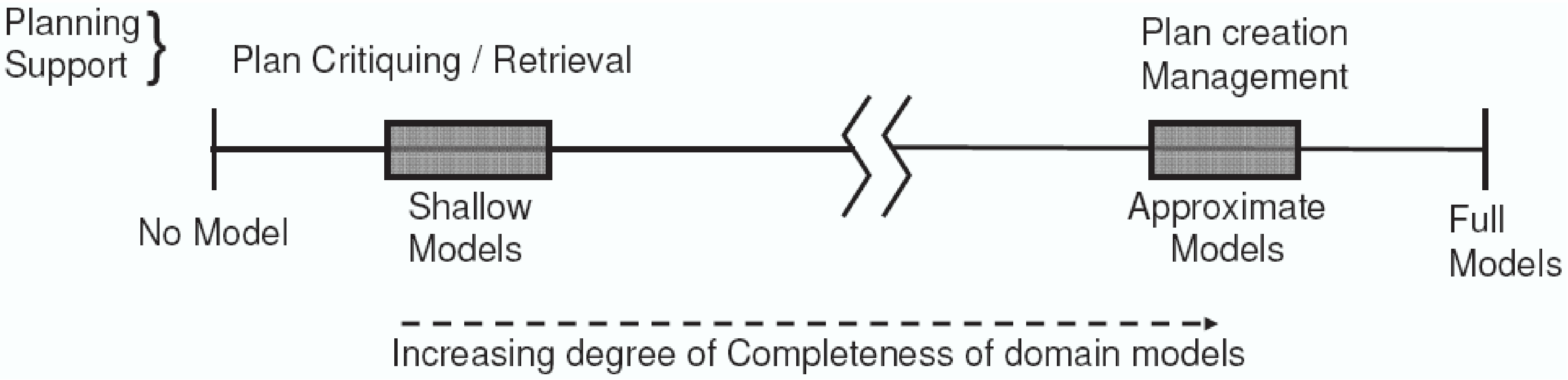
Is the only way to get more applications is to tackle more and more expressive domains?

- There are many scenarios where domain modeling is the biggest obstacle
  - Web Service Composition
    - Most services have very little formal models attached
  - Workflow management
    - Most workflows are provided with little information about underlying causal models
  - Learning to plan from demonstrations
    - We will have to contend with incomplete and evolving domain models..
- ..but our approaches assume complete and correct models..

