LIBACOP and LIBACOP2 Planner

Isabel Cenamor and **Tomás de la Rosa** and **Fernando Fernández**
Departamento de Informática, Universidad Carlos III de Madrid
Avda. de la Universidad, 30. Leganés (Madrid). Spain
icenamor@inf.uc3m.es, trosa@inf.uc3m.es, ffernand@inf.uc3m.es

Abstract

This document describes two planning portfolios developed for the Learning Track of IPC-2014 (*International Planning Competition*). The Learning Instance-Based Configured Portfolios are based on predictive models learnt with training instances gathered from previous executions of the base planners. For solving new planning problems, the portfolios select a group of planners using the predictions of the learnt models. Specifically, LIBACOP uses a classification model to select potential planners assigning the same execution time to each of them. LIBACOP2 uses the same classification model and, in addition, a regression model to predict the time for each planner.

Introduction

In recent years, the notion of portfolio has appeared for improving the performance of automated planners. The portfolios are motivated by the general idea that none of existing planners dominates all other in all cases (problems and domains). In the IPC scope, they consist of assigning the available time (maximum execution time) to a sub-set of available planners and running this configuration. In the state of the art, there are several portfolios that define different ways to combine simple base planners (strategies).

The most common strategy is the static, where the portfolio components and the time for each planner is previously defined and maintained for all domains and problems. FDSS (*Fast Downward Stone Soup*) (Helmert 2006) is an example of this type of portfolio. It has various configurations based on previous planning results, as a function of the track or the component that is considered. It obtained good results in the IPC-7.

The other possible strategy is dynamic, where the portfolio does not commit to the same configuration in all cases. An example of this type is PbP planner. PbP (Gerevini, Saetti, and Vallati 2009) generates domain-specific multi-planners from a set of domain-independent planning techniques. It generates macro-actions, optimizes planner parameters and selects specific planners for each domain. Therefore, it generates a different configuration for each domain. PbP won the learning track in IPC-6 and the following version PbP2 (Gerevini, Saetti, and Vallati 2011) won in IPC-7.

Our portfolios also follow a dynamic strategy, where a different configuration per problem is generated. Planners in the learning track exhibit two behaviours. The first one is a base strategy without knowledge and the second one exploits the domain-knowledge acquired from training problems. Our base strategy is the result of applying the Pareto efficiency technique (Censor 1977) to select a sub-set of planners from all planners in sequential satisficing track in IPC-7 plus LPG-tn (Gerevini et al. 2004) and SGPlan (Hsu and Wah 2008). We select the planners that dominate all others in at least one domain (from a set of training domains), taking into account quality and time. The selected planners are assigned the same running time. The learning-based behaviour is a dynamic strategy based on learning predictive models of the planner performance: one portfolio with classification model (LIBACOP) and the other one with classification and regression models (LIBACOP2).

LIBACOP is a portfolio configurable with a predictive classification model. This portfolio is an evolution of the base strategy (without knowledge). LIBACOP selects a sub-set of planners using a classification model. This model is the result of a learning process, and predicts the behaviour of the planners in future problems, i.e. whether they will be able to solve a given problem. The planners with higher confidence are selected, and they are ordered following such confidence. Then, running time is divided uniformly among them.

LIBACOP2 strategy is an evolution of LIBACOP. It has an extra phase that includes a regression model to predict the running time for the planners selected by the classification model. In contrast LIBACOP2 assigns the time estimation given by the regression model. In the next section we present the general ideas of the portfolios, with their components and how we finally created the portfolios.

General System Description

In this section, we explain the general process to configure LIBACOP planners. For each configuration in our planner, there is a learning phase for each domain. This process is depicted in Figure 1. For each domain, the system gathers some examples of future problems. From these examples we learn which sub-set of planners are likely to solve these problems. The learning phase has two difference parts. The first one is the characterization of the training problems (Extract Fea-

tures) and the second one is building the predictive models (Planner Selection and Time Assignment).

