# Decentralized Multi-agent Plan Repair in Dynamic Environments*

# (Extended Abstract)

Antonín Komenda Peter Novák Michal Pěchouček

Agent Technology Center,
Department of Computer Science and Engineering,
Czech Technical University in Prague
{antonin.komenda,peter.novak,michal.pechoucek}@agents.fel.cvut.cz

## ABSTRACT

Achieving joint objectives by teams of cooperative planning agents requires significant coordination and communication efforts. For a single-agent system facing a plan failure in a dynamic environment, arguably, attempts to repair the failed plan in general do not straightforwardly bring any benefit in terms of time complexity. However, in multi-agent settings the communication complexity might be of a much higher importance, possibly a high communication overhead might be even prohibitive in certain domains. We hypothesize that in decentralized systems, where coordination is enforced to achieve joint objectives, *attempts to repair failed multi-agent plans should lead to lower communication overhead than replanning from scratch*.

## Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*Intelligent agents, Multiagent systems*; I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search—*Plan execution, formation, and generation*

## General Terms

Algorithms, Experimentation.

## Keywords

multi-agent plan repair, decentralized multi-agent planning, communication complexity.

## 1. MOTIVATION

When an agent is situated in a dynamic environment, occurrence of various unexpected events the environment generates might lead to plan invalidation, a failure. A straightforward solution to this problem is to invoke a planning algorithm and compute a new plan from the state the agent found itself in after the failure to a state conforming with its original objective. In many cases, however, a relatively minor fix to the original plan would resolve the failure, possibly at a lower cost.

*An extended version of this paper was also published as [1].

**Appears in:** *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2012)*, Conitzer, Winikoff, Padgham, and van der Hoek (eds.), 4-8 June 2012, Valencia, Spain.


In general, plan repair can be seen as planning with re-use of fragments of the old plan. Even though there is a body of research empirically demonstrating that plan repair in some domains performs better than replanning (e.g., [2]), theoretical analysis concluded concluded that plan re-use (repair) in general does not bring any benefit over replanning in terms of computational time complexity [3]. In situated multi-agent systems often it is not the time complexity which is of a primary importance, the *communication complexity* is often a higher priority concern (consider application domains, such as e.g., undersea operations, where the communication links are extremely constrained and expensive). The motivation for our research is the intuition that multi-agent plan repair, even though not always the fastest approach, should under specific conditions generate lower communication overheads in comparison to replanning.

## 2. MULTI-AGENT PLAN REPAIR

We consider a number of *cooperative* and *coordinated* actors featuring possibly distinct sets of capabilities (actions), which concurrently plan and subsequently execute their local plans so that they achieve a joint goal. An instance of a multi-agent planning problem is defined by: i) an environment characterized by a state space, ii) a finite set of agents, each characterized by a set of primitive actions (or capabilities) it can execute in the environment, iii) an initial state the agents start their activities in and iv) a characterization of the desired goal states. Definitions of the underlying formal framework can be found in Nissim et al. [4]. The core hypothesis of the paper can be then formulated as follows: *multi-agent plan repair approaches producing more preserving repairs than replanning tend to generate lower communication overhead for tightly coupled multi-agent problems.*

We propose three algorithms for solving the plan repair problem. The core idea behind the *back-on-track* (BoT) algorithm is to utilize a multi-agent planner to produce a plan from the failed state to the originally desired state and subsequently follow the rest of the original multi-agent plan from the step in which the failure occurred. In result, the BoT repair tries to preserve a suffix of the original plan and prefix it with a newly computed plan starting in the failure state and leading to some state along the execution of the original plan in the ideal environment.

The second approach, *lazy-repair (LR)*, is designed to to preserve an *executable remainder* of the original multi-agent plan (the actions would remain, if the original plan was executed ignoring non executable actions) and close the gap between the state resulting from the failed plan execution and a goal state of the original planning problem. The lazy approach tries to preserve a partial

prefix of the original plan and complete it by a newly planned plan suffix. The algorithm is incomplete, as it might happen that the execution of the executable remainder diverges to a state from which no plan to some goal state exists.

The shortcoming of the LR algorithm is addressed by the *repeated lazy repair (RL).* The idea is that a failure during execution of an already repaired plan makes the previous repair irrelevant and its result can be discarded, unless the failure occurred already in the fragment appended by the previous repair. Note, the repeated lazy repair algorithms enables a plan execution model which preserves significantly longer fragments of the original plan. That is, upon a failure, instead of trying to repair the failed plan directly, as the previous two algorithms, the system can simply proceed with execution of the remainder of the original plan and only after its complete execution the lazy plan repair is triggered. The approach simply ignores the plan failures during the multi-agent plan execution and postpones the repair to the very end of the process, hence the "*lazy*" label for the two algorithms.

