Hapori Explainable Decision Tree

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

Hapori Explainable Decision Tree[1] is a portfolio planner which participated in the optimal, satisficing, and agile tracks of the International Planning Competition (IPC) 2023. It uses the single-model planner selection via a decision tree by Ferber and Seipp (2022) to predict over IPC 2018 planners which planner to execute for a task.

Introduction

No planner excels at all tasks, but each has its individual strength and weaknesses (Roberts and Howe 2009). Thus, planner portfolios try to combine multiple planners to dip into the strength of each one. Delfi (Katz et al. 2018b; Sievers et al. 2019) is an example of an online portfolio. Given a task, Delfi converts it to an image and then uses a convolutional neural network (CNN) to predict the best planner for the task. Delfi is highly successful and won the classical, optimal track of the International Planning Competition (IPC) 2018. Delfi is not only successful, but also unexplainable. Experts understand neither which properties of a planning tasks can be seen in the constructed images nor which rules the CNN learned. The graph convolution based successor of Delfi (Ma et al. 2020) improved upon the first issue, but the second one remained. Ferber and Seipp (2022) successfully train explainable portfolios with a similar performance to Delfi. One of their portfolios is based on a single decision tree which receives a set of numeric, explainable features as input and outputs the name of the planner to execute. Here, we retrained this portfolio using a larger set of benchmarks tasks and the planners from the IPC 2018.

Method

Let $\mathcal{O}$ be the set of possible observations (in our case the set of all possible PDDL tasks). Let F be a list of *numeric features* such that each feature $f \in F$ is a function $f : \mathcal{O} \to \mathbb{R}$. A feature vector $\vec{x} \in \mathbb{R}^{|F|}$ for an observation o holds the evaluation of the features on the observation, i.e. $\vec{x}_i = F_i(o)$ for $1 \leq i \leq |F|$. A decision tree (Breiman et al. 1984) D trained on the features F is a binary tree where every internal node i is associated with a condition C_i over F and every leaf node l is associated with a prediction P_l (see Figure 1). Every condition C_i is a function $C_i(\vec{x}) = \vec{x}_j \leq t$ where $1 \leq j \leq |F|$ and $t \in \mathbb{R}$ is some threshold value. To evaluate the decision tree T on the feature vector $\vec{x}$, we set the root node as current node n. If n is an internal node, then we traverse to the first child if $C_n(\vec{x})$ holds and to the second child otherwise. If n is a leaf node, the decision tree evaluates to P_n.

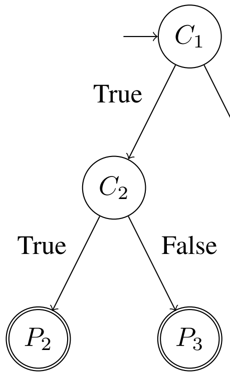

Figure 1: Schema of a decision tree. Each internal node is associated with a condition C_i on the input features. Each leaf node is associated with a prediction outcome P_j.

To train a decision T as portfolio, we specify the list of numeric features F, a set of training tasks T, and a list of planners P. For every task $t \in T$, we compute the feature vector $\vec{x}_t$ and the label vector $\vec{y}_t \in \mathbb{B}^{|P|}$ with $\vec{y}_{t,i} = \top$ iff the planner P_i find a solution for the task t within some resource limits. Our training data D consits of all pairs $\langle \vec{x}_t, i \rangle$ such that $t \in T$ and $\vec{y}_{t,i} = \top$. This data representation allows us to use any standard algorithm to train decision trees. A drawback of the representation is that the more planners solve a task, the more the task appears in the training data. Thus, tasks are not equally impactful during training. As a counter measure, we weight every sample $\langle \vec{x}_t, i \rangle$ by $1/n$ where n is the number of planners which solve t.

[1]Hapori is the Maori word for community.

