Incremental Contingency Planning for Recovering from Uncertain Outcomes

Yolanda E-Martín[1], María D. R-Moreno[2], and David E. Smith[3]

[1] Centre for Automation and Robotics, CSIC-UPM, 28500 Madrid, Spain
yolanda.e.martin@csic.es
[2] Universidad de Alcalá, Ctra Madrid-Barcelona Km 33.6, 28871 Madrid, Spain
mdolores@aut.uah.es
[3] NASA Ames Research Center, Moffett Field, CA 94035 USA
david.smith@nasa.gov

Abstract. Incremental Contingency Planning is a framework that considers all potential failures in a plan and attempts to avoid them by incrementally adding contingency branches to the plan in order to improve the overall probability. The planner focuses its attempts on the higher probability outcomes. Precautionary planning is a form of incremental contingency planning that takes advantage of the speed of replanning for easy contingencies and only considers the unrecoverable outcomes in the plan. In this work, we present an approach to incrementally generating contingency branches to deal with uncertain outcomes. The main idea is to first generate a high probability non-branching seed plan, which is then augmented with contingency branches to handle the most critical outcomes. Any remaining outcomes are handled by runtime replanning.

1 Introduction

Incremental Contingency Planning (ICP) is a framework that considers all potential failures in a plan and attempts to avoid them by incrementally adding contingency branches to the plan in order to improve the overall probability [2]. The planner focuses its attempts on the higher probability outcomes. Precautionary planning [4] is a form of ICP that takes advantage of the speed of replanning for easy contingencies and only considers the unrecoverable outcomes in the plan. In this work, we present an approach to incrementally generating contingency branches to deal with uncertain outcomes. The main idea is to first generate a high probability non-branching seed plan, which is then augmented with contingency branches to handle the most critical outcomes. Any remaining outcomes are handled by runtime replanning. For the most critical outcomes, an attempt will be made to improve the chances of recovery by (1) finding a new plan that avoids or reduces the probability of getting to that outcome, (2) adding precautionary steps that allow recovery, if the failure occurs, or (3) adding a conformant solution that achieves the goal by using a different path. All three strategies can increase the overall probability of the plan. The process is repeated until (1) the resulting contingent plan achieves at least a given probability threshold, (2) the available time is exhausted, or (3) a certain number of branches are added. By critical outcomes, we mean those that are both likely and have poor chances of recovery.

Our incremental approach starts with a high probability seed plan that we generate using PIPSS^I [3]. Section 2 defines the heuristic function used to identify points of failure that potentially improve the total probability of the plan. Section 3 details the different techniques we can apply to improve the chances of recovery, if the failure occurs. Section 4 presents an empirical evaluation. Section 5 discusses the drawbacks of our work and outlines some future work.

2 Recognizing outcomes

Once a non-branching seed plan has been generated, we analyze all its potential unexpected outcomes to estimate how much utility could be gained by improving the chances of recovery for that outcome. We call this estimation *Gain* and it is the maximum probability that the plan could potentially be improved by precautionary planning. This measure is based on the Completion Probability Estimate (CPE)[4]. The CPE is computed by propagating probability and Interaction information through a plan graph [3] for a given state. This information is used to find relaxed plans, which then provide an estimate of the probability of reaching the goal from that state. To illustrate, consider the probabilistic planning problem shown in Figure 1, where there is a package *pkg* and a truck *trk* at location *a*, and the package needs to be delivered to location *c*. The truck can move between different locations, and it may have a flat tire during the trip with 0.4 probability. Locations *a* and *d* have a spare tire.

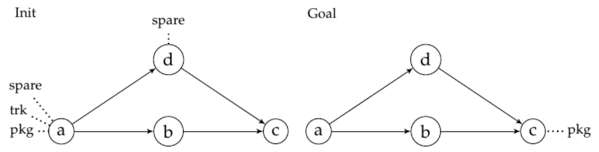


Fig. 1. Initial and goal states for a simple probabilistic Logistics problem.

Figure 2 shows the non-branching seed plan generated by the planner using all-outcomes determinization [7]. Action (drive trk a b) has an alternative outcome o_1 with probability 0.4 and CPE = 0. This means that there is no chance of completing the objective if this outcome actually happens – the tire goes flat and the truck cannot reach the goal. Action (drive trk b c) has an alternative outcome, o_2, with probability 0.4 and CPE = 1 because even though the tire goes flat, the truck still arrives at location c, and the remainder of the plan succeeds. For an alternative outcome (or branch) x of action a, the optimistic possible gain from precautionary planning will be the difference between the *estimated reward with repair* and the *estimated reward without repair*. We compute the latter using the CPE estimation. That is, the probability of reaching the goal

[4] CPE is essentially the same as CCE [3], but expressed in terms of probability instead of in terms of cost.

