STAIRS 2014
U. Endriss and J. Leite (Eds.)
© 2014 The Authors and IOS Press.
This article is published online with Open Access by IOS Press and distributed under the terms of the Creative Commons Attribution Non-Commercial License.
doi:10.3233/978-1-61499-421-3-270

A Cost-Based Relaxed Planning Graph Heuristic for Enhanced Metric Sensitivity

Michal SROKA [a], Derek LONG [a]
[a] *name.surname@kcl.ac.uk*

Abstract. Most applications of planning depend on finding high quality solutions. The quality is evaluated in terms of user defined metric function. We discuss the current state-of-the-art for finding good quality solutions, and its limitations. We determine which planners are metric sensitive and are driven by cost. Following that we present a novel metric sensitivity heuristic using a modified version of the relaxed planning graph. The proposed heuristic helps in generating plans in response to the change of the metrics.

Keywords. planning, multi-objective, metrics, metric sensitive

Introduction

Practical applications of planning depend on finding good quality plans, where the quality is generally not measured simply by the number of actions in the plan. Instead, it is usual for the quality of a plan to be measured by costs associated with different actions, which reflect consumption of resources: typically time, energy or money. PDDL2.1 [4] introduced the extension that allows a user to specify the plan metric by which the quality of a plan is to be measured, but it is still the case that most modern planners ignore this specification and simply focus on generating short plans. Most benchmark domains exhibit a close correlation between plan length and plan cost, so the fact that planners ignore the cost is obscured in empirical evaluations of performance.

In this paper, we explore the property of metric sensitivity [11], which means that a planner responds directly to the plan metric. More specifically, we consider examples of problems in which the optimal solution can vary significantly as the metric changes, and then examine the small set of modern planners that are responsive to the plan metric. User-specified plan metric is used to evaluate the plan quality. The use of metric fluents may cause the action cost to vary depending on the state in which they are applied. A planner can mitigate the cost by appropriately preparing the state.

After a short example we briefly review the state-of-the-art of planning with metric functions, a description of domains which offer trade-offs between resources to achieve the goal. A selection of current state-of-the-art planners, together with evaluation of their metric sensitivity. We conclude that very few demonstrate metric sensitivity. In Section 3, we introduce a novel method for calculating a metric sensitive heuristic function using a cost-based Relaxed Planning Graph. An implementation of the method is then presented using a novel compilation from a non-temporal cost domain to a temporal domain. We conclude with a comparison between our method and relevant state-of-the-art planner.

1. Example

We now present a simple concrete example problem that illustrates some of the issues involved in finding high quality plans. Suppose we want to transport two packages from location $L0$ to location $L5$. We have two drivers and three vehicles available, one electric, *Te*1, and two diesel, *Tf*1 and *Tf*2. The amount of resource used by a vehicle is equal to the distance driven multiplied by the *square* of the loaded truck weight (number of packages plus one) using metric fluents this is $(resource\text{-}used) = (distance\ ?l1\ ?l2) * (load\ ?t)^2$ where resource can be diesel or electricity. This problem is represented in Figure 1. The following plan metric is used for evaluation: $(minimize(+(*A(electricity\text{-}used))(*B(fuel\text{-}used))))$. Where A and B vary, to check how planner responds to that change.

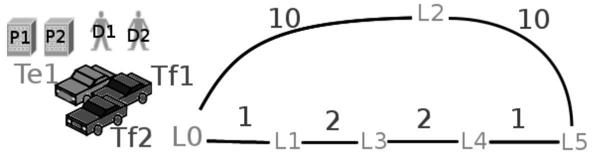


Figure 1. A simple problem with different vehicle types to transport packages $P1$ and $P2$ to $L5$.

Plans solving this problem are summarised below:

1) Load both packages into *Tf*1 and then drive them to $L5$ via $L2$ using driver $D1$. Cost: 180 diesel, 0 electric, length: 6.	2) Load both packages into *Tf*1 and drive to $L5$ via $L1$, $L3$ and $L4$, using $D1$. Cost: 54 diesel, 0 electric, length: 9.
3) Load both packages into *Te*1 and drive to $L5$ via $L1$, $L3$ and $L4$, using $D1$. Cost: 0 diesel, 54 electric, length: 9.	4) Load one package into *Tf*1 and one into *Tf*2 and drive both to $L5$ via $L1$. Cost: 48 diesel, 0 electric, length: 14.

