Motivation

- Ways to improve Planner Scalability
  - Problem Formulation
  - Search Space
  - Reachability Heuristics
    - Domain (Formulation) Independent
    - Work for many search spaces
    - Flexible – work with most domain features
    - Overall compliment other scalability techniques
    - Effective!!
Scalability of Planning

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.

The primary revolution in planning in the recent years has been domain-independent heuristics to scale up plan synthesis.

Topics

- Classical Planning \( \{ \text{Rao} \)\)
- Cost Based Planning \( \{ \text{Dan} \)\)
- Partial Satisfaction Planning \( \{ \text{Rao} \)\)
- Resources (Continuous Quantities) \( \{ \text{Rao} \)\)
- Temporal Planning \( \{ \text{Rao} \)\)
- Non-Deterministic/Probabilistic Planning \( \{ \text{Dan} \)\)
- Hybrid Models \( \{ \text{Dan} \)\)
Classical Planning

Rover Domain

(define (domain rovers_classical)
  (:requirements :strips :typing)
  (:types waypoint data)
  (:predicates
    (at ?x - waypoint)
    (avail ?d - data ?x - waypoint)
    (comm ?d - data)
    (have ?d - data))
  (:action drive
    :parameters (!x !y - waypoint)
    :precondition (at ?x)
    :effect (and (at !y) (not (at ?x)))
  )
  (:action commun
    :parameters (?d - data)
    :precondition (have ?d)
    :effect (comm ?d))
  (:action sample
    :parameters (?d - data ?x - waypoint)
    :precondition (and (at ?x) (avail ?d ?x))
    :effect (have ?d))
)

(define (problem rovers_classical1)
  (:domain rovers_classical)
  (:objects
    soil image rock - data
    alpha beta gamma - waypoint)
  (:init (at alpha)
    (avail soil alpha)
    (avail rock beta)
    (avail image gamma))
  (:goal (and (comm soil)
    (comm image)
    (comm rock)))
)
Classical Planning

- Relaxed Reachability Analysis
- Types of Heuristics
  - Level-based
  - Relaxed Plans
- Mutexes
- Heuristic Search
  - Progression
  - Regression
  - Plan Space
- Exploiting Heuristics

Planning Graph and Search Tree

- Envelope of Progression Tree (Relaxed Progression)
  - Proposition lists: Union of states at kth level
- Lowerbound reachability information
Level Based Heuristics

- The distance of a proposition is the index of the first proposition layer in which it appears
  - Proposition distance changes when we propagate cost functions – described later
- What is the distance of a Set of propositions??
  - Set-Level: Index of first proposition layer where all goal propositions appear
    - Admissible
    - Gets better with mutexes, otherwise same as max
  - Max: Maximum distance proposition
  - Sum: Summation of proposition distances

Example of Level Based Heuristics

\[ \text{set-level}(s_i, G) = 3 \]
\[ \text{max}(s_i, G) = \max(2, 3, 3) = 3 \]
\[ \text{sum}(s_i, G) = 2 + 3 + 3 = 8 \]
**Distance of a Set of Literals**

- \( h(S) = \sum_{p \in S} \text{lev}(\{p\}) \) for Sum Set-Level
- \( h(S) = \text{lev}(S) \) for Set-Level with memos

- \( \text{lev}(p) \): index of the first level at which \( p \) comes into the planning graph
- \( \text{lev}(S) \): index of the first level where all props in \( S \) appear non-mutexed.
  - If there is no such level, then
    - If the graph is grown to level off, then \( \infty \)
    - Else \( k+1 \) (\( k \) is the current length of the graph)

---

**How do Level-Based Heuristics Break?**

The goal \( g \) is reached at level 2, but requires 101 actions to support it.
Relaxed Plan Heuristics

- When Level does not reflect distance well, we can find a relaxed plan.
- A relaxed plan is a subgraph of the planning graph, where:
  - Every goal proposition is in the relaxed plan at the level where it first appears
  - Every proposition in the relaxed plan has a supporting action in the relaxed plan
  - Every action in the relaxed plan has its preconditions supported.
- Relaxed Plans are not admissible, but are generally effective.
- Finding the optimal relaxed plan is NP-hard, but finding a greedy one is easy. Later we will see how “greedy” can change.

Example of Relaxed Plan Heuristic

Support Goal Propositions Individually

Count Actions $\text{RP}(s_I, G) = 8$

Identify Goal Propositions
Results

Figure 4: Runtime curves on large Logistics instances for those six planners that could scale up to them. Time is shown on a logarithmic scale.

June 7th, 2006

Results (cont’d)

Figure 6: Runtime curves on Schedule instances for those planners that could handle conditional effects. Time is shown on a logarithmic scale.

June 7th, 2006
Optimizations in Heuristic Computation

- **Taming Space/Time costs**
  - Bi-level Planning Graph representation
  - Partial expansion of the PG (stop before level-off)
    - It is FINE to cut corners when using PG for heuristics (instead of search)!!

