Subset Selection of Search Heuristics

Chris Rayner[1] Nathan Sturtevant[2] and Michael Bowling[1]

Summary Overview

A **search heuristic** $h(i,j)$ estimates the distance from state i to j.
A **set** of heuristics can be **combined** by maximizing over each:

$$h(i,j) = \max\{h_1(i,j), h_2(i,j), \ldots\}$$

We ask:
Given a **fixed budget** d and a set of **candidate heuristics** C,
what is the best subset of C?

1. Under modest assumptions, we prove that **greedy selection** of highest contributing heuristics is in fact **near-optimal**
2. Provable optimality is retained even when using **sampling**
3. Our strategy outperforms existing methods and leads to new insights into choosing heuristics for **directed domains**

Problem Definition

Define the heuristic lookup as combining
default and **user-defined** heuristics:

$$h^H(i,j) = \max_{h_x \in H \cup D} h_x(i,j)$$

D is a set of "default" pre-defined heuristics (e.g., the zero heuristic)
$H \subseteq C$ is a set of user-defined heuristics (to be defined)
C is a (large) set of candidate heuristics

Goal is to find an **optimal subset** H that minimizes the **loss** between the true distances $\delta(i,j)$ and the heuristics $h^H(i,j)$ across all i,j:

$$\underset{H}{\textbf{minimize}} \quad \sum_{i=1}^{n}\sum_{j=1}^{n} W_{ij}|\delta(i,j) - h^H(i,j)|$$

$$\textbf{subject to} \quad |H| = d$$

(where user-defined W specifies importance of every pair of states)

NP-hard, but solutions can be approximated

Acknowledgments

NSERC of Canada
Alberta Innovates Technology Futures
IJCAI 2013's Anonymous Reviewers

Solutions

Assuming admissibility of all $h \in C$,
negating the loss yields a simpler **utility** function $\mathcal{U}$:

$$-\sum_{i,j} W_{ij}|\delta(i,j) - h^H(i,j)| = -\sum_{i,j} W_{ij}\,\delta(i,j) + \sum_{i,j} W_{ij} h^H(i,j)$$

$$\mathcal{U}(H) \equiv \sum_{i,j} W_{ij}\, h^H(i,j)$$

$\mathcal{U}$ is **monotonic** (adding to H can't hurt)

$\mathcal{U}$ is **submodular** (falling marginal returns)

Theorem: Initialize $H_0 = \emptyset$. Incrementally add heuristics by **greedy selection** from a set of admissible heuristics C:

$$H_t \leftarrow H_{t-1} \cup \left\{ \arg\max_{h \in C} \mathcal{U}(H_{t-1} \cup \{h\}) \right\}$$

$\mathcal{U}(H_d)$ is within 0.63 of optimal

Assuming consistency of all $h \in C$,
it is also safe to use sampling to approximate $\mathcal{U}(H)$:

Divide the state space into mutually exclusive and collectively exhaustive regions: $Z_1, \ldots, Z_m$ each with an associated representative: $z_i \in Z_i$

This yields a **sample utility** function $\overline{\mathcal{U}}$:

$$\overline{\mathcal{U}}(H) \equiv \sum_{p=1}^{m}\sum_{q=1}^{m} \overline{W}_{pq}\, h^H(z_p, z_q)$$

$$\text{where } \overline{W}_{pq} = \sum_{r \in Z_p}\sum_{s \in Z_q} W_{rs}$$

Theorem: Initialize $H_0 = \emptyset$. Incrementally add heuristics by **greedy selection** from a set of consistent heuristics C:

$$H_t \leftarrow H_{t-1} \cup \left\{ \arg\max_{h \in C} \overline{\mathcal{U}}(H_{t-1} \cup \{h\}) \right\}$$

$\mathcal{U}(H_d)$ is within $0.63\,\frac{\overline{\mathcal{U}}(H_d) - \epsilon}{\overline{\mathcal{U}}(H_d) + \epsilon - \epsilon/e}$ of optimal

