Parallel Motion Planning: Coordinating a Swarm of Labeled Robots with Bounded Stretch*

Erik D. Demaine† Sándor P. Fekete‡ Phillip Keldenich‡ Henk Meijer§ Christian Scheffer‡

Abstract

We present a collection of results for parallel motion planning, in which the objective is to reconfigure a swarm of labeled disk-shaped objects into a given target arrangement. This problem is of significant importance for a wide range of practical challenges, with potential applications to coordinated motion planning of ground robots, self-driving cars, and/or drone swarms, in addition to air traffic control, and human team coordination (e.g., in sports, military, or fire fighting).

We solve an open problem by Overmars dating back to 2006 by designing a constant-factor approximation algorithm for minimizing the execution time of a *parallel motion plan* for a rectangular grid of robots, and a desired permutation of those robots, where, in each round, every robot can move to any neighboring location whose robot is simultaneously leaving to another location. In fact, our algorithm achieves constant *stretch factor*: if all robots ultimately want to move to a location at most d units away, then the computed parallel motion plan requires only $\mathcal{O}(d)$ rounds.

Furthermore, we provide lower and upper bound results for the corresponding continuous and unlabeled versions of the problem setting.

1 Introduction

Since the beginning of computational geometry, robot motion planning and especially multi-robot coordination has received a considerable amount of attention. Even in the groundbreaking work by Schwartz and Sharir [11] from the early 1980s, one of the challenges was coordinating the motion of *several* disk-shaped objects among obstacles. Their algorithms run in time polynomial in the complexity of the obstacles, but exponential in the number of disks; moreover, it was shown by Hopcroft et al. [5] that the reachability of a given target configuration is PSPACE-complete to decide. This illustrates that a major aspect of the complexity arises not just from dealing with obstacles, but from interaction between the individual robots. In addition, a growing number of applications focus solely on robot interaction, even in settings in which obstacles are of minor importance, such as air traffic control or swarm robotics, where the goal is overall efficiency, rather than individual navigation.

With the hardness of multi-robot coordination being well known, there is still a huge demand for positive results with provable performance guarantees. In this paper, we provide significant progress in this direction, with a broad spectrum of results.

1.1 Our Results

For the problem of minimizing the total time needed to reconfigure a system of labeled circular robots in a grid environment, we give an $\mathcal{O}(1)$-approximation, i.e. bounded *stretch*, for optimal parallel motion planning, solving an open problem stated by Overmars [9] in 2006. See Theorem 1.

We extend our approach to establish constant stretch for the generalization of *colored* case, for which unlabeled disks are another special case; see Theorem 2. For the continuous case of N disks and arbitrary density, we establish a lower bound of $\Omega(N^{1/4})$ and an upper bound of $\mathcal{O}(\sqrt{N})$ on the achievable stretch, see Theorem 3 and Theorem 4.

1.2 Related Work

Multi-object motion planning problems have received a tremendous amount of attention from a wide spectrum of areas. Due to limited space, we focus on algorithmic work with a focus on geometry.

In the presence of obstacles, Aronov et al. [2] demonstrate that for up to three robots, a path can be constructed efficiently, if one exists. Schwartz and Sharir [11] consider the case of several disk-shaped objects between polygonal obstacles. They give algorithms for deciding reachability of a given target configuration. The algorithms run in time polynomial in the complexity of the obstacles, but exponential in the number of disks. Hopcroft et al. [5] prove that it is PSPACE-complete to decide reachability of a given target configuration, even when restricted to rectan-

*This work was partially supported by the DFG Reserach Unit Controlling Concurrent Change, funding number FOR 1800, project FE407/17-2, Conflict Resolution and Optimization.

