A Theory of Intra-Agent Replanning

Kartik Talamadupula[†] and **David E. Smith**[§] and **William Cushing**[†] and **Subbarao Kambhampati**[†]

[†]**Dept. of Computer Science and Eng.**
Arizona State University
Tempe, AZ 85287
{krt,rao} @ asu.edu, william.cushing @ gmail.com

[§]**NASA Ames Research Center**
Moffet Field
CA 94035
david.smith @ nasa.gov

Abstract

When autonomous agents execute in the real world, the world state as well as the objectives may change from the agent's original conception of those things. In such cases, the agent's planning process must modify the existing plan to make it amenable to the new conditions, and to resume execution. The need for inter-agent replanning, in terms of commitments to other agents, is understood in the multi-agent systems community. Such inter-agent replanning also motivates an intra-agent replanning problem for each individual agent. However, the single-agent planning community has mostly limited its view of replanning to reducing the computational effort involved, by minimally perturbing the current plan structure to replan. This is not very appropriate as a general model for intra-agent replanning, which may consist of various techniques that are employed according to the scenario at hand. In this paper, we present a general replanning problem that is built on various types of replanning constraints. We show that these constraints can model different types of replanning, including the similarity-based approaches used in the past and sensitivity to commitments made to other agents. Finally, we show that partial satisfaction planning provides a good substrate for modeling this general replanning problem.

1 Introduction

Many tasks require handling dynamic objectives and environments. Such tasks are characterized by the presence of highly complex, incomplete, and sometimes inaccurate specifications of the world state, the problem objectives and even the model of the domain dynamics. These discrepancies may come up due to factors like plan executives or other agents that are executing their own plans in the world. Due to this divergence, even the most sophisticated planning algorithms will eventually fail unless they offer some kind of support for replanning. These dynamic scenarios are non-trivial to handle even when planning for a single agent, but the introduction of multiple agents – automated or otherwise – introduces further complications. All these agents necessarily operate in the same world, and the decisions made and actions taken by an agent may change that world for all the other agents as well. Moreover, the various agents' published plans may introduce commitments between them, due to shared resources, goals or circumstances. The need for *inter-agent replanning* in terms of these commitments is understood in the multi-agent systems (MAS) community (c.f. Section 2). However, these inter-agent commitments may evolve as the world itself changes, and may in turn affect a single agent's internal planning process.

Given the importance of replanning in dealing with all these issues, one might assume that the single-agent planning community has studied the issues involved in depth. This is particularly important given the difference between agency and execution, and the real-world effectors of those faculties: a single agent need not necessarily limit itself to planning just for itself, but can generate plans that are carried out by multiple executors in the world. Unfortunately, most previous work in the single-agent planning community has looked upon replanning as a *technique* whose goal is to reduce the computational effort required in coming up with a new plan, given changes to the world. The focus in such work is to use the technique of minimally perturbing the current plan structure as a solution to the replanning problem. However, neither reducing replanning computation nor focusing on minimal perturbation are appropriate techniques for intra-agent replanning in the context of multi-agent scenarios.

In this work, we argue for a better, more general, model of the replanning problem. This model considers the central components of a planning problem – the initial state, the set of goals to be achieved, and the plan that does that, along with *constraints* imposed by the execution of that plan in the world – in creating the new replan. These replanning constraints take the form of commitments for an agent, either to an earlier plan and its constituent actions, or to other agents in its world. We will show that this general commitment sensitive planning architecture subsumes past replanning techniques that are only interested in minimal perturbation – the "commitment" in such cases is to the structure of the previously executing plan. We will also show that partial satisfaction planning (PSP) techniques provide a good substrate for this general model of replanning.

In the next section, we discuss some prior and related work from the multi-agent systems (MAS) and single-agent planning communities in order to motivate our work. We then present our formulation of the replanning problem in terms of the problem instance (composed of the initial state and the goals), the plan to solve that particular instance, and the dependencies or constraints that are introduced into the world by that plan, and three models associated with the handling of these *replanning constraints* that are defined in that formulation. Subsequently, we delve deeper into the compo-

sition of those constraints, and discuss the various solution techniques that can be used to satisfy these constraints while synthesizing a new replan. We then describe our experimental setup and present our results.

2 Related Work

Replanning has been an early and integral part of automated planning and problem solving work in AI. The STRIPS robot problem-solving system (Fikes, Hart, and Nilsson 1972), one of the earliest applications of planning and AI, used an execution monitoring system known as PLANEX to recognize plan failures in the world, and replan if direct re-execution was not an option. The replanning mechanism worked by sending the change in state back to the STRIPS system, which returned a sequence of actions that brought the state back to one from which the execution of the original plan could be resumed. This relatively simple procedure encoded the idea that would come to dominate replanning work within the planning community for the next few decades – the notion of *commitment to a plan*. The principle underlying the concept of minimally changing an existing plan is christened plan *stability* by Fox et al. (Fox et al. 2006). In that work, two approaches – replanning from scratch, and repairing the existing plan – and their respective impacts on plan stability are considered. Stability itself is defined as the measure of the difference a process induces between an original plan and a new plan, and is closely related to the idea of *minimal perturbation planning* (Kambhampati 1990) used in past replanning and plan re-use (Nebel and Koehler 1995) work. Fox et al. argue that plan stability as a property is desirable both from the standpoint of measurable quantities like plan generation time and plan quality, as well as intangibles like the cognitive load on human observers of planned activity and the strain on the plan executive.