Model-Lite Planning is Planning with incomplete models

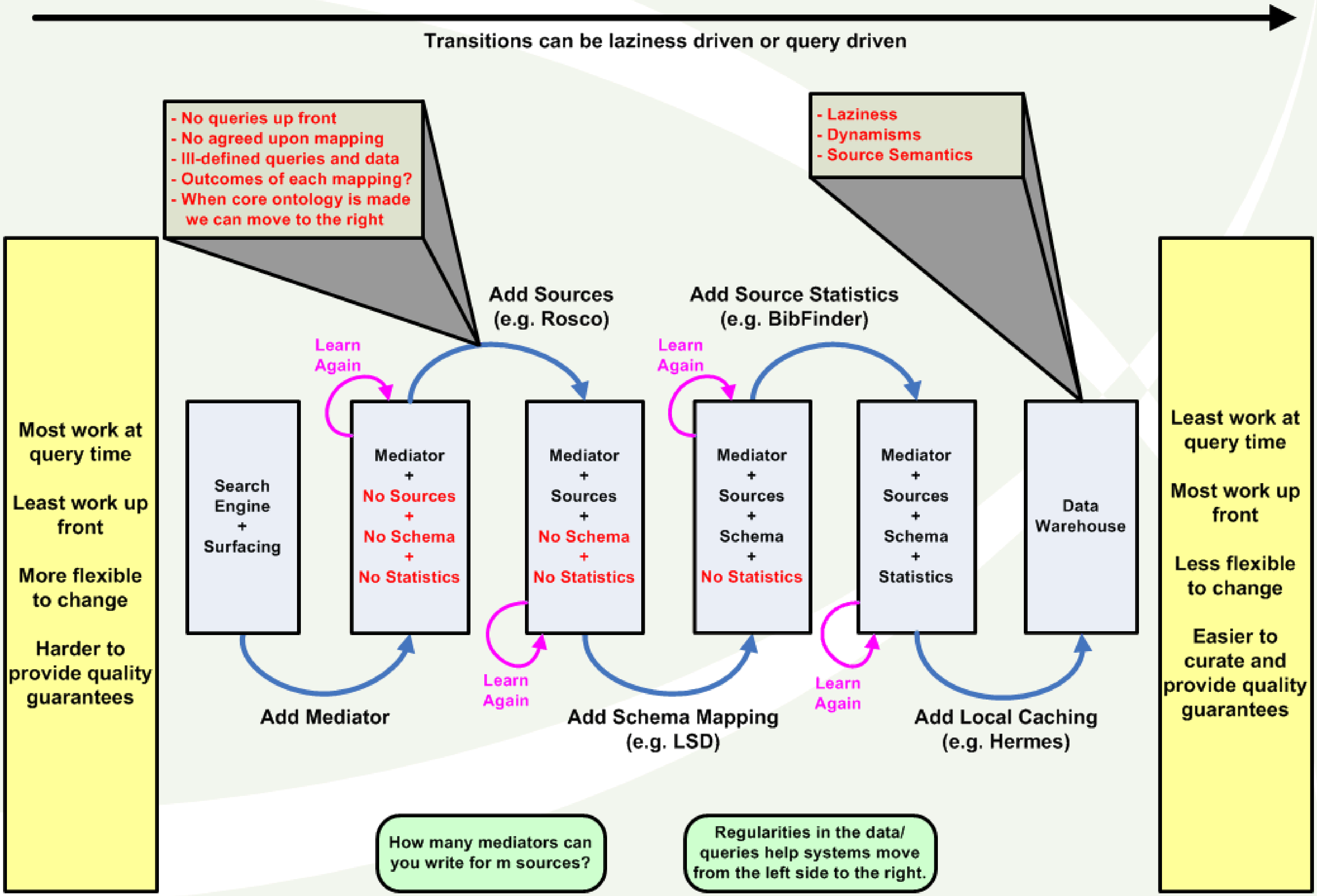
From "Any Time" to "Any Model" Planning

- .."incomplete" ➔ "not enough domain knowledge to verify correctness/optimality"
- How *incomplete* is incomplete?
  - Knowing no more than I/O types?
  - Missing a couple of preconditions/effects or user preferences?



Mediator Systems

Moving from Ad hoc Integration to Data Warehouses



Transitions can be laziness driven or query driven

- No queries up front
- No agreed upon mapping
- Ill-defined queries and data
- Outcomes of each mapping?
- When core ontology is made we can move to the right

- Laziness
- Dynamisms
- Source Semantics

Most work at query time

Least work up front

More flexible to change

Harder to provide quality guarantees

Search Engine + Surfacing

Add Mediator

Learn Again

Mediator + No Sources + No Schema + No Statistics

Add Sources (e.g. Rosco)

Learn Again

Mediator + Sources + No Schema + No Statistics

Add Schema Mapping (e.g. LSD)

Learn Again

Mediator + Sources + Schema + No Statistics

Add Source Statistics (e.g. BibFinder)

Learn Again

Mediator + Sources + Schema + Statistics

Add Local Caching (e.g. Hermes)