- **Branching factor can still be quite high**
  - Use actions appearing in the PG (complete)
    - Select actions in lev(S) vs Levels-off (incomplete)
  - Consider action appearing in RP (incomplete)

---

Adjusting for Negative Interactions

- Until now we assume actions only positively interact, but they often conflict
- Mutexes help us capture some negative interactions
  - Types
    - Actions: Interference/Competing Needs
    - Propositions: Inconsistent Support
  - Binary are the most common and practical
  - $|A| + 2|P|$-ary will allow us to solve the planning problem with a backtrack-free GraphPlan search
    - An action layer may have $|A|$ actions and $2|P|$ noops
  - Serial Planning Graph assumes all non-noop actions are mutex
Binary Mutexes

- sample needs at(α), drive negates at(α) --Interference--
- have(soil) only supporter is mutex with at(β) only supporter --Inconsistent Support--

Set-Level(s_i, {at(B), have(soil)}) = 2
Max(s_i, {at(B), have(soil)}) = 1

Adjusting the Relaxed Plans

- Start with RP heuristic and adjust it to take subgoal interactions into account
  - Negative interactions in terms of “degree of interaction”
  - Positive interactions in terms of co-achievement links
    - Ignore negative interactions when accounting for positive interactions (and vice versa)

\[
H_{AdjSum2M}(S) = \text{length}(\text{RelaxedPlan}(S)) + \max p, q \in S \delta(p, q)
\]
Where \(\delta(p, q) = \text{lev}\{p, q\} - \max\{\text{lev}(p), \text{lev}(q)\}\) /*Degree of –ve Interaction*/

June 7th, 2006 ICAPS'06 Tutorial T6
Anatomy of a State-space Regression planner

Problem: Given a set of subgoals (regressed state) estimate how far they are from the initial state

[AAAI 2000; AIPS 2000; AIJ 2002; JAIR 2003]

Rover Example in Regression

Should be $\infty$, $s_4$ is inconsistent, how do we improve the heuristic??

Sum($s_1$, $s_4$) = 0+0+1+1+2=4

Sum($s_1$, $s_3$) = 0+1+2+2=5
AltAlt Performance

Problem sets from IPC 2000

June 7th, 2006 ICAPS'06 Tutorial T6 23

Plan Space Search

Then it was cruelly UnPOPped

The good times return with Re(vived)POP

In the beginning it was all POP.

June 7th, 2006 ICAPS'06 Tutorial T6 24
POP Algorithm

1. **Plan Selection**: Select a plan $P$ from the search queue
2. **Flaw Selection**: Choose a flaw $f$ (open cond or unsafe link)
3. **Flaw resolution**:
   - If $f$ is an open condition, choose an action $S$ that achieves $f$
   - If $f$ is an unsafe link, choose promotion or demotion
     Update $P$
     Return NULL if no resolution exist
4. If there is no flaw left, return $P$

**Choice points**
- Flaw selection (*open condition? unsafe link? Non-backtrack choice*)
- Flaw resolution/Plan Selection (*how to select (rank) partial plan?*)

---

**PG Heuristics for Partial Order Planning**

- **Distance heuristics** to estimate cost of partially ordered plans (and to select flaws)
  - If we ignore negative interactions, then the set of open conditions can be seen as a regression state
- **Mutexes used to detect indirect conflicts in partial plans**
  - A step threatens a link if there is a mutex between the link condition and the steps’ effect or precondition
  - Post disjunctive precedences and use propagation to simplify

\[
\begin{align*}
S_k & \prec S_j \lor S_j \prec S_k \\
\text{if } & \text{mutex}(p,q) \text{ or } \text{mutex}(p,r)
\end{align*}
\]
Regression and Plan Space

RePOP’s Performance

- RePOP implemented on top of UCPOP
  - Dramatically better than any other partial order planner before it
  - Competitive with Graphplan and AltAlt
- VHPOP carried the torch at ICP 2002

You see, pop, it *is* possible to Re-use all the old POP work!

Written in Lisp, runs on Linux, 500MHz, 250MB

[IJCAI, 2001]
Exploiting Planning Graphs

- Restricting Action Choice
  - Use actions from:
    - Last level before level off (complete)
    - Last level before goals (incomplete)
    - First Level of Relaxed Plan (incomplete) – FF’s helpful actions
    - Only action sequences in the relaxed plan (incomplete) – YAHSP

- Reducing State Representation
  - Remove static propositions. A static proposition is only ever true or false in the last proposition layer.

Classical Planning Conclusions

- Many Heuristics
  - Set-Level, Max, Sum, Relaxed Plans

- Heuristics can be improved by adjustments
  - Mutexes

- Useful for many types of search
  - Progression, Regression, POCL
Cost-Based Planning

Cost-based Planning

- Propagating Cost Functions
- Cost-based Heuristics
  - Generalized Level-based heuristics
  - Relaxed Plan heuristics
Rover Cost Model

Cost Propagation

Cost reduces because of different supporter at a later level.
Cost Propagation (cont’d)

Terminating Cost Propagation

- Stop when:
  - goals are reached (no-lookahead)
  - costs stop changing (∞-lookahead)
  - k levels after goals are reached (k-lookahead)
Guiding Relaxed Plans with Costs

Cost-Based Planning Conclusions

- **Cost-Functions:**
  - Remove false assumption that level is correlated with cost
  - Improve planning with non-uniform cost actions
  - Are cheap to compute (constant overhead)
Partial Satisfaction (Over-Subscription) Planning

- Selecting Goal Sets
  - Estimating goal benefit
- Anytime goal set selection
- Adjusting for negative interactions between goals
Partial Satisfaction (Oversubscription) Planning

In many real world planning tasks, the agent often has more goals than it has resources to accomplish.