Plans 1, 2, 3 and 4 are all different, both qualitatively and quantitatively. It is clear that the optimal cost plan, under any combination of metric fluents, will involve using the shorter path. The fact that this uses more actions is a problem for many planners, which attempt to minimise plan length as a proxy for the plan quality.

Using a single plan metric combining metric fluents like fuel cost and electricity cost with respective weights of 9 and 8, the optimal plan uses one diesel truck and the electric truck, each carrying one package (cost 24 diesel and 24 electricity, with total weighted cost of 408, while both plans 3 and 4 have weighted cost of 432). This example illustrates how interesting choices arise according to the trade-offs between resources being used.

2. Background

For generation of good quality plans under different plan metrics we require the planner to be metric sensitive [11]. A planner is metric sensitive if, presented with two different plan metrics (cost), m_1 and m_2, it produces different plans, π_1 and π_2 for the same problem instance, such that $m_1(\pi_1) < m_1(\pi_2)$ and $m_2(\pi_1) > m_2(\pi_2)$. In practice, a metric sensitive planner will not always be able to find two different plans that are each better under their respective plan metrics. In their specification of PDDL2.1, Fox and Long [4] introduced plan metrics, which are functions of the metric fluent parameters of a planning problem used to evaluate the cost or reward of a solution plan. For example:

```
(:metric minimize (+(energy)(+(*5 (labour))(*4 (pollution)))))
```

Radzi [9] distinguishes different interactions between plan lengths and plan metrics. In the following we use π and $\pi;\sigma$ to stand for executable sequences of actions, where the latter is a concatenation of two sub-sequences, $c(\pi)$ as the cost of π and $|\pi|$ as the length of π. We say a plan metric, c, is *strictly length-correlated* if $\forall \pi_1 \pi_2 \cdot c(\pi_1) \leq c(\pi_2) \rightarrow |\pi_1| \leq |\pi_2|$, *monotonic* if $\forall \pi\sigma \cdot c(\pi) \leq c(\pi;\sigma)$, *non-monotonic* if $\exists \pi\sigma \cdot c(\pi) > c(\pi;\sigma)$. While most current planners optimise quality only when the plan metric is strictly length-correlated, only metric sensitive planners can effectively deal with monotonic metrics. There are currently no planners, including our own, that deal effectively with non-monotonic metrics.

Keyder and Geffner [7] present one approach to solving domains with monotonic plan metric. Their heuristic is calculated backward from the goal taking the cheapest achievers of each open facts, where the achievers are taken from the RPG constructed. This gives the cheapest relaxed plan which is found when constructing the RPG, however, it is still prone to bias demonstrated in the example as the actions which does not appear in the RPG, like driving via l1, are not considered in the construction of the relaxed plan.

2.1. Domains

Current benchmark planning domains do not usually provide us with interesting plan metric functions that are unrelated to the plan makespan. In our experiments we use domains which contain the possibility of trade-off between various resources, introduced in [9], Bread and Production. They contain the property that the length of the plan does not correspond with the quality and, therefore, this property forces the planner to look for a better rather than shorter solutions in order to increase the quality. We also introduce a modified Driverlog domain [8], decoupling the plan length and quality.

The Bread domain describes a process of baking bread and buns. We start with flour which we turn into a mix. The mix is used to create a dough. Dough is either created by hand or using a machine. Using the machine increases energy consumption and the machine needs cleaning, but can create twice more dough comparing to doing it by hand. The next stage is to make a bun or a bread from the dough. From the same amount of dough we can form either two loaves of bread or five buns. They can be baked using either an electric oven or on charcoal. An electric oven uses one unit of energy and charcoal increases the pollution by one unit. An electric oven can bake ten buns or four loaves while charcoal only two buns or two loaves. Our plan metric function is composed of weighted sum of the following metric fluents: energy, labour, pollution.

The aim in the Production domain is to obtain a certain amount of materials and ready products in stock. There are various ways of creating the same product or obtaining materials. The planner has flexibility to use various methods to arrive at the same goal which differ in labour, hazard or machine-cost values.