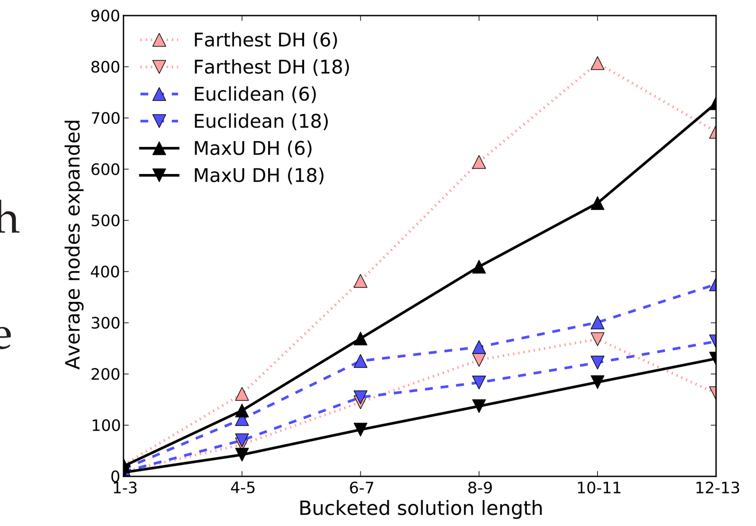
(where ϵ is an easily computable constant)

Example Applications

Comparison against existing methods are favorable:

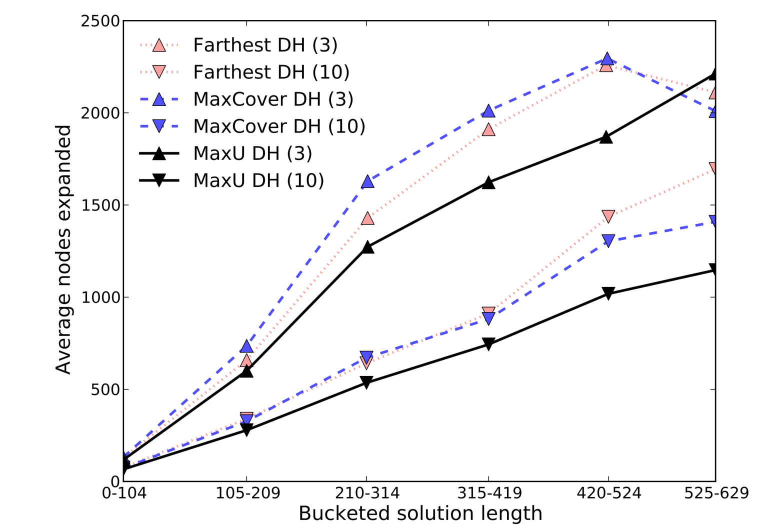
Word Graph

- Undirected graph of 4,820 four-letter words
- Agent can change 1 letter at a time
- *Farthest* struggles due to graph's intrinsic high dimensionality [Rayner *et al.* 2011]
- Multi-dimensional (*Euclidean*) heuristics more effective given 6 and 8 dimensions
- Greedy utility maximization (*MaxU*) shows DHs *can* in fact succeed, when budget is 18

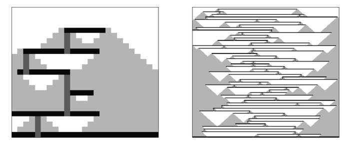


Game Maps

- Standard pathfinding benchmarks (http://www.movingai.com/)
- Undirected graphs with 168–18,890 states
- Octile connectivity; diagonal moves cost 1.5
- *Farthest* and *MaxCover* are effective because maps are like corridors, i.e. one-dimensional
- *MaxU* still significantly outperforms these methods – even when sample utility is used