**Example: Rover Mission Planning (MER)**

Need automated support for Over-subscription/Partial Satisfaction Planning

Actions have execution costs, goals have utilities, and the objective is to find the plan that has the highest net benefit.

Adapting PG heuristics for PSP

- **Challenges:**
  - Need to propagate costs on the planning graph
  - The exact set of goals are not clear
    - Interactions between goals
    - Obvious approach of considering all $2^n$ goal subsets is infeasible

- **Idea:** Select a subset of the top level goals upfront
- **Challenge:** Goal interactions
  - Approach: Estimate the net benefit of each goal in terms of its utility minus the cost of its relaxed plan
    - Bias the relaxed plan extraction to (re)use the actions already chosen for other goals
Goal Set Selection In Rover Problem

- Found By RP
- Found By Cost Propagation
- Found By Biased RP

<table>
<thead>
<tr>
<th>comm(soil)</th>
<th>comm(rock)</th>
<th>comm(image)</th>
<th>Cost</th>
<th>Utility</th>
<th>Net Benefit</th>
</tr>
</thead>
<tbody>
<tr>
<td>√</td>
<td></td>
<td></td>
<td>35</td>
<td>20</td>
<td>-15</td>
</tr>
<tr>
<td>√</td>
<td></td>
<td></td>
<td>40</td>
<td>20</td>
<td>-20</td>
</tr>
<tr>
<td>√</td>
<td></td>
<td></td>
<td>65</td>
<td>80</td>
<td>15</td>
</tr>
<tr>
<td>√</td>
<td></td>
<td></td>
<td>25</td>
<td>50</td>
<td>0</td>
</tr>
<tr>
<td>√</td>
<td></td>
<td></td>
<td>60</td>
<td>70</td>
<td>0</td>
</tr>
<tr>
<td>√</td>
<td></td>
<td></td>
<td>65</td>
<td>110</td>
<td>45</td>
</tr>
<tr>
<td>√</td>
<td></td>
<td></td>
<td>90</td>
<td>130</td>
<td>40</td>
</tr>
</tbody>
</table>

SAPAP\textsuperscript{PS} (anytime goal selection)

- A* Progression search
  - g-value: net-benefit of plan so far
  - h-value: relaxed plan estimate of best goal set
    - Relaxed plan found for all goals
    - Iterative goal removal, until net benefit does not increase
  - Returns plans with increasing g-values.
Some Empirical Results for AltAlt\textsuperscript{PS}

Exact algorithms based on MDPs don’t scale at all

[AAAI 2004]

Adjusting for Negative Interactions (AltWlt)

- Problem:
  - What if the apriori goal set is not achievable because of negative interactions?
  - What if greedy algorithm gets bad local optimum?

- Solution:
  - Do not consider mutex goals
  - Add penalty for goals whose relaxed plan has mutexes.
    - Use interaction factor to adjust cost, similar to adjusted sum heuristic
      \[
      \max_{g_1, g_2 \in G} \{ \text{lev}(g_1, g_2) - \max(\text{lev}(g_1), \text{lev}(g_2)) \}
      \]
  - Find Best Goal set for each goal
The Problem with Plangraphs [Smith, ICAPS 04]

Assume independence between objectives

For rover: all estimates from starting location

Approach

– Construct *orienteering* problem
– Solve it
– Use as search guidance
Orienteering Problem

– Given:
  - network of cities
  - rewards in various cities
  - finite amount of gas

– Objective:
  - collect as much reward as possible before running out of gas

Orienteering Graph
The Big Question:

How do we determine which propositions go in the orienteering graph?

Propositions that:
are changed in achieving one goal
impact the cost of another goal

Sensitivity analysis

Recall: Plan Graph
Sensitivity Analysis

June 7th, 2006
ICAPS'06 Tutorial T6

Sample1
Image1
Sample3
Image2
Loc0
Loc1
Loc2
Loc3
Rover

Sensitivity Analysis

June 7th, 2006
ICAPS'06 Tutorial T6
Basis Set Algorithm

For each goal:
Construct a relaxed plan
For each net effect of relaxed plan:
   Reset costs in PG
      Set cost of net effect to 0
      Set cost of mutex initial conditions to $\infty$
   Compute revised cost estimates
   If significantly different,
      add net effect to basis set

25 Rocks
PSP Conclusions

- Goal Set Selection
  - Apriori for Regression Search
  - Anytime for Progression Search
  - Both types of search use greedy goal insertion/removal to optimize net-benefit of relaxed plans

