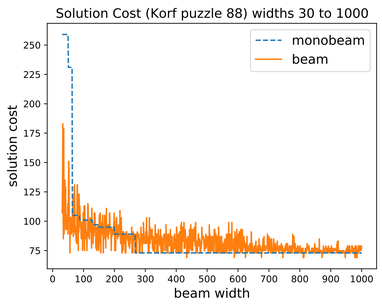
Beam Search: Faster & Monotonic

Sofia Lemons,[1, 2] Carlos Linares López,[3] Robert C. Holte,[4] Wheeler Ruml[1]

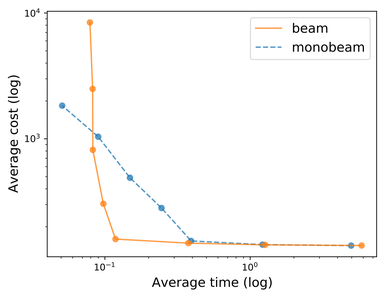
[1] University of New Hampshire, [2] Earlham College, [3] Universidad Carlos III de Madrid,
[4] University of Alberta, Alberta Machine Intelligence Institute (Amii)

Non-monotonicity

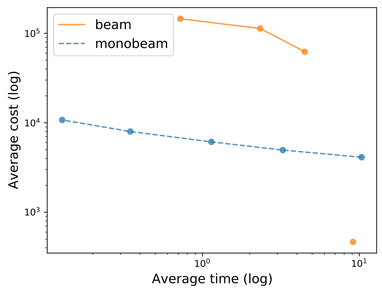


When increasing beam width, beam search sometimes returns worse solutions.

Trouble in Non-Unit Cost



15-puzzle (sqrt cost)

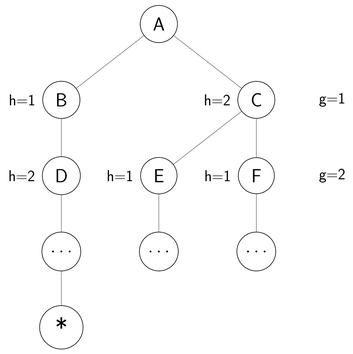


15-puzzle (reverse cost)

In non-unit cost, beam and monobeam often find poor solutions or fail to solve.

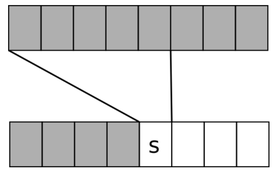
Root Cause: Cuckoo Nodes

The children of nodes generated at the larger beam width may have incorrectly low cost-to-go estimates due to heuristic error. We refer to these as *cuckoo nodes* because they cause truly better nodes to be pushed off the beam, just as, when a cuckoo bird lays its eggs in another bird's nest, the cuckoo chicks push the other bird's eggs out of the nest. In the example, a beam search with width 2 returns a solution of lower quality than a beam search with width 1. When the width of the beam is increased, node C serves as a cuckoo node, with its successors pushing the successors of B off the beam.



Monobeam

In order to avoid cuckoo nodes as the beam is widened, **monobeam** considers each specific position in the beam, or *slot*, sequentially. When selecting a node to fill slot s on the beam, considers only successors of the nodes currently in slots 1 through s. This preserves the search order in earlier slots, regardless of increases in beam width.



Monobeam with width w returns equal or better solution than any width lower than w.

Searching on Distance-to-go

For best-first search, searching using distance-to-go d instead of cost-to-go h is known to yield faster search in non-unit cost domains. It is natural to ask whether guiding beam search using d might help. We introduce **bead** search, which is beam search using a purely distance-based measurement $l(n) = depth(n) + d(n)$ (estimated length of solution.) We also introduce **monobead** search, which is monotonic beam search, except using $l(n)$ to select nodes for the beam.

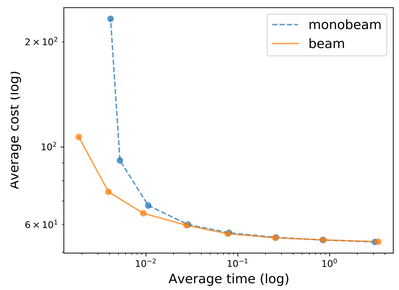
Conclusions

Monobeam guarantees solution cost monotonic in beam width.

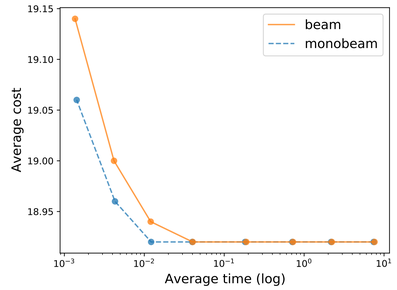
The guarantee of monotonicity sometimes comes at the expense of slightly worse average solution cost, but sometimes better.

Using distance-to-go instead of cost-to-go provided significant improvement in beam searches.

Results (Monobeam)

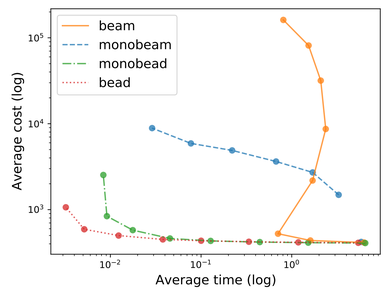


15-puzzle (unit cost)

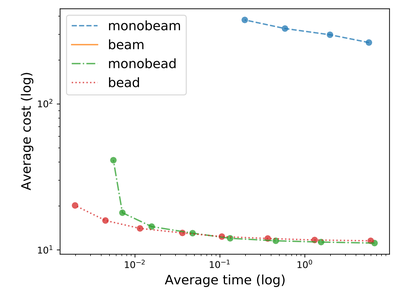


20-pancake problem (unit cost)

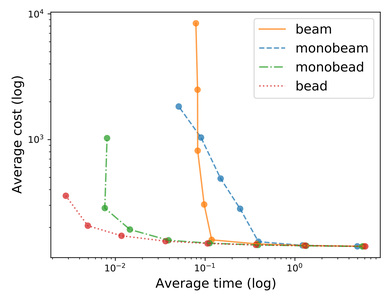
Results (Bead & Monobead)



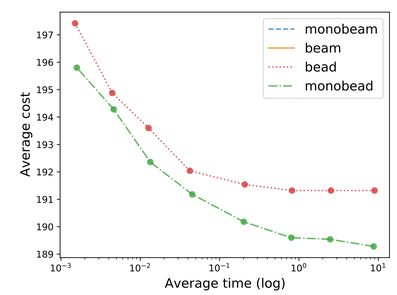
15-puzzle (heavy cost)



15-puzzle (inverse cost)



15-puzzle (sqrt cost)



20-pancake problem (heavy cost)