Components and Training Data

As the pool of planners for our portfolios to choose from, we use all planners from the IPC 2018 and a selection of planners from IPC 2014. If an IPC 2018 planner is itself a portfolio, we use its component planners instead. We only consider each planner once. (Some IPC 2018 portfolios include planners that were also submitted separately and several portfolios included the same planners.)

Optimal Planners. For the optimal track, we exclude the planners MAPlan-1, MAPlan-2 and Meta-Search Planner because they use CPLEX, and Complementary1 because it may generate suboptimal solutions. Furthermore, the FDMS planners and Metis1 are covered by the Delfi portfolio already. This results in the following list of planners (or their components):

- Complementary2 (Franco, Lelis, and Barley 2018)
- components of DecStar (Gnad, Shleyfman, and Hoffmann 2018)
- components of Delfi (Delfi1 and Delfi2 have the same components; Katz et al., 2018b)
- Metis2 (Sievers and Katz 2018)
- Planning-PDBs (Moraru et al. 2018)
- Scorpion (Seipp 2018b)
- SymBA*1 (IPC 2014; Torralba et al., 2014)
- Symple-1 and Symple-2 (Speck, Geißer, and Mattmüller 2018)

Satisficing Planners. All planners participating in the IPC 2018 satisficing track also participated in the agile track (except for Fast Downward Stone Soup 2018), with an identical code base but possibly with different configurations. We thus only have one set of planners but multiple configurations for these two tracks. We exclude the Alien planner because we could not get it to run, and Freelunch-Doubly-Relaxed, FS-blind and FS-sim because they have a large number of dependencies which results in planner images too large to be included in our planner pool. Furthermore, IBaCoP-2018 and IBaCoP2-2018 use a large number of planners or portfolios of which newer and stronger versions participated in IPC 2018 as standalone planners, or which we failed to get to run, so we only cover the component planners Jasper, Madagascar, Mercury, and Probe. This results in the following list of planners (or their components):

- Cerberus and Cerberus-gl (Katz 2018)
- components of DecStar (Gnad, Shleyfman, and Hoffmann 2018)
- components of Fast Downward Remix (Seipp 2018a)
- components of Fast Downward Stone Soup 2018 (Seipp and Röger 2018)
- Jasper (IPC 2014; Xie, Müller, and Holte, 2014)
- Dual-BFWS, BFWS-preference, BFWS-polynomial and DFS$^+$ (Francès et al. 2018)
- Madagascar (IPC 2014; Rintanen, 2014)
- Mercury2014 (Katz and Hoffmann 2014)
- MERWIN (Katz et al. 2018a)
- OLCFF (Fickert and Hoffmann 2018)
- Probe (IPC 2014; Lipovetzky et al., 2014)
- Grey Planning configuration of Saarplan (Fickert et al., 2018; rest covered by DecStar)
- Symple-1 and Symple-2 (Speck, Geißer, and Mattmüller 2018)

Benchmarks and Runtimes. For training the portfolios, we use all tasks and domains from previous IPCs, from the Delfi training set (Katz et al. 2018b), and from the Autoscale 21.11 collection Torralba, Seipp, and Sievers (2021), leading to a set of 92 domains with 7330 tasks. We use Downward Lab (Seipp et al. 2017) to run all planners across 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 store its outcome (plan found, out of memory, out of time, task not supported by planner, unexpected 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 available online.[2] As training data for our optimal (respectively satisficing/agile) portfolios, we select from each domain the 30 tasks which are solved by the fewest optimal (or satisficing/agile) planners, which results in 1926 (optimal) and 2377 (satisficing/agile) tasks.

Features. Ferber and Seipp (2022) showed that their models performed best when trained on the 49 PDDL features of Fawcett et al. (2014). Thus, we also train our models on those features. Among others, those include the number of objects, the number of actions, and the mean number of parameters per predicate. For each task in our benchmark collection, the PDDL features are available at https://github.com/ipc2023-classical/planner19/tree/latest/learners/explainable_planner_selection in the files `features_opt.csv` and `features_sat.csv`.