- Orienteering Problem
  - Interactions between goals apparent in OP
  - Use solution to OP as heuristic
  - Planning Graphs help define OP
Planning with Resources

- Propagating Resource Intervals
- Relaxed Plans
- Handling resource subgoals

June 7th, 2006 ICAPS'06 Tutorial T6

Rover with power Resource

(define (domain rovers_resource)
  (:requirements :strips :typing)
  (:types waypoint data)
  (:predicates (comm ?d - data)
    (have ?d - data)
    (at ?x - waypoint)
    (avail ?d - data ?x - waypoint))
  (:functions (power)
    (effort ?x ?y - waypoint)
    (effort ?d - data))

  (:action drive
    :parameters (?x ?y - waypoint)
    :precondition (and (at ?x) (> (power) (effort ?x ?y)))
    :effect (and (not (at ?x))
               (decrease (power) (effort ?x ?y))))

  (:action common
    :parameters (?d - data)
    :precondition (and (have ?d) (> (power) 5))
    :effect (and (decrease (power) 5)))

  (:action sample
    :parameters (?d - data ?x - waypoint)
    :precondition (and (at ?x) (avail ?d ?x)
                       (> (power) (effort ?d)))
    :effect (and (decrease (power) (effort ?d)))

  (:action recharge
    :parameters ()
    :precondition (and (avail ?d ?x) (<= (power) 75))
    :effect (increase (power) 25)))

(define (problem rovers_resource1)
  (:domain rovers_resource)
  (:objects
    soil image rock - data
    alpha beta gamma - waypoint)
  (:init (at alpha)
    (avail soil alpha)
    (avail rock beta)
    (avail image gamma)
    (= (effort alpha beta) 10)
    (= (effort beta alpha) 5)
    (= (effort alpha gamma) 30)
    (= (effort gamma alpha) 5)
    (= (effort beta gamma) 15)
    (= (effort gamma beta) 10)
    (= (effort soil) 20)
    (= (effort rock) 25)
    (= (effort image) 5)
    (= (power) 25))
  (:goal (and (comm soil)
               (comm image)
               (comm rock)))
)

Resource Usage, Same as costs for This example
Resource Intervals

Resource Interval assumes independence among Consumers/Producers

UB: UB(noop) + recharge = 25 + 25 = 50
LB: LB(noop) + sample + drive = 25 + (-20) + (-10) = -5

UB: UB(noop) + recharge = 50 + 25 = 75
LB: LB(noop) + drive + sample + drive + sample + commun + drive + drive
= -5 + (-30) + (-20) + (-10) + (-25) + (-5) + (-15) + (-5) = -115

Resource Intervals (cont’d)
Relaxed Plan Extraction with Resources

Start Extraction as before
Track “Maximum” Resource Requirements For actions chosen at each level
May Need more than One Supporter For a resource!!

Results

Figure 13: Runtime curves on Settlers instances for the planners favoring speed. Time is shown on a logarithmic scale, instance size scales from left to right.

June 7th, 2006 ICAPS'06 Tutorial T6
Results (cont’d)

Planning With Resources Conclusion

- Resource Intervals allow us to be optimistic about reachable values
  - Upper/Lower bounds can get large
- Relaxed Plans may require multiple supporters for subgoals
- Negative Interactions are much harder to capture

Figure 8: Runtime curves on Depots instances for the planners favoring speed. Time is shown on a logarithmic scale; instance size scales from left to right.
Temporal Planning

- Temporal Planning Graph
  - From Levels to Time Points
  - Delayed Effects
- Estimating Makespan
- Relaxed Plan Extraction
Rover with Durative Actions

SAPA

Planning Problem
Generate start state

Queue of Time-
Stamped states

Select state with
lowest f-value

Satisfies
Goals?

No

Expand state
by applying
actions

Yes

Build RTPG
Propagate Cost
functions
Extract relaxed plan
Adjust for
 Mutexes; Resources

f can have both
Cost & Makespan
components

Partialize the
 p.c. plan
Return
 o.c and p.c plans

[EC2 2001; AIPS 2002; ICAPS 2003; JAIR 2003]
Search through time-stamped states

- **Goal Satisfaction:**
  \( S = (P, M, \Pi, Q, t) \Rightarrow G \) if \( \forall <p_i, t_i> \in G \) either:
  - \( \exists <p_i, t_j> \in P, t_j < t_i \) and no event in \( Q \) deletes \( p_i \).
  - \( \exists e \in Q \) that adds \( p_i \) at time \( t_e < t_i \).

- **Action Application:**
  Action \( A \) is applicable in \( S \) if:
  - All instantaneous preconditions of \( A \) are satisfied by \( P \) and \( M \).
  - \( A \)'s effects do not interfere with \( \Pi \) and \( Q \).
  - No event in \( Q \) interferes with persistent preconditions of \( A \).
  - \( A \) does not lead to concurrent resource change

- **When \( A \) is applied to \( S \):**
  - \( P \) is updated according to \( A \)'s instantaneous effects.
  - Persistent preconditions of \( A \) are put in \( \Pi \).
  - Delayed effects of \( A \) are put in \( Q \).