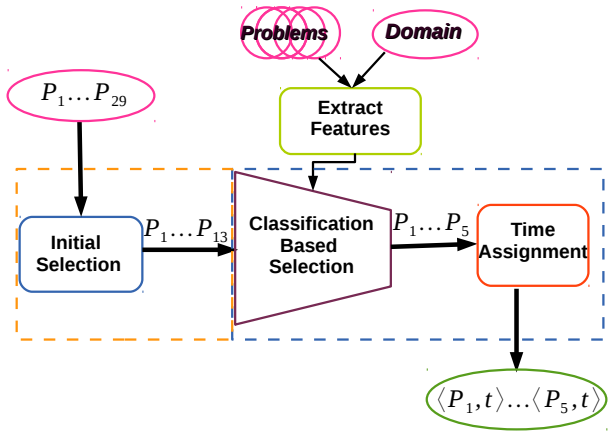

Figure 1: System Diagram of LIBaCoP Planners

As base planners we considered the competitors from IPC-7 plus LPG-tn and SGPlan. However, the initial set of planners is large and we preferred to do a pre-selection of them. The technique to implement the selection is Pareto efficiency.

Portfolio Components

The list of base planners included in our portfolios are:

- ARVAND (Nakhost, Valenzano, and Xie 2011)
- FD-AUTOTUNE 1 & 2 (Fawcett et al. 2011)
- FD STONE SOUP (FDSS) 1 & 2 (Helmert et al. 2011)
- LAMA 2008 & 2011 (Richter, Westphal, and Helmert 2011)
- PROBE (Lipovetzky and Geffner 2011)
- MADAGASCAR (Rintanen 2011)
- RANDWARD (Olsen and Bryce 2011)
- YAHSP2-MT (Vidal 2011)
- LPG-TN (Gerevini et al. 2004)
- LAMAR (Olsen and Bryce 2011)
- DAE-YAHSP (Dréo et al. 2011)
- SGPLAN (Hsu and Wah 2008)

The following step is problem characterization. We create some features to characterize the planning executions. In this phase, we also choose the output attribute of the learning process. There are two different task in this process, one for selecting the promising planners, which is whether the planner will find a solution for the problem in a 900 seconds. And, the other one, for estimating the run time each selected planner will spend in solving the problem.

Data Preparation

The first step is the characterization of the problem. For this task, we consider some features in the planning task previously used (Roberts and Howe 2009) and include others for a better particularization of the problem difficulty (Cenamor, de la Rosa, and Fernández 2012). These features show good accuracy for configuring portfolios (Cenamor, de la Rosa, and Fernández 2013; Roberts et al. 2008). In addition, we create some new features to improve the characterization of the initial state of the problem. Such features are:

Previous Features From the previous work (Cenamor, de la Rosa, and Fernández 2012), we include the number of objects and the number of goals, which are directly extracted from the PDDL files. Also a group of elaborated features are generated from the problem translation to the SAS$^+$ formalism (Backstrom and Nebel 1995) like the number of variables in the causal graph (CG), the ratio between the high level variable and all variables in the CG, the standard deviation of the number of input edges in the CG, the average number of output edges in the CG, the maximum and the average weight of the output edges in the CG, the standard deviation of the number of output edges in high level variables in the CG, the maximum weight of input edges in high level variables in the CG, the number of variables in the domain transition graph (DTG), the number of edges in the same graph and the maximum weight of input edges in the DTG.

New Features We include information that appears in the translation and preprocess of Fast Downward (Helmert 2006). The features are: the number of types of objects, the number of functions, the number of auxiliary atoms in the translation from PDDL to SAS$^+$, the number of implied effects removed groups, all resulting from the translation process. In addition, we include the most representative heuristic functions computed for the initial state with unit cost: h_{add}, h_{max} (Bonet and Geffner 2001), Context enhanced additive (Helmert and Geffner 2008), h_{FF} (Hoffmann and Nebel 2011), Goal count (i.e., the number of unsatisfied goals), Landmark count (Richter, Helmert, and Westphal 2008) and Landmark cut (Helmert and Domshlak 2009), the ratio h_{FF}/h_{max} and a set of features to characterize the *fact balance* of the relaxed plan (RP). We define the fact balance for fact p, as the number of times p appears as an add effect of an action belonging to RP, minus the number of times p is a delete effect of an action in RP, considering original actions where deletes are not ignored. The intuition behind fact balances is that high positive values would characterize easier (relaxed) problems for a given domain, since achieved facts do need to be deleted many times. Given that the number of relevant facts of a planning task is variable, we compute statistics (i.e., min, max, average and variance) for the fact balance of the relevant facts. Additionally, we compute statistics only considering facts that are goals, following the same procedure. Besides, the number of relevant facts and the number of actions are included in the feature selection.

In summary, we use 35 features, and the time to extract features is negligible given that features wrt. graphs imply

basic arithmetic computations and heuristic functions are only called once for the initial state.

