NuCeLaR

Sergio Núñez and **Isabel Cenamor** and **Jesús Virseda**
Departamento de Informática, Universidad Carlos III de Madrid
Avda. de la Universidad, 30. 28911 Leganés (Madrid). Spain
sergio.nunez@uc3m.es, icenamor@inf.uc3m.es, jvirseda@inf.uc3m.es

Abstract

In this document we describe the techniques used to configure NuCeLaR, a sequential portfolio submitted and adapted to the three deterministic sequential tracks of the International Planning Competition 2014: sequential optimal, sequential satisficing and sequential multi-core. This portfolio has been configured using the combination of Machine Learning techniques and Mixed-Integer Programming.

Introduction

Since none of the existing planners dominates all others in every domains, the combination of some of them intuitively should improve their performance. Different approaches to combine existing planners have been proposed; i.e. using different components of different planners during the same search. Specifically, the combination of several planners independently executed in sequence with short timeouts are usually named portfolios. Works like (Helmert, Röger, and Karpas 2011; Gerevini, Saetti, and Vallati 2009) have shown that portfolios are a useful approach, since they achieved quite successful results in the previous International Planning Competitions (IPCs).

In this work, we apply a new strategy to combine existing planners using a portfolio approach: we configure a sequential portfolio for each kind of problem. To determine these kinds of problems, we apply machine learning to a set of planning problems from past IPCs. Particularly, we split these training problems in different groups using clustering techniques. These techniques are applied over a set of problem features extracted from the training problems. Once the set of training problems is split into different subsets (clusters), we compute a different portfolio configuration for each subset using a technique based on Mixed-Integer Programming (MIP) (Núñez, Borrajo, and Linares López 2012). Finally, we analyze the features of the problem to be solved and we run the corresponding portfolio configuration.

Figure 1 shows the two phases of the system: learning and deployment. The learning phase is also subdivided in two steps: clustering and portfolio generation.

The next sections describe in more detail both phases and provide specific information about the planners in the different tracks.

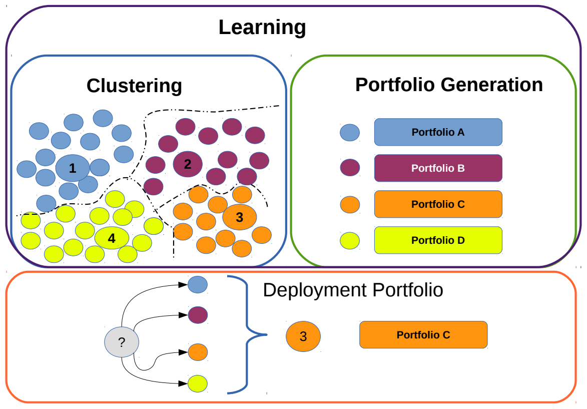

Figure 1: General System Diagram

Clustering Phase

The goal of data clustering (Jain 2010), also known as cluster analysis, is to discover the natural grouping of a set of patterns or points. Thus, this statistical classification technique determines whether the individuals of a population fall into a group or another by making quantitative comparisons of multiple characteristics. An operational definition of clustering can be defined as a representation of n objects, where the objective is to find k groups based on a measure of similarity. The similarities between instances that belong to the same group are high while the similarities between instances in different groups are low.

This methodology has fundamental challenges associated (Jain and Dubes 1988). The most relevant to achieve our objective are the feature selection, the data normalization, the number of clusters and whether the discovered clusters and partitions are valid for a portfolio configuration. We define an appropriate characterization of instances to find good solutions based in previous works (Roberts and Howe 2009; Cenamor, de la Rosa, and Fernández 2012; Virseda, Borrajo, and Alcázar 2013) and we evaluate them to obtain the best combination of features.

Therefore, we split the initial set of problems in several sub-groups using the selected features. However, the key is to know how many groups is the best choice. To do that, we configure the corresponding portfolio for each possible number of clusters (within a range) and select the best one in order to find the best performance. Consequently, the selected k value is the one that solves more problems and achieves the best quality in the evaluation phase.

Portfolio Generator

The portfolio configuration for each cluster has been generated using MIP (Wolsey 2008), which computes the portfolio with the best achievable performance with respect to a selection of training planning tasks (Núñez, Borrajo, and Linares López 2012). The resulting portfolio configuration is a linear combination of candidate planners defined as a sorted set of pairs <planner, time>. The MIP model considers an *objective function* that maximizes a weighted sum of different parameters including: overall running time and quality.

Since the MIP model takes into account two different criteria (time and quality), it could be viewed and solved as a multi-objective maximization problem. Instead, we solve two MIP tasks in sequence while preserving the cost of the objective function from the solution of the first MIP. Specifically, we first run the MIP task to optimize only *quality* —i. e., sum of the plan quality of each solved problem for the satisficing track and the total number of solved planning tasks for the optimal track. If a solution exists, then a second execution of the MIP model is performed to find the combination of candidate planners that achieves the same quality (denoted as Q) while minimizing the overall running time. To enforce a solution with the same quality an additional constraint is added: $\sum_{i=0}^{n} quality_i \geq Q - \epsilon$, where ϵ is just any small real value used to avoid floating-point errors. Clearly, a solution is guaranteed to exist here, since a first solution was already found in the previous step. Algorithm 1 shows the steps followed to generate all the submitted portfolios where quality was maximized first, and then running time was minimized among the combinations that achieved the optimal quality. In our experiments, $\epsilon = 0.001$.

Algorithm 1 Build a portfolio optimizing quality and time

```
set weights to optimize only quality
$portfolio_1$ := solve the MIP task
$Q$ := the resulting value of the objective function
**if** a solution exists **then**
    add constraint $\sum_{i=0}^{n} quality_i \geq Q - 0.001$
    set weights to optimize only overall running time
    $portfolio_2$ := solve the MIP task
    **return** $portfolio_2$
**else**
    exit with no solution
**end if**
```

Implementation of the Portfolio

Every submitted portfolio runs a fixed portfolio configuration for each problem to be solved. However, the runtime assigned to each component planner can change in unexpected ways during its execution when the component planner finishes prematurely: planner bugs, terminating cleanly without solving the instance, running out of memory, etc. Therefore, the total runtime of the executed portfolio can be lower than the available time. In this case, the submitted portfolio will run a default planner using the remaining time (RT). This default planner is picked up among the set of candidate planners which had a remarkable performance in the IPC 2011.

Sequential Optimization Track

In the design of NUCELAR, we have used all the problems defined in the optimal track of the IPC 2011. Also, we considered all the participant planners in that competition but FORKINIT, IFORKINIT and LMFORK because the organizers of the IPC 2014 had problems with the license of the Mosek LP solver [1]. Since the set of candidate planners was too small, we discarded the participant portfolios and added their component planners instead. Moreover, we included all the planners considered in the design of FDSS (Helmert, Röger, and Karpas 2011).

Table 1 shows the configuration of the NUCELAR portfolio for the sequential optimization track. This configuration is composed of six portfolio configurations in turn, one for each cluster, since we defined six clusters in the clustering phase. The execution sequence of the component planners has been sorted by increasing order of the allotted time.

Planner	Allotted time (s)
GAMER	1800
CPT4	550
RHW LANDMARKS	598
M&S-BISIM 1	652
FD AUTOTUNE	191
M&S-BISIM 2	326
GAMER	1282
LM-CUT	105
M&S-BISIM 1	188
M&S-BISIM 2	220
GAMER	1237
SELMAX	1800
CPT4	131
M&S-BISIM 2	331
h^{max} LANDMARKS	1356
SELMAX	RT

Table 1: Configuration of NUCELAR for the sequential optimization track.

Sequential Satisficing Track

NUCELAR for the satisficing track has been configured applying our technique over all the satisficing planning tasks

[1]MOSEK is a tool for solving mathematical optimization problems. http://mosek.com/

defined for the IPC 2011. Also, we have used all the participant planners in that competition and the component solvers of the participant portfolios. Moreover, we included all the planners considered in the design of FDSS. The FDSS planners are defined by a search algorithm, an evaluation method and a set of heuristics. Specifically, FDSS only considered weighted-A* w=3 (WA*) and greedy best-first search (GBFS), with EAGER (standard) and LAZY (deferred evaluation) variants of both search algorithms. Also, only four heuristics were considered: Additive heuristic ADD (Bonet and Geffner 2001), FF/additive heuristic FF (Hoffmann and Nebel 2001; Keyder and Geffner 2008), Causal Graph heuristic CG (Helmert 2004), and Context-Enhanced Additive heuristic CEA (Helmert and Geffner 2008).

The resulting portfolio is shown in Table 2, which contains one portfolio configuration for each one of the six clusters defined in the cluster analysis.