Other work on replanning has taken a strong stand either for or against the idea of plan repair. Van Der Krogt et al. (Van Der Krogt and De Weerdt 2005) fall firmly into the former category, as they outline a way to extend state-of-the-art planning techniques to accommodate plan repair. For the purposes of this paper, it suffices to note that this work looks at the replanning problem as one of commitment to and maintenance of a broken plan. This work has a strong parallel (and precursor) in planning for autonomous space exploration vehicles, a proven real world application of planning technology. The Casper system (Knight et al. 2001), which was designed to autonomously control a spacecraft and its activities, was designed as a system with a high level of responsiveness, enabled through a technique called *iterative repair* – an approach that fixes flaws in an existing plan repeatedly until an acceptable plan is found. At the other end of the spectrum, Fritz et al. (Fritz and McIlraith 2007) deal with changes to the state of the world by replanning from scratch. Their approach provides execution monitoring capabilities by formalizing notions of plan validity and optimality using the situation calculus; prior to execution, each step in the (optimal) plan is annotated with conditions that are sufficient for the plan's optimality to hold. When a discrepancy or unexpected change occurs during execution, these conditions are re-evaluated in order to determine the optimality of the executing plan. When one of the conditions is violated, the proposed solution is to come up with a completely new plan that satisfies the optimality (or validity) conditions.

In contrast, the MAS community has looked at replanning issues more in terms of multiple agents and the conflicts that can arise between these agents when they are executing in the same dynamic world. Wagner et al. (Wagner et al. 1999) proposed the twin ideas of *inter-agent* and *intra-agent* conflict resolution. In the former, agents exchange commitments between each other in order to do team work. These commitments in turn may affect an agent's local controller, and the feasibility of the agent's individual plan – this brings up the process of intra-agent conflict resolution. Inter-agent commitments have been variously formalized in different work in the MAS community (Komenda et al. 2008; Bartold and Durfee 2003; Wooldridge 2000), but the focus has always been on the interactions between the various agents, and how changes to the world affect the declared commitments. The impact that these changes have *within* an agent's internal planning process has not received significant study. The closest work in the multi-agent planning community to ours is by (Komenda, Novák, and Pěchouček 2012), where the multi-agent plan repair problem is introduced and reduced to the multi-agent planning problem; and (Meneguzzi, Telang, and Singh 2013), where a first-order representation and reasoning technique for modeling commitments is introduced.

In this work, we propose to bring these two approaches from two different communities – single-agent planning, and multi-agent systems – together in a unified theory of agent replanning. Our central argument is that it should be the single-agent planning community's brief to heed the changes to the world state and inter-agent commitments, and to generate a new (single-agent) plan that remains consistent with the larger multi-agent commitments in the world. The first step in this endeavor is to re-define the replanning problem such that both single and multi-agent commitments can be represented under a unified framework.

3 The Replanning Problem

We posit that replanning should be viewed not as a technique, but as a *problem* in its own right – one that is distinct from the classical planning problem. Formally, this idea can be stated as follows. Consider a plan Π_P that is synthesized in order to solve the planning problem $P = \langle I, G \rangle$, where I is the initial state and G, the goal description. The world then changes such that we now have to solve the problem $P' = \langle I', G' \rangle$, where I' represents the changed state of the world, and G' a changed set of goals (possibly different from G). We then define the *replanning problem* as one of finding a new plan Π'_P that solves the problem P' subject to a set of constraints ψ^{Π_P}. This model is depicted in Figure 1. The composition of the constraint set ψ^{Π_P}, and the way it is handled, can be described in terms of specific *models* of this newly formulated replanning problem. Here, we present three such models based on the manner in which the set ψ^{Π_P} is populated.

1. M_1 | **Replanning as Restart**: This model treats replanning as 'planning from restart' – i.e., given changes in the world $P = \langle I, G \rangle \rightarrow \ P' = \langle I', G' \rangle$, the old plan Π_P is

completely abandoned in favor of a new plan Π'_P which solves P'. Thus the previous plan induces no constraints that must be respected, meaning that the set ψ^{Π_P} is empty.

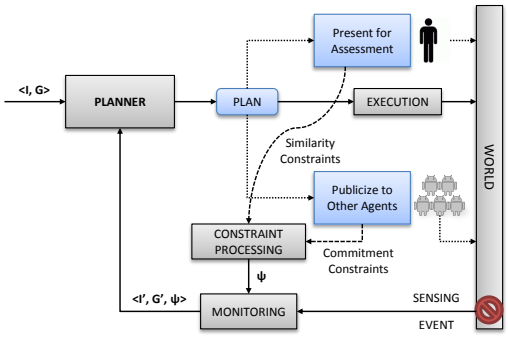

Figure 1: A model of replanning

2. M_2 | **Replanning to Reduce Computation**: When the state of the world forces a change from a plan Π_P to a new one Π'_P, in the extreme case, Π'_P may bear no relation to Π_P. However, it is most desirable that the cost of comparing the differences between the two plans with respect to execution in the world be reduced as far as possible. The problem of minimizing this cost can be re-cast as one of minimizing the differences between the two plans Π'_P and Π_P using *syntactic constraints* on the form of the new plan. These syntactic constraints are added to the set ψ^{Π_P}.
3. M_3 | **Replanning for Multi-agent Scenarios**: In many real world scenarios, there are multiple agents $A_1 \ldots A_n$ that share an environment and hence a world state.[1] The individual plans of these agents, $\Pi_1 \ldots \Pi_n$ respectively, affect the common world state that the agents share and must plan in. This leads to the formation of dependencies, or *commitments*, by other agents on an agent's plan. These commitments can be seen as special types of constraints that are induced by an executing plan, and that must be obeyed when creating a new plan as a result of replanning. The aggregation of these commitments forms the set ψ^{Π_P} for this model.

In the following section, we explore the composition of the constraint set ψ^{Π} (for any given plan Π) in more detail. First, however, we consider a real world application scenario and the application of the three replanning models described above to it, in order to illustrate that these models are broad enough to capture the various kinds of replanning techniques.