†MIT Computer Science and Artificial Intelligence Laboratory, Cambridge, Massachusetts, USA, edemaine@mit.edu

‡Department of Computer Science, TU Braunschweig, Germany, {s.fekete,p.keldenich,c.scheffer}@tu-bs.de

§Science Department, University College Roosevelt, Middelburg, The Netherlands, h.meijer@ucr.nl

gular objects in a rectangular region. Dumitrescu and Jiang [4] consider minimizing the *number* of moves of a set of disks into a target arrangement without obstacles. They prove that the problem remains NP-hard for congruent disks even when the motion is restricted to sliding.

In both discrete and continuous variants of the problem, the objects can be *labeled*, *colored* or *unlabeled*. In the *colored* case, the objects are partitioned into k groups and each target position can only be covered by an object with the right color. This case was recently considered by Solovey and Halperin [12], who present and evaluate a practical sampling-based algorithm. In the *unlabeled* case, the objects are indistinguishable and each target position can be covered by any object. This scenario was first considered by Kloder and Hutchinson [6], who presented a practical sampling-based algorithm. Turpin et al. [15] prove that it is possible to find a solution in polynomial time, if one exists. This solution is optimal with respect to the longest distance traveled by any one robot. However, their results only hold for disk-shaped robots under additional restrictive assumptions on the free space. For unit disks and simple polygons, Adler et al. [1] provide a polynomial-time algorithm under the additional assumption that the start and target positions have some minimal distance from each other. Under similar distance assumptions, Solovey et al. [14] provide a polynomial-time algorithm that produces a set of paths that is no longer than $\mathrm{OPT} + 4m$, where m is the number of robots. However, they do not consider the makespan, but only the total path length. On the negative side, Solovey and Halperin [13] prove that the unlabeled multiple-object motion planning problem is PSPACE-hard, even when restricted to unit square objects in a polygonal environment.

On grid graphs, approaches for the problem (see Kunde [8] and Cheung and Lau [3]) typically assume that at least a constant number of packets can be held at any processor which means that a constant number of robots may overlap in the context of our problem setting. On the other hand, on grid graphs, the problem resembles the generalization of the 15-puzzle, for which Wagner [16] and Kornhauser et al. [7] give an efficient algorithm that decides reachability of a target configuration and provide both lower and upper bounds on the number of moves required. Ratner and Warmuth [10] prove finding a shortest solution for this puzzle remains NP-hard.

2 Preliminaries

In the grid setting considered in Section 3, robots are arranged in an $n \times m$-rectangle P which is dual to a grid graph $G = (V, E)$. A *configuration* of P is an injective mapping $C : V \to \{1, \dots, k, \perp\}$, where $\{1, \dots, k\}$ are the labels of the $k \leqslant |P|$ robots to be moved, and C does not have to be injective with respect to the empty squares denoted by $\perp$. The inverse image of a robot's label ℓ is denoted by $C^{-1}(\ell)$. d is the maximum distance between a robot's start and target position.

A configuration $C_1 : V \to \{1, \dots, k, \perp\}$ can be *transformed* into another configuration $C_2 : V \to \{1, \dots, k, \perp\}$, denoted $C_1 \to C_2$, if $C_1^{-1}(\ell) = C_2^{-1}(\ell)$ or $(C_1^{-1}(\ell), C_2^{-1}(\ell)) \in E$ holds for all $\ell \in \{1, \dots, k\}$, i.e., if each robot does not move or moves to one of the four adjacent squares. Furthermore, two robots cannot exchange their squares in one transformation step. The number of steps in a sequence of transformations is called its *makespan*. Given a start configuration C_s and a target configuration C_t, the *optimal makespan* is the minimum number of steps in a transformation sequence starting with C_s and ending with C_t.