Data Warehouse

Least work at query time

Most work up front

Less flexible to change

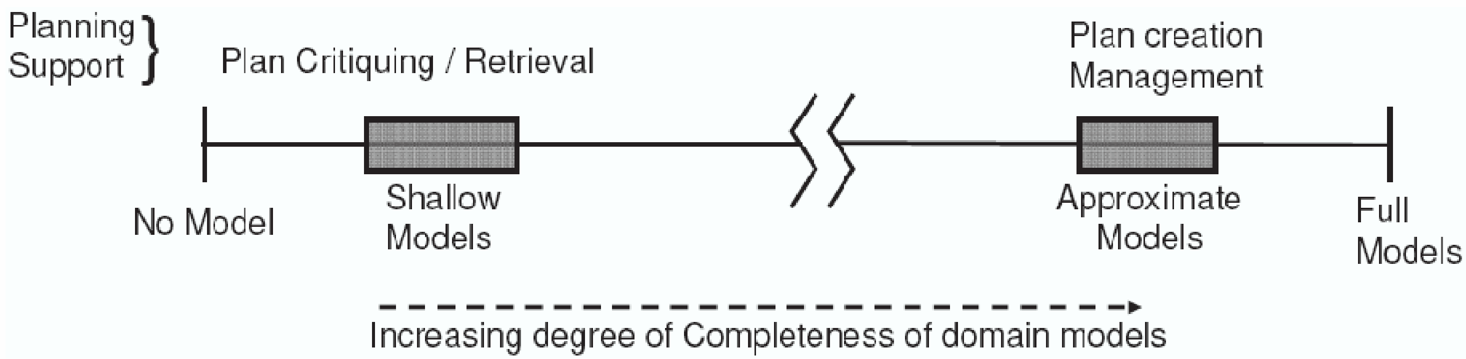
Easier to curate and provide quality guarantees

How many mediators can you write for m sources?

Regularities in the data/ queries help systems move from the left side to the right.

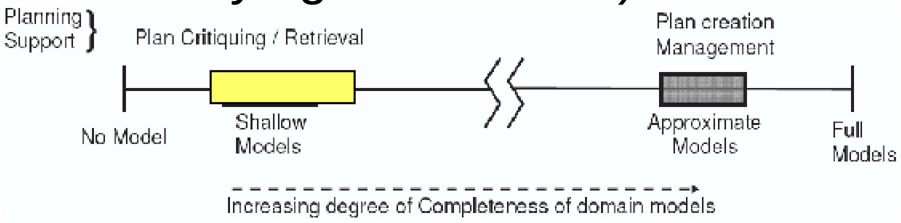
Challenges in Realizing Model-Lite Planning

1. Planning support for shallow domain models [ICAC 2005]
2. Plan creation with approximate domain models [IJCAI 2007, ICAPS Wkshp 2007]
3. Learning to improve completeness of domain models [ICAPS Wkshp 2007]



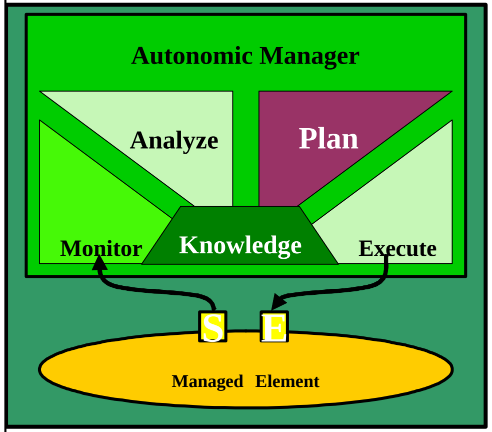
Challenge: Planning Support for Shallow Domain Models

- Provide planning support that exploits the shallow model available
- Idea: Explore wider variety of domain knowledge that can either be easily specified interactively or learned/mined. E.g.
    - I/O type specifications (e.g. Woogle)
    - Task Dependencies (e.g. workflow specifications)
  - Qn: Can these be compiled down to a common substrate?
- Types of planning support that can be provided with such knowledge
  - Critiquing plans in mixed-initiative scenarios
  - Detecting incorrectness (as against verifying correctness)



Planning in Autonomic Computing (AC)