The Driverlog domain [8] used in experiments is the standard benchmark but differentiates electric and diesel trucks which creates a trade off between diesel and electricity consumption used for transportation of packages. Driverlog_metric was further modified by an addition of multiple shortcuts for driving between cities where, if used, the resource consumption is much smaller but the number of drive actions is greater. This change is a good test for planners which are truly metric sensitive. In our experiments we have used two versions of Driverlog domain: Basic, where no changes are made to the infrastructure, metric fluents, and plan metrics, where three locations, $s0$, $s1$ and $s2$, are connected by very short routes with two intermediate steps.

2.2. Satisficing planners

Satisficing planners, due to their speed, offer a more promising approach to solving larger problem instances. We briefly present some state-of-the-art planners which are capable of reasoning with plan metrics as specified in PDDL2.1 [4]. Many satisficing planners can generate high quality plans for domains containing metrics. Some of them, like Lama [10]; aim at generating high quality solutions, however, they typically do not support :FLUENTS. The only metrics which are allowed are the metrics which affect the (TOTAL-COST) function which is assumed to be the cost of an action. Our work focuses on providing the flexibility for the user of the planning system to define functions against which the plan is evaluated. Therefore these approaches are not applicable, since the planner must generate good quality plans depending on the metric. The candidates which are promising for exhibiting metric sensitive behaviour are presented below.

MetricFF [6] is a very successful domain independent planner working with PDDL2.1. For each state which it evaluates it solves a relaxed version of the planning problem to obtain the cost of arriving to the goal state. The most recent version is capable of handling metric fluents and optimising a cost function, so it is a good candidate for a metric sensitive planner.

LPG [5], is a domain independent, local search, stochastic planner. It creates its search space based on a graph with interleaved proposition and action layers. Its heuristic consists of two elements, estimating the expected search cost and the expected execution cost to complete the plan from an evaluated search state, where search cost is the cost to resolve all the flaws in the current search state and execution cost is the estimated total cost of executing actions in the plan. Execution cost therefore represents plan quality. There are two weights on these two components, which allow trade-off between finding solutions quickly or searching for good quality solutions.

POPF2 [2] is a deterministic temporal planner that attempts to find minimum makespan plans. It uses a Simple Temporal Network (STN) to handle temporal constraints between actions and schedule them in a feasible way. It handles metric fluents, as for PDDL2.1, and therefore we believe it is a good candidate.

LPRPG [1] uses relaxed planning graph (RPG) heuristics combined with linear programming (LP) methods. It solves a number of LP for every decision it makes to calculate bounds on resources and to improve its numeric reasoning. Doing this gives LPRPG more precise information about bounds on resources than other planners and therefore is designed for use in domains with numeric resource.

2.3. Metric sensitivity test

Here we present results of the planners discussed in Section 2.2. Each planner is presented with the same problem, in eleven different versions, where each version is only different in the plan metric function to be minimised. An instance of a problem from the modified Driverlog domain, described in Section 2.1, is used. In order to show metric sensitivity, we expect the planners to generate different solutions. This can be achieved by using electric or diesel vehicles, depending on the cost of each of the resources (diesel and electricity). The plan metric function to minimise in each of the examples is ((10-i) * (fuel-used) + i * (electricity-used)), for i=1,2,...,10. Figure 2 presents the results. It is clear that the only planner which generates different results is LPG.

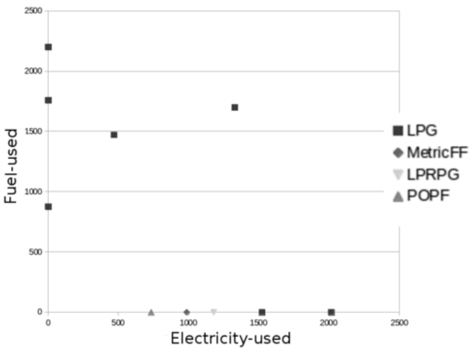


Figure 2. Results for running each of the planners 11 times on the same problem file, but with different plan metric function to minimise. For MetricFF, POPF2 and LPRPG marks on the figure represent 11 identical plans generated for 11 different plan metrics.

POPF2, LPRPG and MetricFF perform poorly in this test and do not exhibit metric sensitive behaviour. We attribute this fact to the RPG-based heuristic which prefers plans that achieve the goal in fewer action layers, instead of cheaper in terms of plan metrics.

LPG performed well by generating different plans for different plan metric function for the same problem. It is therefore our choice for later comparison.

POPF2 will produce the same plan for a given problem instance, regardless of the plan metric presented to it, since it does not respond to the plan metric. This does not seem a very promising planner to consider in the context of the current work, but we will show that its approach to optimising makespan can be harnessed to allow it to be metric sensitive in solving non-temporal problems.