Modelling the data

To continue with the process, we select and apply a variety of modelling techniques to find the model with higher accuracy. The output models should be evaluated in the context of the objective: planning capability of the developed portfolio.

In this step, a classification model is created to predict whether a planner will find a solution for a problem. Besides, a regression model to predict the running time for each planner is created. We trained with 25 algorithms (for different model types: trees, rules, support vector machines and instance based learning) using WEKA (Witten and Frank 2005). WEKA is a Data Mining toolkit that provides a standard format for running machine learning algorithms.

The last phase, the deployment, is the part of the process that verifies previously held hypotheses through the knowledge discovered in the earlier phases of the Data Mining methodology. The final system gets a new problem and domain, calculates the features, queries the models, and returns the sub-set of planners with their running time.

Deployment

In this section, we report an algorithmic version of the whole system. We show the general behaviour of the portfolio as a function of the learnt knowledge and the portfolio version.

The general working process appears in the algorithm 1. The input of this algorithm is the problem (i), the domain (d), the planners ($\mathcal{P}_{ini}$), the classification model ($\mathcal{C}$), the regression model ($\mathcal{R}$) and the available time (T). The output is a sorted list with the planners and their running time ($Portfolio = [< p_1, t_1 > \ldots < p_c, t_c >]$). If the system has the configuration by default (which means there is not classification neither regression model) the output is a base strategy. The time per the initial planners is the same for all components. In other case, the features (f) are extracted from the problem (i) and the domain (d). For each planner, the algorithm asks for the confidence to the classification model. After that, the best 5 planners are extracted with their parameters (i.e., their confidence, the set of initial planners and the number of the selected planners). If the regression model exists (LIBACOP2 planner), these planners query for their running time, otherwise the planners get the same time (180 seconds).

LIBACOP

The submitted version of LIBACOP planner uses a RandomForest (Breiman 2001) in the classification task. This model is a combination of tree predictors such that each tree depends on the values of a random vector and with the same distribution. It selects the 5 planners with the highest confidence of solving the problem. The execution order of the planners is based on their confidence. The running time is assigned uniformly to each planner (180 seconds).

Data: Problem (i), Domain (d), Set of base planners ($\mathcal{P}_{ini}$), Classification model ($\mathcal{C}$), Regression model ($\mathcal{R}$), Available time (T)
Result: *Portfolio Configuration*: Sorted list of planners with their running time, $Portfolio = [< p_1, t_1 > \ldots < p_c, t_c >]$

if $\mathcal{C}$ == *null and* $\mathcal{R}$ == *null* **then**
  (No classification nor regression models available)
  $n = size(\mathcal{P}_{ini})$;
  $Portfolio = []$;
  for p *in* $\mathcal{P}_{ini}$ **do**
    $append(< p, \frac{T}{n} >, Portfolio)$
  end
else
  (Classification model available)
  f = ExtractFeatures(d, i);
  $Portfolio = []$;
  for p *in* $\mathcal{P}_{ini}$ **do**
    c_p = GetConfidence $(\mathcal{C}, f, p)$
  end
  $n = 5$;
  $\mathcal{P}_{best}$ = ExtractBestConfidence $(\mathcal{P}_{ini}, \vec{c}, n)$;
  if $\mathcal{R} ==$ *null* **then**
    $Portfolio = []$;
    for p *in* $\mathcal{P}_{best}$ **do**
      $append(< p, \frac{T}{n} >, Portfolio)$
    end
  else
    (Regression model available)
    $Portfolio = []$;
    for p *in* $\mathcal{P}_{best}$ **do**
      $append(<$
      $p, TimeAssignment(\mathcal{R}, f, p) >$
      $, Portfolio)$
    end
  end
end

Algorithm 1: LIBACOP planners

LIBACOP2

The submitted version of LIBACOP2 planner uses the same classification model as LIBACOP (RandomForest) with the 5 planners with the highest confidence of solving the problem. The second step uses a DecisionTable (Kohavi 1995) model to decide the running time for the selected planners. The execution order of the planners is based on their confidence.

Acknowledgements

We generated sequential portfolios of existing planners to be submitted to the International Planning Competition 2014. Thus, we would like to acknowledge and thank the authors of the individual planners for their contribution and hard work. This work has been partially supported by the Spanish project TSI-090302-2011-6, TIN2012-38079 and TIN2012-38079-C03-02.

