Learning a Heuristic Function
Automated Planning Term Project Proposal

Douglas Souza
Faculdade de Informática
Pontifícia Universidade Católica do Rio Grande do Sul
Av. Ipiranga, 6681, Porto Alegre, RS
douglas.souza.002@acad.pucrs.br

Abstract

This paper aims to present the proposal for the final term project of the Automated Planing course. We propose the use of machine learning to learn a heuristic function to guide the planner search. We intend to achieve this result by exporting data of intermediate planning search, like literals, for example, and its associated cost to the objective state. Finally, this data is used to approximate a function that can be used as a new heuristic function. We expect this new function to present a better performance for the domain it was trained for compared to a standard heuristic function.

Introduction

We can observe a plan as a set of nodes. A node n can be viewed as a combination of the state s, its parent node, the action that resulted in this state, and the cost from the initial state to the current node. In this sense, we can relate the planing problem to a search problem, where we need to find the best path from the initial node to the goal node. In order to select the next node to visit, it is common the use of *heuristics*. Therefore, a heuristic is a function $h(n)$ that evaluates numerically each adjacent node to n (Ghallab, Nau, and Traverso 2004). Heuristics for node selection are often based on the *relaxation principle*: to define how desirable an adjacent node is, it simplifies the problem by relaxing constraints. The closer the relaxed problem is to the original problem the more informative the heuristic is. Similarly, the more relaxed the problem is, the easier to compute the heuristic. Thus, it is hard to find a good trade-off between informativeness and relaxation of the heuristic, it often requires a deep knowledge of the problem. (Ghallab, Nau, and Traverso 2004).

Machine learning can help in automated planning in two ways: learning **action models** to feed planners and **search control** to guide planners (Jiménez et al. 2012). The goal of this proposal is to apply machine learning help planners on the search control task. In other words, we could use machine learning to approximate a good heuristic function for each domain. A similar work was proposed by Yoon et al. in 2006 (Yoon, Fern, and Givan 2006), in which they defined an approximation of a heuristc funcion of the form $H(s, A, g) = \sum_i w_i \times f(s, A, g)$, where s is the state, A is the set of possible actions in that state, and g is the goal. It is basically a linear weighted combination of the features $f(s, A, g)$. In the context of learning a heuristic function, one of the main challenges is how to arrange the features that the learning algorithm will use as input. In the work mentioned above, their main contribution was the encoding of the features $f(s, A, g)$, where they tried to use more information than just the relaxed plan length.

Technical Approach

To address the problem, a solution will be developed that consists of two main tasks: 1) export data from planner, 2) use the data to estimate a heuristic function. The idea is to run the planner for a set of different problems of the same domain, for each plan found, we export the literals of each state s and the cost for the goal g at that state. The cost is simply the number of steps necessary to reach the goal from a state s.

After exporting data from the planner, it is necessary to prepare the data before we use it to estimate the heuristic function. Here lays the difficult part, the way the data is formatted can lead to very different results. This step will require a set of experiments to choose what works better. A very promising way is to require typed variables from the domain file and build a data set with these variable as features.

Once the data is ready, we can then estimate a function using a regression algorithm. In the work of Yoon et. al (Yoon, Fern, and Givan 2006), they used a linear function, which may not capture the behavior of the data in the best way. In this step it would be interesting to try different regression algorithms, perform experiments not only with linear functions, but also with polynomial functions and neural networks.The idea is that this learned function will have a better performance for any problem on the domain it was trained in than a standard function.

This project will be developed using the Python programming language.

Project Management

The project development will extend to the rest of the semester, which has about 7 weeks left. The duration of each

task can be seen in Table 1 and its description can be seen in Table 2.

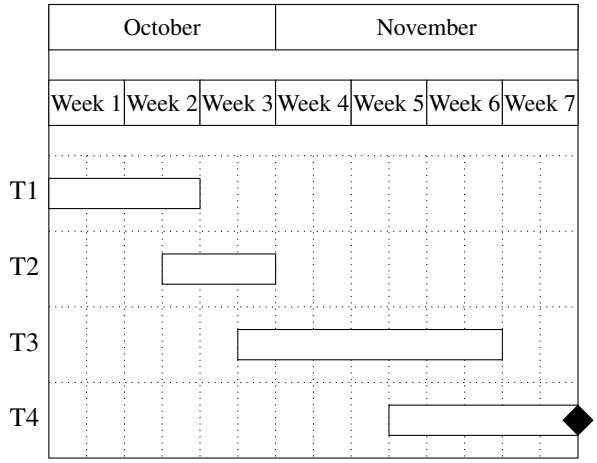

Table 1: Task schedule

ID	Task
T1	Investigate related work
T2	Adapt planner to export data
T3	Perform experiments
T4	Write report

Table 2: Task description

Conclusion

Finding a good heuristic function is not an easy task. Even though simple heuristics work well for most problems, for some of them they have a poor performance. Using a different set of problems for a specific domain, we can collect and store a reasonable amount of data. Assuming that the values of the literals in some state have a relation to the cost related to the goal state, we can leverage the power of machine learning to capture this relation. Finally, we can evaluate the performance of the planners using the learned heuristic *versus* planners using standard heuristic to measure if there is a real gain in performance.

References

[Ghallab, Nau, and Traverso 2004] Ghallab, M.; Nau, D.; and Traverso, P. 2004. *Automated planning: theory & practice*. Elsevier.

[Jiménez et al. 2012] Jiménez, S.; De La Rosa, T.; Fernández, S.; Fernández, F.; and Borrajo, D. 2012. A review of machine learning for automated planning. *The Knowledge Engineering Review* 27(04):433–467.

[Yoon, Fern, and Givan 2006] Yoon, S. W.; Fern, A.; and Givan, R. 2006. Learning heuristic functions from relaxed plans. In *ICAPS*, 162–171.