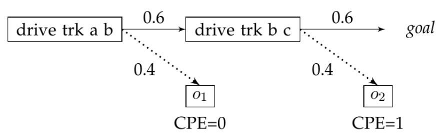


Fig. 2. Example of a non-branching seed plan with potential outcomes to be repaired

from that state. On the other hand, to compute the *estimated reward with repair*, we propagate probability and Interaction in the plan graph only considering the outcome x, but allowing other actions in the plan to change. By doing this, we force x to be in the plan and, therefore, the new probability and Interaction information can be used to construct a relaxed plan. We call this estimation *Optimistic Probability Estimation* (OPE). More formally, for a branch x, the gain is a measure of how much the total plan probability could potentially be increased by precautionary planning and is computed by the difference between the OPE of branch x and the CPE of x:

$$\mathrm{Gain}(x) = \mathrm{OPE}(x) - \mathrm{CPE}(x) \tag{1}$$

Following the previous example, for branches o_1 and o_2, the gains from precautionary planning are:

$$\mathrm{Gain}(o_1) = \mathrm{OPE}(o_1) - \mathrm{CPE}(o_1) = 0.36 - 0 = 0.36$$
$$\mathrm{Gain}(o_2) = \mathrm{OPE}(o_2) - \mathrm{CPE}(o_2) = 0.36 - 1 = -0.64$$

This means that by repairing branch o_1, the total plan probability will improve more than through branch o_2. Therefore, we would prefer to recover o_1 since it seems that it is possible to gain more probability mass, and o_2 might be recoverable by using runtime replanning.

The calculation of gains allows us to create a ranking to start the recovery of alternative paths.

3 Repairing outcomes

Given the recognized undesirable outcomes ranking, the next step is to repair the plan in order to increase the overall probability of success. For each outcome, the idea is to look for the best improvement. In the next subsections, we present three methods to do that. The first method is called *Confrontation*, which tries to find a plan that avoids the problematic action's outcome when its execution depends on a condition. The second method is called *Precautionary Steps*, which adds precautionary actions before the problematic action to increase the probability of recovery in case it happens. The third method is called *Conformant Augmentation*, which increases the total probability by adding conformant steps to the contingency plan solution.

3.1 Confrontation

A probabilistic outcome of an action may be subject to different conditions. In our example, it might be that for the action (unload pkg trk c), proposition ¬(at c pkg) occurs when, for instance, the store in c where the package needs to be delivered is closed. Confrontation on this condition will avoid ¬(at c pkg) by ensuring that the store is open before the start of driving.

The idea is to find a new plan that avoids or reduces the probability of getting to that branch, and then replace the old seed plan with the new plan. More precisely, suppose that a is the action in the seed plan with an unrecoverable outcome conditioned by c. We force the planner to find a new seed plan that achieves $\neg c$ to prevent the failure from occurring. The way we do that is by creating a new version of the action a, *a'*, that keeps its original preconditions plus a new additional precondition $\neg c$, and its original effects plus an additional unique effect. The unique effect is also added to the set of goals. We then add the new action to the set of operators and call the deterministic planner to find a plan for the goals. If a new plan is found and it has higher probability than the old seed plan, the new plan replaces the old seed plan.

3.2 Precautionary Steps

Precautionary Steps consists of repairing an undesirable action's outcome by adding precautionary actions to the plan before the problematic action. For example, picking up a spare tire before driving in case you have a flat tire. This method improves the chance of recovery, if the seed plan fails, and makes it possible to reach the goal when the unexpected outcome of the problematic action happens during runtime. The idea is to force the planner to find a plan that uses each alternative outcome, but does not lose any precondition needed to reach the goal when the action has the desired outcome. In other words, for each alternative outcome o of action a, the method:

1. Divides the initial seed plan into two parts: a *prefix*, which contains all actions preceding a, and a *suffix*, which contains all actions following a.
2. Creates a new action a' that keeps its original preconditions and effects except for the new predicate (unique-effect), which is added to its effects.
3. Analyzes the causal structure of the suffix to collect all the preconditions needed by the suffix, and adds them to the set of preconditions of a'.
4. Adds the predicate (unique-effect) to the goal state to force a' into the plan.
5. Adds a' to the set of operators and calls the deterministic planner to find a plan for the new goal state. If a plan is found and the overall probability of the plan is higher, the prefix is replaced with the prefix of the new plan and the suffix is added to it.