---

**Temporal Planning**

- **Record First time Point Action/Proposition is First Reachable**
- **Assume Latest Start Time for actions in RP**
SAPA at IPC-2002

- Satellite (complex setting)
- Rover (time setting)

Temporal Planning Conclusion

- Levels become Time Points
- Makespan and plan length/cost are different objectives
- Set-Level heuristic measures makespan
- Relaxed Plans measure makespan and plan cost
Non-Deterministic Planning

- Belief State Distance
- Multiple Planning Graphs
- Labelled Uncertainty Graph
- Implicit Belief states and the CFF heuristic
Conformant Rover Problem

(define (domain rovers_conformant)
  (:requirements :strips :typing)
  (:types waypoint data)
  (:predicates
    (at ?x - waypoint)
    (avail ?d - data ?x - waypoint)
    (comm ?d - data)
    (have ?d - data))
  (:action drive
    :parameters (?x ?y - waypoint)
    :precondition (at ?x)
    :effect (and (at ?y) (not (at ?x))))
  (:action commun
    :parameters (?d - data)
    :precondition (have ?d)
    :effect (comm ?d))
  (:action sample
    :parameters (?d - data ?x - waypoint)
    :precondition (at ?x)
    :effect (when (avail ?d ?x) (have ?d)))
)

(define (domain rovers_conformant1)
  (:domain rovers)
  (:objects
    soil image rock - data
    alpha beta gamma - waypoint)
  (:init (at alpha)
    (oneof (avail soil alpha) (avail soil beta) (avail soil gamma))
  )
  (:goal (comm soil)))

Search in Belief State Space

| avail(soil, α) at(α) have(soil) | sample(soil, β) |
| avail(soil, β) at(α) | avail(soil, α) at(β) have(soil) |
| avail(soil, γ) at(α) | drive(α, β) |
| sample(soil, α) | drive(α, γ) |
| drive(α, γ) | drive(b, γ) |
| | drive(α, β) | sample(soil, β) |
| | drive(α, γ) | drive(b, γ) |
Belief State Distance

---

**Compute Classical Planning Distance Measures**
Assume can reach closest goal state
Aggregate State Distances

---

**Estimate Plans for each state pair**
Capture Positive Interaction & Independence

---

Max = 15, 20
Sum = 20, 20
Union = 17, 20

---

[ICAPS 2004]
State Distance Aggregations

[Bryce et al., 2005]

Multiple Planning Graphs

- Build A planning Graph
- For each State in the belief state
- Extract Relaxed Plans from each
- Step-wise union relaxed plans

Figure 16: Ratio of heuristic estimates for distance between $BS_P$ and $BS_T$ to optimal plan length. Rv = Rovers, L = Logistics, B = BT, BC = BTC, C = Cube Center, R = Ring.
**Labelled Uncertainty Graph**

- Action labels are the Conjunction (intersection) of their Precondition labels.
- Effect labels are the Disjunction (union) of supporter labels.
- Stop when goal is Labeled with every state.

**Labelled Relaxed Plan**

- Subgoals and Supporters: Need not be used for Every state where they are reached (labeled).
- Must Pick Enough Supporters to cover the (sub)goals.
Comparison of Planning Graph Types

[LJAIR, 2006]

LUG saves Time
Sampling more worlds, Increases cost of MG
Sampling more worlds, Improves effectiveness of LUG

Figure 15: Total Time (ms) to solve all problems when sampling worlds to use in heuristic computation.

State Agnostic Planning Graphs (SAG)

- LUG represents multiple explicit planning graphs
- SAG uses LUG to represent a planning graph for every state
- The SAG is built once per search episode and we can use it for relaxed plans for every search node, instead of building a LUG at every node
- Extract relaxed plans from SAG by ignoring planning graph components not labeled by states in our search node.
Ignore irrelevant labels
➢ Largest LUG == all LUGs

Build a LUG for all states (union of all belief states)

June 7th, 2006 ICAPS'06 Tutorial T6

CFF (implicit belief states + PG)

sample(soil, α)

at(α)
drive(α, β)

at(β)
drive(α, γ)

drive(α, β)

avail(soil, α)
at(α)

drive(b, γ)

at(β)
sample(soil, β)

drive(α, γ)

drive(α, β)

drive(α, γ)

[ AAAI, 2005]
Conditional Planning

- Actions have Observations
- Observations branch the plan because:
  - Plan Cost is reduced by performing less “just in case” actions – each branch performs relevant actions
  - Sometimes actions conflict and observing determines which to execute (e.g., medical treatments)
- We are ignoring negative interactions
  - We are only forced to use observations to remove negative interactions
  - Ignore the observations and use the conformant relaxed plan
    - Suitable because the aggregate search effort over all plan branches is related to the conformant relaxed plan cost

June 7th, 2006
ICAPS'06 Tutorial T6
Non-Deterministic Planning Conclusions

- Measure positive interaction and independence between states co-transitioning to the goal via overlap
  - Labeled planning graphs and CFF SAT encoding efficiently measure conformant plan distance
- Conformant planning heuristics work for conditional planning without modification

Stochastic Planning
Stochastic Rover Example

```
(define (domain rovers_stochastic)
  (:requirements :strips :typing)
  (:types waypoint data)
  (:predicates
    (at ?x - waypoint)
    (avail ?d - data)
    (comm ?d - data)
    (have ?d - data))
  (:action drive
    :parameters (?x ?y - waypoint)
    :precondition (at ?x)
    :effect (and (at ?y) (not (at ?x))))
  (:action commun
    :parameters (?d - data)
    :precondition (have ?d)
    :effect (probabilistic 0.8 (comm ?d))
  (:action sample
    :parameters (?d - data ?x - waypoint)
    :precondition (at ?x)
    :effect (when (avail ?d ?x)
               (probabilistic 0.9 (have ?d))))
)
```

Search in Probabilistic Belief State Space
Handling Uncertain Actions

- Extending LUG to handle uncertain actions requires label extension that captures:
  - State uncertainty (as before)
  - Action outcome uncertainty
    - Problem: Each action at each level may have a different outcome. The number of uncertain events grows over time – meaning the number of joint outcomes of events grows exponentially with time
    - Solution: Not all outcomes are important. Sample some of them – keep number of joint outcomes constant.

Monte Carlo LUG (McLUG)

- Use Sequential Monte Carlo in the Relaxed Planning Space
  - Build several deterministic planning graphs by sampling states and action outcomes
  - Represent set of planning graphs using LUG techniques
    - Labels are sets of particles
    - Sample which Action outcomes get labeled with particles
    - Bias relaxed plan by picking actions labeled with most particles to prefer more probable support
McLUG for rover example

Sample States for Initial layer
-- avail(soil, γ) not sampled

Particles in action
Label must sample action outcome

P0 → A0
P1 → A1
P2 → A2
P3 → A3

% of particles
Support goal, need At least ½
Okay to stop

Logistics Domain Results

Scalable, w/ reasonable quality
Grid Domain Results

Grid Domain Results

Grid Domain Results

Grid Domain Results

Grid(0.8) time(s)

Grid(0.5) time(s)

Grid(0.8) length

Grid(0.5) length

Again, good scalability and quality!

Need More Particles for broad beliefs

Direct Probability Propagation

- Alternative to label propagation, we can propagate numeric probabilities
  - Problem: Numeric Propagation tends to assume only independence or positive interaction between actions and propositions.
    - With probability, we can vastly under-estimate the probability of reaching propositions
  - Solution: Propagate Correlation – measures pair-wise independence/pos interaction/neg interaction
    - Can be seen as a continuous mutex
Correlation

- \( C(x, y) = \frac{\Pr(x, y)}{\Pr(x)\Pr(y)} \)

- If:
  - \( C(x, y) = 0 \), then \( x, y \) are mutex
  - \( 0 < C(x, y) < 1 \), then \( x, y \) interfere
  - \( C(x, y) = 1 \), then \( x, y \) are independent
  - \( 1 < C(x, y) < 1/\Pr(x) \), then \( x, y \) synergize
  - \( C(x, y) = 1/\Pr(x) = 1/\Pr(y) \), then \( x, y \) are completely correlated

Probability of a set of Propositions

- \( \Pr(x_1, x_2, \ldots, x_n) = \prod_{i=1..n} \Pr(x_i|x_1\ldots x_{i-1}) \)
  - Chain Rule
  - \( \Pr(x_i|x_1\ldots x_{i-1}) = \Pr(x_1\ldots x_{i-1} | x_i) \Pr(x_i) \)
    - Bayes Rule
    - \( \Pr(x_1\ldots x_{i-1} | x_i) \Pr(x_i) \approx \Pr(x_1|x_i) \ldots \Pr(x_{i-1}|x_i)\Pr(x_i) \)
      - Assume Independence
      - Bayes Rule
      - \( = \Pr(x_i|x_1) \ldots \Pr(x_i|x_{i-1}) \Pr(x_i) \)
        - Correlation
        - \( = \Pr(x_i) \Pi_{j=1..i-1} C(x_i, x_j) \)
  - \( \Pr(x_1, x_2, \ldots, x_n) = \prod_{i=1..n} \Pr(x_i) \Pi_{j=1..i-1} C(x_i, x_j) \)
Probability Propagation

- The probability of an Action being enabled is the probability of its preconditions (a set of propositions).
- The probability of an effect is the product of the action probability and outcome probability.
- A single (or pair of) proposition(s) has probability equal to the probability it is given by the best set of supporters.
- The probability that a set of supporters gives a proposition is the sum over the probability of all possible executions of the actions.