3.1 Example: Planetary Rovers

Planning for planetary rovers is a scenario that serves as a great overarching application domain for describing the motivations behind the various models of replanning that we propose in this work. Automating the planning process is central to this application for three reasons: (1) the complex checks and procedures that are part of large-scale or critical applications can often only be fully and correctly satisfied by automation; (2) there are limited communication opportunities between the rover and and the control station; and (3) the distances involved rule out immediate tele-operation, since there is a considerable communication lag between a rover operating on the surface of a distant planet and the control center.

1. M_1: This model is frequently used by planning algorithms that create path and motion plans for the rover's operation. Often, changes to the environment (e.g. the detection of an obstacle such as a rock ahead) will render the currently executing plan useless; in cases where the system needs to react immediately and produce a new plan, creating a completely new plan works better than trying to salvage some version of an existing plan.
2. M_2: In the case of planetary rovers, both computational and cognitive costs are present when it comes to comparing Π and Π'. Changes to an executing plan Π must pass muster with human mission controllers on Earth as well as mechanical and electrical checks on-board the rover itself. It is thus imperative that the replanning model is aware of the twin objectives of minimizing cognitive load on the mission controllers as well as minimizing the computation required on board the rover when vetting a new plan Π' that replaces Π. In this case, the set ψ^{Π_P} will contain constraints that try to minimize the effort needed to reconcile Π' with Π, and the metric used in the reconciliation determines the contents of ψ^{Π_P}. These can be seen as a *syntactic* version of plan stability constraints, as against the *semantic* stability constraints (based on commitments) that we go on to propose.
3. M_3: In a typical scenario, it is also possible that there may be multiple rovers working in the same environment, with knowledge (complete or partial) of the other rovers' plans. This knowledge in turn leads to dependencies which must be preserved when the plans of one (or more) of the rovers change – for example, rover *Spirit* might depend on rover *Opportunity* to transmit (back to base) the results of a scientific experiment that it plans to complete. If *Opportunity* now wishes to modify its current plan Π_O, it must pay heed to the commitment to communicate with *Spirit* – and pass on the data that results from that communication – when devising its new plan Π'_O.

4 Replanning Constraints

As outlined in the previous section, the replanning problem can be decomposed into various *models* that are defined by the constraints that must be respected while transitioning from the old plan Π to the new plan Π'. In this section, we define those constraints, and explore the composition of the set ψ for each of the models defined previously.

4.1 Replanning as Restart

By the definition of this model, the old plan Π_P is completely abandoned in favor of a new one. There are no constraints induced by the previous plan that must be respected, and thus the set ψ^{Π_P} is empty. Instead, what we have is a new problem instance P' whose composition is completely independent of the set ψ^{Π_P}.

[1] Note that this is the case regardless of whether the planner models these agents explicitly or chooses to implicitly model them in the form of a dynamic world.

4.2 Replanning to Reduce Computation

It is often desirable that the replan for the new problem instance P' resemble the previous plan Π_P in order to reduce the computational effort associated with verifying that it still meets the objectives, and to ensure that it can be carried out in the world. We name the effort expended in this endeavor as the *reverification complexity* associated with a pair of plans Π_P and Π'_P, and informally define it as the amount of effort that an agent has to expend on comparing the differences between an old plan Π_P and a new candidate plan Π'_P with respect to execution in the world.

This effort can either be computational, as is the case with automated agents like rovers and robots; or cognitive, when the executor of the plans is a human. Real world examples where reverification complexity is of utmost importance abound, including machine-shop or factory-floor planning; planning for assistive robots and teaming; and planetary rovers (see Section 3.1). Past work on replanning has addressed this problem via the idea of *plan stability* (Fox et al. 2006). The general idea behind this approach is to preserve the stability of the replan Π'_P by minimizing some notion of difference with the original plan Π_P. In the following, we examine two such ways of measuring the difference between pairs of plans, and how these can contribute constraints to the set ψ^{Π_P} that will minimize reverification complexity.

Action Similarity Plans are defined, first and foremost, as sequences of actions that achieve specified objectives. The most obvious way to compute the difference between a given pair of plans then is to compare the actions that make up those plans. (Fox et al. 2006) defines a way of doing this - given an original plan Π and a new plan Π', they define the difference between those plans as the number of actions that appear in Π and not in Π' plus the number of actions that appear in Π' and not in Π. If the plans Π and Π' are seen as sets comprised of actions, then this is essentially the symmetric difference of those sets, and we have the following constraint:[2] $\min\ |\Pi \triangle \Pi'|$.

This method of gauging the similarity between a pair of plans suffers from some obvious pitfalls; a very simple one is that it does not take the ordering of actions in the plans into account at all. Consider the simple plans $\Pi : \langle a_1, a_2 \rangle$ and $\Pi' : \langle a_2, a_1 \rangle$; the difference between these two plans is $\Pi \triangle \Pi' = \emptyset$. However, from a replanning perspective, it seems obvious that these two plans are really quite different, and may lead to different results if the actions are not commutative. In order to account for such cases, we would need to consider the ordering of actions within a plan, and more generally, the causal structure of a plan.

Causal Link Similarity The next step in computing plan similarity is to look not just at the actions that constitute the plans under comparison, but to take the causal structure of those plans into account as well. Work on partial order planning (POP) has embedded a formal notion of causal links quite strongly within the planning literature. Past partial order planning systems (Penberthy and Weld 1992; Joslin and Pollack 1995) have looked at the idea of different serializations of the same partial order plan. Given plans Π and Π', and CL(Π) and CL(Π') the sets of causal links on those plans respectively, a simple constraint to enforce causal similarity would be: $\min |\text{CL}(\Pi) \triangle \text{CL}(\Pi')|$. Note that this number may be non-zero even though the two plans are completely similar in terms of action similarity; i.e. $(\Pi \triangle \Pi') = \emptyset$. This analysis need not be restricted to causal links alone, and can be extended to arbitrary ordering constraints of a non-causal nature too, as long as they can be extracted from the plans under consideration.