Executing Predictive Portfolios

Given a task, the portfolio selector computes the values of its input features. Then, it evaluates the output of the trained model with respect to the values of the features. Next, it interprets the model output, e.g., if the model directly predicts a planner, then this planner is selected; if it predicts for each planner the probability that it solves the given task, then the planner with highest probability is selected. Finally, it executes the the selected planner for the whole time limit.

Post-IPC Analysis

The IPC 2023 used 7 domains with 20 tasks each, resulting in a benchmark set of 140 planning tasks, for all three tracks. Each planner was limited to 30 minutes of CPU time and 8 GiB of memory.

[2]https://github.com/ipc2023-classical/planner19/tree/latest/experiments/data/01-opt-planners-eval and https://github.com/ipc2023-classical/planner19/tree/latest/experiments/data/02-sat-planners-eval

In the optimal track, there were 22 competing planners. The objective was to optimally solve the tasks. The best planner solved 77 tasks, the blind baseline 50, the LM-cut baseline 34, and our portfolio only 31 tasks, ranking 20th. Unfortunately, our planner had many technical problems, such as writing to inaccessible temp directories. In those cases where it could run, it selected only four out of all available component planners.

In the satisficing track, there were 22 competing planners. The objective was to find plans of high quality. The best planner achieved a summed score of 71.86, only closely beating the baseline LAMA with a score of 68.76, and our planner scored 33.57, ranking 17th. Here, too, our planner had some bugs that lead to selecting a planner by the wrong name or choosing a planner which did not support some PDDL features. In the successful cases, our planner selected only five out of all available component planners.

In the agile track, there were 22 competing planners. The objective was to find plans as quickly as possible. The best planner achieved a score of 40.25, closely below the baseline LAMA-first with a score 40.28, and our planner scored 13.03, ranking 16th. Since the tasks used for the agile track are identical to those of the satisficing track, and the set of component planners available to our portfolios is identical for both tracks, we faced the exact same problems as in the satisficing track and selected the same component planners.

Compared to Hapori Linear Regression, our planner made a better selection in the satisficing and agile tracks, but not in the optimal one.

We are actively working on fixing the bugs of our planners, and aim to do a thorough comparison of the Hapori portfolios in a journal article.

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. We also thank Daniel Fišer and Florian Pommerening for organizing the competition and taking on the time-consuming task of running our numerous planner submissions.

References

Breiman, L.; Friedman, J. H.; Olshen, R. A.; and Stone, C. J. 1984. *Classification and Regression Trees*. Wadsworth.

Fawcett, C.; Vallati, M.; Hutter, F.; Hoffmann, J.; Hoos, H.; and Leyton-Brown, K. 2014. Improved Features for Runtime Prediction of Domain-Independent Planners. In Chien, S.; Fern, A.; Ruml, W.; and Do, M., eds., *Proceedings of the Twenty-Fourth International Conference on Automated Planning and Scheduling (ICAPS 2014)*, 355–359. AAAI Press.

Ferber, P.; and Seipp, J. 2022. Explainable Planner Selection for Classical Planning. In Honavar, V.; and Spaan, M., eds., *Proceedings of the Thirty-Sixth AAAI Conference on Artificial Intelligence (AAAI 2022)*, 9741–9749. AAAI Press.

Fickert, M.; Gnad, D.; Speicher, P.; and Hoffmann, J. 2018. SaarPlan: Combining Saarland's Greatest Planning Techniques. In *Ninth International Planning Competition (IPC-9): Planner Abstracts*, 11–16.

Fickert, M.; and Hoffmann, J. 2018. OLCFF: Online-Learning h^{CFF}. In *Ninth International Planning Competition (IPC-9): Planner Abstracts*, 17–19.

Francès, G.; Geffner, H.; Lipovetzky, N.; and Ramiréz, M. 2018. Best-First Width Search in the IPC 2018: Complete, Simulated, and Polynomial Variants. In *Ninth International Planning Competition (IPC-9): Planner Abstracts*, 23–27.