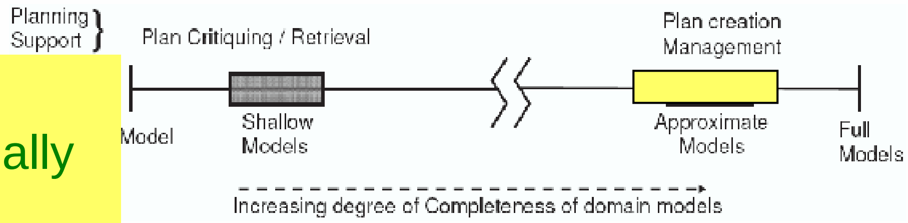
- The ‘P’ of the M-A-P-E loop in an Autonomic Manager
- Planning provides the policy engine for goal-type policies
  - Given expected system behavior (goals), determine actions to satisfy them
- Synthesis, Analysis & Maintenance of plans of action is a vital aspect of Autonomic Computing
  - Example 1: Taking high-level behavioral specifications from humans, and control the system behavior in such a way as to satisfy the specifications
    - Change requests (e.g., INSTALL, UPDATE, REMOVE) from administrator in managing software on a machine (Solution Install scenarios)
  - Example 2: Managing/propagating changes caused by installations and component changes in a networked environment
    - Remediation in the presence of failure



Challenge: Plan Creation with Approximate Domain Models

- Support plan creation despite missing details in the model. The missing details may be (1) action models (2) cost/utility models
- Example: Generate robust "line" plans in the face of incompleteness of action description
  - View model incompleteness as a form of uncertainty (e.g. work by Amir et. al.)
- Example: Generate Diverse/Multi-option plans in the face of incompleteness of cost model
  - Our IJCAI-2007 work can be viewed as being motivated this way..

Note: Model-lite planning aims to reduce the modeling burden; the planning itself may actually be harder

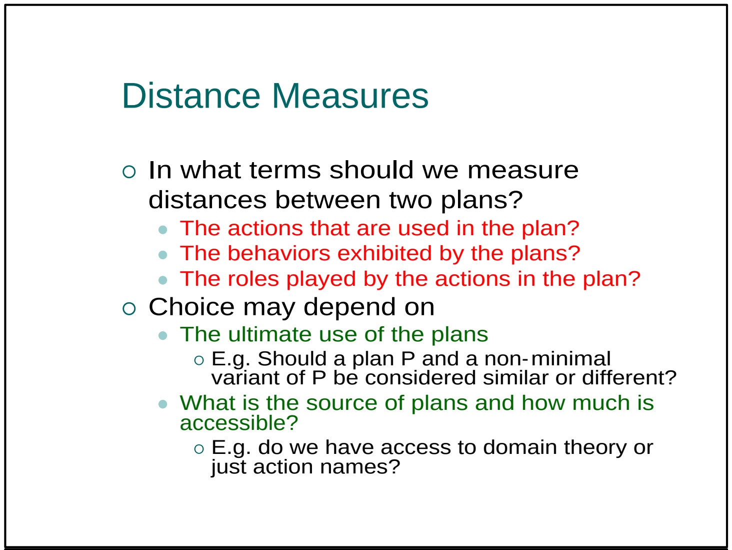


Generating Diverse Plans

- Formalized notions of bases for plan distance measures
- Proposed adaptation to existing representative, state-of-the-art, planning algorithms to search for diverse plans
  - Showed that using action-based distance results in plans that are likely to be also diverse with respect to behavior and causal structure
  - LPG can scale-up well to large problems with the proposed changes