3.3 Conformant Plan

It is possible that there are several plans that reach the goal, which are not initially generated because they have lower probability. In some cases, one or more of these plans may be executable with the original seed plan and will raise the probability of the plan. Conformant plans may happen when the Precautionary Steps method is applied. This is the case when the plan that is generated contains action *a'* (the one forced to be in the plan), but it is only in the plan to achieve the unique effect.

4 Experimental evaluation

We conducted an experiment on IPPC-06 and IPPC-08 fully observable probabilistic planning domains, as well as on the *probabilistically interesting domains* (PID) [5]. The test consists of running the planner and using the resulting plan in the MDP Simulator [10]. The planner and the simulator communicate by exchanging messages. The simulator first sends the planner the initial state. Then, the interaction between planner and simulation consists of the planner sending an action and the simulator sending the next state to the planner.

The planners used for this test were FPG [1], FF-Replan [7], FHH [8], FHH$^+$ [9], and RFF [6]. We compare these with two variants of our planner:

- PIPSS^I_r [3], which generates a high-probability non branching seed plan and does runtime replanning to deal with unexpected states at execution time.
- C-PIPSS^I_r, a modified PIPSS^I_r planner that incrementally augments the plan solution using confrontation, precautionary steps, and conformant augmentation. It does runtime replanning to deal with unexpected states at execution time.

The experiments were conducted on a 2.4 GHz Pentium dual core processor. For the rest of the planners, given that we were not able to obtain and run them ourselves, data were collected from work done by Yoon et al. [9]. For all the planners, 30 trials per problem were performed with a total time limit of 30 minutes for the 30 trials. There are 15 problems for each domain, except for the 2-Tireworld domain that has 10, and the PID domains that have one each. Therefore, the maximum number of successful rounds for each domain is $15 \times 30 = 450$, $10 \times 30 = 300$, and 30 respectively.

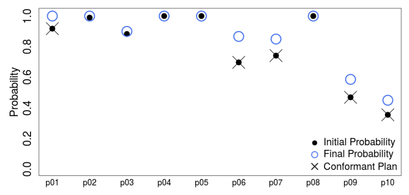
Table 1 (left panel) shows the number of successful rounds for each planner in each IPPC-06 domain. C-PIPSS^I_r gets good results in two of the three domains. The highest success rates are obtained in Exploding-Blocks and Tireworld domains. In fact, C-PIPSS^I_r is the planner that achieves the highest rate in the Exploding-Blocks domain. We expected that C-PIPSS^I_r would perform better than PIPSS^I_r. However, in the Elevator domain, C-PIPSS^I_r performs much poorer than PIPSS^I_r, and it is only slightly better in the Tireworld domain. Figure 3 shows data on the plan solution after applying ICP to the initial non-branching seed plan. The left column presents a plot for each domain in the IPPC-06 that shows the increase in probability after repairing the plan outcomes, and if a conformant branch has been added on the plan solution. The right column presents a scatter plot for each domain in the IPPC-06, where each dot in the plot represents the relationship between the *total number of outcomes in the plan* and the number of *recoverable outcomes*, and the *total number of outcomes in the plan* and the number of *unrecoverable outcomes*.

Figure 3(a) shows that for the Exploding-Blocks domain the overall probability of the plan increases in all the problems. The reason is the high number of recoverable outcomes, which is shown in Figure 3(b). Therefore, the performance of C-PIPSS^I_r is better than the performance of PIPSS^I_r. Figure 3(c) shows that for the Elevator domain the overall probability of the plan does not increase

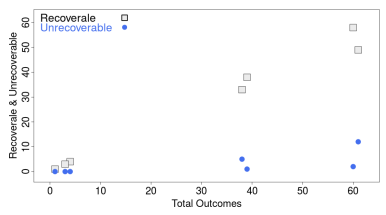
Table 1. Total number of successful rounds on the IPPC-06 and IPPC-08 using ICP.