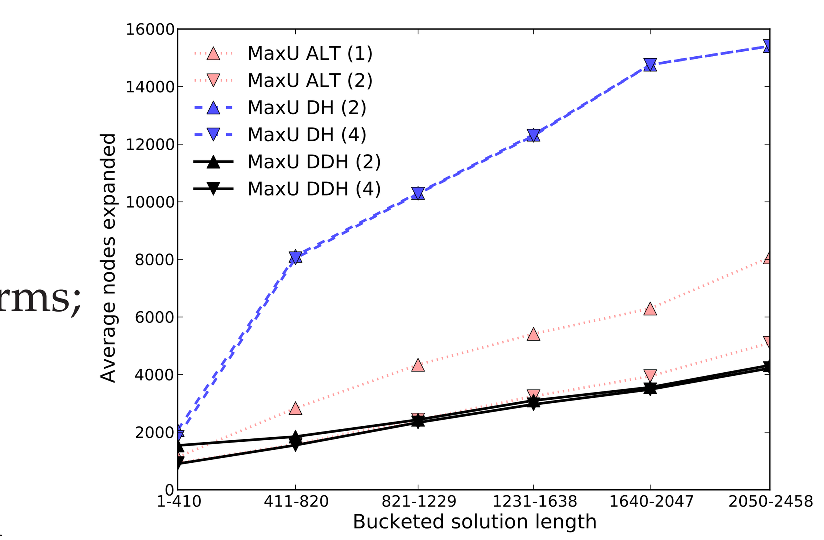


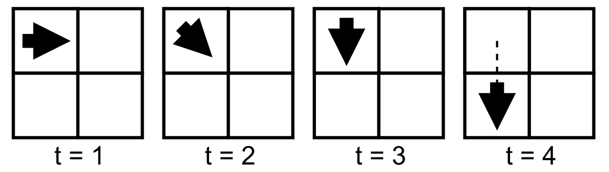
We introduce **directed differential heuristics** (DDHs), a simple (but untested!) type of heuristic for directed domains. Greedy utility maximization provides a fair and principled way to evaluate them against alternatives:



Platformer

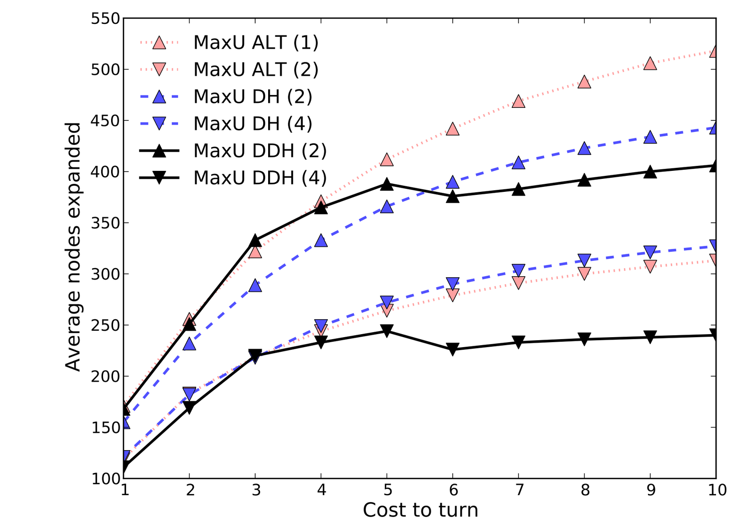
- Directed graph with 16,248 states
- Agent can fall; move left or right on platforms; or up and down onto ladders
- The space of DDHs encapsulates and is exponentially larger than the space of ALT heuristics, but greedy utility maximization only experiences a doubling in effort





Oriented Game Map

- Directed graph with 7,342 states
- Agent can advance along its current heading (cost 1) or turn left/right (at variable cost)
- By varying the cost to turn, a changing preference for each heuristic is revealed
- DDH performance suggests future application



1. Department of Computing Science University of Alberta

2. Computer Science Department University of Denver