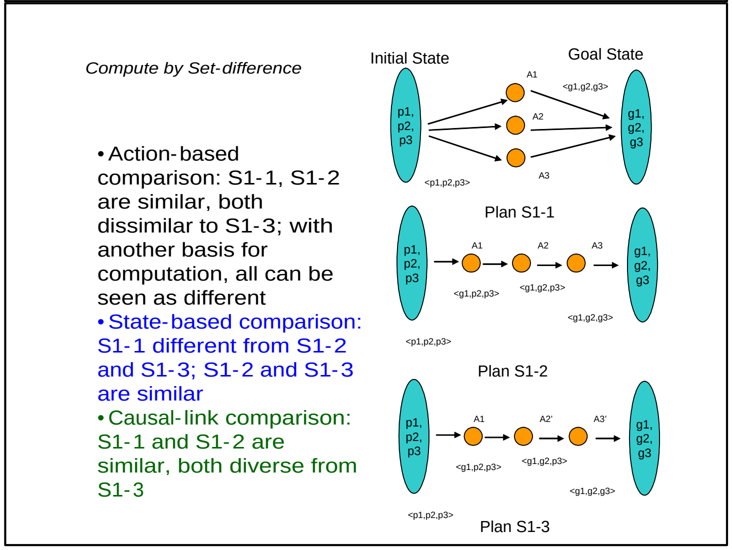
[IJCAI 2007]

- *d*DISTANT*k*SET
  - Given a distance measure δ(.,.), and a parameter k, find k plans for solving the problem that have guaranteed minimum pair-wise distance *d* among them in terms of δ(.,.)



Distance Measures

- In what terms should we measure distances between two plans?
  - The actions that are used in the plan?
  - The behaviors exhibited by the plans?
  - The roles played by the actions in the plan?
- Choice may depend on
  - The ultimate use of the plans
    - E.g. Should a plan P and a non-minimal variant of P be considered similar or different?
  - What is the source of plans and how much is accessible?
    - E.g. do we have access to domain theory or just action names?