	Planners				
IPPC-06	FFH	FFH$^+$	FPG	PIPSS$_r^I$	C-PIPSS$_r^I$
Exploding-Blocks	205	**265**	193	239	262
Elevators	214	292	342	**396**	382
Tireworld	343	**364**	337	360	356
Total	762	921	872	995	**1000**

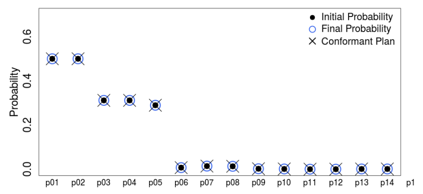
	Planners				
IPPC-08	FFH	FFH$^+$	RFF	PIPSS$_r^I$	C-PIPSS$_r^I$
Exploding-Blocks	131	**214**	58	171	176
2-Tireworld	**420**	**420**	382	21	68
Total	551	**634**	440	192	244



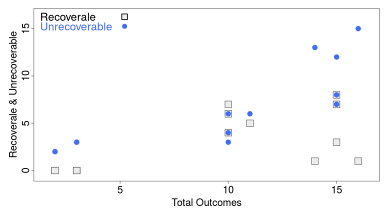
(a) Probability difference in the Exploding-Blocks Domain



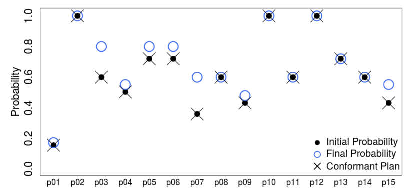
(b) Recoverable and unrecoverable outcomes in the Exploding-Blocks Domain



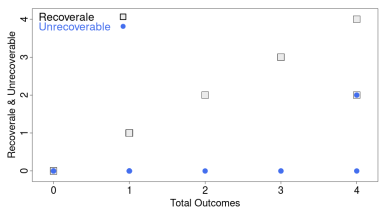
(c) Probability difference in the Elevator Domain



(d) Recoverable and unrecoverable outcomes in the Elevator Domain



(e) Probability difference in the Tireworld Domain



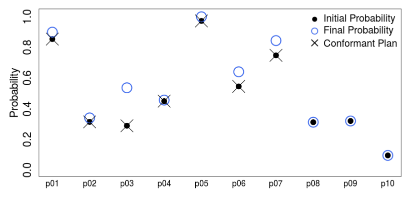
(f) Recoverable and unrecoverable outcomes in the Tireworld Domain

Fig. 3. Comparison between initial and final plan probability (left column), and recoverable and unrecoverable plan outcomes (right column) of IPPC-06.

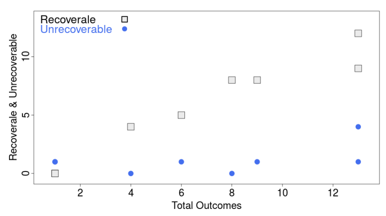
in any of the problems, but a conformant plan is added to each of them. Figure 3(d) shows that the number of unrecoverable outcomes is higher than the number of recoverable outcomes. This is why the overall probability does not increase. Although each problem has a conformant plan added, this additional plan has lower probability than the initial plan. Therefore, the chances of failure during execution are higher. However, the success rate of C-PIPSS$_r^I$ is higher than FFH and close to FFH$^+$. Figure 3(e) shows that for the Tireworld domain the overall probability of the plan increases in half of the problems, and a conformant plan is added to each of them. Figure 3(d) shows that the number of recoverable outcomes is higher than the number of unrecoverable outcomes. However, C-PIPSS$_r^I$ performs just a bit better than PIPSS$_r^I$, where we expected better performance. The success rate of C-PIPSS$_r^I$ is higher than FFH and FPG. FFH$^+$ performs slightly better.

Table 1 (right panel) shows the number of successful rounds for each planner in each IPPC-08 domain. C-PIPSS$_r^I$ has a lower success rate than we expected. For the Exploding-Blocks domain, even though from Figure 4(a) we can see the overall probability increase for some of the problems, the success rate of C-PIPSS$_r^I$ does not improve. For the 2-Tireworld domain, the success rate of C-PIPSS$_r^I$ increases considerably compare to PIPSS$_r^I$, but it is still very low.

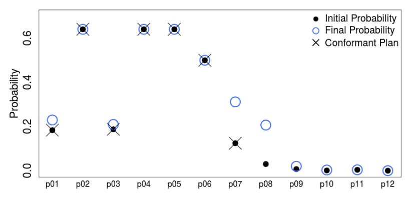
Table 2 shows the number of successful rounds for each planner in each PID domain. Figure 5 shows that surprisingly there is no improvement in the overall probability for any of the tested domains. PIPSS$_r^I$, and C-PIPSS$_r^I$ have relatively high success rates in the Climb and River domains. However, we expected some improvement in the Tire1 and Tire10 domains after ICP was applied.