[2] Given this constraint, the similarity and difference of a pair of plans are inverses, and hence the name 'Action Similarity'.

4.3 Replanning for Multi-agent Scenarios

In a multiperson situation, one man's goals may be another man's constraints. – Herb Simon (Simon 1964)

In an ideal world, a given planning agent would be the sole center of plan synthesis as well as execution, and replanning would be necessitated only by those changes to the world state that the agent cannot foresee. However, in the real world, there exist multiple such agents, each with their own disparate objectives but all bound together by the world that they share. A plan Π_P that is made by a particular agent affects the state of the world and hence the conditions under which the other agents must plan – this is true for every agent. In addition, the publication of a plan Π_P^A by an agent A leads to other agents predicating the success of their own plans on parts of Π_P^A, and complex dependencies are developed as a result. Full multi-agent planning can resolve the issues that arise out of changing plans in such cases, but it is far from a scalable solution for real world domains currently. Instead, this multi-agent space filled with dependencies can be projected down into a single-agent space with the help of *commitments* as defined by (Cushing and Kambhampati 2005). These commitments are related to an agent's current plan Π, and can describe different requirements that come about:

1. when Π changes the world state that other agents have to plan with
2. when the agent decides to execute Π, and other agents predicate their own plans on certain aspects of it
3. due to cost or time based restrictions imposed on the agent
4. due to the agent having paid an up-front setup cost to enable the plan Π

A simple travel example serves to demonstrate these different types of commitments. Consider an agent A_1 who must travel from Phoenix (PHX) to Los Angeles (LAX). A travel plan Π that is made for agent A_1 contains actions that take it from PHX to LAX with a long stopover at Las Vegas (LAS). A_1 is friends with agent A_2, who lives in LAS, and thus publicizes the plan of passing through LAS. A_2 then makes its own plan to meet A_1 – this depends on A_1's presence at the airport in LAS. If there are changes to the world (for e.g., a lower airfare becomes available), there are several commitments that a planner must respect while creating a new plan Π' for A_1. First, there are commitments to other agents – in this case, the meeting with A_2 in LAS. There are also setup and reservation costs associated with the previous plan; for example, A_1 may have paid a non-refundable airfare as part of Π. Finally, there may be a deadline on getting

to LAX, and any new plan has to respect that commitment as well.

At first blush, it seems that the same kinds of constraints that seek to minimize reverification complexity between plans Π and Π' (minimizing action and causal link difference between plans) will also serve to preserve and keep the most commitments in the world. Indeed, in extreme cases, it might even be the case that keeping the structures of Π and Π' as similar as possible helps keep the maximum number of commitments made due to Π. However, this is certainly not the most natural way of keeping commitments. In particular, this method fails when there is any significant deviation in structure from Π to Π'; unfortunately, most unexpected changes in real world scenarios are of a nature that precludes retaining significant portions of the previous plan. For example, in the (continuing) air travel example from above, agent A_1 has a commitment not to the plan Π itself, but rather to the event of meeting A_2. This suggests modeling commitments natively as state conditions (as opposed to casting them as extraneous constraints on plan structure) as goals that must be either achieved or preserved by a plan as a possible replanning constraint. We elaborate on this in Section 5.3.

5 Solution Techniques

So far, we have looked at three different ways in which the replanning problem can be represented, and delineated the differences between these models via the constraints that need to be considered when making new plans in a changed world. We now turn our attention to the planning *techniques* that are (or can be) used to solve these variants.

5.1 T1: Classical Planning

For the *replanning as restart* model, the problem is defined as one of going from a plan Π_P that solves the problem instance $P = \langle I, G \rangle$ to the best new plan Π'_P that is valid for the new problem instance $P' = \langle I', G' \rangle$. I' is the state of the world at which Π_P stops executing to account for the change that triggered replanning; that is, replanning commences from the current state of the world. G' is the same as G unless new goals are explicitly added as part of the changes to the world. The replanning constraint set ψ^{Π_P} is empty, since replanning is being performed from scratch. This new instance is then given to a standard classical planner to solve, and the resulting plan is designated Π'_P.

5.2 T2: Specialized Replanning Techniques

When it comes to *replanning to reduce computation* and associated constraints, techniques that implement solutions that conform to these constraints must necessarily be able to compile them into the planning process in some way. This can be achieved by implementing plan stability metrics – either explicitly by comparing each synthesized plan candidate with the existing plan Π_P, or implicitly by embedding these metrics within the search process. One way of doing the latter is to use a planner such as LPG (Gerevini, Saetti, and Serina 2003), which uses local search methods, and to structure the evaluation function such that more syntactic similarity between two plans – similar actions, for example – is preferred. Such an approach is used by (Srivastava et al. 2007) in the generation of a set of diverse plans where the constituent plans differ from each other by a defined metric; for replanning where search re-use is of importance, the objective can instead be to produce *minimally different* plans within that set. An earlier version of this approach can be seen in the Casper system's iterative repair approach (Knight et al. 2001).