3. A cost-based relaxed planning graph

Most current planners use Relaxed Planning Graph (RPG) as the base for their heuristics. This approach proved to provide a very good heuristic estimates efficiently for STRIPS domains. When dealing with numeric domains where the plan quality is evaluated using metric fluents RPG based approaches suffer from its bias towards shorter plans. Although for strictly length correlated plan metrics, this is not a problem as shorter plans are also plans of better quality, when faced with real life problems who are commonly monotonic or non-monotonic, this approach performs poorly. We propose a novel approach to constructing a relaxed planning graph based on cost. This novel approach aims at preferring cheaper, in terms of the cost, rather than shorter plans.

Figure 3 demonstrates how an example cost-RPG is constructed for the problem described in Figure 1 and metric function $(: metricminimize(+(*1(electricity - used))(+(*2(fuel-used))(*1(walked))))$ to minimise. Construction of cost-RPG layers is done in the following steps:

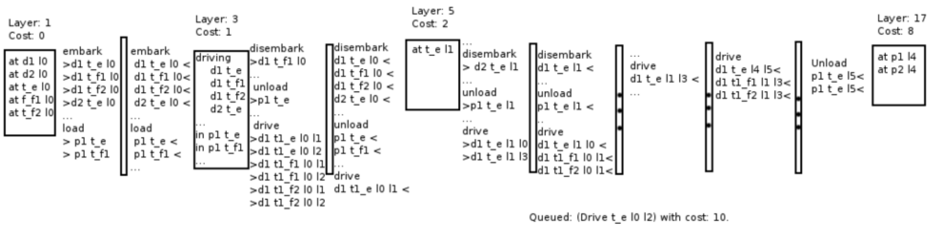


Figure 3. cost-RPG constructed from the initial state for the problem shown on Figure 1.

- Insert all facts from the initial state into Fact Layer 0.
- Then new action starts are applied, and their ends queued based on cost, until no new start action is applicable. Resulting layers are separated by ε.
- Then, lowest cost action ends are applied, this forms a new cost-RPG layer with the cost determined by the action cost.
- Then all new applicable action starts are applied.
- The process repeats until a goal is found or no new facts can be achieved.

Figure 3, demonstrates how cheap actions, in terms of the plan metrics, appear in cost-RPG and the expensive actions, in this case using fuel or driving via location 2, are postponed. It is interesting to notice that action driving to location 2 is not finished even when the goal is reached. Driving via location two is expensive and undesirable, yet the standard version of RPG uses it. In cost domains, when non-temporal action is split into start and end action, the start of an action contains no effects which can enable other actions. All positive effects, which can enable other actions, are placed at end. This ensures that the positive effects are available only after the cost is paid and, at the same time, prevents unnecessary branching.

For the purpose of handling context dependent action cost, in every state before creation of the cost-RPG, action costs are computed. These calculated action cost, within the context of the current state, are then used inside the cost-RPG for estimating the total cost of arriving to the goal state. The costs of single actions could change between different layers from cost-RPG but for simplicity we keep them constant from the time they have been calculated within the context of the state which is evaluated.

The basis of our use of POPF2 to achieve metric sensitivity lies in the way that it constructs the Temporal Relaxed Plan Graph (TRPG) [2]. The idea in the temporal setting, which is similar to the construction used in Sapa [3], is to extend the plan graph, during construction, with applicable action *start* points, queueing their end points to occur at the corresponding duration interval after the start. Once all applicable action starts have been identified and applied, arriving at a fixed point for this stage, the *earliest* queued action end point is applied and then the process is repeated, until this queue is empty (or the goals are achieved). Critically, each layer of the TRPG is labelled with the time at which it is reached. This means that the makespan of the relaxed plan can be identified directly within the TRPG and this leads to an informative temporal heuristic.

3.1. Compiling the cost-RPG

In order to exploit POPF2, without the need to modify it directly, we propose a novel compilation approach, in which non-temporal domains are compiled into 'temporal' do-

Table 1. An original action, and one of the actions resulting from a compilation.