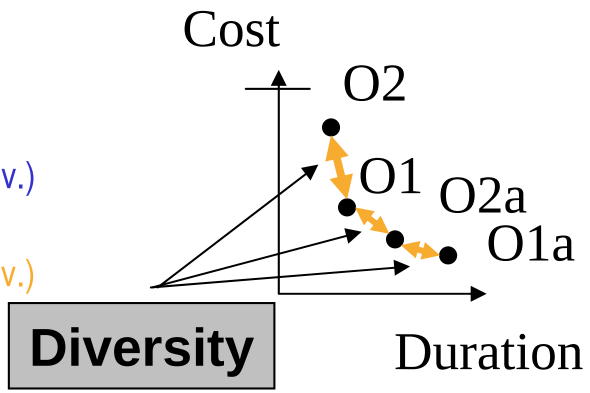


Compute by Set-difference

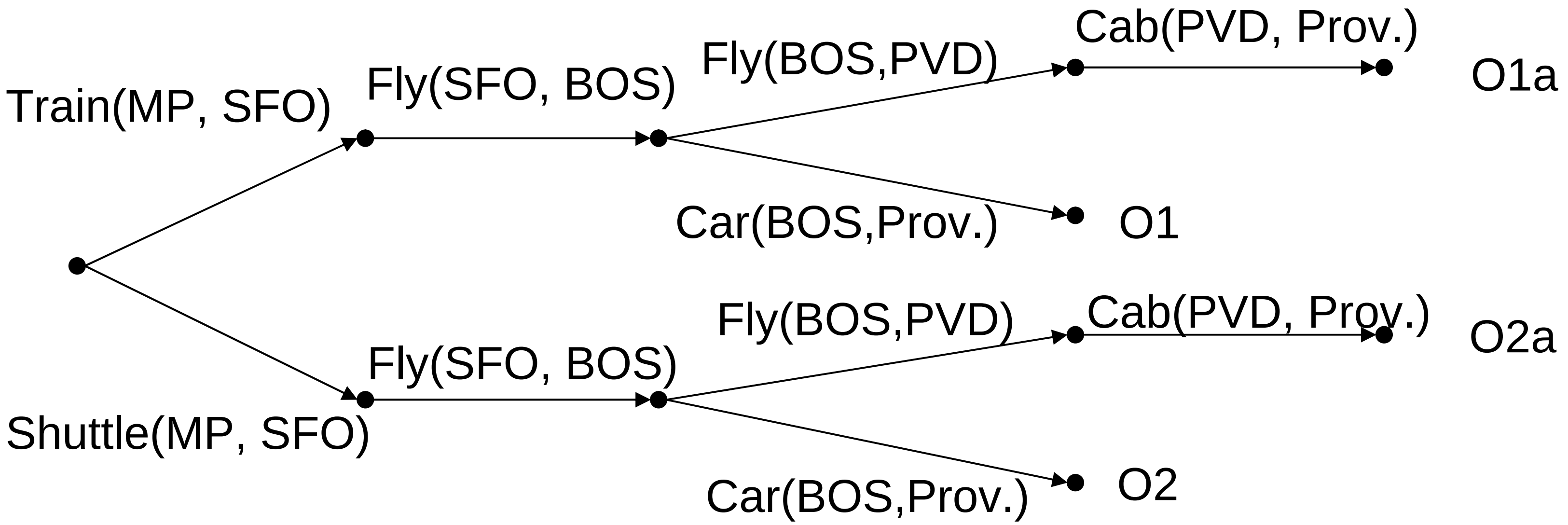
- Action-based comparison: S1-1, S1-2 are similar, both dissimilar to S1-3; with another basis for computation, all can be seen as different
- State-based comparison: S1-1 different from S1-2 and S1-3; S1-2 and S1-3 are similar
- Causal-link comparison: S1-1 and S1-2 are similar, both diverse from S1-3

Diverse Multi-Option Plans

- Each plan step presents several diverse choices
  - Option 1: Train(MP, SFO), Fly(SFO, BOS), Car(BOS, Prov.)
  - Option 1a: Train(MP, SFO), Fly(SFO, BOS), Fly(BOS, PVD), Cab(PVD, Prov.)
  - Option2: Shuttle(MP, SFO), Fly(SFO, BOS), Car(BOS, Prov.)
  - Option2a: Shuttle(MP, SFO), Fly(SFO, BOS), Fly(BOS, PVD), Cab(PVD, Prov.)
- A type of conditional plan
  - Conditional on the user's objective function
- An algorithm (MOLAO*)
  - Each generated (belief) state has an associated Pareto set of "best" sub-plans
  - Dynamic programming (state backup) combines successor state Pareto sets
    - Yes, its exponential time per backup per state ☹
      - There are approximations ☺



Diversity through Pareto Front w/ High Spread



[ICAPS 2007 Execution Wkshp]

Challenge: Learning to Improve Completeness of Domain Models

- In traditional “model-intensive” planning learning is mostly motivated for speedup
  - ..and it has gradually become less and less important with the advent of fast heuristic planners
- In model-lite planning, learning (also) helps in model acquisition and model refinement.
  - Learning from a variety of sources
    - Textual descriptions; plan traces; expert demonstrations
  - Learning in the presence of background knowledge
    - The current model serves as background knowledge for additional refinements for learning
- Example efforts
  - Much of DARPA IL program (including our LSP system); PLOW etc.
  - *Stochastic Explanation-based Learning* (ICAPS 2007 wkhop)

