Hapori IBaCoP2

Patrick Ferber[1], Michael Katz[2], Jendrik Seipp[3], Silvan Sievers[1], Daniel Borrajo[4], Isabel Cenamor, Tomas de la Rosa, Fernando Fernandez-Rebollo[4], Carlos Linares López[4], Sergio Nuñez, Alberto Pozanco, Horst Samulowitz[2], Shirin Sohrabi[2]

[1] University of Basel, Switzerland
[2] IBM T.J. Watson Research Center, Yorktown Heights, USA
[3] Linköping University, Sweden
[4] Universidad Carlos III de Madrid, Spain

patrick.ferber@unibas.ch, michael.katz1@ibm.com, jendrik.seipp@liu.se, silvan.sievers@unibas.ch, dborrajo@ia.uc3m.es, icenamorg@gmail.com, tomdelarosa@gmail.com, ffernand@inf.uc3m.es, clinares@inf.uc3m.es, sergio.nunez@repsol.com, alberto.pozanco@gmail.com, samulowitz@us.ibm.com, shirin.sohrabi@gmail.com

Abstract

We describe Hapori-IBaCoP2, the instance-based configured portfolio we submitted to the classical tracks of the deterministic International Planning Competition held in 2023. The portfolio is the integration of IBaCoP2 (Cenamor, De La Rosa, and Fernández 2016) into the Hapori family of portfolios where the base planners and the training benchmarks are shared by all variants.

Introduction

The configuration of a sequential planning portfolio consists of a *(time, algorithm)* schedule empirically derived from the performance of base planners on a set of benchmarks. Usually, these configurations remain fixed regardless of the planning task being evaluated. The key idea of IBaCop portfolios is that we train a performance model that predicts the behaviour of planners when they solve a particular task, Thus, for new problems, we compute the features that characterize the task and query the model to select the subset of planners that are better candidates for finding a solution. IBaCoP2, the variant with the (*solved, unsolved*) classification model, was declared the winner of the satisficing track of the International Planning Competition (IPC) in 2014 (Vallati et al. 2015). The system and some further enhancements are described in IBaCoP papers (Cenamor, De La Rosa, and Fernández 2016; De la Rosa, Cenamor, and Fernández 2017).

Here, we focus on providing the general overview and the remarkable details considered to prepare the submission to IPC-2023 under the Hapori family of portfolios.

Feature Computation

To integrate IBaCoP2 into Hapori portfolios we have separated the feature computation from the system. Given a planning task (domain and problem) in PDDL, this module computes the following set of features:

- PDDL: Basic features extracted from the PDDL files, for instance, the number of objects or actions
- Instantiation: Features resulting from the task grounding into the finite domain representation, for instance the number of instantiated actions or relevant facts
- SAS+: Statistics and ratios from the properties of the causal graph and domain-transition graphs
- Fact Balance: Statistics over a set of propositions with the intention of capturing the relaxed plan structure
- Initial state heuristics: Different alternatives to estimate the hardness of the task through heuristic functions
- Landmarks: Statistics from the pre-computed landmarks
- Red-Black: Statistics and properties from the variables used to compute the Red-Black heuristic

Each of this feature sub-set is computed independently in separated programs. So, if one of them fails (e.g., running out of memory) we assign missing values in this subset. This allow us to use the output of the feature extraction even with partial information.

Data and Model Training

As pool of planners for our portfolios to choose from, we used all planners from IPC 2018. If a 2018 planner was already a portfolio, we used its component planners instead. For the particular case of IBaCoP2-2018, it uses planners and portfolios of which newer and stronger versions participated in IPC 2018 as standalone planners. Therefore, the initial pool consists of the newest individual components that we manage to compile and run.

As benchmarks, we used all tasks and domains from previous IPCs, from Delfi (Katz et al. 2018), and from the 22.03 Autoscale collection (Torralba, Seipp, and Sievers 2021), leading to a set of 92 domains with 7330 tasks. We used Downward Lab (Seipp et al. 2017) to run all planners on all benchmarks on AMD EPYC 7742 2.25GHz processors, imposing a memory limit of 8 GiB and a time limit of 30 minutes for optimal planners and 5 minutes for satisficing and agile planners. For each run, we stored its outcome (plan found, out of memory, out of time, task not supported by planner, error), the execution time, the maximum resident memory, and if the run found a plan, the plan length and plan cost. This data set is online available[1]. As training data for our optimal (respectively satisficing and agile) portfolios, we

[1] URL to be published

IPC	Planner	Configuration
ipc2014	Jasper	
ipc2014	MPC	
ipc2014	Probe	
ipc2018	Madagascar	
ipc2018	olcff	
ipc2018	Saarplan	
ipc2018	FD-2018	(config 43)
ipc2018	FD-2018	(config 55)
ipc2018	lapkt-bfws	bfws-pref-agl, bfws-pref-sat
ipc2018	lapkt-bfws	dual-bfws-agl, dual-bfws-sat
ipc2018	lapkt-bfws	poly-bfws
ipc2018	lapkt-dfs-plus	
ipc2018	symple1	