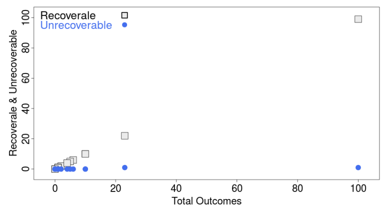
(a) Probability difference in the Exploding-Blocks Domain



(b) Recoverable and unrecoverable outcomes in the Exploding-Blocks Domain



(c) Probability difference in the 2-Tireworld Domain



(d) Recoverable and unrecoverable outcomes in the 2-Tireworld Domain

Fig. 4. Comparison between initial and final plan probability, and recoverable and unrecoverable plan outcomes of IPPC-08.

5 Discussion and Conclusions

This work goes beyond what Foss [4] did by computing a high-probability seed plan and a *Gain* value that evaluates which outcomes will improve the overall

Table 2. Total number of successful rounds on the PID using ICP.

	Planners					
Domains	FF-Replan	FFH	FFH^{+}	FPG	PIPSS$_r^I$	C-PIPSS$_r^I$
Climb	19	**30**	**30**	**30**	**30**	**30**
River	20	20	20	20	**23**	20
Tire1	15	**30**	**30**	**30**	21	19
Tire10	0	6	**30**	0	0	0
Total	54	86	**110**	80	74	69

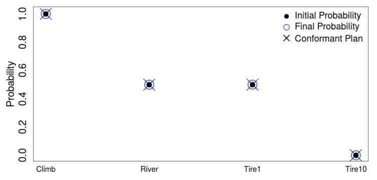


Fig. 5. Probability difference on the PID.

seed plan probability. In addition, we included the *Confrontation* technique to repair outcomes subject to a condition. In general, ICP provides little additional benefit using all-outcomes determinization for finding the seed plan. In a few domains, Incremental Precautionary Planning can help; the success rates are higher, which means that the planner has been able to reach the goal in a larger percentage of problems. However, we expected that the combination of Incremental Precautionary Planning and runtime replanning would increase the success rate for all the tested domains. Our hypothesis for the poor performance of our framework is the classical all-outcomes determinization approach.

Our approach consists of generating a deterministic planning domain from a probabilistic planning domain. The probability information in the domain description is used in a heuristic function that propagates probability and Interaction information through a plan graph. This heuristic estimator is used to guide the search toward high probability non-branching seed plans. The resulting plans are then analyzed to find potential points of failure that can be identified as recoverable or unrecoverable. Recoverable failures will be left in the plan and will be repaired through replanning at execution time. For each unrecoverable failure, we attempt to incrementally improve the chances of recovery by applying confrontation, adding precautionary steps, and adding conformant plans. The final plan is a contingency plan that has a highest probability of success during execution time. However, we observed that the probability and Interaction information underestimates the actual probability of propositions and actions in the plan graph, and therefore, the probability of each state in the search space. To illustrate, consider the simple probabilistic Logistics domain defined in Figure 1. Assume we use all-outcomes determinization, then the probabilistic domain results in a deterministic domain with two deterministic actions created from the probabilistic action *drive*. Figure 6 shows that determinization process. The most likely outcome of the action implies that the car successfully drives between locations with probability 0.6. This results in action *drive-1*. For the other outcome, the car achieves the destination, but it gets a flat tire with a probability of 0.4. This results in action *drive-2*.

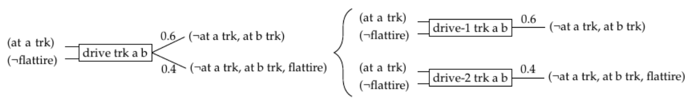


Fig. 6. Example of all-outcomes determinization of a probabilistic action.

Figure 7 shows the plan solution that is generated by our planner. If we compute the probability of the resulting state after performing (drive-1 trk a b), the probability of (at b trk) and ¬(flattire) is 0.6. As a result, the probability of performing (drive b c) is equal to the product of the probability of its preconditions. That is, $pr(\text{at b trk})\, pr(\neg\text{flattire}) = 0.6(0.6) = 0.36$. As a consequence, the probability of (at c trk) is the product of the probability of performing (drive b c) and the probability of achieving (at c trk) through (drive b c). That is, $0.36(0.6) = 0.216$. This means that the probability of success of the plan is 0.216. However, both outcomes of the probabilistic action (drive trk a b) result in the car at the next location. This means that, the probability of achieving (at b trk) is dependent on the outcome. The all-outcomes determinization technique does not consider the dependence between propositions and outcomes since it does not consider the overall probability of those propositions that are common in all the outcomes. Therefore, by considering propositions individually instead of as a result of an action's outcome, we can compute more accurate probability estimates.