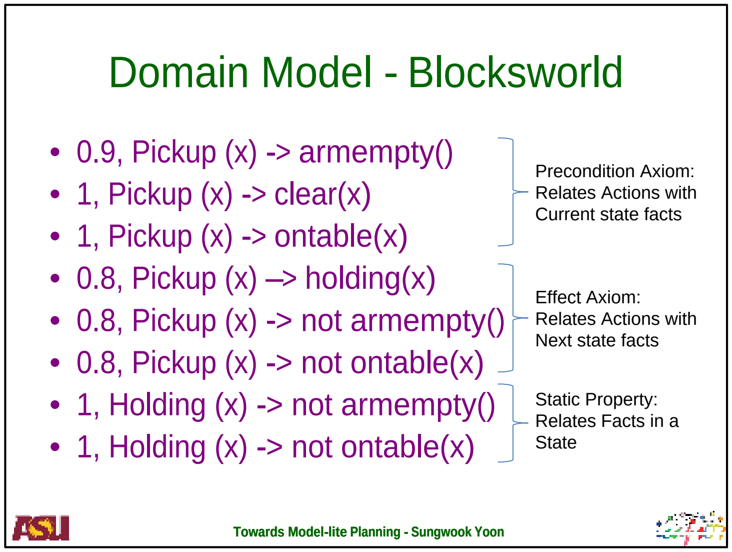
Make planning Model-lite ←→ Make learning knowledge (model) rich

Learning & Planning with incomplete models: A proposal..

- Represent incomplete domain with (relational) probabilistic logic
  - Weighted precondition axiom
  - Weighted effect axiom
  - Weighted static property axiom

- Address learning and planning problem
  - Learning involves
    - Updating the prior weights on the axioms
    - Finding new axioms
  - Planning involves
    - Probabilistic planning in the presence of precondition uncertainty
    - Consider using MaxSat to solve problems in the proposed formulation

DARPA Integrated Learning Project



Domain Model - Blocksworld

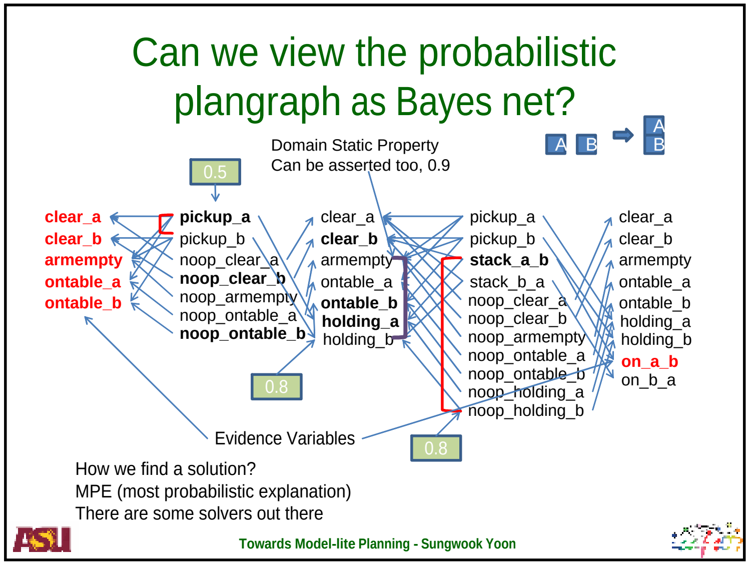
- 0.9, Pickup (x) -> armempty()
- 1, Pickup (x) -> clear(x)
- 1, Pickup (x) -> ontable(x)
- 0.8, Pickup (x) –> holding(x)
- 0.8, Pickup (x) -> not armempty()
- 0.8, Pickup (x) -> not ontable(x)
- 1, Holding (x) -> not armempty()
- 1, Holding (x) -> not ontable(x)