Franco, S.; Lelis, L. H. S.; and Barley, M. 2018. The Complementary2 Planner in the IPC 2018. In *Ninth International Planning Competition (IPC-9): Planner Abstracts*, 32–36.

Gnad, D.; Shleyfman, A.; and Hoffmann, J. 2018. DecStar – STAR-topology DECoupled Search at its best. In *Ninth International Planning Competition (IPC-9): Planner Abstracts*, 42–46.

Katz, M. 2018. Cerberus: Red-Black Heuristic for Planning Tasks with Conditional Effects Meets Novelty Heuristic and Enchanced Mutex Detection. In *Ninth International Planning Competition (IPC-9): Planner Abstracts*, 47–51.

Katz, M.; and Hoffmann, J. 2014. Mercury Planner: Pushing the Limits of Partial Delete Relaxation. In *Eighth International Planning Competition (IPC-8): Planner Abstracts*, 43–47.

Katz, M.; Lipovetzky, N.; Moshkovich, D.; and Tuisov, A. 2018a. MERWIN Planner: Mercury Enchanced With Novelty Heuristic. In *Ninth International Planning Competition (IPC-9): Planner Abstracts*, 53–56.

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

Lipovetzky, N.; Ramirez, M.; Muise, C.; and Geffner, H. 2014. Width and Inference Based Planners: SIW, BFS(f), and PROBE. In *Eighth International Planning Competition (IPC-8): Planner Abstracts*, 6–7.

Ma, T.; Ferber, P.; Huo, S.; Chen, J.; and Katz, M. 2020. Online Planner Selection with Graph Neural Networks and Adaptive Scheduling. In Conitzer, V.; and Sha, F., eds., *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence (AAAI 2020)*, 5077–5084. AAAI Press.

Moraru, I.; Edelkamp, S.; Martinez, M.; and Franco, S. 2018. Planning-PDBs Planner. In *Ninth International Planning Competition (IPC-9): Planner Abstracts*, 69–73.

Rintanen, J. 2014. Madagascar: Scalable Planning with SAT. In *Eighth International Planning Competition (IPC-8): Planner Abstracts*, 66–70.

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

Seipp, J. 2018a. Fast Downward Remix. In *Ninth International Planning Competition (IPC-9): Planner Abstracts*, 74–76.

Seipp, J. 2018b. Fast Downward Scorpion. In *Ninth International Planning Competition (IPC-9): Planner Abstracts*, 77–79.

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

Seipp, J.; and Röger, G. 2018. Fast Downward Stone Soup 2018. In *Ninth International Planning Competition (IPC-9): Planner Abstracts*, 80–82.

Sievers, S.; and Katz, M. 2018. Metis 2018. In *Ninth International Planning Competition (IPC-9): Planner Abstracts*, 83–84.

Sievers, S.; Katz, M.; Sohrabi, S.; Samulowitz, H.; and Ferber, P. 2019. Deep Learning for Cost-Optimal Planning: Task-Dependent Planner Selection. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence (AAAI 2019)*, 7715–7723. AAAI Press.

Speck, D.; Geißer, F.; and Mattmüller, R. 2018. SYMPLE: Symbolic Planning based on EVMDDs. In *Ninth International Planning Competition (IPC-9): Planner Abstracts*, 91–94.

Torralba, Á.; Alcázar, V.; Borrajo, D.; Kissmann, P.; and Edelkamp, S. 2014. SymBA*: A Symbolic Bidirectional A* Planner. In *Eighth International Planning Competition (IPC-8): Planner Abstracts*, 105–109.

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.

Xie, F.; Müller, M.; and Holte, R. 2014. Jasper: the art of exploration in Greedy Best First Search. In *Eighth International Planning Competition (IPC-8): Planner Abstracts*, 39–42.