

Cost Based Search Considered Harmful (Extended Abstract)

William Cushing and J. Benton and Subbarao Kambhampati

Department of Computer Science and Engineering
Arizona State University
Tempe, AZ 85281

Abstract

Planning research has returned to the issue of optimizing *costs* (rather than *sizes*) of plans. A prevalent perception, at least among non-experts in search, is that graph search for optimizing the size of paths generalizes more or less trivially to optimizing the cost of paths. While this kind of generalization is usually straightforward for graph *theorems*, graph algorithms are a different story. In particular, implementing a search evaluation function by substituting cost for size is a Bad Idea. Though experts have stated as much, cutting-edge practitioners are still learning of the consequences the hard way; here we mount a forceful indictment on the inherent dangers of cost-based search.¹

Introduction

Planning, and combinatorial search in general, is being wedged ever tighter into the proverbial “rock and a hard place”. On the one hand, satisficing approaches return arbitrarily poor solutions in theory, and (*sometimes*) practice (Openstacks, Satellite); on the other, optimal approaches ‘guarantee non-termination’. Recent results from Helmert and Röger tighten the ‘non-termination guarantee’ of optimal planning considerably: even given $\max(h^* - e, 0)$ for free, optimal planners will take exponential time both in theory and on *standard benchmarks*. In contrast, enforced hill climbing is both polynomial ($O(b^e)$) and optimal given such a powerful heuristic (as is best-first search).

Many tackle this dilemma head on: attempting to develop planners with guarantees on solution quality, terminating in both theory and practice. Instead we believe it is better to separate the problems of *discovering* good solutions and *proving* bounds about them. Standard formalizations of these problems are both PSPACE-complete, so it would seem there is little to gain. However, an important difference arises when considering cost instead of size.

(Proof by example) Suppose a (cost-)optimal and second-best solution to a problem exist on 10 and 1000 unspecified actions. *The optimal solution may be the larger one.* How long should it take just to find the 10 action plan? How

long should it take to prove (or disprove) its optimality? In general (presuming PSPACE \neq P):

1. Discovery should require time exponential in 10.
2. Proof should require time exponential in, at least, 1000.

That is, in principle the only way to (domain-independently) prove that the 10 action plan is better than the 1000 action one is to in fact go and discover the 1000 action plan. That the problem is PSPACE-complete then gives the lower bound by assumption. (A full proof of optimality might have to find yet larger near-optimal solutions.)

For discovery alone, size-based (branch-and-bound) search demonstrates the upper bound. Moreover (we argue), it sets a reasonable baseline in general — against which cost-based fails to measure up. By “size-based (branch-and-bound)” we mean: The search proceeds as if all costs were 1 (“size-based”), but also tracks real costs in order to prune nodes worse than the currently best known solution (“branch-and-bound”). (One can incorporate admissible heuristics into the pruning.) Then discovery is the printing of solutions, and proof of optimality is termination of the search. So by “branch-and-bound” we mean something like “anytime optimal” or Zilberstein’s *interruptible*, but more specific.

In the following we present 3 critiques of cost-based (branch-and-bound) search. First, it is (easily) possible to construct ‘traps’ to make it perform extremely poorly. Second, we argue that of all possible topological surfaces (an evaluation function) to choose for search, cost-based is the worst. Lastly, we carry out some tests of our own with real planners, in order to confirm that such traps arise in practice. These tests are deliberately quite limited: many others have already encountered these pernicious effects of cost-based search in their own (comprehensive) experimental studies.

Yet it rarely seems to merit more than a plaintive mention. The community seems resigned to the view that these difficulties are just the cost of doing business in the area (pun intended). We want to question that resignation, and explicitly take the position that cost-based search is harmful.

Trapping Cost-Based Search

Very low cost actions are a common feature of real world planning problems: boarding versus flying (ZenoTravel), mode-switching versus machine operation (Job-Shop), labor

Copyright © 2010, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

¹Full version: <http://rakaposhi.eas.asu.edu/socs-full.pdf>

This research is supported in part by ONR grants N00014-09-1-0017 and N00014-07-1-1049, and the NSF grant IIS-0905672.

versus (precious) material cost, and so forth.² In addition, planning formalisms are often stretched to encompass additional complex concepts via compilation. So, we expect problems to contain actions with very small (or very large) costs. If we normalize all costs to $[0, 1]$ by dividing through by $\max_a \text{cost}(a)$, then $\varepsilon = \frac{\min_a \text{cost}(a)}{\max_a \text{cost}(a)}$ is the cost of the least cost action(s) after normalization.

ε -cost Trap:³ Consider the problem of making some counter on k bits contain one less than its maximum value ($2^k - 2$), starting from 0, using only the operations of increment and decrement. There are 2 minimal solutions: incrementing $2^k - 2$ times, or decrementing twice (exploiting overflow). Set the cost of incrementing and decrementing to 1, except that overflow (in either direction) costs, say, 2^{k-1} . Then the 2 minimal solutions cost $2^k - 2$ and $2^{k-1} + 1$, or, normalized, $2(1 - \varepsilon)$ and $1 + \varepsilon$.