For the continuous setting of Section 4, we consider N robots $R := \{1, \dots, N\} \subseteq \mathbb{N}$. A *movement* of a robot r is a curve $m_r : [0, T_r] \to \mathbb{R}^2$, such that $||m_r'(t)'|_2 \leqslant 1$ holds for all points in time $t \in [0, T_r]$. Let $m_i : [0, T_i] \to \mathbb{R}^2$ and $m_j : [0, T_j] \to \mathbb{R}^2$ be two movements; m_i and m_j are *compatible* if the corresponding robots do not intersect at any time.A *movement* of R is a set of compatible movements $\{m_1, \dots, m_N\}$, one for each robot. The *(continuous) makespan* of a movement $\{m_1, \dots, m_N\}$ is defined as $\max_{r \in R} T_r$. A movement $\{m_1, \dots, m_N\}$ *realizes* a pair of start and target configurations $\mathcal{S} := (\{s_1, \dots, s_N\}, \{t_1, \dots, t_N\})$ if $m_r(0) = s_r$ and $m_r(T_r) = t_r$ hold for all $r \in R$. We are searching for a movement $\{m_1, \dots, m_N\}$ realizing $\mathcal{S}$ with minimal makespan.

3 Labeled Grid Permutation

Let $n \geqslant m \geqslant 2$, $n \geqslant 3$ and let P be a $n \times m$-rectangle. By filling possibly empty squares with dummy robots, we may assume $k = |P| = nm$.

Our main result is the following:

Theorem 1 *There is an algorithm with runtime $\mathcal{O}(dmn)$ that, given an arbitrary pair of start and target configurations of an $n \times m$-rectangle with maximum distance d between any start and target position, computes a schedule of makespan $\mathcal{O}(d)$, i.e., an approximation algorithm with constant stretch.*

On a high level, our algorithm first computes the maximal Manhattan distance d between a robot's start and target position. Then we partition P into a set T of pairwise disjoint rectangular *tiles*, where each tile $t \in T$ is an $n' \times m'$-rectangle for $n', m' \leqslant 24d$. We then use an algorithm based on flows to guarantee that all robots are in their target tile, see Figure 1. Once all robots are in the correct tile, we use a sorting algorithm, *rotate sort*, for meshes simultaneously on all tiles to move each robot to the correct position within its target tile.

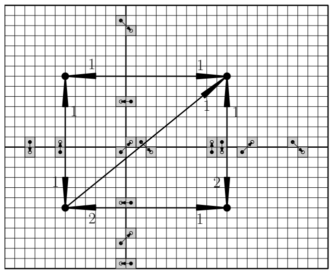


Figure 1: A tiling of an 26×32-rectangle into four tiles with $d = 1$ and the corresponding dual graph. Robots not in their target tile are illustrated by small dots. Their target positions are depicted as white disks.

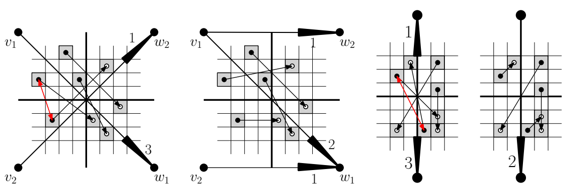
3.1 Outline of the Approximation Algorithm

We model the movements of robots between tiles as a flow f_T, using the weighted directed graph $G_T = (T, E_T, f_T)$, which is dual to the tiling T defined in the previous section. In G_T, we have an edge $(v, w) \in E_T$ if there is at least one robot that has to move from v into w. Furthermore, we define the weight $f_T((v, w))$ of an edge as the number of robots that move from v to w. As P is fully occupied, f_T is a *cyclic flow*, i.e., a flow with no sources or sinks, in which flow conservation has to hold at all vertices. We observe that G_T is a grid graph with additional diagonal edges and thus has degree at most 8. This is due to the fact that the side lengths of the tiles are larger than d as enforced by construction of the tiling.

While the maximum edge value of f_T may be $\Theta(d^2)$, only $\mathcal{O}(d)$ robots can possibly leave a tile within a single transformation step. Therefore, we decompose the flow f_T of robots into a *partition* consisting of $\mathcal{O}(d)$ *subflows*, where each individual robot's motion is modeled by exactly one subflow and each edge in the subflows has value at most d. Each subflow is then *realized* in a single transformation step. To facilitate the decomposition into subflows, we first preprocess G_T. The algorithm consists of the following subroutines:

Step 1: Compute d, the tiling T and the flow G_T, see Figure 1 for the basic idea.

