



CENTRO DE CIENCIAS EXATAS E TECNOLOGICAS - CCE
DEPARTAMENTO DE INFORMATICA

THESIS PROJECT:

**ON SELECTING HEURISTIC FUNCTIONS FOR
DOMAIN-INDEPENDENT PLANNING.**

A Thesis Project submitted by Marvin Abisrro for the degree of Master to the PPG

Supervised by:
Levi Enrique Santana de Lelis, Santiago Franco

0.1 Project

We present a greedy method based on the theory of supermodular optimization for selecting a subset of heuristic functions, without relying on domain knowledge, to guide the A* search algorithm. If the heuristics are consistent, our method selects a subset which is guaranteed to be near optimal to the resulting A* search tree size. Furthermore to being consistent, if all heuristics have the same evaluation time, we expect the subset would be near-optimal with respect to the resulting A* search tree size.

0.2 Introduction

This theses project is concerned with cost-optimal state-space planning using the A* algorithm [Hart and Raphael, 1968]. We assume that a pool, ζ , of hundreds or even thousands of heuristics is available, and that the final heuristic used to guide A*, h_{max} , will be defined as the maximum over a subset ζ' of those heuristics ($h_{max(s, \zeta')} = \max_{h \in \zeta'} h(s)$). The choice of the subset ζ' can greatly affect the efficiency of A*. For a given size N and planning task ∇ , a subset containing N heuristics from ζ is optimal if no other subset containing N heuristics from ζ results in A* expanding fewer nodes when solving ∇ .

0.3 Contributions

The first contribution of this research is to show that the problem of finding the optimal subset of ζ of size N for a given problem task is supermodular, and therefore a greedy algorithm, which we call Greedy Heuristic Selection (GHS), that selects heuristics from ζ one at a time is guaranteed to produce a subset ζ' such that the number of nodes expanded by A* is no more than approximately 1.36 times optimal. Moreover, if all heuristics in ζ have the same evaluation time, then the A* running time using ζ' is also guaranteed to be no more than approximately 1.36 times optimal. An optimization procedure which is similar to ours is presented by [Rayner et al., 2013], but their procedure maximizes the average heuristic value. By contrast, GHS minimizes the search tree size.

GHS requires a prediction of the number of nodes expanded by A* using any given subset. Although there are methods for accurately predicting the number of nodes expanded by Iterative Deepening-A* [Korf et al., 2001] (IDA*), Stratified Sampling [Lelis et al., 2013] (SS), these methods can't be easily adapted to A* because A*'s duplicate pruning makes it very difficult to predict how many nodes will occur at depth d of A*'s search tree (the tree of nodes expanded by A*). In this research we want to make another contribution showing that the Stratification Sampling(SS) generates very good predictions for the A*'s search tree.

The second contribution of this research is an empirical evaluation of GHS in optimal domain-independent planning problems. Namely, we implemented and evaluated GHS on the planning problems from the 2011 International Planning Competition (IPC).

The main aim of this research is to develop an approach to resolves problems quickly. The approach we are proposing can be applied for solving optimal domain problems in the Fast-Downward Planner [Helmert, 2006]. A few of these domains are: Elevators opt08 and opt11, Floortile opt11, Nomystery opt11, etc.

0.4 Domain Independent Problems.

The Domain Independent Problems are well know as general problem solving. In order to avoid cycles and dead ends during the search we have modified the Stratification Sampling (SS) to solve this issues using BFS each time the cost operator being zero.

We categorize these domains in the following:

0.4.1 Unit cost Domains

In these domains all the cost operators have the value of one. For example:

1. nomystery-opt11-strips
2. openstacks-opt08-strips
3. openstacks-opt11-strips
4. parking-opt11-strips
5. pegsol-opt11-strips
6. tidybot-opt11-strips
7. visitall-opt11-strips

0.4.2 Non Unit cost Domains

In these domains all the cost operators have the value greather than one. For example:

1. barman-opt11-strips
2. floortile-opt11-strips
3. scanalyzer-opt11-strips
4. transport-opt08-strips
5. transport-opt11-strips
6. woodworking-opt08-strips
7. woodworking-opt11-strips

0.4.3 Zero cost Domains

In these domains all or some of the cost operators have the value equal to zero. For example:

1. elevators-opt08-strips
2. elevators-opt11-strips
3. parcprinter-opt11-strips
4. sokoban-opt08-strips
5. sokoban-opt11-strips

0.5 Problem Formulation

When solving ∇ using the consistent function $h_{\max}(\zeta')$, for $\zeta' \subseteq \zeta$, A^* expands in the worst case $J(\zeta', \nabla)$ nodes, where:

$$J(\zeta', \nabla) = |\{s \in V \mid f_{\max}(s, \zeta') \leq C^*\}| \quad (1)$$

$$J(\zeta', \nabla) = |\{s \in V \mid h_{\max}(s, \zeta') \leq C^* - g(s)\}| \quad (2)$$

We write $J(\zeta')$ instead of $J(\zeta', \nabla)$ whenever ∇ is clear from context.

We present a greedy algorithm for approximately solving the following optimization problem.

$$\min_{\zeta' \subseteq \zeta} J(\zeta', \nabla) \quad (3)$$

$$\text{subject to } |\zeta'| = N \quad (4)$$

where N could be determined by a hard constraint such as the maximum number of PDBs one can store in memory.

0.6 Greedy Heuristic Selection

Greedy Heuristic Selection (GHS), is an approximation algorithm for selecting a subset $\zeta' \subseteq \zeta$. The algorithm receives as input a planning problem ∇ , a set of heuristics ζ a cardinality size N , and it returns a subset $\zeta' \subseteq \zeta$ of size N . In each iteration GHS greedily selects from ζ the heuristic h which will result in the largest reduction of the value of J . GHS returns ζ' once it has the desired cardinality size N .

Input: problem ∇ , set heuristics ζ , cardinality N .

Output: heuristic subset $\zeta' \subseteq \zeta$ of size N

$\zeta' \leftarrow \emptyset$;

while $|\zeta'| < N$ **do**

$h \leftarrow \arg \min_{h \in \zeta} J(\zeta' \cup \{h\}, \nabla)$

$\zeta' \leftarrow \zeta' \cup \{h\}$

end

return ζ'

Algorithm 1: Greedy Heuristic Selection

0.6.1 GHS Approximation Analysis

In the following analysis all heuristic functions are assumed to be consistent. We also assume that A^* expands all nodes n with $f(n) \leq C^*$ while solving ∇ , as shown in Equation 1.

Bibliography

- N. J.; Hart, P. E.; Nilsson and B. Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics SSC*, 4(2):100–107, 1968.
- Chris Rayner, Nathan Sturtevant, and Michael Bowling. Subset selection of search heuristics. *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence*, pages 637–643, 2013.
- Richard E Korf, Michael Reid, and Stefan Edelkamp. Time complexity of iterative-deepening-a*. *Artificial Intelligence*, 129(1):199–218, 2001.
- Levi HS Lelis, Sandra Zilles, and Robert C Holte. Predicting the size of ida* search tree. *Artificial Intelligence*, 196:53–76, 2013.
- Malte Helmert. The fast downward planning system. *J. Artif. Intell. Res.(JAIR)*, 26:191–246, 2006.