k-Robust Multi-Agent Path Finding

Dor Atzmon, Ariel Felner
Roni Stern
Ben-Gurion University
Israel

Glenn Wagner
Computer Science Department
Carnegie Mellon University
USA

Roman Barták
Charles University
Czech

Neng-Fa Zhou
City University of New York
USA

k-Robust MAPF

In the multi-agent path-finding (MAPF) problem a plan is needed to move a set of agents from their initial location to their goals without collisions. In this paper we introduce and study the k-robust MAPF problem, where we seek a plan that is robust to k unexpected delays per agent.

Definition 1 (k-delay Conflict) *A k-delay conflict $\langle a_i, a_j, t\rangle$ in a plan π occurs iff there exists $\Delta \in [0, k]$ such that agents a_i and a_j are located in the same location in time steps t and $t + \Delta$, respectively, i.e, when $\pi_i(t) = \pi_j(t + \Delta)$.*

We say that a plan π is k-robust if it does not have any k-delay conflicts. Informally, this means that no conflicts will occur even if some of the agents are delayed by up to k time steps. The problem we address in this paper is how to find optimal sum-of-costs k-robust plans. Namely, the minimal sum of all paths in which all agents can delay up to k times without causing a collision.

Conflict-Based Search Solutions

This paper presents a planner that can solve the k-robust MAPF planner which is based on the *Conflict-based search* (CBS) (Sharon et al. 2015) MAPF solver. CBS does not explicitly search the n-agent state space. Instead, agents are associated with constraints of the form $\langle a_i, v, t\rangle$, which would prohibit agent a_i from occupying vertex v at time step t. A *consistent path* for agent a_i is a path that satisfies *all* of a_i's constraints, and a *consistent plan* is a plan composed only of consistent paths. Note that a consistent plan can be *invalid* if it contains conflicts despite each path satisfying the individual agent constraints.

CBS works by searching a *constraint tree* (CT) for a set of constraints such that a consistent plan w.r.t. this set of constraints is optimal. The CT is a binary tree, in which each node N contains: **(1)** a set of constraints imposed on the agents ($N.constraints$), **(2)** a single plan ($N.\pi$) consistent with these constraints, and **(3)** the cost of $N.\pi$ ($N.cost$).The root of the CT contains an empty set of constraints (thus, every plan is consistent with the root). Each vertex N has two successors, which are generated by first finding a conflict in $N.\pi$. Each successor is then generated by adding a new constraint to $N.constraints$ that prohibits one of the agents involved in the collision from occupying the conflicting vertex at the time of the conflict. A new plan consistent with the augmented constraints is then computed for that agent. A CT node N is a goal node when $N.\pi$ is valid. To search the CT for a goal node CBS runs a best-first search where nodes are ordered by their costs.

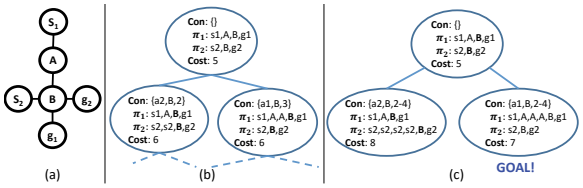

Figure 1: (a) The graph (b) CT using the original time/location constraints (c) CT using range constraints

k-Robust CBS

Next, we describe *k-robust CBS* (kR-CBS), an adaptation of CBS designed to return optimal k-robust plans. kR-CBS differs from CBS in that a valid path cannot contain any k-delay conflicts and how constraints are added to avoid said conflicts.

Resolving k-delay Conflicts. Let N be a non-goal node in the CT selected to be expanded next by kR-CBS, and let $\langle a_i, a_j, t\rangle$ be a k-delay conflict in N. Note that there is no k-robust plan in which a_i is at v at time t while a_j is at v at time $t+\Delta$. Therefore, kR-CBS generates two children for n, adding the constraint $\langle a_i, v, t\rangle$ to one child, and $\langle a_j, v, t+\Delta\rangle$ to the other.

Proving that kR-CBS is sound and complete is straightforward. It is sound because it only halts when generating a CT node that has no k-delay conflicts. It is complete because when splitting a CT node we do not lose any valid plans. Similarly, kR-CBS returns optimal plans, as it searches the CT in order of the nodes' costs, and the cost of a node N is a lower bound on the cost of any optimal plan consistent with $N.constraints$.

Example. Consider a 2-robust MAPF problem on the graph in Figure 1(a), with two agents whose start-goal pairs are s_1-g_1 and s_2-g_2, respectively. Figure 1(b) shows the first

	Plan cost			Plan time (ms)						
				k=0	k=1			k=2		
m	k=0	k=1	k=2	All	KR	IKR(A)	IKR(S)	KR	IKR(A)	IKR(S)
4	21	22	22	6	15	14	15	193	110	67
6	31	32	32	5	28	26	20	990	388	94
7	36	37	39	7	31	26	17	1,618	826	184
8	41	41	43	6	29	23	20	2,625	1,051	229
9	48	49	51	9	379	218	76	20,006	4,408	556
10	49	51	53	41	162	124	78	22,464	7,097	875

Table 1: Average plan cost and planning runtime for different CBS-based k-robust solvers, on an 8x8 open grid

two levels of the CT generated by kR-CBS, where every node N shows $N.constraints$ (labeled Con), $N.\pi_1$, $N.\pi_2$, and $N.cost$. Observe that the plan in the root is valid, but is not 2-robust, having a 2-delay conflict $\langle a_2, a_1, 2\rangle$ at location B for $\Delta = 1$, since $\pi_1(3) = \pi_2(2) = B$. To try to resolve this conflict, kR-CBS adds the constraint $\langle a_2, B, 2\rangle$ to the left child and the constraint $\langle a_1, B, 3\rangle$ to the right child. Both children of the root node are also not goal nodes. In fact, in this example we will need to generate a total of 7 CT nodes before finding an optimal plan.

Improved k-Robust CBS

Next, we introduce the Improved kR-CBS (I-kR-CBS) that resolves conflicts in a CT node N by imposing *range constraints* on its successors. A *range constraint* is defined by the tuple $\langle a_i, v, [t_1, t_2]\rangle$ and represents the constraint that agent a_i must avoid vertex v from time t_1 to time t_2.

Definition 2 (Sound Range Constraints) *A pair of range constraints are called* sound *iff all k-robust plans satisfy at least one of these constraints.*

Corollary 1 *A kR-CBS variant that uses range constraints is sound, complete, and returns optimal k-robust plans if it resolves conflicts only with sound pairs of range constraints.*

Corollary 2 (Symmetric range constraints) *For any time step t, vertex v, and agents a_i and a_j, the range constraints $\langle a_i, v, [t, t+k]\rangle$, $\langle a_j, v, [t, t+k]\rangle$ are sound for solving a k-robust MAPF problem.*

A pair of sound range constraints can also be *asymmetric*, i.e., constrain one agent to a longer time range than the other agent. For example, consider a conflict $\langle a_i, a_j, t\rangle$ at vertex v and pair of range constraints $R_1 = \langle a_i, v, [t-k, t+k]\rangle$ and $R_2 = \langle a_j, v, [t]\rangle$. R_1 and R_2 are a sound pair of constraints, because a solution must satisfy either R_1 or R_2, since violating both results in a k-delay conflict. R_1 and R_2 are extremely asymmetric, but one can imaging asymmetric range constraints that are more balanced. An open question for asymmetric range constraints is how to choose which agent to impose the more restricted constraint upon.

Experimental Results

We experimented with kR-CBS and I-kR-CBS using symmetric and asymmetric pairs of range constraints. Asymmetric constraints placed a constraint of one time step on one arbitrarily chosen agent, and a constraint on a $2k+1$ time range on the other agent. Random MAPF problem instances were generated in an open 8x8 grid. The kR-MAPF solvers were run with $k \in \{0, 1, 2\}$ and the resulting plan cost and the CPU runtime were measured.

Table 1 shows the average plan cost and average CPU runtime when finding k-robust solutions using kR-CBS (labeled KR) and I-kR-CBS with the asymmetric and with the symmetric range constrains (labeled IKR(A) and IKR(S), respectively) for 4, 6, 7, 8, 9, and 10 agents (different rows). Note that the plan cost was identical for all solvers, so we only show this once. Note also that $k = 0$ is standard CBS.

First, consider the plan costs. As can be seen, the k robust plans are not much more costly than a plan for the basic definition of MAPF (i.e., for $k = 0$), which indicates that k-robust solvers provide additional robustness at little increase in cost. Next, as expected, both I-kR-CBS variants runs much faster than kR-CBS and this improvement increases when increasing k and when more agents exist. Symmetric constrains clearly outperform asymmetric constraints. We conjuncture that this is due to the arbitrary way in which we choose which agent to constrain more when using the asymmetric range constraints. Future work will investigate a more intelligent way of doing so.

We also performed experiments on a larger map from Dragon Age Origins (Sturtevant 2012). Specifically, we used 90 randomly generated instances with 30 agents on the `brc202d` map, which has 43,151 vertices. This map contains many possible optimal paths for each agent, allowing the k-robust plan to often have the same cost as a plan that is not robust. When averaging over 50 random instances, the average plan cost was 3,818.35, 3,818.43, and 3,818.53 for $k = 0$, 1, and 2, respectively. Indeed, the plan cost grows with k, but negligibly. This emphasizes the usefulness of finding a k-robust plan, as one can be found in such a domain without extensive cost increase. That being said, finding k-robust plans is more time consuming. In the above experiments, finding the k-robust plans required an average of 213, 284, and 381 seconds, for $k = 0$, 1, and 2, respectively.

Discussion and Conclusion

In this paper we studied how to modify MAPF planners to cause them to generate multi-agent plans where each agent can experience until k delays while still preserving the ability to follow the generated plan.

Another direction for future work is to develop MAPF solvers that generate plans that can be followed with probability greater than a parameter (Wagner and Choset 2017), and to study more reactive execution policies.

Acknowledgments: This research was supported by the Israel and Czech Ministries of Science #8G15027.

References

Sharon, G.; Stern, R.; Felner, A.; and Sturtevant, N. R. 2015. Conflict-based search for optimal multi-agent pathfinding. *Artif. Intell.* 219:40–66.

Sturtevant, N. R. 2012. Benchmarks for grid-based pathfinding. *Computational Intelligence and AI in Games* 4(2):144–148.

Wagner, G., and Choset, H. 2017. Path Planning for Multiple Agents Under Uncertainty. In *IROS (to appear)*.