Heuristic Search Planning: Progress and Challenges

Hector Geffner

Universidad Simón Bolívar
Caracas, Venezuela

Universitat Pompeu Fabra
Barcelona, Spain

9/2001

Planning

- Planning is a form of **general problem solving**

$$\textit{Problem} \longrightarrow \textit{Language} \longrightarrow \boxed{\begin{array}{c}\textit{Representation}\\ + \\ \textit{Algorithms}\end{array}} \longrightarrow \textit{Solution}$$

- **Idea:** problems **described** at high-level and **solved** automatically
- **Goal:** facilitate modeling without penalty in performance

General Problem Solving: How general?

For which **class of problems** (classical) planning should work?

For problems that map into **deterministic state models**

Deterministic State Models are models given by

- finite and discrete state space S
- an initial state $s_0 \in S$
- a set $S_G \subseteq S$ of goal states
- actions $A(s) \subseteq A$ applicable in each state $s \in S$
- a transition function $s' = f(a, s)$ for $s \in S$ and $a \in A(s)$
- action costs $c(a, s) > 0$

A **solution** is a sequence of applicable actions that map s_0 into goal state $s \in S_G$

An **optimal** solution minimizes sum of action costs

Temporal planning, planning with **sensing**, ... involve different models

Action Languages

Action languages play **two roles** in planning:

- **specification:** concise description of state models
- **computation:** reveal useful heuristic information

- **Strips** is one of the oldest, simplest, and most used planning language, even if not too flexible
- A **problem** in Strips is a tuple $\langle A, O, I, G \rangle$ where
  - A stands for set of all **atoms** (boolean vars)
  - O stands for set of all **operators** (ground actions)
  - $I \subseteq A$ stands for **initial situation**
  - $G \subseteq A$ stands for **goal situation**
- Operators $o \in O$ **represented** by three lists
  - the **Add** list $Add(o) \subseteq A$
  - the **Delete** list $Del(o) \subseteq A$
  - the **Precondition** list $Pre(o) \subseteq A$

Semantics of Strips

Strips problem $P = \langle A, O, I, G \rangle$ determines **state model** $\mathcal{S}(P)$ where

- the states $s \in S$ are **collections of atoms**
- the initial state s_0 is I
- the goal states $s \in S_G$ are such that $G \subseteq s$
- the actions in s are the $op \in O$ s.t. $Prec(op) \subseteq s$
- the state that results from doing op in s is $s' = s - Del(op) + Add(op)$
- action costs $c(op, s)$ are all 1

The (optimal) **solution** of planning problem P is the (optimal) solution of State Model $\mathcal{S}(P)$

Heuristic Search Planning

- Explicitly **searches** state space $\mathcal{S}(P)$ with an heuristic $h(s)$ that estimates cost from s to Goal
- **Key idea:** heuristic h extracted automatically from problem encoding P (McDermott 96, Bonet et al. 97)
- **Variations:** h, search algorithm, search direction, ...

Why relevant?

- Possibly most **effective** current approach for optimal and non-optimal classical (sequential) planning; did particularly well in AIPS2000 Planning Competition
- **Several planners:** HSP (Bonet & Geffner), GRT (Refanidis & Vlahavas), FF (Hoffman & Nebel), ALTALT (Kambhampati *et al.*), plus hybrids like STAN (Fox & Long), MIPS (Edelkamp & Helmert), ...
- **Graphplan**, a planner with strong influence on planning since 95, best understood as an heuristic search planner too!

Outline for the rest of the talk

- **Heuristics** for Strips planning
- **Extensions** for
  - **temporal planning:** concurrent actions with durations (relation to constraint-based approaches)
  - planning with **uncertainty** and **sensing** (feedback)
- Useful distinctions to keep in mind:
  - 'domain-independent' vs. 'knowledge rich' planning
  - optimal vs. non-optimal planning

Heuristic Functions

- Heuristics often derived as **optimal** cost function of **relaxed** problems
- In planning, some simple relaxations obtained by
  - ignoring delete lists
  - approximating 'large' atom sets by smaller subsets
  - ignoring certain atoms (making them 'don't care's)

Relaxation 1: Ignoring Delete Lists

Problem: relaxation still intractable ...

- solve relaxation suboptimally

  FF *heuristic*

- assume that atoms are independent

  additive heuristic in UNPOP, HSP

Heuristics not admissible but informative, and can be computed fast in every state

- For all **atoms** p:

$$g(p;s) \stackrel{\text{def}}{=} \begin{cases} 0 & \text{if } p \in s \\ \min_{a \in O(p)}[1 + g(Prec(a);s)] & \text{otherwise} \end{cases}$$

- For **sets** of atoms C:

$$g(C;s) \stackrel{\text{def}}{=} \sum_{r \in C} g(r;s)$$

- Additive **heuristic** h_{add}

$$h_{add}(s) \stackrel{\text{def}}{=} g(Goals;s)$$

Relaxation 2: Approximating 'large' atom sets by smaller subsets

- **Idea:** For a given $m = 1, 2, \ldots$ assume cost of achieving set of atoms A given by cost of **most costly subset of size** m (Haslum & Geffner 2000)
- For $m = 1$, heuristic h^1 given by
  - For all **atoms** p:

$$g(p; s) \stackrel{\text{def}}{=} \begin{cases} 0 & \text{if } p \in s \\ \min_{a \in O(p)} [1 + g(Prec(a); s)] & \text{otherwise} \end{cases}$$