Original Action	One of compiled temporal actions
(**:action** drive-electrictruck **:parameters** (?v - electrictruck ?f ?t - location ?d - driver) **:precondition** (and (at ?v ?f) (driving ?d ?v)(link ?f ?t)) **:effect** (and (not (at ?v ?f)) (at ?v ?t) (increase (electricity-used) (* (time-to-drive ?f ?t) (* (load ?t) (load ?t))))))	(**:durative-action** drive-electrictruck_l0-l2 **:parameters** (?v - electrictruck ?d - driver) **:duration** (= ?duration (* 100.0 (* (load ?t) (load ?t)) **:condition** (and (at start(at ?v s0)) (at start(driving ?d ?v)) (at start(link s0 s2))) **:effect**(and (at start(not (at ?v s0)))(at end (at ?v s2)) (at end (increase (electricity-used) (* 10.0 (* (load ?t) (load ?t)))))))

mains. Our method substitutes action 'durations' for action costs and, therefore, the makespan of the plan is a proxy for the plan cost. This compilation exploits the similarity between the construction of TRPG to the cost-based RPG. We now describe this process.

Faced with a non-temporal planning domain and problem instance with a single plan metric, we compile the problem into a temporal model, in which the durations are determined by the costs of the corresponding actions under the specific plan metric. The positive effects of the action are then placed at the end of the action, while the preconditions and negative effects are placed at the start.

There are several issues in the compilation. Firstly, we need to ensure that resource consumption, not to be confused with cost, occurs as soon as an action is applied, otherwise the gap between starting and ending the action can be exploited by the planner to try to execute an action that uses the same resource. Although the attempt will fail, because it will prevent the end of the original action from being applied, it will significantly impact on the search for a plan. We avoid this by ensuring that delete effects occur at the start, and add effects at the end. For numeric variables representing resources, we apply consumption effects (inhibiting) at the start and production effects (enabling) at the end. This is achieved by identifying the way in which numeric variables appear in preconditions of other actions. For example, in the Bread domain, increasing the metric fluent $(ready\text{-}dough\ ?k - kitchen)$ at the start allows actions to start baking bread before one of the $kneed\text{-}dough\text{-}machine$ or $kneed\text{-}hand$ action finishes. In this case we have to increase the $(ready\text{-}dough\ ?k)$ resource as an end effect.

It is usually necessary to translate an action from the metric domain into multiple actions in the temporal domain. One approach is to generate partially grounded versions of each action from the original domain and calculate their costs separately. In this case, we ground actions intelligently (checking static preconditions) to avoid unnecessary increase in the domain size. The compilation supports actions that do not have a constant cost and where cost depends on the context in which they are applied. Handling of this is delegated to underlying planner, which means that a planner must be able to handle context dependent action duration.

We now present an example of the compilation, consider the action presented in Table 1. For the aggregated plan metric function: $(: metricminimize(+(*10(electricity-used))(+(*1(fuel-used))(*1(walked))))$ This action is compiled into multiple durative actions. By grounding the location variables we can calculate the cost of the action given (time-to-drive ?f ?t), we ground the action for each value of (time-to-drive ?f ?t) given in the problem file. One of the translated actions is present in Table 1.

The duration of the action, 100 units multiplied by the load of the truck, is calculated using the aggregated plan metric function and grounded location functions. Knowing that we increase electricity-used by 10 units multiplied by the load of the truck, the plan metric function increases by: $10 \times (10 \times (load?t)^2) + 1 \times 0 + 1 \times 0 = 100 \times (load?t)^2$.

4. Experiments and results

In our experiments we will show that using the cost-based approach to build the RPG can generate better quality solutions, evaluated by the plan metrics, than the closest state of the art planner. Based on our earlier experiments with the current state of the art planners, discussed in Section 2.2, LPG is selected for comparison.

Each domain defines a set of metric fluents that combined, with different weights, form a plan metric in each problem instance. For each domain and each problem instance, we generate problem instances that combine three metric fluents with weightings chosen from $\{0, ..., 10\}$ so that the sum of weights is equal to 10. this generates sixty six different weightings. Each time a planner is run it is given a 100 second window to find a solution. After 100 seconds a planner is stopped whether it finds a solution or not. This repeats for all the weighting schemes. Time of the planners reported in Section 4 is total time for generating solutions to all of the sixty six problem instances, averaged over all problems.

