CENTRO DE CIENCIAS EXATAS E TECNOLOGICAS - CCE

DEPARTAMENTO DE INFORMATICA

PROJECT:

ON SELECTING HEURISTIC FUNCTIONS FOR DOMAIN-INDEPENDENT PLANNING.

Marvin Abisrror

Supervised by:
Levi Henrique Santana de Lelis, Santiago Franco

Contents

0.1 Introduction 2
0.2 Problem 2
    0.2.1 Formalizing the Problem 3
0.3 Objective 3
    0.3.1 Specific Objectives 3
0.4 Literature 3
0.5 Budget 4
0.6 Timetable 5

0.1 Introduction

Solving domain-independent problems is a very hard task when we do planning. Sometimes we will find instances that takes too much time to find the solution, and the measure of that time could be hours, days or even years. Users of search algorithms usually do not know *a priori* how long it will take to solve a problem instance − some instances could be solved quickly while others could take long time. Many researchs in the area of heuristic search explains why this could be happening, some of those reasons are: That the size of the tree is very large, the inconsistency of the heuristic used to do the search or the complexity of the instance that could contains cycles or even dead end areas which avoid to make any movement in the search space tree, etc.

Furthermore, domain-independent planning is related to the general problems solving where we basically require a very inteligent domain-independent search algorithm or heuristic to do the search in order to solve the instance without knowing in detail how is the problem and the search space it produces.

In addition, the search for solving domain-independent planning is critical in areas such as manufacturing, space systems, software engineers, robotics, education, and entertainment. On the whole, finding an strategy which allow us to solve the major number of problems using domain indepenent planning is a very important task in AI.

In the last few decades, AI has made significant strides in domain-independent planning. Some of the progress has resulted from adopting the heuristic search approach to problem-solving, where use of an appropiate heuristic often means substantial reduction in the time needed to solve hard problems. Initially heuristics were hand-crafted. These heuristics required domain-specific expertise and often much trial-and-error effort.

Recently, techniques have been developed to automatically generate heuristics for domain and problem specifications. The component that generate these heuristics are called *heuristic generators.*. And we find a good oportunity to use it in this project. We pretend to use the *heuristic generator* in order to create a large set of heuristics.

Finally, the proposal of this project is to develop an approach of selecting heuristics functions from a large set of heuristics. Not only to optimize the search of the solution applied by the domain-independent planning but also reduze the resources used of amount of memory and time.

0.2 Problem

Instances of domain-independent problems that requires a lot of time to find the solution are processes that require a lot of resources. The machines that execute this kind of process allocates a considerable amount of memory for this purpose. However, the main culprit for this behavior is the approach used to solve the problem. That is the reason why is needed an new approach based on selection that help us to solve the problems quickly and without spend a lot of time.

When we try to solve problems using only one heuristic. Even if the heuristic is *admissible* and *consistent* is not enough to find the solution quickly, because there are other variables that affect the search, for example: The distribution of the search tree, or the existence of nodes that is not possible to obtain their heuristic value. So, we thought that using hundreds or even thousands of heuristics instead of only one in the search will help us to find the solution quickly and resolves more problems.

So, the idea we have in mind is based in two steps: First, we make selection of the better heuristics from a bunch of heuristics. Then, we evaluate each node with the subset of heuristics in order to obtain the optimal path to the solution during the process of search. In this way, we assume that we can solve problems quickly and without spend too much time.

0.2.1 Formalizing the Problem

This project is concerned with cost-optimal state-space planning using the A* algorithm [Hart and Raphael, 1968]. We asume that a pool, ζ, of hundreds or even thousands of heuristics is available, and that the final heuristic we assume 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 $\triangledown$, 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 $\triangledown$.

0.3 Objective

The general objective of this project is to develop an approach of selection of heuristics functions in order to optimize the process of search the solution of domains problems. In addition, we pretend to find the best way to get the subset of heuristics that allow us to obtain the optimal path to the solution.

0.3.1 Specific Objectives

What we expect to achieve as a specific objectives are:

1. Finding the optimal subset of ζ of size N for a given problem task can be obtained using optimization process.
2. We are going to try different strategies to obtain the size N that fits the best the subset of heuristics.
3. We know that Stratification Sampling (SS) does not make even reasonable predictions for the number of nodes expanded by A*. Nevertheless, even though SS produces poor predictions for the number of nodes expanded by A*, we would like to verify whether these predictions can be helpful in selecting a subset of heuristics to guide the A* search.
4. For optimally solving domain-independent problems. We are going to use the Fast Downward planner as our base implementation.

0.4 Literature

Domain-Indepentent automatic planning require the basic knowledge of State-space search algorithms, such as A* [Hart and Raphael, 1968], which fundamental in many AI applications. A* uses the $f(s) = g(s) + h(s)$ cost function to guide its search. Here, $g(s)$ is the cost of the path from the start state to state s, and $h(s)$ is the estimated cost-to-go from s to a goal; $h(.)$ is known as the heuristic function.

A heuristic h is *admissible* if $h(s) \leq dist(s, goal)$ for every state **s** and goal state *goal*, where *dist(n, m)* is the cost of at least-cost path from n to m. If $h(n)$ is admissible, *i,e.* always returns a lower bound estimate of the optimal cost, these algorithms are guaranteed to find an optimal path from the start state to a goal state if one exists.

In addition to being admissible the heuristic also could be *consistent*. A heuristic h is *consistent* if for every pair of states, m and n, $h(m) - h(n) \leq dist(m, n)$

The system most similar to our proposal of selecting heuristics is RIDA* [Barley et al., 2014]. RIDA* also selects a subset from a pool of heuristics to guide the A* search. In RIDA* this is done by starting with an empty subset and trying all combinations of size one before trying the combination of size two and so on. RIDA* stops after evaluating a fixed number of subsets. While RIDA* is able to evaluate a set of heuristics with tens of elements, we pretend that our approach could evaluate a set of heuristics with thousands of elements.

0.5 Budget

Table 1: Total Cost of the Master

Cost Specification	Description	Cost ($R)	Source of Income
1.- Personal	Stipend to conduct the research (Scholarship amount X 24)	36,000	CAPES
Subtotal 1		**36,000**	
2.- Bibliographic Material	Books, Technical Journal, etc	400	Own Resources
Subtotal 2		**400**	
3.- Materials	Sheets of papers	50	Own Resources
	Printing	100	Own Resources
	USB	20	Own Resources
Subtotal 3		**170**	
Total = Subtotal 1 + Subtotal 2 + Subtotal 3		**36,570**	

0.6 Timetable

Table 2: Schedule from June to March

Activities	June	July	August	September	October	November	December	January	February	March
Present the project to the PPG	✓									
Implementing the *heuristic generator*	✓									
Implementing the prediction strategy A* or SS		✓								
Test which strategy of prediction fits the best the subset of heuristics		✓	✓							
Testing with the domain-independent problems of Fast-Downward			✓	✓						
Analyze the results and prepare a paper to submit				✓						
Organize and Analyze the results of SS compared with IDA* and Selection of heuristics functions					✓	✓				
Writing the theses						✓	✓	✓	✓	
Dissertation theses										✓

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.

Michael W. Barley, Santiago Franco, and Patricia J. Riddle. Overcoming the utility problem in heuristic generation: Why time matters. *Proceedings of the Twenty-Fourth International Conference on Automated Planning and Scheduling, ICAPS 2014, Portsmouth, New Hampshire, USA, June 21-26, 2014*, 2014.