Cost-based search is the clear loser on this problem. To analyze such problems, assume $h = 0$ (or complicate the problem until it appears as given despite the heuristic). While both approaches prove optimality in exponential time ($O(2^k)$), size-based discovers the optimal plan in constant time. Of course $2^k - 2$ is chosen to best illustrate the trap. So consider the discovery problem for other goals: from $2^k[0, \frac{1}{2}]$ cost-based search is twice as fast, from $2^k[\frac{1}{2}, \frac{2}{3}]$ the performance gap narrows to break-even, and from $2^k[\frac{2}{3}, 1]$ the size-based approach takes the lead — *by an enormous margin*. Note that between $2^k[\frac{2}{3}, \frac{3}{4}]$ there is a tradeoff: size-based finds a solution before cost-based, but cost-based finds the optimal solution first.

Then, even across all goals, cost-based search is still quite inferior: the margins of victory either way are extremely lopsided. To illustrate, consider ‘large’ k , say, $k = 1000$. Even the most patient reader will have forcibly terminated either search *long* before receiving any useful output — except if the goal is of the form $0 \pm f(k)$ for some sub-exponential $f(k)$. Both approaches discover and prove the optimal solution in the positive case in time $O(f(x))$. In the negative case, only the size-based approach manages to discover a solution before being killed (and does so in time $O(f(k))$). Moreover, while *it* will fail to produce a proof before death, *we*, based on superior understanding of the domain, can show it to be *posthumously* correct (and already have: $2^k - f(k) + 1 > 2^k \frac{3}{4}$ for large k).

Search Topology

We view evaluation functions (f), as topological surfaces over states, so that states are expanded in order of f -altitude.⁴ Fix some admissible estimate of cost-to-go, $h_c()$ (and let $f_c(s) = g_c(s) + h_c(s)$). Consider the minimum altitude of a goal — all states with lower altitude comprise the *cost-optimal footprint*. Enumerating all such states *is* a

²Even *negative* weights arise naturally: selling versus buying.

³This problem may appear contrived, even pathological, as it breaks the semantics of a counter. Consider that the space is just a simple cycle with one expensive edge — a model of, for example, *Bait-and-Switch* or paying in installments rather than up-front.

⁴With inconsistent heuristics, it is more accurate to think in terms of a slow flood of the surface.

(minimum) proof of optimality, relative to using (only) $h_c()$. Observe that every branch-and-bound search is equivalent if the optimal solution is known: all that remains is to exhaust the cost-optimal footprint. *This can be done in any order*. By using $f_c(s)$ to prune, one is free to use any other f to direct search.

Given this, admissible cost-based topology is the worst possible choice: such a search cannot be usefully interrupted since its first solution is also its final solution. In contrast the size-based approach will take guesses at the optimal solution long before terminating; indeed, after discovering the optimal solution the numerous *longer* but *cheaper* plans still need to be investigated. So size-based branch-and-bound search is *interruptible* — and asymptotically best possible assuming PSPACE \neq P. That is, the approach is a reasonable *baseline*, but there remains lots of room for improvement. (As a first step, let the size-based heuristic estimate the size of the *best*, rather than smallest, continuation.)

Plateaus in g are important; though not due to implying search plateaus (an inconsistent h could create a very bumpy f -surface). Instead, the minimum gradient in g imposes an upper bound on search effort: the number of states that could conceivably fit beneath a given f -altitude. For uniformly branching trees this upper bound is $b^{f(s) \min \nabla g}$. So, for size-based f , $O(b^d)$ bounds the discovery problem. For cost-based, we instead arrive at $O(b^{d\varepsilon^{-1}})$ in the worst case. Tighter bounds (and more appropriate models than uniform trees) can be formulated, but the basic idea is that cost-based topology (admissible or otherwise) allows search to wander a factor of $\frac{1}{\varepsilon}$ deeper into the space — much too deep.⁵

Practice

Actual planners are much more complicated than A^* with some heuristic. Conceivably one of the many other techniques being employed interacts better with a cost-based (rather than size-based) evaluation function. We tested variants of SapaReplan and LAMA against a limited set of simple ZenoTravel tasks — many other (comprehensive) studies are available, e.g., the IPC results.

Mode	2 Cities		3 Cities	
	Score	Rank	Score	Rank
Hybrid	88.8%	1	43.1%	2
Size	83.4%	2	43.7%	1
Cost	77.8%	3	33.3%	3

Table 1: IPC metric on SapaReplan variants in ZenoTravel.

Results: Size-based evaluation functions find better plans faster than Cost-based. Hybrid evaluation functions (size + *normalized* cost) also do relatively well.

In conclusion, cost-based branch-and-bound search digs its own ($\frac{1}{\varepsilon}$ deep) grave. The size-based approach can be taken as a baseline; albeit, *balancing* exploration (size) with exploitation (cost) is far more attractive.

⁵In the full version we elaborate on “why” search becomes trapped. Key concepts are *fairness* and *exploitation vs exploration*.