The main focus of this work is increasing the quality of the solutions. The method used when comparing the performance of planners with each other is based on the scoring metric used in recent International Planning Competitions. After a set of planners is run on a problem instance derived from a multi-objective problem by combining the metric fluents into a single weighted sum, the plan metric, the solutions are gathered and compared with each other. For each weighting scheme we calculate the value of the best plan across all the planners, Θ_{best} and then we assign a score to each planner, i, producing a plan for this problem with value Θ_i, using the relative quality score:

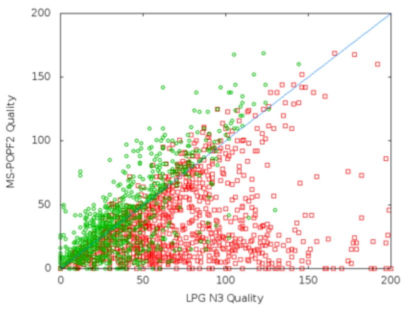
Definition 1 ***Relative Quality*** *is* $\Theta_i^{relative} = \frac{\Theta_i - \Theta_{best}}{\Theta_{best}}$
Where: Θ_i*- plan metric value;* Θ_{best}*- Best value across planners.*

This method is used to evaluate the quality of solutions for multiple runs over different plan metrics for each problem file for each domain described in Section 2.1.

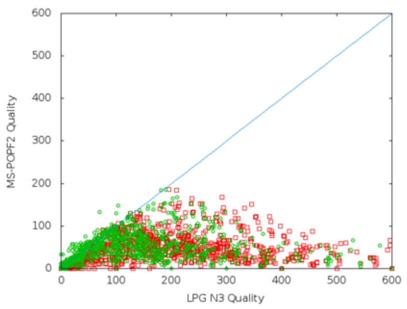
Results. Experiments were conducted on an Intel®Core^{TM}i7-2600 CPU @ 3.40GHz 8 machine with 4GB or RAM memory for each planner.

Table 2 A) shows results for 13 problems from the modified Driverlog domain. It is clear that the approach based on the cost-RPG outperforms current state of the art in terms of quality. This approach takes far more time, as we depicted by average time results in Table 2 B). This can be partly explained by time contributed towards the compilation of the domain, larger amount of action in the compiled domain and the performance of POPF2 without the compilation which is also slower than LPG.

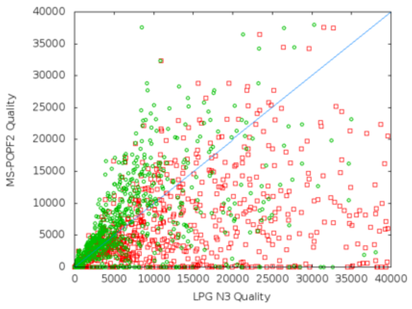
An interesting observation with regards to the role of stochasticity comes from the difference between the number of best plans found by each planner and their relative quality. This is especially visible for the Bread domain where LPG N3 generated more plans of higher quality, but its average relative quality is very low. We attribute this to its stochastic behaviour which in many cases leads to good quality solutions, but it also forces the planner to search in areas of the search space with poorer quality solutions.



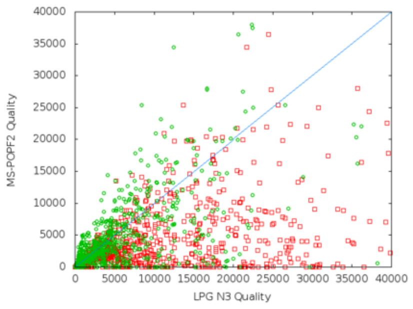
(a) Results for domain bread



(b) Results for domain production



(c) Results for domain driverlog



(d) Results for domain driverlog metric

Figure 4. Results for all weights and all problems by domain aggregated together. Squares compare quality of LPG n1 and circles LPG n2 with the quality of MS-POPF2. Lower cost values are better.

5. Conclusion

We have presented a novel cost-based expansion of RPG which is a new approach to cost oriented planning. The method is analogical to temporal RPG mentioned earlier.

The comparison of performance shows that although cost-RPG requires more time to find a solution, it finds a solution of higher quality than LPG. The comparison is limited to LPG because most planner do not handle context/metric dependent action costs.

The cost-RPG heuristic works particularly well in finding plans where the number of actions does not correlate well with the change of cost. In the example presented in Section 1 there are two routes, A: $L0 - L2 - L5$ and B: $L0 - L1 - L3 - L4 - L5$. It is common for planners to favour the shorter action sequence, but more expensive, route A. In constructing the cost-RPG, the end of the action (drive-electrictruck ?v L0 L2 ?d) would not be added to the plan graph until after the cheaper route via $L1$ is already in place.