Table 1: List of IBaCoP2 candidate planners for the satisficing track. Please refer to the source code to see configuration parameters

IPC	Planner	Configuration
ipc2014	opt-symba1	
ipc2018	decstar	(config04)
ipc2018	decstar	(config06)
ipc2018	complementary2	
ipc2018	Delfi	h2-simpless-dks-celmcut
ipc2018	Delfi	simpless-dks-masb50kmiasmdfp
ipc2018	Delfi	simpless-oss-masb50kmiasmdfp
ipc2018	metis	metis2
ipc2018	planning-pdbs	
ipc2018	Scorpion	
ipc2018	symple1	

Table 2: List of IBaCoP2 candidate planners for the optimal track. Please refer to the source code to see configuration parameters

selected from each domain the 30 tasks which are solved by the fewest optimal (or satisficing or agile) planners, which results in 2377 remaining tasks.

From the pool of planners we selected a subset of good performing planners with the following procedure. From the training data, and for each benchmark we selected the planner that solved most problems (i.e., ties broken in alphabetical order). Then, from the list of 92 benchmarks we picked the top 15 planners that appear most often as top performer. The subset of selected planners for the satisficing and optimal tracks are listed in Table 1 and Table 2.

Regarding the training step, we did not care about whether the planners run out of time, out of memory or had an unspecified error. Thus, we re-labelled the training data to mark as ”unsolved” all tasks that did not get a solution for a given planner. Then, we trained a Random Forest classifier (Breiman 2001) using the scikit-learn python library. The forest contains 120 trees with max-depth set to 17.

Portfolio Configurations

For a given task, we compute the task features and query the model for each one of the candidate planners. The classifier provides the probability of solving the task. Based on the sorted list of probabilities, we select a subset of the best k planners. We set k=*5* in the satisficing track and k=*3* in the optimal track. The schedule for running the portfolio consists of assigning equal time to the k selected planners. The time for running the planners is the time limit minus the time spent in computing the features.

IBaCoP2 was initially not designed to work with optimal planners. However, now that we have modular components and a set of optimal planners is available under the Hapori pool, we were able to set a per-instance configuration with optimal planners. The learning task is the same as in the satisficing track, but the classifier is trained using data from optimal planners. Besides, for the optimal track it makes more sense to run 3 planners rather than 5 because (1) there are fewer candidate planners, and (2) in the particular case of Scorpion, it needs 300 seconds to run its pre-computation step.

Notes on Executing Sequential Portfolios

In the previous sections, we assumed that a portfolio simply assigns a runtime to each algorithm, leaving their sequential order unspecified. With the simplifying assumption that all planner runs use the full assigned time and do not communicate information, the order is indeed irrelevant. In reality the situation is more complex.

We do not know upfront how long a selected planner will really run. Therefore, we treat per-algorithm time limits defined by the portfolio as relative, rather than absolute values: whenever we start an algorithm, we compute the total allotted time of this and all following algorithms and scale it to the actually remaining computation time. We then assign the respective scaled time to the run. As a result, the last algorithm is allowed to use all of the remaining time. We use the driver component of Fast Downward (Helmert 2006) which implements the above described mechanic for running portfolios.

Acknowledgments

The success of a portfolio planner must be primarily attributed to the developers of the portfolio components. Therefore, we would like to express our gratitude to the numerous authors of the components on which our portfolios are based.

References

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

Cenamor, I.; De La Rosa, T.; and Fernández, F. 2016. The IBaCoP planning system: Instance-based configured portfolios. *Journal of Artificial Intelligence Research*, 56: 657–691.

De la Rosa, T.; Cenamor, I.; and Fernández, F. 2017. Performance modelling of planners from homogeneous problem sets. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 27, 425–433.

Helmert, M. 2006. The Fast Downward Planning System. 26: 191–246.

Katz, M.; Sohrabi, S.; Samulowitz, H.; and Sievers, S. 2018. Delfi: Online Planner Selection for Cost-Optimal Planning. In *Ninth International Planning Competition (IPC-9): Planner Abstracts*, 57–64.

Seipp, J.; Pommerening, F.; Sievers, S.; and Helmert, M. 2017. Downward Lab. https://doi.org/10.5281/zenodo.790461.

Torralba, Á.; Seipp, J.; and Sievers, S. 2021. Automatic Instance Generation for Classical Planning. In Goldman, R. P.; Biundo, S.; and Katz, M., eds., *Proceedings of the Thirty-First International Conference on Automated Planning and Scheduling (ICAPS 2021)*, 376–384. AAAI Press.

Vallati, M.; Chrpa, L.; Grześ, M.; McCluskey, T. L.; Roberts, M.; Sanner, S.; et al. 2015. The 2014 international planning competition: Progress and trends. *Ai Magazine*, 36(3): 90–98.