Step 2: Remove intersecting and bidirectional edges from G_T, see the Figure below for the basic idea.



Step 3: Compute a partition of G_T into $\mathcal{O}(d)$ subflows with edge flows upper bounded by d.

Step 4: Realize the $\mathcal{O}(d)$ subflows using $\mathcal{O}(d)$ transformation steps, see Figures 2 and 3 for the basic ideas.

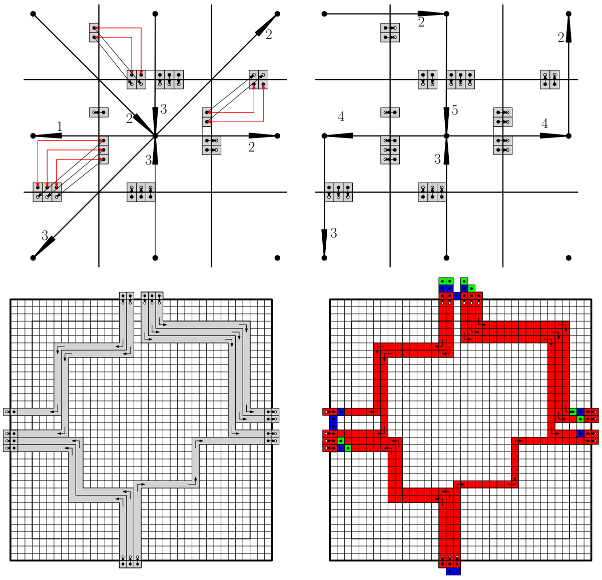


Figure 2: Remove diagonal edges (top) and than apply the main approach (bottom) for realizing a subflow.

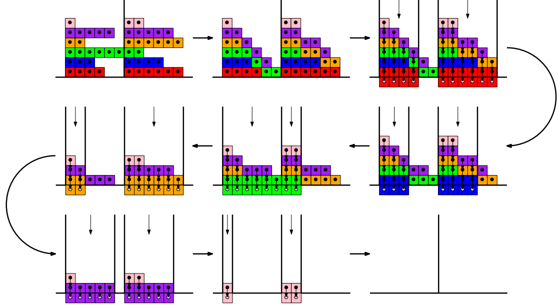


Figure 3: Realizing a sequence of subflows by stacking the rows of robots to be moved onto each other in the order in which the subflows are realized. Each color indicates a separate subflow.

Step 5: Simultaneously apply a sorting algorithm, *rotate sort*, for meshes to all tiles, moving each robot to its target position.

4 Variants on Labeling

A different version is the unlabeled variant, in which all robots are the same. A generalization of both this and the labeled version arises when robots belong to one of k color classes, with robots from the same color class being identical.

Theorem 2 *There is an algorithm with running time $\mathcal{O}(k(mn)^{1.5}\log(mn) + dmn)$ that computes, given start and target images I_s, I_t with maximum distance d between any start and target position, an $\mathcal{O}(1)$-approximation of the optimal makespan M and a corresponding sequence of transformation steps.*

The basic idea is to transform the given labeled problem setting into an unlabeled problem setting by solving a geometric bottleneck problem.

5 Continuous Motion

The continuous geometric case considers N unit disks that have to move into a target configuration in the plane; the velocity of each robot is bounded by 1, and we want to minimize the makespan. For dense arrangements of disks, we can show that constant stretch can *not* be achieved.

