Subdimensional Expansion and Optimal Task Reassignment*

Glenn Wagner and **Howie Choset**
The Robotics Institute
Carnegie Mellon University
5000 Forbes Avenue, Pittsburgh, PA 15213

Nora Ayanian
Distributed Robotics Laboratory
Massachusetts Institute of Technology
77 Massachusetts Ave Cambridge, MA 02139

Abstract

Multirobot path planning and task assignment are traditionally treated separately, however task assignment can greatly impact the difficulty of the path planning problem, and the ultimate quality of solution is dependent upon both. We introduce *task reassignment*, an approach to optimally solving the coupled task assignment and path planning problems. We show that task reassignment improves solution quality, and reduces planning time in some situations.

Introduction

Multirobot systems are attractive for surveillance, search and rescue, and warehouse automation applications. Unfortunately, the flexibility and redundancy that make multirobot systems appealing also make assigning robots to tasks and planning collision free paths to perform those tasks difficult. In our prior work, we developed an approach called *subdimensional expansion* for efficiently generating optimal collision free paths (Wagner and Choset 2011). Subdimensional expansion seeks to decouple planning between robots, which is difficult in environments that feature bi-directional traffic passing through narrow spaces. Coupling the task assignment problem with the path planning problem allows such situations to be avoided, reducing time to find a solution and increasing solution quality.

Work on coupling path planning with task assignment focuses on distributed execution and resolving execution time conflicts (Golfarelli, Maio, and Rizzi 1997; Zheng and Koenig 2009). We are interested in decreasing computational costs for finding optimal multirobot paths. Operator Decomposition (Standley 2010) and increasing cost tree search (Sharon et al. 2011) both exploit decoupling that arises from taking the cost of the joint path as the sum of the cost of the individual robot paths. M* exploits the same decoupling at a finer level (Wagner and Choset 2011).

M*

M* is an implementation of subdimensional expansion for solving the multirobot path planning when the configuration of each robot is represented by a graph. An *individually optimal policy* is computed for each robot, which specifies the best action at a given configuration for a single robot, ignoring all other robots. A low dimensional search space is constructed by initially restricting each robot to its individually optimal policy. The search space is explored using A*, which maintains an open list of candidate vertices sorted by *f-value*, the sum of the cost to reach the vertex plus a heuristic cost to go. A robot is permitted to depart from its individual policy, thus increasing the local dimensionality of the search space, at a given joint configuration only if the A* search has found a path from the current joint configuration to a collision involving said robot. The *collision set* is the set of robots for which such a path exists from a given configuration. rM* is a variant that separates planning for disjoint subsets of colliding robots, in a manner similar to Independence Detection (Standley 2010).

Task Reassignment

M* becomes computationally expensive when large numbers of robots interact in a small area, such as narrow bottlenecks. *Task reassignment* generates alternate task assignments for robots involved in collisions to reduce the density of robot-robot interactions, which reduces planning time and improves solution quality. In this paper, we describe a method for finding the optimal combination of task assignment and path.

Task reassignment functions by maintaining a set of active task assignments $\Gamma' = \{\gamma_1, \ldots\}$, each conflated with an M* planner to solve the associated path planning problem. Search proceeds for all active assignments in an iterative manner. At each planning step t the minimum f-value in the open list of any active assignment, $f_{\text{min}}(t) = \min_{\gamma \in \Gamma'} f(\gamma(t))$, is computed. Planning for each assignment in Γ' then proceeds until the open list of the associated M* planner is *exhausted* of vertices with f-value less than or equal to $f_{\text{min}}(t)$. If new robot-robot collisions are found, additional assignments may be added to Γ' and exhausted as described above. If no collision free path has been found after all new assignments are exhausted, then there is no collision free path of cost $f_{\text{min}}(t)$. The search loop repeats with a new, larger $f_{\text{min}}(t+1)$. Task assignment will return the optimal combination of task assignment and robot paths.

Task reassignment seeks to minimize the number of tasks

*This work is supported by the MURI SMARTS and MURI ANTIDOTE programs.

Algorithm 1 Add_Assignments(γ_k)

```
colbots ← C(γ_k)
γ_l ← γ_{k+1}
while colbots ≠ ∅ do
    {γ_k^i is task assignment for i'th robot in γ_k}
    diff ← {i ∈ colbots | γ_k^i ≠ γ_l^i}
    if diff ≠ ∅ then
        Γ' ← Γ' ∪ γ_l
    colbots ← colbots\diff
    γ_l ← γ_{l+1}
```

that must be added to Γ' by exploiting lower bounds on optimal path cost that are placed by assignments in Γ' on similar assignments.

Let $f(\gamma(t))$ be the minimal f-value of all vertices in the open list of γ at the start of planning step t, which is a lower bound of the cost of the optimal path for the assignment, $c(\gamma)$. Let $h(\gamma)$ be the sum of the costs of the optimal paths for all robots ignoring robot-robot collisions. $f(\gamma(t))$ can then be written as the sum of $h(\gamma)$ and a coordination cost $a(\gamma(t))$, which is the additional cost incurred by robots deviating from their individually optimal policies to avoid collisions. As such, $a(\gamma(t))$ depends solely on the paths of the robots in the collision set of the root of the search tree $C(\gamma(t))$.

If two assignments γ_k and γ_l, with $h(\gamma_k) \leq h(\gamma_l)$, are identical for each robot in $C(\gamma_k(t))$, then $C(\gamma_k(t)) \subset C(\gamma_l(t))$, which implies that $a(\gamma_k(t)) \leq a(\gamma_l(t))$. Since $f(\gamma(t))$ is a lower bound on $c(\gamma)$, $f(\gamma_k(t)) \leq c(\gamma_l)$. In this case, we say that γ_k *bounds* γ_l at step t.

Adding new robots to $C(\gamma_k)$ during exhaustion weakens the bounds γ_k places on other assignments, which requires that new assignments be added to Γ'. The algorithm then steps along the set of all possible task assignments Γ, sorted in order of increasing $h(\gamma)$ (Kuhn 1955; Murty 1968), starting at γ_{k+1}. An assignment is added to Γ' if it is not bounded by γ_k or any assignments added earlier in the update process. Pseudo-code is given in Algorithm 1.

Task Reassignment Results

We compared the results of using M* and rM* to find a path for the heuristically cheapest task assignment to running M* and rM* with task reassignment, in problems of up to 40 robots, (see Figure 1). When M* is used as the underlying planner, the ability of task reassignment to find easily solvable assignments offsets the additional overhead of task reassignment and results in an optimal path and task assignment being found more quickly than a path can be found for the heuristically cheapest assignment. Furthermore, path cost decreased by an average of 0.31%. rM* is a more powerful planner which benefits less from discovering more easily solved task assignments than M*. Furthermore, the hierarchical nature of rM* reduces the efficiency of task reassignment, which can only occur at the highest level of the hierarchy. As a result, finding a path for the heuristically cheapest assignment with rM* is faster running rM* with task reassignment for more than 20 robots.

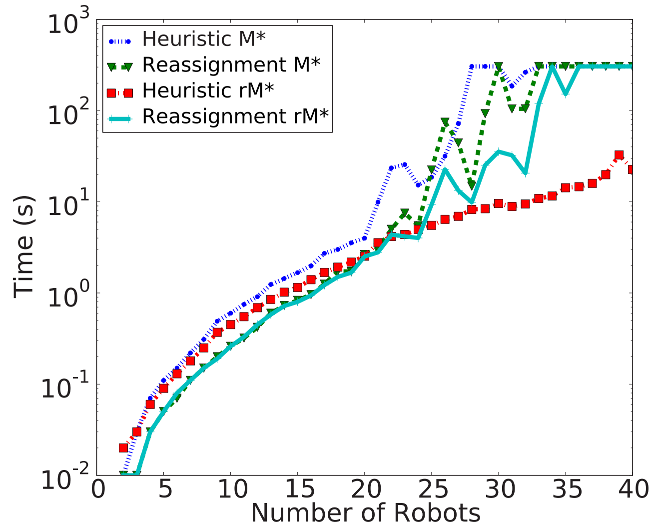

Figure 1: Median time to find a path using M* and rM*, for both the single heuristically cheapest task assignment and with task reassignment. The plateauing is the result of the planners reaching the 5 minute time limit.

Conclusions

We present task reassignment, a method for finding the optimal combination of task assignment and path for multirobot systems. We show that task reassignment imposes moderate overhead to guarantee that an optimal pair of task assignment and path will be found, which in some situations can be offset by the benefits of finding easier task assignments.

References

Golfarelli, M.; Maio, D.; and Rizzi, S. 1997. A task-swap negotiation protocol based on the contract net paradigm. Technical Report 005-97, CSITE.

Kuhn, H. 1955. The hungarian method for the assignment problem. *Naval research logistics quarterly* 2(1-2):83–97.

Murty, K. 1968. An algorithm for ranking all the assignments in order of increasing cost. *Operations Research* 682–687.

Sharon, G.; Stern, R.; Goldenberg, M.; Felner, A.; and Beer-Sheva, I. 2011. The increasing cost tree search for optimal multi-agent pathfinding. In *Twenty-Second International Joint Conference on Artificial Intelligence*.

Standley, T. 2010. Finding optimal solutions to cooperative pathfinding problems. In *Proceedings of the 24th AAAI Conference on Artificial Intelligence (AAAI)*.

Wagner, G., and Choset, H. 2011. M*: A complete multirobot path planning algorithm with performance bounds. In *IROS 2011*.

Zheng, X., and Koenig, S. 2009. K-swaps: Cooperative negotiation for solving task-allocation problems. In *International Joint Conference on Artificial Intelligence*, 373–379.