## 3. EXPERIMENTAL VALIDATION

To verify the core hypothesis, we conducted a series of experiments with implementations of the proposed multi-agent plan repair algorithms. Firstly, a multi-agent plan was computed by a distributed multi-agent planner authored by Nissim et al. [4]. Secondly, we executed the multi-agent plan. In the course of the plan execution, we simulated the environment dynamics by producing various plan failures according to a variable failure probability $P$ (with a uniform distribution). The plan execution was monitored and upon a failure detection a plan repair algorithm was invoked. Before execution of each plan step, the joint action of all actions of the particular agents is checked for applicability in the current state. In the case it is not applicable, a plan repair algorithm is invoked and the execution continues on the repaired plan.

We distinguished two types of plan failures: *action failures* and *state perturbations*. An *action failure* is simulated by omitting a randomly chosen (with a uniform distribution) individual agent action from the actual plan step. The other simulated failure type, *state perturbation,* is parametrized by a positive non-zero integer $c$, which determines the number of randomly chosen (again with a uniform distribution) state terms, which are removed from the current state, as well as the number of terms which are added to it.

The experiments were conducted on three planning domains originating in the standard benchmark single-agent ICP planning domains. Similarly to [4], we chose domains, which are straightforwardly modifiable to multi-agent planning problems: LOGISTICS (3 agents), ROVERS (3 agents), and SATELLITES (2–5 agents). The metrics were i) *execution length* (number of joint actions executed), ii) *planning time* (cumulative time consumed by the underlying planner), iii) *communication* (number of messages passed between the agents).

## 4. RESULTS AND FINAL REMARKS

The first batch of experiments directly targets validation of the core hypothesis. We used LOGISTICS as a tightly coupled domain (the resulting personal plans often depend on each other) and dynamics of the simulated environment modeled as action failures. Figure 1 depicts the results of the experiment, which *support* our hypothesis. The overall planning time was at 54% (34% at best) and at 51% (12% at best) for Back-on-Track-Repair and Repeated-Lazy-Repair against replanning respectively. The execution length was lower being in average 96% (72% at best, 130% at worst) by Back-on-Track-Repair and significantly lower being

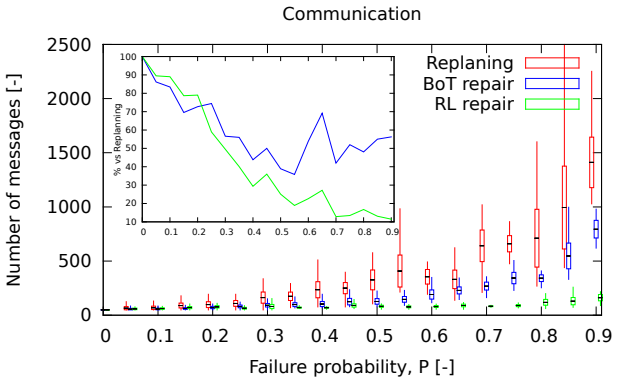


**Figure 1: Experimental results for LOGISTICS domain with 3 agents and action failures.**

81% (34% at best, 132% at worst) for Repeated-Lazy-Repair.

The second batch of experiments focused on boundaries of validity of the positive result presented above, i.e., *with decreasing coupling of the domain, the communication efficiency gains of repairing techniques should decrease*. Experiments performed with the loosely-coupled domain of ROVERS support the claim.

The third batch of experiments targeted the perturbation magnitude of the plan failures, i.e., *communication efficiency gain of plan repairing should decrease as the difference between a nominal and related failed state increases*. The underlying intuition that in the case the dynamic environment generates only relatively small state perturbations and the failed states are "not far" from the actual state was positively supported by results of another LOGISTICS experiment employing state perturbations as the model of the environment dynamics ($c = 1$).

Finally, we conducted a series of experiments with an uncoupled SATELLITES domain. The results show the anticipated lower plan repair communication efficiency in contrast to replanning.

The main difference between our approach and the related work (partial ordered plan monitoring and repairing, conformant and contingency planning, plan re-use and plan adaptation, and finally Markov decision processes) is that the state perturbations utilized in our experiments have *a priori* unknown probabilities.

## Acknowledgements

This work was supported by the *U.S. Army CERDEC* grant W911NF-11-1-0252 and by the *Grant Agency of the Czech Technical University in Prague* grant SGS10/189/OHK3/2T/13 .

## 5. REFERENCES

[1] Antonín Komenda, Peter Novák, and Michal Pěchouček. Decentralized Multi-agent Plan Repair in Dynamic Environments. *CoRR*, abs/1202.2773, February 2012.

[2] Roman van der Krogt and Mathijs de Weerdt. Self-interested planning agents using plan repair. In *Proceedings of the ICAPS 2005 Workshop on Multiagent Planning and Scheduling*, pages 36–44, 2005.

[3] B. Nebel and Koehler J. Plan reuse versus plan generation: a theoretical and empirical analysis. *Artificial Intelligence*, 76(1-2):427–454, July 1995.

[4] Raz Nissim, Ronen I. Brafman, and Carmel Domshlak. A general, fully distributed multi-agent planning algorithm. In *Proceedings of AAMAS'10*, pages 1323–1330, 2010.