5.3 T3: Partial Satisfaction Planning

We now turn our attention to replanning techniques that can be used when the dependencies or commitments towards other agents due to an agent A's original plan Π (solving the problem instance P) must be maintained. The constraint set $\psi^{\Pi^A_P}$ now contains all those commitments to other agents that were made by the plan Π. We follow Cushing et al. (Cushing and Kambhampati 2005) in modeling commitments as *soft* constraints that an agent is not mandated to necessarily achieve for plan success. More generally, commitments – as reservations, prior dependencies or deadlines – can be modeled as soft trajectory constraints on any new plan Π' that is synthesized. Modeling commitments as soft constraints (instead of hard) is essential because not all commitments are equal. A replan Π' may be valid even if it flouts a given commitment; indeed, it may be the *only* possible replan given the changed state of the world. Soft goals allow for the specification of different priorities for different commitments by allowing for the association of a *reward* for achieving a given goal, and a *penalty* for non-achievement. Both of these values are optional, and a commitment may either be seen as an opportunity (accompanied by a reward) or as a liabiity (when assigned a penalty). The quality of a replan Π' – in terms of the number of commitment constraints that it satisfies – can then be discussed in terms of the *net-benefit*, which is a purely arithmetic value.

An added advantage of modeling commitments as soft goals is that the constraints on plan structure discussed previously in Section 4.2 can be cast as commitments too. These constraints are commitments to the *structure* of the original plan Π, as against commitments to other agents or to other extraneous phenomena like deadlines etc. The advantage in doing this is that new plans and their adherence to commitments can be evaluated solely and completely in terms of the net-benefit of those plans; this makes the enforcement of the replanning constraints during the planning process more amenable to existing planning methods. We thus devise a natural way of combining two distinct quality issues in replanning: (1) how good a replan Π' is for solving the changed problem instance $\langle I', G' \rangle$; and (2) how much Π' respects and balances the given replanning constraints, which may be in service of completely different objectives like reducing the computation involved in verifying a new plan, or commitments to other agents in the world.

To obtain the new problem instance P' from the original problem P, we perform the following transformations: I' is, as before, the state of the world at which execution is stopped because of the changes that triggered replanning. G' consists of all outstanding goals in the set G as well as any other explicit changes to the goal-set; in addition, the constraints from the set $\psi^{\Pi^A_P}$ are added to G' as soft goals, using the compilations described below. The new problem

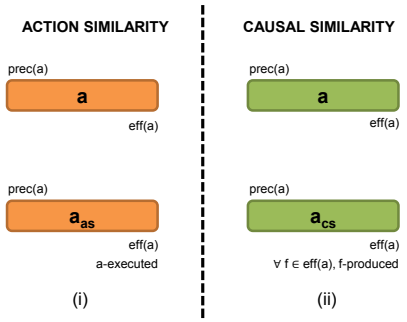

Figure 2: Compiling action and causal similarity to PSP by creating new effects, actions that house those effects, and soft goals on those effects.

instance is then given to a PSP planner to solve for the plan with the best net-benefit, which is then designated Π'^{A}_{P}.

The syntactic plan similarity constraints discussed at length in Section 4.2 can be cast as PSP constraints, in the form of soft goals. In the following, we describe a general compilation of the constraints in $\psi^{\Pi^A_P}$ to a partial satisfaction planning problem instance. We follow (van den Briel et al. 2004) in defining a PSP Net Benefit problem as a planning problem $P = (F, O, I, G_s)$ (where F is a finite set of fluents, O is a finite set of operators and $I \subseteq F$ is the initial state as defined earlier in our paper) such that each action $a \in O$ has a "cost" value $C_a \geq 0$ and, for each goal specification $g \in G$ there exists a "utility" value $U_g \geq 0$. Additionally, for every goal $g \in G$, a 'soft' goal g_s with reward r_g and penalty p_g is created; the set of all soft goals thus created is added to a new set G_s.

The intuition behind casting these constraints as goals is that a new plan (replan) must be constrained in some way towards being similar to the earlier plan. However, making these goals *hard* would over-constrain the problem – the change in the world from I to I' may have rendered some of the earlier actions (or causal links) impossible to preserve. Therefore the similarity constraints are instead cast as *soft* goals, with rewards or penalties for preserving or breaking (respectively) the commitment to similarity with the earlier plan. In order to support these goals, new fluents need to be added to the domain description that indicate the execution of an action, or achievement of a fluent respectively. Further, new copies of the existing actions in the domain must be added to house these effects. Making copies of the actions from the previous plan is necessary in order to allow these actions to have different costs from any new actions added to the plan.

Compiling Action Similarity to PSP The first step in the compilation is converting the action similarity constraints in $\psi^{\Pi^A_P}$ to soft goals to be added to G_s. Before this, we examine the structure of the constraint set $\psi^{\Pi^A_P}$; for every ground action $\bar{a}$ (with the names of the objects that parameterize it) in the old plan Π, the corresponding action similarity constraint is $\Psi_{\bar{a}} \in \psi^{\Pi^A_P}$, and that constraint stores the name of the action as well as the objects that parameterize it.

Next, a copy of the set of operators O is created and named O_{as}; similarly, a copy of F is created and named F_{as}. For each (lifted) action $a \in O_{as}$ that has an instance in the original plan Π, a new fluent named "a-executed" (along with all the parameters of a) is added to the fluent set F_{as}. Then, for each action $a \in O_{as}$, a new action a_{as} which is a copy of the action a that additionally also gives the predicate a-executed as an effect, is created. The process of going from the original action a to the new one a_{as} is depicted graphically in Figure 2(i). In the worst case, the number of actions in each O_{as} could be twice the number in O.

Finally, for each constraint $\Psi_{\bar{a}} \in \psi^{\Pi^A_P}$, a new soft goal $g_{\bar{a}}$ is created with corresponding reward and penalty values $r_{g_{\bar{a}}}$ and $p_{g_{\bar{a}}}$ respectively, and the predicate used in $g_{\bar{a}}$ is $\bar{a}$-executed (parameterized with the same objects that $\bar{a}$ contains) from O_{as}. All the $g_{\bar{a}}$ goals thus created are added to G_s. In order to obtain the new compiled replanning instance P' from P, the initial state I is replaced with the state at which execution was terminated, I'; the set of operators O is replaced with O_{as}; and the set of fluents F is replaced with F_{as}. The new instance $P' = (F_{as}, O_{as}, I', G_s)$ is given to a PSP planner to solve.

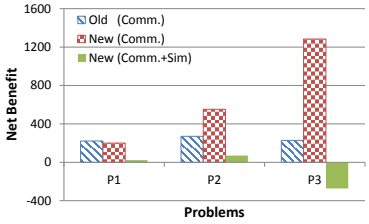

Figure 3: Net-benefit of plans for zenotravel problems, Experiment 1.

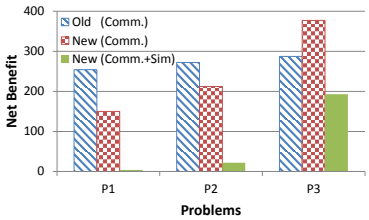

Figure 4: Net-benefit of plans for driverlog problems, Experiment 1.

Compiling Causal Similarity to PSP Causal similarity constraints can be compiled to PSP in a manner that is very similar to the above compilation. The difference that now needs to be considered is that the constraints are no longer on actions, but on the grounded fluents that comprise the causal links between the actions in a plan instead.

The first step is to augment the set of fluents; a copy of F is created and named F_{cs}. For every fluent $f \in F$, a new fluent named "f-produced" is added to F_{cs}, along with all the original parameters of f. A copy of the set of operators O is created and named O_{cs}. Then, for each action in $a \in O_{cs}$, a new action a_{cs} is added; a_{cs} is a copy of action a, with the additional effects that for every fluent f_a that is in the add effects of the original a, a_{cs} contains the effect f_a-produced – this process is shown in Figure 2(ii). Thus in the worst case, the number of effects of every action a_{cs} is twice the number of effects of the original action a, and the size of O_{cs} is twice that of O.

Finally, the causal constraints in $\psi^{\Pi^A_P}$ must be converted to soft goals that can be added to G_s. The constraints $\Psi \in$

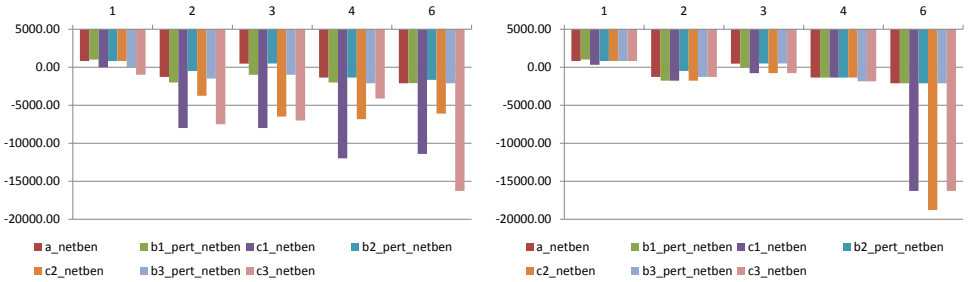

Figure 5: Net-benefit of plans for zenotravel problems, Experiment 2.

$\psi^{\Pi^A_P}$ are obtained by simulating the execution of Π from I using the operators in O. Each ground add-effect $\bar{f}_e$ of each ground action $\bar{a_\Pi}$ in Π is added as a new constraint $\Psi_{\bar{f}_e}$. Correspondingly, for each such new constraint added, a new soft goal $g_{\bar{f}_e}$ is created whose fluent corresponds to $\bar{f}_e$, with reward and penalty values $r_{g_{\bar{f}_e}}$ and $p_{g_{\bar{f}_e}}$ respectively.[3] All the goals thus created are added to G_s. The new planning instance to be provided to the PSP planner is thus given as $P' = (F_{cs}, O_{cs}, I', G_s)$, where I' is the state of the fluents when execution was previously suspended.

6 Empirical Study

Generally, work on specialized (single-agent) replanning techniques claims the support of experiments that exhibit either: (1) an advantage over from-scratch replanning in terms of speed or efficiency; or (2) greater plan stability when compared to other techniques. Unfortunately, our improved understanding of replanning as a general problem rather than as any single technique renders such an evaluation unsatisfactory. Since different kinds of replanning problems can be realized from different instantiations of the constraint set ψ, and these constraints can all be weighted as desired, one thing that we *should* evaluate is whether our single general model can model any problem (and technique) in the replanning spectrum. Given access to such a general model, it is also rather straightforward to set up problem instances that favor a specific technique over all other replanning techniques. Such results can be attributed either to the fact that the other techniques ignore the constraints that the first takes into account, or that they use surrogate constraints in order to mimic them. In the following, we describe the setup of three such experiments, and present preliminary results from the first two.

Setting Rewards & Penalties – The compilations outlined in Section 5.3, as well as the results that follow, are sensitive to the actual value of the rewards and the penalties that are assigned to the goals that the planner must achieve. We are currently in the process of conducting a theoretical as well as empirical analysis of the effect of these values on the plans that are produced, as well as the time taken to replan. For the experiments outlined below, we assign a reward of 500 units and a penalty of 1000 units to the regular, state space goals. Similarity goals are given a reward of 0 units, and a penalty of 1000 units (since they can be seen as commitments to the form of the previous plan).

[3] Note that in the general case, we would need to consider *consumers* – i.e., actions that consume the causal link – apart from the producers of those links, in order to avoid over-constraining the new problem. However, we assume here that the original plan does not contain any superfluous actions.

6.1 Planning System

Since PSP is key to the success of our approach, we used the planner Sapa Replan (Talamadupula et al. 2010), a planner that has been used previously to support applications that require replanning. Sapa Replan additionally handles temporal planning and partial satisfaction. The system contains an execution monitor that oversees the execution of the current plan in the world, which focuses the planner's attention by performing objective (goal) selection, while the planner in turn generates a plan using heuristics that are extracted by supporting some subset of those objectives. Unfortunately, Sapa Replan's support for all of these varied functionalities renders it less scalable to an increase in the number of soft goals that must concurrently be pursued by the planner. This rules out extensive experimentation, as well as the testing of theories such as the one proposed in Experiment 4 (see Section 6.2). We are currently working on using faster planners that can handle larger numbers of soft goals for our experiments.

6.2 Experiments

For our experiments, we compared similarity based replanning against commitment sensitive replanning (Experiment 1) and against replanning from scratch (Experiment 2). In order to set up a compilation from similarity based replanning into PSP, we followed the procedure outlined in Section 5.3 to alter the IPC domains that we considered, and obtain the respective O_{as} operator sets and the P' problems (which we denote P_a for the experiments). We then ran all the problem instances for all our experiments on the same planner.

Experiment 1 – We ran the planner with O_{as} and P_a, and recorded the net-benefit values of the resulting plan π_a. We then created two different versions of this problem, P_b and P_c respectively, to simulate the effect of changes in the world (and hence force "replanning"). The perturbations used to generate P_b from P_a involved deleting facts from the initial state, deleting goals, or both. P_c was a copy of P_a that additionally contained new "similarity" goals on the execution predicates of every action $o \in \pi_a$. Each of these similarity goals carried a penalty with it – one that is levied for every action in π_c that deviates from π_a (and hence lowers π_c's similarity to π_a). We ran the problem on P_b and P_c as well and recorded the net-benefit and makespan values from these runs.

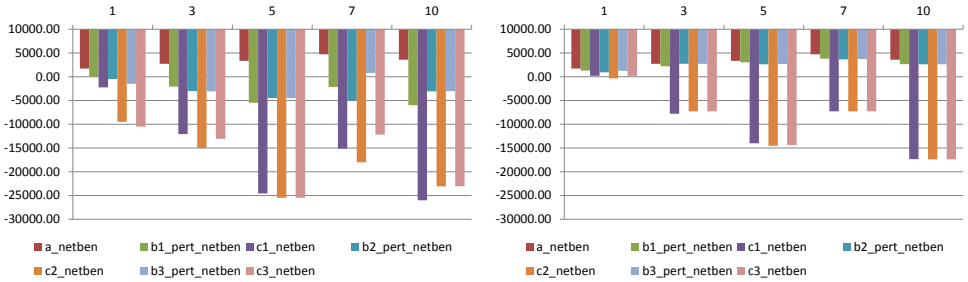

Figure 6: Net-benefit of plans for driverlog problems, Experiment 2.

Experiment 2 – For this experiment, we generated more perturbations to each P_a in a randomized fashion. These perturbations involved deleting facts from the initial state, deleting goals, or both. From each problem instance P_a we produced three perturbed instances $P_{b_1} \ldots P_{b_3}$ by deleting (at random) up to a maximum of 4 facts from the original initial state, and a maximum of 2 goals from the original set of goals to be achieved. Then, once the perturbed instances were generated, we generated three corresponding "similarity" instances $P_{c_1} \ldots P_{c_3}$ by copying $P_{b_1} \ldots P_{b_3}$ respectively, to force the planner to adopt a minimal perturbation approach. This was achieved by adding an additional set of "similarity" soft goals to the instances P_{c_i}; these similarity goals were generated using the same process as for Experiment 1. The addition of a penalty ensured that the planning process tried, as far as possible, to include the actions from the previous plan in the new plan. Due to the way the IPC problem instances are set up, even the smallest random perturbation to the initial state can render the perturbed problem unsolvable. To account for this problem, we created copies of all the P_{b_i} and P_{c_i} instances with only the goals deleted; that is, the initial state was unchanged, and only goal perturbations were introduced.

The resulting net-benefit values are plotted in Figure 5 and Figure 6; the plot on the left denotes those instances where the perturbation was performed on both the initial state as well as the goals, whereas the plot on the right represents those with only goal perturbations. The numbers along the X-axis are the IPC problem number, and the columns (from left to right) denote the instance that produced that value – respectively, P_a, P_{b_1}, P_{c_1}, P_{b_2}, P_{c_2}, P_{b_3} and P_{c_3}. Every pair of columns after the first one (there are three such pairs) signifies a direct comparison between the net-benefit for the perturbed problem, and the perturbed problem with similarity constraints added. Since the net-benefit values of many of the resulting plans are negative, the X-axis is shifted further up along the Y-axis. Due to this, smaller columns indicate better plan quality, since those net-benefit values are higher. We preserve the same axes for a given domain, so direct comparisons may be made between the plots on the left and the right.

Experiment 3 – Experiment 2 can also be modified in order to induce a direct comparison between similarity and commitment based replanning. Such an experiment would require a component that extracts commitments from a given plan π in an automated and unbiased manner; these commitments (or some randomized subset therein) would then be added in as goals for the perturbed version of the commitment based problem. The similarity based problem would be perturbed as before, and similarity goals added to it based on the actions in the original plan. Both of these instances can then be run on the same planning system, and the net-benefit values returned may then be compared directly. We have constructed a preliminary version of such a module, and are in the process of collecting results for this experiment.

Experiment 4 – Another experiment that can be performed is to contrast replanning methods that preserve similarity (either action or causal) against methods that instead preserve (state space) commitments. This can be done by fixing the number of commitments C that must be adhered to when transitioning from the original plan Π to a new plan Π'. Suppose that the state space of the original plan Π is given by S, where each element $s \in S$ is a state that results from the execution of actions $a \in \Pi$ starting from the initial state I. When the value of C is zero – that is, no commitments need be preserved when replanning – the net benefit of the new plan Π' will be at its highest value. Then, as the value of C tends toward $|S|$, the net benefit of the plan generated by the commitment preserving approach will steadily change, until at $C = |S|$ it is the same as the previous plan Π (indeed, at this stage, the new plan Π' is the same as Π). We are currently evaluating this experiment across various domains.

7 Conclusion

In this paper, we presented a general model of the single-agent replanning problem, and described three replanning paradigms that are distinguished by the constraints that they are bound to satisfy during the replanning process: replanning as restart, replanning to reduce computation, and replanning for multi-agent scenarios. In particular, we showed how commitment to certain constraints – whether they be from the structure of the previous plan, or understandings with other agents – can influence replanning. We then looked at solution techniques from the single-agent planning community for these three paradigms. Finally, we presented an evaluation of our main claims based on a compilation of the previously defined replanning constraints into a partial satisfaction planning framework.

8 Acknowledgements

We wish to thank anonymous reviewers for helpful comments and suggestions on past versions of this paper. Kambhampati's research is supported in part by the ARO grant W911NF-13-1-0023, the ONR grants N00014-13-1-0176, N00014-09-1-0017 and N00014-07-1-1049, and the NSF grant IIS201330813.

References

[Bartold and Durfee 2003] Bartold, T., and Durfee, E. 2003. Limiting disruption in multiagent replanning. In *Proceedings of the Second International Joint Conference on Autonomous Agents and Multiagent Systems*, 49–56. ACM.

[Cushing and Kambhampati 2005] Cushing, W., and Kambhampati, S. 2005. Replanning: A New Perspective. In *Proc. of ICAPS 2005*.

[Fikes, Hart, and Nilsson 1972] Fikes, R.; Hart, P.; and Nilsson, N. 1972. Learning and executing generalized robot plans. *Artificial intelligence* 3:251–288.

[Fox et al. 2006] Fox, M.; Gerevini, A.; Long, D.; and Serina, I. 2006. Plan stability: Replanning versus plan repair. In *Proc. of ICAPS 2006*.

[Fritz and McIlraith 2007] Fritz, C., and McIlraith, S. 2007. Monitoring plan optimality during execution. In *Proc. of ICAPS 2007*, 144–151.

[Gerevini, Saetti, and Serina 2003] Gerevini, A.; Saetti, A.; and Serina, I. 2003. Planning through stochastic local search and temporal action graphs in lpg. *J. Artif. Intell. Res. (JAIR)* 20:239–290.

[Joslin and Pollack 1995] Joslin, D., and Pollack, M. E. 1995. Least-cost flaw repair: A plan refinement strategy for partial-order planning. In *Proceedings of the National Conference on Artificial Intelligence*, 1004–1009.

[Kambhampati 1990] Kambhampati, S. 1990. Mapping and retrieval during plan reuse: a validation structure based approach. In *Proceedings of the Eighth National Conference on Artificial Intelligence*, 170–175.

[Knight et al. 2001] Knight, S.; Rabideau, G.; Chien, S.; Engelhardt, B.; and Sherwood, R. 2001. Casper: Space exploration through continuous planning. *Intelligent Systems, IEEE* 16(5):70–75.

[Komenda et al. 2008] Komenda, A.; Pechoucek, M.; Biba, J.; and Vokrinek, J. 2008. Planning and re-planning in multi-actors scenarios by means of social commitments. In *Computer Science and Information Technology, 2008. IMCSIT 2008. International Multiconference on*, 39–45. IEEE.

[Komenda, Novák, and Pěchouček 2012] Komenda, A.; Novák, P.; and Pěchouček, M. 2012. Decentralized multi-agent plan repair in dynamic environments. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 3*, 1239–1240. International Foundation for Autonomous Agents and Multiagent Systems.

[Meneguzzi, Telang, and Singh 2013] Meneguzzi, F.; Telang, P. R.; and Singh, M. P. 2013. A first-order formalization of commitments and goals for planning.

[Nebel and Koehler 1995] Nebel, B., and Koehler, J. 1995. Plan reuse versus plan generation: a complexity-theoretic perspective. *Artificial Intelligence* 76:427–454.

[Penberthy and Weld 1992] Penberthy, J., and Weld, D. 1992. UCPOP: A sound, complete, partial order planner for ADL. In *Proceedings of the Third International Conference on Knowledge Representation and Reasoning*, 103–114. Citeseer.

[Simon 1964] Simon, H. 1964. On the concept of organizational goal. *Administrative Science Quarterly* 1–22.

[Srivastava et al. 2007] Srivastava, B.; Nguyen, T.; Gerevini, A.; Kambhampati, S.; Do, M.; and Serina, I. 2007. Domain independent approaches for finding diverse plans. In *Proc. of IJCAI*, volume 7, 2016–2022.

[Talamadupula et al. 2010] Talamadupula, K.; Benton, J.; Kambhampati, S.; Schermerhorn, P.; and Scheutz, M. 2010. Planning for human-robot teaming in open worlds. *ACM Transactions on Intelligent Systems and Technology (TIST)* 1(2):14.

[van den Briel et al. 2004] van den Briel, M.; Sanchez, R.; Do, M.; and Kambhampati, S. 2004. Effective approaches for partial satisfaction (over-subscription) planning. In *Proceedings of the National Conference on Artificial Intelligence*, 562–569.

[Van Der Krogt and De Weerdt 2005] Van Der Krogt, R., and De Weerdt, M. 2005. Plan repair as an extension of planning. In *Proc. of ICAPS 2005*.

[Wagner et al. 1999] Wagner, T.; Shapiro, J.; Xuan, P.; and Lesser, V. 1999. Multi-level conflict in multi-agent systems. In *Proc. of AAAI Workshop on Negotiation in Multi-Agent Systems*.

[Wooldridge 2000] Wooldridge, M. 2000. *Reasoning about rational agents*. MIT press.