Precondition Axiom: Relates Actions with Current state facts

Effect Axiom: Relates Actions with Next state facts

Static Property: Relates Facts in a State

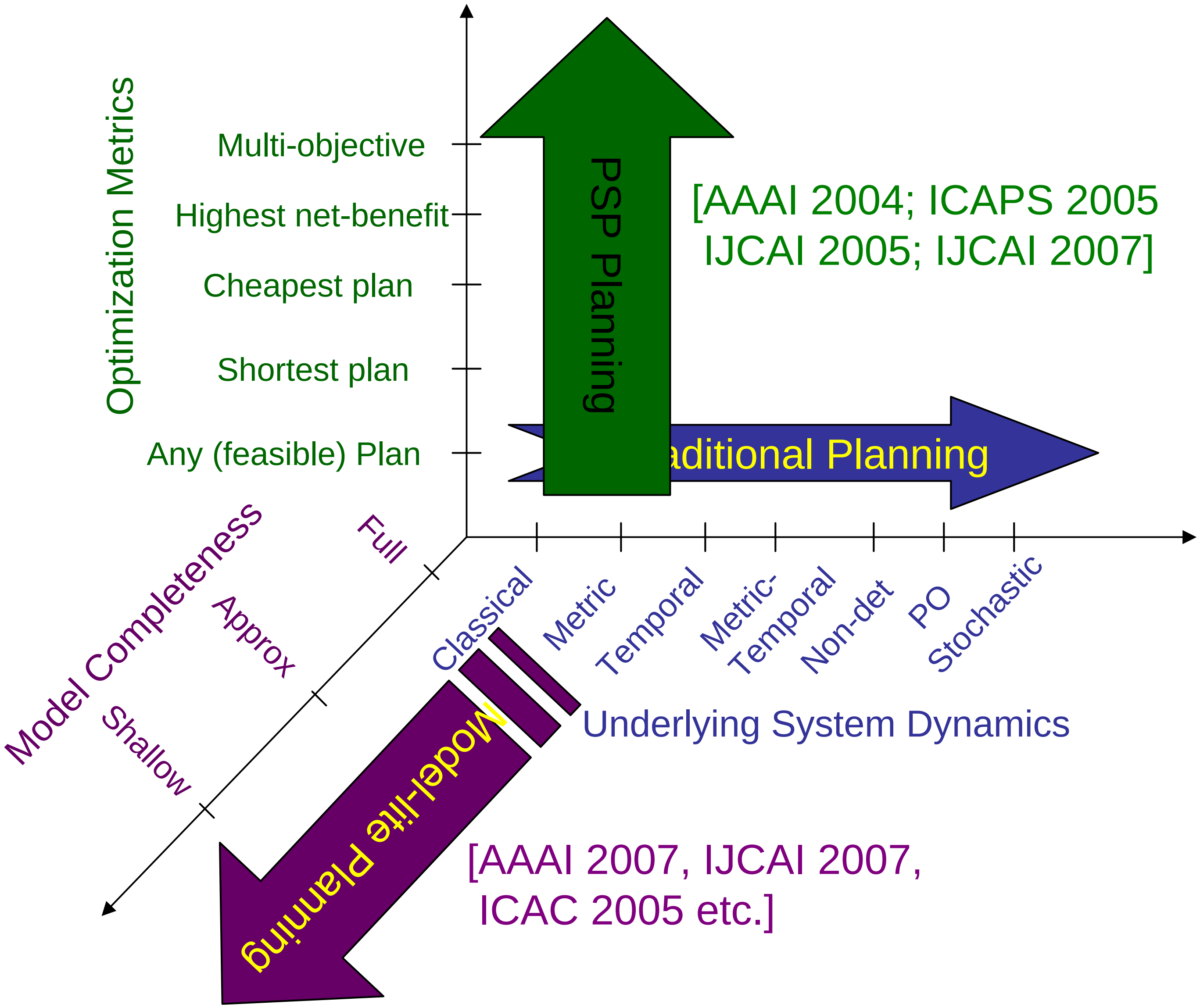
Towards Model-lite Planning - Sungwook Yoon



Can we view the probabilistic plangraph as Bayes net?

How we find a solution?
MPE (most probabilistic explanation)
There are some solvers out there

Towards Model-lite Planning - Sungwook Yoon



[AAAI 2004; ICAPS 2005
IJCAI 2005; IJCAI 2007]

[AAAI 2007, IJCAI 2007,
ICAC 2005 etc.]

Google “Yochan” or “Kambhampati” for related papers

This document was created with Win2PDF available at http://www.daneprairie.com.
The unregistered version of Win2PDF is for evaluation or non-commercial use only.