Proceedings of the Fifteenth International Symposium on Combinatorial Search (SoCS 2022)

A Jeep Crossing a Desert of Unknown Width (Extended Abstract)*

Richard E. Korf
Computer Science Department
University of California, Los Angeles
Los Angeles, CA 90095
korf@cs.ucla.edu

Abstract

The classic jeep problem concerns crossing a desert wider than the range of the jeep, with the aid of preplaced fuel caches. There has been a lot of work on this problem and its variations, and the optimal strategy is well known, but all previous work assumes that we know the width of the desert. We consider the case where we don't know the distance in advance. We evaluate a strategy by its competitive ratio, which is the worst-case ratio of the cost of the strategy, divided by the cost of an optimal solution had we known the distance in advance. We show that no strategy with a fixed sequence of caches can achieve a finite competitive ratio. The optimal strategy is an iterative one that uses the optimal known-distance strategy to reach a sequence of target distances, emptying all caches between iterations. One optimal strategy doubles the cost of each successive iteration, and achieves a competitive ratio of four. The full paper was published in the American Mathematical Monthly.

Introduction

We have a jeep with a given fuel capacity, and a range it can travel on a full load of fuel. This includes fuel in the tank plus fuel cans in the jeep. Without loss of generality, we assume that the jeep can carry one gallon of fuel, and can travel one mile per gallon. We have unlimited fuel at the start, and can cache unlimited fuel along the way. We want to minimize the total fuel cost to travel a given distance.

In the one-way version of the problem, we just have to cross the desert, while in the two-way version we have to return to the start as well. We adopt the two-way version, since for the unknown-distance case we always have to be able to return to the start. It's much easier to determine the maximum distance we can go with a given integer amount of fuel, than to determine the amount of fuel needed to go a given integer distance.

Previous Work: The Known Distance Case

The one-way problem was first solved in (Fine 1947), and the round-trip problem in (Phipps 1947), both in 1947. With one gallon of fuel, we can go $1/2$ mile and return. With two gallons, we go $1/4$ mile, cache $1/2$ gallon, and return. Then we go $1/4$ mile, pick up $1/4$ gallon, go $1/2$ mile further and back, pick up the remaining $1/4$ gallon, and return to the start. With three gallons, we can go $1/6 + 1/4 + 1/2 \approx .91667$ miles, with caches at $1/6$ and $1/6 + 1/4$ miles from the start. In general, with n gallons, we can go $1/2n + 1/2(n-1) + \cdots + 1/4 + 1/2$ miles, with $n-1$ caches, spaced at these distances from each other, except for the last $1/2$ mile. We never carry back more fuel than needed to reach the previous cache or start, and every forward trip starts with a full gallon of fuel. The amount of fuel needed to go d miles and return is slightly less than $e^{2(d-1)} \approx 7.389^{d-1}$ gallons.

The Unknown Distance Case

To evaluate a strategy for the unknown distance case, we use its competitive ratio. The competitive ratio of an algorithm for this problem is the cost of reaching a given distance using the algorithm, divided by the cost of an optimal solution for that distance, had we known the distance in advance. The worst-case competitive ratio is the maximum of this ratio over all possible distances.

Using a Fixed Sequence of Caches

The simplest strategy stocks a fixed sequence of caches, until we reach the goal. An optimal strategy never carries back more fuel than necessary to reach the previous cache. If there is enough fuel at the current cache to go forward with a full load, we do so, and otherwise we go back to the start. We show that for a given sequence of caches, this is an optimal way to schedule the hops.

With caches $1/3$ of a mile apart, the cost of going d miles and returning is $O(27^d)$ gallons. With caches $1/4$ of a mile apart, this decreases to $O(16^d)$ gallons, and for caches $1/5$ of a mile apart it decreases to $O(12.86^d)$. In general, for evenly spaced caches $1/k$ miles apart, the asymptotic cost of reaching a given distance grows exponentially with the distance, and the base of the exponent is $\left(\frac{k}{k-2}\right)^k$. This is always greater than the e^2 base for the optimal known-distance strategy, and only approaches e^2 in the limit as k goes to infinity. This would require an infinite number of caches in any finite distance, and an infinite amount of fuel. Since the exponential growth rate of these strategies is greater than for

*This is an extended abstract version of (Korf 2022).

the optimal known-distance strategy, their competitive ratios are unbounded as the goal distance increases.

For the unknown-distance case, given three adjacent caches, the most efficient placement of the intermediate cache is halfway between the other two. Thus, for any fixed sequence of caches for the unknown-distance case, spacing them evenly apart is the most efficient placement. When combined with the result for evenly-spaced caches, this implies that any fixed sequence of caches results in an unbounded competitive ratio.

This result was very surprising. When I began this work, I was sure there was an optimal strategy with a fixed sequence of caches, and set out to find it.