Our results, presented in Section 4, show that use of the cost-RPG leads to a metric sensitive performance that generates plans of much higher quality than the closest competitor. We have shown that LPG, as other planners, favours shorter plans over cheaper in terms of plan metrics. Our novel cost-RPG heuristic overcomes this limitation. The heuristic have been successfully implemented using a current temporal planner, POPF2, and a novel compilation from metric to temporal domain.

Table 2. A) Average relative quality results for different plan metrics for each of the problems from the modified Driverlog domain. B) Average results for relative quality, time and number of best plans generated over 20 problem files for domain Production, Bread and 13 problems for Driverlog.

	LPG1	LPG2	LPG3	MSPOPF
1	292	117	0.24	**0**
2	725	336	1.85	**0.64**
3	1149	308	136	**0.02**
4	1367	466	458	**0.25**
5	205	196	175	**0.21**
6	8	0.69	**0.2**	2.39
7	745	500	3.13	**1**
8	2915	2594	448	**0**
9	668	632	1.24	**0.61**
10	3395	1572	526	**0.32**
11	7253	5149	2956	**0.34**
12	1996	3.97	136	**0.31**
13	7272	4530	2734	**0.4**
Avg	2153	1262	583	**0.5**

Quality	LPG1	LPG2	LPG3	MSPOPF
Production	12.52	7.34	6	**0.29**
Bread	8.94	2.75	1.32	**0.55**
Driver	2024	1059	443	**0.65**
Driver_m	1699	688	257	**2.11**
Time	LPG1	LPG2	LPG3	MSPOPF
Production	**6.23**	56.91	305.5	602
Bread	**5.14**	151.37	1254.71	22.6
Driver	**8.09**	13.88	64.27	262.47
Driver_m	**8.92**	13.92	63.92	439.5
# Best	LPG1	LPG2	LPG3	MSPOPF
Production	33	220	401	**966**
Bread	35	323	**707**	584
Driver	99	302	**554**	413
Driver_m	47	186	358	**436**
Total	214	1031	2020	**2399**

References

[1] A. Coles, M. Fox, D. Long, and A. Smith, ‘A hybrid relaxed planning graphlp heuristic for numeric planning domains’, *Proc. Int. Conf. on Automated Planning and Scheduling (ICAPS)*, (2008).

[2] A. J. Coles, A. I. Coles, M. Fox, and D. Long, ‘Forward-chaining partial-order planning’, in *Proc. Int. Conf. on Automated Planning and Scheduling (ICAPS)*, (2010).

[3] Minh Binh Do and S. Kambhampati, ‘Sapa: A multi-objective metric temporal planner’, *J. Artif. Intell. Res.*, **20**, 155–194, (2003).

[4] M. Fox and D. Long, ‘PDDL 2.1: An Extension to PDDL for Expressing Temporal Planning Domains’, *J. Artif. Int. Res.*, **20**, 61–124, (2003).

[5] A. Gerevini, A. Saetti, and I. Serina, ‘LPG-TD: a Fully Automated Planner for PDDL2.2 Domains’, *Proc. Int. Conf. on Automated Planning and Scheduling (ICAPS)*, (2004).

[6] J. Hoffmann and B. Nebel, ‘The FF Planning System: Fast Plan Generation Through Heuristic Search’, *J. Artif. Int. Res.*, **14**, 253–302, (2001).

[7] Emil Keyder and Hector Geffner, ‘”heuristics for planning with action costs revisited”’, in *ECAI*, pp. 588–592, (2008).

[8] D. Long and M. Fox, ‘The 3rd International Planning Competition: Results and analysis’, *J. Artif. Int. Res.*, **20**, 1–59, (2003).

[9] Nor Haizan Mohamed Radzi, *Multi-Objective Planning using Linear Programming*, Ph.D. dissertation, University of Strathclyde, 2011.

[10] S. Richter and M. Westphal, ‘The LAMA Planner: Guiding Cost-based Anytime Planning with Landmarks’, *J. Artif. Int. Res.*, **39**, 127–177, (2010).

[11] M. Sroka and D. Long, ‘Exploring Metric Sensitivity of Planners for Generation of Pareto Frontiers.’, in *Starting AI Research Symposium (STAIRS)*, pp. 306–317, (2012).