Results

Figure 2: Run times (s), Plan lengths, and Expanded Nodes vs. probability threshold for sandcastle-67

Figure 3: Run times (s), Plan lengths, and Expanded Nodes vs. probability threshold for slippery gripper
Stochastic Planning Conclusions

- Number of joint action outcomes too large
  - Sampling outcomes to represent in labels is much faster than exact representation
- SMC gives us a good way to use multiple planning graph for heuristics, and the McLUG helps keep the representation small
- Numeric Propagation of probability can better capture interactions with correlation
  - Can extend to cost and resource propagation

Hybrid Planning Models
Hybrid Models

- Metric-Temporal w/ Resources (SAPA)
- Temporal Planning Graph w/ Uncertainty (Prottle)
- PSP w/ Resources (SAPA\(\text{MPS}\))
- Cost-based Conditional Planning (CLUG)

Propagating Temporal Cost Functions

- Shuttle (Tempe, Phx): Cost: $20; Time: 1.0 hour
- Helicopter (Tempe, Phx): Cost: $100; Time: 0.5 hour
- Car (Tempe, LA): Cost: $100; Time: 10 hour
- Airplane (Phx, LA): Cost: $200; Time: 1.0 hour
Heuristics based on cost functions

- If we want to minimize makespan:
  - $h = t_0$
- If we want to minimize cost:
  - $h = CostAggregate(G, t_\infty)$
- If we want to minimize a function $f(time, cost)$ of cost and makespan:
  - $h = \min f(t, Cost(G, t)) \ s.t. \ t_0 \leq t \leq t_\infty$
  - E.g. $f(time, cost) = 100 \cdot \text{makespan} + \text{Cost}$ then $h = 100 \cdot 2 + 220$ at $t_0 \leq t = 2 \leq t_\infty$

Using Relaxed Plan

- Extract a relaxed plan using $h$ as the bias
  - If the objective function is $f(time, cost)$, then action $A$ (to be added to RP) is selected such that:
    - $f(t(RP+A), Cost(RP+A)) + f(t(G_{\text{new}}), Cost(G_{\text{new}}))$
    - is minimal
    - $G_{\text{new}} = (G \cup \text{Precond}(A)) \setminus \text{Effects}$

Phased Relaxation

- The relaxed plan can be adjusted to take into account constraints that were originally ignored

Adjusting for Mutexes:
- Adjust the make-span estimate of the relaxed plan by marking actions that are mutex (and thus cannot be executed concurrently)

Adjusting for Resource Interactions:
- Estimate the number of additional resource-producing actions needed to make-up for any resource short-fall in the relaxed plan
  - $C = C + \sum_{R} \left[ \frac{(\text{Con}(R) - (\text{Init}(R) + \text{Pro}(R)))}{\Delta_R} \right] \ast C(A_R)$
Handling Cost/Makespan Tradeoffs

Results over 20 randomly generated temporal logistics problems involve moving 4 packages between different locations in 3 cities:

\[ O = f(\text{time}, \text{cost}) = \alpha \cdot \text{Makespan} + (1 - \alpha) \cdot \text{TotalCost} \]

Prottle

- SAPA-style (time-stamped states and event queues) search for fully-observable conditional plans using L-RTDP
- Optimize probability of goal satisfaction within a given, finite makespan
- Heuristic estimates probability of goal satisfaction in the plan suffix
Prottle planning graph

Time Stamped Propositions

Probabilistic Outcomes at Different Times

June 7th, 2006  ICAPS'06 Tutorial T6  113

Probability Back-Propagation

Vector of probability Costs for each top-level Goal:
\[ 1 - \text{Pr}(\text{Satisfy } g_i) \]