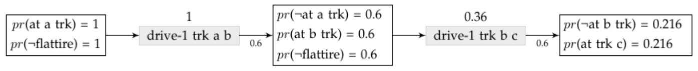


Fig. 7. Plan solution probability using Determinization.

To illustrate, consider the example in Figure 8 that shows the same plan solution as before. In this case, the probability for each state is computed by considering probabilistic actions and propositions individually. Therefore, after (drive trk a b) is performed, there is a probabilistic state where ¬(at a trk) and (at b trk) have a probability of 1 – since both propositions are in both outcomes of the action; ¬(flattire) has a probability of 0.6 and (flattire) has a probability of 0.4. Consequently, the probability of (drive trk b c) is 0.6, instead of 0.36 for the all-outcomes determinization. This results in (at c trk) having a probability of 0.6. Consequently, the probability of success of the plan is 0.6 where before it was 0.216. The underestimation that is caused by all-outcomes determinization is, therefore, harmful to our probability propagation, yielding seed plans that do not have high probability of success. To illustrate this, consider the simple probabilistic Logistic problem in Figure 1. It has two possible paths that achieve the goal (1) driving from a to b and from b to c, which has a probability of 0.6

of reaching the goal, and (2) driving from a to d and from d to c, which has a probability of 1 of reaching the goal since location d has a spare tire – if the truck got a flat tire in d, it would be able to change the tire and successfully continuing the drive. This means that we start off with the wrong seed plan and, therefore, do not recover. For this reason, we are working on a new approach that computes estimates of probability, which consider the overall probability of each proposition across all of the action's outcomes, and the dependence between those propositions. These estimates will generate high probability seed plans.

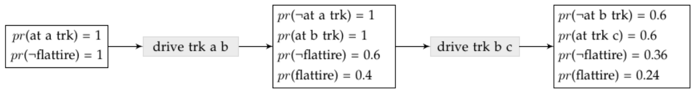


Fig. 8. Plan solution probability considering the overall probability of propositions across action outcomes.

Acknowledgments

This work was supported by the NASA Safe Autonomous Systems Operations (SASO) project, the MINECO project EphemeCH TIN2014-56494-C4-4-P, and UAH project 2015/00297/001.

References

1. O. Buffet and D. Aberdeen. The factored policy-gradient planner. volume 173, pages 722–747, 2009.
2. R. Dearden, N. Meuleau, S. Ramakrishnan, D. E. Smith, and R. Washington. Incremental contingency planning. In *Proc. of ICAPS-03 Workshop on Planning under Uncertainty*, Trento, Italy, 2003.
3. Y. E-Martín, M. D. R-Moreno, and D. E. Smith. Progressive heuristic search for probabilistic planing based on interaction estimates. *Expert Systems*, 31(5):421–436, 2014.
4. J. Foss, N. Onder, and D. E. Smith. Preventing unrecoverable failures through precautionary planning. In *Proc. of the ICAPS'07 Workshop on Moving Planning and Scheduling Systems into the Real World*, Providence, RI, USA, 2007.
5. I. Little and S. Thiébaux. Probabilistic planning vs replanning. In *Proc. of the ICAPS'07 Workshop on Planning Competitions*, Providence, RI, USA, 2007.
6. F. Teichteil-Königsbuch, U. Kuter, and G. Infantes. Incremental plan aggregation for generating policies in mdps. In *Proc. of AAMAS*, Toronto, Canada, 2010.
7. S. Yoon, A. Fern, and R. Givan. Ff-replan: a baseline for probabilistic planning. In *Proc. of the International Conference on Automated Planning and Scheduling*, Providence, RI, USA, 2007.
8. S. Yoon, A. Fern, R. Givan, and S. Kambhampati. Probabilistic planning via determinization in hindsight. In *Proc. of AAAI*, Chicago, IL, USA, 2008.
9. S. Yoon, W. Ruml, J. Benton, and M. Do. Improving determinization in hindsight for on-line probabilistic planning. In *Proc. of ICAPS*, Toronto, Ontario, Canada, 2010.
10. H. L. S. Younes, M. L. Littman, D. Weissman, and J. Asmuth. The first probabilistic track of the international planning competition. *JAIR*, 24:841–887, 2005.