References

Backstrom, C., and Nebel, B. 1995. Complexity results for SAS+ planning. *Computational Intelligence* 11:625–655.

Bonet, B., and Geffner, H. 2001. Planning as heuristic search. *Artificial Intelligence* 129(1):5–33.

Breiman, L. 2001. Random forests. *Machine learning* 45(1):5–32.

Cenamor, I.; de la Rosa, T.; and Fernández, F. 2012. Mining ipc-2011 results. In *Proceedings of the Third Workshop on the International Planning Competition*.

Cenamor, I.; de la Rosa, T.; and Fernández, F. 2013. Learning predictive models to configure planning portfolios. In *Proceedings of the Workshop on the Planning and Learning*.

Censor, Y. 1977. Pareto optimality in multiobjective problems. *Applied Mathematics and Optimization* 4(1):41–59.

Dréo, J.; Savéant, P.; Schoenauer, M.; and Vidal, V. 2011. Divide-and-evolve: the marriage of descartes and darwin. *Proceedings of the 7th international planning competition (IPC). Freiburg, Germany*.

Fawcett, C.; Helmert, M.; Hoos, H.; Karpas, E.; Röger, G.; and Seipp, J. 2011. Fd-autotune: Automated configuration of fast downward. *The 2011 International Planning Competition* 31–37.

Gerevini, A.; Saetti, A.; Serina, I.; and Toninelli, P. 2004. Lpg-td: a fully automated planner for pddl2. 2 domains. In *In Proc. of the 14th Int. Conference on Automated Planning and Scheduling (ICAPS-04) International Planning Competition abstracts*. Citeseer.

Gerevini, A.; Saetti, A.; and Vallati, M. 2009. An automatically configurable portfolio-based planner with macro-actions: PbP. In *Proceedings of the 19th International Conference on Automated Planning and Scheduling (ICAPS-09)*.

Gerevini, A.; Saetti, A.; and Vallati, M. 2011. Pbp2: Automatic configuration of a portfolio-based multi-planner. *The 2011 International Planning Competition*.

Helmert, M., and Domshlak, C. 2009. Landmarks, critical paths and abstractions: What's the difference anyway? In *ICAPS*.

Helmert, M., and Geffner, H. 2008. Unifying the causal graph and additive heuristics. In *ICAPS*, 140–147.

Helmert, M.; Röger, G.; Seipp, J.; Karpas, E.; Hoffmann, J.; Keyder, E.; Nissim, R.; Richter, S.; and Westphal, M. 2011. Fast downward stone soup. *The 2011 International Planning Competition* 38.

Helmert, M. 2006. The fast downward planning system. *Journal of Artificial Intelligence Research* 26(1):191–246.

Hoffmann, J., and Nebel, B. 2011. The ff planning system: Fast plan generation through heuristic search. *arXiv preprint arXiv:1106.0675*.

Hsu, C.-W., and Wah, B. W. 2008. The sgplan planning system in ipc-6. In *Proceedings of IPC*.

Kohavi, R. 1995. The power of decision tables. In *8th European Conference on Machine Learning*, 174–189. Springer.

Lipovetzky, N., and Geffner, H. 2011. Searching with probes: The classical planner probe. *The 2011 International Planning Competition* 30(29):71.

Nakhost, H.; Valenzano, R.; and Xie, F. 2011. Arvand: the art of random walks. *The 2011 International Planning Competition* 15.

Olsen, A., and Bryce, D. 2011. Randward and lamar: Randomizing the ff heuristic. *The 2011 International Planning Competition* 55.

Richter, S.; Helmert, M.; and Westphal, M. 2008. Landmarks revisited. In *AAAI*, volume 8, 975–982.

Richter, S.; Westphal, M.; and Helmert, M. 2011. Lama 2008 and 2011. *The 2011 International Planning Competition* 50.

Rintanen, J. 2011. Madagascar: Efficient planning with sat. *The 2011 International Planning Competition* 61.

Roberts, M., and Howe, A. 2009. Learning from planner performance. *Artificial Intelligence* 173(5):536–561.

Roberts, M.; Howe, A. E.; Wilson, B.; and desJardins, M. 2008. What makes planners predictable?. In *ICAPS*, 288–295.

Vidal, V. 2011. Yahsp2: Keep it simple, stupid. *The 2011 International Planning Competition* 83–90.

Witten, I. H., and Frank, E. 2005. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann.