# Generating and Exploiting Cost Predictions in Heuristic State-Space Planning

Francesco Percassi[1], Alfonso E. Gerevini[2], Enrico Scala[2], Ivan Serina[2] & Mauro Vallati[1]

[1]University of Huddersfield [2]Università degli studi di Brescia

## Abstract

This paper proposes and investigates a novel way of combining machine learning and heuristic search to improve domain-independent planning. On the learning side, we use learning to predict the plan cost of a good solution for a given instance. On the planning side, we propose a bound-sensitive heuristic function that exploits such a prediction in a state-space planner. Our function combines the input prediction (derived inductively) with some pieces of information gathered during search (derived deductively). As the prediction can sometimes be grossly inaccurate, the function also provides means to recognise when the provided information is actually misguiding the search. Our experimental analysis demonstrates the usefulness of the proposed approach in a standard heuristic best-first search schema.

## The challenge

- Suppose to have a **cost prediction** of a solution for a planning problem
- This prediction could be made by a domain expert, or automatically generated
  - It could be **very inaccurate**

***Challenge:***

*How can we exploit a prediction of the cost of a plan to **improve search performance**?*

To address this question, here we introduce:

- a **domain-independent** approach for predicting the cost of a "good" solution
- a best-first search schema that can take advantage of the provided prediction
- an **adaptive mechanism** to cope with inaccurate predictions

## The considered search schema

- We used a standard best-first search schema
- We used two heuristics $h_\delta(n)$ (**admissible**) and $h_s(n)$ (**inadmissible**)
- Our search prioritizes the expansion of nodes having the lower *f-value*:

$$f(n) = g(n) + w * h^B(n)$$

where the heuristic term is a function involving $h_s(n), h_\delta(n), B$

## Proposed estimate-sensible heuristic

- We design an estimate-sensible heuristic to speed up the search

$$h^B(n) = h_s(n) * \left(\frac{B}{g(n)+h_\delta(n)}\right)$$

- **Multiplicative term**: nodes are **penalized** and **rewarded** in term of priority of extraction from the queue
- It accelerates the search towards nodes that are close to predicted value B:
  - the nodes such that $g(n) + h_\delta(n) \ll B$ are penalized
  - all nodes lying beyond the estimate are discounted (prioritized)

## How to cope with inaccurate predictions

- There may be cases where the predicted cost is a **serious underestimation** of an optimal plan: this could negatively affect search time
- We design a **reactive** strategy to avoid this problem:

$$penalty_{rate} = \frac{\#\{n \in exp.nodes \mid g(n)+h_\delta(n) > B\}}{\#exp.nodes}$$

- this metric is used to **smoothly deactivate** the strategy in favour of the search heuristic

$$h^B(n) = h_s(n) * \left(\frac{B}{g(n)+h_\delta(n)}\right)^{(1-penalty_{rate})}$$

## How do we predict costs?

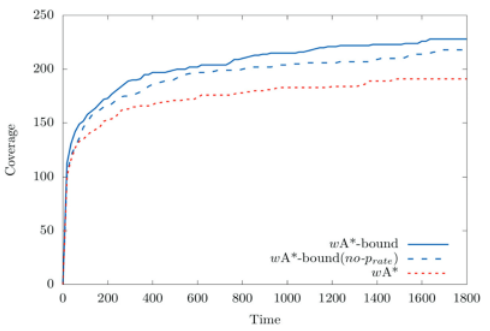
- Given a planning instance, we can extract useful information under the form of **features**
- Large set of features defined by the community
- Features comes from:
  - **Different encodings** of a planning problem (PDDL, SAS+)
  - **Pre-processing statistics** coming from relevant planning systems (LPG, FastDownward)
  - **Search space topology** (Torchlight)
  - FastDownward **probing features** –brief runs of a planner on the considered instance, in order to extract information from its search trajectories
- Large dataset to train the model (≈7000 planning instances)
- Training done using Auto-WEKA to generate a domain-independent predictor

## Experimental Results

**Experimental setting:**

- We compared a baseline (i.e. wA* with w=5) with the proposed search schema in FastDownward
- The search is guided by a couple of heuristics $\{h^{FF}, h^{LM}\}$ of which is made estimate-sensible $h^{FF}$
- $h_\delta = h^{max}$
- As benchmark, we considered all the instances used in the satisficing track of IPCs 2014 and 2018 (which are not included in the training set)
- As estimates we used those calculated using the proposed predictor
- Time = 1800sec, Memory = 8GB

| Domain | $\delta$ | score-Q *prediction* | score-Q *baseline* | score-T *prediction* | score-T *baseline* |
|---|---|---|---|---|---|
| Barman | -0.08 | 19.59 | 3.0 | 20.0 | 1.29 |
| Caldera | 7.02 | 12.83 | 14.72 | 8.16 | 15.0 |
| Caldera-split | 0.5 | 6.0 | 5.84 | 4.75 | 5.88 |
| Childsnack | -0.39 | 2.87 | 2.0 | 3.0 | 1.45 |
| CityCar | -0.36 | 9.17 | 7.0 | 11.76 | 5.64 |
| Data-network | 0.54 | 1.0 | 0.0 | 1.0 | 0.0 |
| GED | 3.14 | 18.97 | 18.47 | 18.9 | 16.07 |
| Hiking | -0.44 | 18.97 | 19.46 | 17.93 | 17.49 |
| Maintenance | 3.91 | 1.0 | 0.0 | 1.0 | 0.0 |
| Nurikabe | 1.49 | 11.82 | 10.66 | 11.09 | 11.0 |
| Openstacks | -0.0 | 20.0 | 20.0 | 19.47 | 17.12 |
| Organic-synthesis | 64.75 | 3.0 | 3.0 | 3.0 | 3.0 |
| Organic-synthesis-split | 0.48 | 6.0 | 5.0 | 6.0 | 3.76 |
| Parking | 0.0 | 15.44 | 19.91 | 17.41 | 18.57 |
| Settlers | -0.83 | 4.98 | 7.0 | 4.32 | 7.0 |
| Snake | 0.22 | 6.59 | 3.0 | 7.0 | 1.11 |
| Spider | 3.57 | 15.67 | 15.83 | 15.03 | 14.25 |
| Termes | -0.54 | 3.98 | 3.96 | 4.0 | 3.57 |
| Tetris | 0.37 | 6.33 | 3.0 | 6.45 | 2.68 |
| Thoughtful | 0.09 | 14.84 | 14.86 | 15.0 | 12.62 |
| Transport | 0.01 | 1.36 | 2.0 | 1.75 | 2.0 |
| Visitall | -0.18 | 14.76 | 8.81 | 14.6 | 7.24 |
| TOTAL | 1.82 | **215.17** | 187.52 | **211.62** | 166.74 |




## Conclusions

In this paper we have addressed the problem of exploiting plan cost predictions, computed at preprocessing through machine learning techniques, in order to improve planning performance for propositional domains with action costs. An effective use of these predictions during planning should take into account that the predicted cost can be (even grossly) inaccurate with respect to the best quality plan that a planning approach can found within a certain time limit. We have developed a cost prediction model that is based on standard machine learning techniques using a large set of in-stance features, and we have proposed a method to exploit the predictions made by this model in the context of wA*.Our method can be seen as a way to dynamically adjust the input weight w of the heuristic during search, taking into ac-count the predicted plan cost and preventing the search to be misguided when the prediction is severely inaccurate.