Planner	**Allotted time (s)**
YAHSP2 MT	2
LAMA 2011	3
FD AUTOTUNE 2	4
MADAGASCAR P	5
FD AUTOTUNE 1	5
YAHSP2	6
DAE YAHSP	27
ROAMER	30
GBFS - EAGER - FF, CG	78
GBFS - EAGER - CG	109
GBFS - LAZY - CG	116
WA* - LAZY - CG	227
PROBE	339
ARVAND	849
LAMA 2011	218
GBFS - LAZY - FF, CG	295
GBFS - LAZY - ADD, FF	1287
WA* - LAZY - FF	1800
FD AUTOTUNE 2	762
RANDWARD	1037
YAHSP2	5
YAHSP2 MT	5
FDSS 2	48
LAMA 2008	73
GBFS - EAGER - CG	142
FD AUTOTUNE 2	280
LAMA 2011	432
FORKUNIFORM	813
LAMAR	47
WA* - LAZY - FF	91
FDSS 1	1660
ROAMER	RT

Table 2: Configuration of NUCELAR for the sequential satisficing track.

Sequential Multi-Core Track

The NUCELAR portfolio for the multi-core track has been configured using the same training data (candidate planners and training planning tasks) and the same technique (adding the concept of core processor to the MIP model) used to configure the sequential satisficing portfolio.

The resulting portfolio is shown in Table 3, which is composed of six portfolio configurations since we defined six clusters in the clustering phase. Each portfolio configuration uses the four cores available and respects the wall-clock time limit defined in the competition.

Planner	**Allotted Time (s)**
PROBE	1356
ARVAND	1800
YAHSP2 MT	8
LAMA 2011	12
FD AUTOTUNE 2	16
MADAGASCAR P	20
FD AUTOTUNE 1	20
YAHSP2	24
DAE YAHSP	108
ROAMER	120
GBFS - EAGER - FF, CG	312
GBFS - EAGER - CG	436
GBFS - LAZY - CG	464
WA* - LAZY - CG	908
LAMA 2011	1800
GBFS - LAZY - FF, CG	1800
GBFS - LAZY - ADD, FF	1800
PROBE	1800
WA* - LAZY - FF	1800
PROBE	1800
LAMA 2011	1800
ARVAND	1800
FD AUTOTUNE 2	1800
RANDWARD	1800
PROBE	1800
LAMA 2011	1800
FORKUNIFORM	1800
LAMA 2011	1728
YAHSP2	20
YAHSP2 MT	20
FDSS 2	192
LAMA 2008	292
GBFS - EAGER - CG	142
FD AUTOTUNE 2	1120
LAMAR	1800
WA* - LAZY - FF	1800
FDSS 1	1800
PROBE	1800
LAMAR	RT
LAMA 2011	RT

Table 3: Configuration of the NUCELAR portfolio for the multi-core track.

Acknowledgments

We automatically generated sequential portfolios of existing planners in other competitions. 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 and the Planinteraction project TIN2011-27652-C03-02.

References

Bonet, B., and Geffner, H. 2001. Planning as Heuristic Search. *Artificial Intelligence* 129(1-2):5–33.

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*.

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 2009)*. AAAI.

Helmert, M., and Geffner, H. 2008. Unifying the Causal Graph and Additive Heuristics. In *Proceedings of the Eighteenth International Conference on Automated Planning and Scheduling (ICAPS 2008)*, 140–147.

Helmert, M.; Röger, G.; and Karpas, E. 2011. Fast Downward Stone Soup: A Baseline for Building Planner Portfolios. *In ICAPS 2011 Workshop on Planning and Learning* 28–35.

Helmert, M. 2004. A Planning Heuristic Based on Causal Graph Analysis. In *Proceedings of the Fourteenth International Conference on Automated Planning and Scheduling (ICAPS 2004)*, 161–170. AAAI Press.

Hoffmann, J., and Nebel, B. 2001. The FF Planning System: Fast Plan Generation Through Heuristic Search. *Journal of Artificial Intelligence Research (JAIR)* 14:253–302.

Jain, A. K., and Dubes, R. C. 1988. *Algorithms for clustering data*. Prentice-Hall, Inc.

Jain, A. K. 2010. Data clustering: 50 years beyond k-means. *Pattern Recognition Letters* 31(8):651–666.

Keyder, E., and Geffner, H. 2008. Heuristics for Planning with Action Costs Revisited. In *Proceedings of the 18th European Conference on Artificial Intelligence (ECAI 2008)*, 588–592.

Núñez, S.; Borrajo, D.; and Linares López, C. 2012. Performance Analysis of Planning Portfolios. In *Proceedings of the Fifth Annual Symposium on Combinatorial Search, SOCS, Niagara Falls, Ontario, Canada, July 19-21, 2012*. AAAI Press.

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

Virseda, J.; Borrajo, D.; and Alcázar, V. 2013. Learning heuristic functions for cost-based planning. *Planning and Learning* 6.

Wolsey, L. A. 2008. Mixed integer programming. *Wiley Encyclopedia of Computer Science and Engineering*.