Iterative Strategies

If the sequence of caches is not fixed, then the only other option is an iterative strategy that chooses a sequence of target distances, uses the optimal strategy for each target distance, and empties all the caches between iterations. We want a sequence of target distances that minimizes the worst-case competitive ratio.

Computing the Optimal Competitive Ratio

The competitive ratio of a strategy for the unknown-distance case is the total cost to find the goal divided by the optimal cost to find it, had we known its distance in advance. The goal is reached in the last iteration, but where it is found in that iteration has little effect on the cost of the iteration. The worst-case competitive ratio occurs when the goal is found just past the range of the penultimate iteration. Thus we compute the competitive ratio as the total cost of all iterations, including the last one, divided by the cost of the penultimate iteration. The worst-case competitive ratio is the maximum value of this ratio over all possible goal positions.

If we assume that the cost of each successive iteration is a constant multiple m of the cost of the previous iteration, then the optimal value of m is easy to compute. If the goal is found on the ith iteration, then the cost of the penultimate iteration is m^{i-1}, the total cost is $1+m+m^2+\cdots m^i$, and the competitive ratio is

$$\frac{1+m+m^2+\cdots+m^i}{m^{i-1}} = \frac{1}{m^{i-1}} + \frac{m}{m^{i-1}} + \cdots + 1 + m$$
$$= 1 + m + \frac{1}{m} + \frac{1}{m^2} + \cdots + \frac{1}{m^{i-1}}$$

The worst-case competitive ratio is the infinite sum

$$1 + m + \frac{1}{m} + \frac{1}{m^2}+ \cdots = \frac{m^2}{m-1}$$

To find the value of m that minimizes this expression, we take its derivative:

$$\frac{d}{dm}\frac{m^2}{m-1} = \frac{m^2-2m}{(m-1)^2}$$

Setting this to zero and solving for m yields $m = 2$, which doubles the cost of each successive iteration, and results in a competitive ratio of 4. Here we assumed a constant multiplier for simplicity, but the full paper shows that no sequence of multipliers can achieve a competitive ratio less than 4.

Cost Bounds for Iterative Deepening

This result is applicable to the problem of choosing cost bounds for depth-first search algorithms, such as Iterative-Deepening-A* (IDA*) (Korf 1985). In problems with many unique cost bounds, IDA* can perform poorly due to only a small number of new nodes being expanded in each iteration. A number of algorithms have been designed to address this issue by using larger increments for successive cost bounds, starting with IDA_CR (Sarkar et al. 1991), and including Iterative Budgeted Exponential Search(Helmert et al. 2019) most recently. A common strategy is to choose cost bounds to try to double the cost of each successive iteration. Our analysis of the optimal strategy for the jeep problem provides a theoretical basis for this strategy.

Assume that every node has a unique cost, and that we can predict how many nodes will be expanded for each cost bound. Then we can choose cost bounds that double the size of each successive iteration. In fact, we can more than double the size of each iteration without exceeding a competitive ratio of 4. The first iteration expands 1 node. The second can expand 7 nodes, because in the worst case the goal could be found with just 2 node expansions, the total cost can't exceed $1 + 7 = 8$, and $8/2 = 4$. The third iteration can expand 24 nodes, because the total cost can't exceed $1 + 7 + 24 = 32$, the goal could be found after 8 node expansions and $32/8 = 4$. In general, if x_i is the number of nodes expanded by the ith iteration, we can compute x_n as $4(x_{n-1}+1) - \sum_{i=1}^{n-1} x_i$. If we look at the ratios of the number of nodes expanded in a given iteration divided by the number of nodes expanded in the previous iteration, we get a sequence of multipliers that approach 2 from above in the limit of an infinite number of iterations. The same thing happens with the target distances for the jeep problem, but it is easier to explain in this discrete context.

References

Fine, N. 1947. The Jeep Problem. *American Mathematical Monthly*, 54(1): 24–31.

Helmert, M.; Lattimore, T.; Lelis, L. H. S.; Orseau, L.; and Sturtevant, N. R. 2019. Iterative Budgeted Exponential Search. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI-19)*, 1249–1257. Macau.

Korf, R. E. 1985. Depth-first iterative-deepening: An optimal admissible tree search. *Artificial Intelligence*, 27(1).

Korf, R. E. 2022. A jeep crossing a desert of unknown width. *American Mathematical Monthly*, 129(5): 435–444.

Phipps, C. 1947. The jeep problem: A more general solution. *American Mathematical Monthly*, 54(8): 458–462.

Sarkar, U.; Chakrabarti, P.; Ghose, S.; and DeSarkar, S. 1991. Reducing reexpansions in iterative-deepening search by controlling cutoff bounds. *Artificial Intelligence*, 50(2): 207–221.