  - For **sets** of atoms C:

$$g(C; s) \stackrel{\text{def}}{=} \max_{r \in C} g(r; s)$$

$$h^1 \stackrel{\text{def}}{=} g(Goals; s)$$

- For $m = 2$ need to keep track of **atom pairs** (or pairwise **mutexes**); resulting heuristic h^2 similar to heuristic underlying GRAPHPLAN
- Heuristics h^m **admissible** and **polynomial**, often **informative** for $m \geq 2$ but **expensive**

Relaxation 3: Ignoring Certain Atoms

- Remove certain atoms from problem ($Init$, $Goal$ and Ops), making resulting state space manageable
- Heuristic given by optimal cost of 'relaxed' problem
- More informed heuristic obtained by max-combining several such abstractions
- Closely related to idea of pattern databases in search (Culberson & Schaeffer)

– Resulting **pattern db heuristics** are **admissible** and can be **more informed** than heuristics h^m for low m (Edelkamp ECP-2001)

– **Quality** of pattern db heuristics dependent on choice of 'abstractions' (which atoms/variables to ignore)

Heuristics: Summary

- number of ways for extracting **heuristics** from representations
- state-of-the-art planners obtained by plugging some of these heuristics into standard **search algorithms**
- better results for **non-optimal planning** sometimes through non-standard search schemes (e.g. FF)

Extending the Heuristic Search Approach in Planning

- Time and concurrency
- Uncertainty & Sensing
- (Cost Structures)

Time and Concurrency; Branching

- Simplest case: **parallel planning** where all actions have same duration (Graphplan, Blackbox, . . .)
- Best approach for **optimal parallel planning** appears to be SAT and CSP; **not** Heuristic Search

Why

- **Branching factor** in parallel/temporal planning is exponential in number of applicable actions either forward or backward
- In those cases, need to **branch** differently . . .

Branching in Search and Problem Solving

- **15-puzzle,Rubik,. . . :**
  - **Branching:** Forward/Backward ok
- **TSP (Travelling Salesman Problem)**
  - **Branching:** include/exclude edge i–j in tour
  - **Termination;** when relaxed 'assignment problem' yields single tour
- **JSP (Job Shop Scheduling)**
  - **Branching:** $i \prec j$, $j \prec i$ for pairs of tasks
  - **Termination:** when no overlap among actions requiring same resource

Planning as Branch and Bound

- Forward/Backward branching ok for **sequential planning** but Partial Order (POP) and CSP branching better for **parallel/temporal** planning
- All branching schemes compatible with use of **heuristics/lower bounds**
- States σ no longer progression (regression) of plan heads (tails) but sets of **commitments**
- $h^*(\sigma)$ = cost of best plan compatible with commitments σ
- Heuristics such as h^m can be transformed to provide useful lower bounds on h^* ...
- **Yet effective branch and bound temporal planner still a challenge**

CSP & Heuristic Search closely related

For each node in Depth First traversal with bound B

Heuristic Search:

– compute lower bound $f \leq f^*$

– if $f > B$ **prune**; else ...

SAT/CSP:

– propagate constraints

– if inconsistency **prune**; else ... where

$f^* \leq B$ represented as an **explicit constraint**

Complete evaluation of pruning condition $f^* > B$ **intractable**

- **HS** approximates by evaluating $f > B$ for lower bound f
- **CSP** approximates by (incomplete) constraint prop.

Lower bounds and **constraint propagation** compatible and synergistic; proposals for combining **planning and scheduling** effectively likely to integrate both ...

Extending the Heuristic Search Approach in Planning

- Time and concurrency
- Uncertainty & Sensing
- (Cost Structures)

Planning with Uncertainty

- Simplest case: **conformant** planning
  - *initial situation partially known*
  - *non-deterministic transitions*
  - *no feedback*
- Mathematical model: **non-deterministic state model**
  - finite and discrete state space S
  - an initial **set** of states $S_0 \in S$
  - a set $S_G \subseteq S$ of goal states
  - actions $A(s) \subseteq A$ applicable in each state $s \in S$
  - a **non-deterministic** transition function $S' = F(a, s) \subseteq S$ for $s \in S$ and $a \in A(s)$
- A (conformant) **plan** must achieve goal for **any** possible initial state and transition

Conformant Planning as Search in Belief Space

Belief state b = a set of states deemed possible

Model for conformant planning reformulated as **deterministic** search problem over **belief space**:

- a finite and discrete (belief) state space B
- an **initial belief state** $b_0 = S_0 \in B$
- a set of final belief states $B_G = \{b \mid b \subseteq S_G\}$
- actions $A(b) \subseteq A$ applicable in each belief state $b \in B$ ...
- a **deterministic** transition function $f(a, b)$ on **beliefs**

$$f(a, b) = b' = \{s' \mid s' \in F(a, s) \& s \in b\}$$

- Same techniques apply; need good **heuristic function** $h(b)$ **over beliefs**
- Any admissible $h(s)$ over states can be **lifted** to beliefs

$$h(b) = \max_{s \in b} h(s)$$

- Similar ideas for **probabilistic planning and sensing** (**MDPs, POMDPs**)
- See our **GPT** planner and **Model Checking** planners by Cimatti *et al.*

Summary

- **Planning as Heuristic Search** formulation has produced powerful systems for **sequential planning**
  - **heuristics** are key
- For **parallel and temporal planning**
  - **branching** as important as lower bounds
- For **planning with incomplete information**
  - search is in **belief space**
- **Future:** convergence of a number of threads concerned with **modeling and problem solving** (KR, Search, Planning and Problem Solving, Constraint Programming, Uncertainty, Comb. Optimization, . . .)