Outcome cost is max of affected proposition costs

Action Cost is expectation of outcome costs

Proposition cost is Product of supported Action costs

Heuristic is product of Relevant Node costs

June 7th, 2006  ICAPS'06 Tutorial T6  114
Prottle Results

<table>
<thead>
<tr>
<th>problem</th>
<th>horizon</th>
<th>ε</th>
<th>time1</th>
<th>time2</th>
<th>cost1</th>
<th>cost2</th>
<th>states1</th>
<th>states2</th>
</tr>
</thead>
<tbody>
<tr>
<td>AI</td>
<td>100</td>
<td>0.3</td>
<td>-</td>
<td>103</td>
<td>-</td>
<td>0.344</td>
<td>-</td>
<td>346,100</td>
</tr>
<tr>
<td>AI</td>
<td>120</td>
<td>0.6</td>
<td>-</td>
<td>404</td>
<td>-</td>
<td>0.222</td>
<td>-</td>
<td>1,319,229</td>
</tr>
<tr>
<td>MS</td>
<td>15</td>
<td>0.0</td>
<td>-</td>
<td>272</td>
<td>-</td>
<td>0.027</td>
<td>-</td>
<td>496,096</td>
</tr>
<tr>
<td>MS</td>
<td>15</td>
<td>0.1</td>
<td>-</td>
<td>171</td>
<td>-</td>
<td>0.114</td>
<td>-</td>
<td>309,826</td>
</tr>
<tr>
<td>MS</td>
<td>15</td>
<td>0.2</td>
<td>2,431</td>
<td>21</td>
<td>0.119</td>
<td>0.278</td>
<td>13,627,753</td>
<td>6,759</td>
</tr>
<tr>
<td>MS</td>
<td>15</td>
<td>0.3</td>
<td>367</td>
<td>235</td>
<td>0.278</td>
<td>0.278</td>
<td>1,950,134</td>
<td>434,772</td>
</tr>
<tr>
<td>MZ</td>
<td>10</td>
<td>0.0</td>
<td>195</td>
<td>10</td>
<td>0.178</td>
<td>0.178</td>
<td>1,374,541</td>
<td>13,037</td>
</tr>
<tr>
<td>MZ</td>
<td>10</td>
<td>0.1</td>
<td>185</td>
<td>2</td>
<td>0.193</td>
<td>0.178</td>
<td>1,246,159</td>
<td>2,419</td>
</tr>
<tr>
<td>MZ</td>
<td>10</td>
<td>0.2</td>
<td>64</td>
<td>1</td>
<td>0.197</td>
<td>0.193</td>
<td>436,876</td>
<td>669</td>
</tr>
<tr>
<td>MZ</td>
<td>10</td>
<td>0.3</td>
<td>62</td>
<td>2</td>
<td>0.202</td>
<td>0.193</td>
<td>414,414</td>
<td>1,812</td>
</tr>
<tr>
<td>TP</td>
<td>20</td>
<td>0.0</td>
<td>442</td>
<td>&lt; 1</td>
<td>0.798</td>
<td>0.798</td>
<td>3,565,698</td>
<td>3,676</td>
</tr>
<tr>
<td>TP</td>
<td>20</td>
<td>0.1</td>
<td>456</td>
<td>&lt; 1</td>
<td>0.798</td>
<td>0.798</td>
<td>3,628,300</td>
<td>2,055</td>
</tr>
<tr>
<td>TP</td>
<td>20</td>
<td>0.2</td>
<td>465</td>
<td>&lt; 1</td>
<td>0.798</td>
<td>0.798</td>
<td>3,672,348</td>
<td>2,068</td>
</tr>
<tr>
<td>TP</td>
<td>20</td>
<td>0.3</td>
<td>464</td>
<td>&lt; 1</td>
<td>1.000</td>
<td>0.798</td>
<td>3,626,404</td>
<td>1,256</td>
</tr>
</tbody>
</table>

PSP w/ Resources

- Utility and Cost based on the values of resources

- Challenges:
  - Need to propagate cost for resource intervals
  - Need to support resource goals at different levels
Resource Cost Propagation

- Propagate reachable values with cost

![Diagram showing cost propagation](image)

Cost Propagation on Variable Bounds

- Bound cost dependent upon
  - action cost
  - previous bound cost
    - current bound cost
    - adds to the next
    - Cost of all bounds in expressions
Results – *Modified* Rovers (numeric soil)

![Graph showing modified rovers with and without bound cost]

- Average improvement: 3.06

Anytime A* Search Behavior

![Graph showing rovers utility over time with and without bound cost]

June 7th, 2006
Results – *Modified* Logistics (#of packages)

![Graph showing the net benefit for modified logistics](image)

- **With Bound Cost**
- **Without Bound Cost**

- **Average improvement**: 2.88

---

**Cost-Based Conditional Planning**

- **Actions may reduce uncertainty, but cost a lot**
  - Do we want more “just in case” actions that are cheap, or less that are more expensive

- **Propagate Costs on the LUG (CLUG)**
  - **Problem:** LUG represents multiple explicit planning graphs and the costs can be different in each planning graph.
    - A single cost for every explicit planning assumes full positive interaction
    - Multiple costs, one for each planning graph is too costly
  - **Solution:** Propagate cost for partitions of the explicit planning graphs
Cost Propagation on the LUG (CLUG)

Relaxed Plan (bold)
Cost = c(B) + C(R) = 10 + 7 = 17

The Medical Specialist [Bryce & Kambhampati, 2005]

Average Path Cost Total Time

Using Propagated Costs Improves Plan Quality, Without much additional time
Overall Conclusions

- **Relaxed Reachability Analysis**
  - Concentrate strongly on positive interactions and independence by ignoring negative interaction
  - Estimates improve with more negative interactions
- **Heuristics can estimate and aggregate costs of goals or find relaxed plans**
- **Propagate numeric information to adjust estimates**
  - Cost, Resources, Probability, Time
- **Solving hybrid problems is hard**
  - Extra Approximations
  - Phased Relaxation
  - Adjustments/Penalties

Why do we love PG Heuristics?

- They work!
- They are “forgiving”
  - You don't like doing mutex? okay
  - You don't like growing the graph all the way? okay.
- Allow propagation of many types of information
  - Level, subgoal interaction, time, cost, world support, probability
- Support phased relaxation
  - E.g. Ignore mutexes and resources and bring them back later…
- Graph structure supports other synergistic uses
  - e.g. action selection
- Versatility…
Versatility of PG Heuristics

- **PG Variations**
  - Serial
  - Parallel
  - Temporal
  - Labelled

- **Propagation Methods**
  - Level
  - Mutex
  - Cost
  - Label

- **Planning Problems**
  - Classical
  - Resource/Temporal
  - Conformant

- **Planners**
  - Regression
  - Progression
  - Partial Order
  - Graphplan-style