Theorem 3 *There is an instance with optimal makespan $M \in \Omega(N^{1/4}) = \Omega(dN^{1/4})$ where $d \in \Theta(1)$, see Figure 4.*

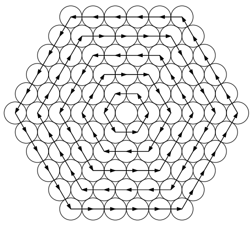


Figure 4: The start and target configurations of our lower-bound construction.

The basic idea of the proof of Theorem 3 is the following. Let $\{m_1, \dots, m_N\}$ be an arbitrary movement with makespan M. We show that there must be a point in time $t \in [0, M]$ where the area of $\text{Conv}(m_1(t), \dots, m_N(t))$ is lower-bounded by $cN + \Omega(N^{3/4})$, where cN is the area of $\text{Conv}(m_1(0), \dots, m_N(0))$. Assume $M \in o\left(N^{1/4}\right)$ and consider the area of $\text{Conv}(m_1(t'), \dots, m_N(t'))$ at some point $t' \in [0, M]$. This area is at most $cN + \mathcal{O}(\sqrt{N}) \cdot o\left(N^{1/4}\right)$ which is a contradiction.

On the other hand, we can give a non-trivial but non-constant upper bound on the possible stretch.

Theorem 4 *There is an algorithm that computes a movement plan with continuous makespan in $\mathcal{O}(d + \sqrt{N})$. If $d \in \Omega(1)$, this implies a $\mathcal{O}(\sqrt{N})$-approximation algorithm.*

The approach of Theorem 4 applies an underlying grid with mesh size $2\sqrt{2}$. Our algorithm (1) moves the robots to vertices of the grid, (2) applies our $\mathcal{O}(1)$-approximation for the discrete case, and (3) moves the robots from the vertices of the grid to their targets.

References

[1] A. Adler, M. de Berg, D. Halperin, and K. Solovey. Efficient multi-robot motion planning for unlabeled discs in simple polygons. In *Algorithmic Foundations of Robotics XI*, pages 1–17. Springer, 2015.

[2] B. Aronov, M. de Berg, A. F. van der Stappen, P. Švestka, and J. Vleugels. Motion planning for multiple robots. *Discrete & Computational Geometry*, 22(4):505–525, 1999.

[3] S. Cheung and F. C. M. Lau. Mesh permutation routing with locality. *Information Processing Letters*, 43(2):101–105, 1992.

[4] A. Dumitrescu and M. Jiang. On reconfiguration of disks in the plane and related problems. *Computational Geometry: Theory and Applications*, 46:191–202, 2013.

[5] J. E. Hopcroft, J. T. Schwartz, and M. Sharir. On the complexity of motion planning for multiple independent objects; PSPACE-hardness of the warehouseman's problem. *Int. J. Robotics Research*, 3(4):76–88, 1984.

[6] S. Kloder and S. Hutchinson. Path planning for permutation-invariant multi-robot formations. In *IEEE Trans. Robotics*, volume 22, pages 650–665. IEEE, 2006.

[7] D. Kornhauser, G. Miller, and P. Spirakis. Coordinating pebble motion on graphs, the diameter of permutation groups, and applications. In *Annual Symposium on Foundations of Computer Science, 1984*, SFCS '84, pages 241–250, 1984.

[8] M. Kunde. Routing and sorting on mesh-connected arrays. In *VLSI Algorithms and Architectures: 3rd Aegean Workshop on Comp. (AWOC 88)*, pages 423–433. Springer, 1988.

[9] M. Overmars. Contributed open problem. In S. P. Fekete, R. Fleischer, R. Klein, and A. Lopez-Ortiz, editors, *Algorithmic Foundations of Programmable Matter, Dagstuhl Seminar 06421*, 2006. http://www.dagstuhl.de/de/programm/kalender/semhp/?semnr=06421.

[10] D. Ratner and M. K. Warmuth. Finding a shortest solution for the $N \times N$ extension of the 15-puzzle is intractable. In *Proc. AAAI Conf. Artificial Intelligence*, pages 168–172, 1986.

[11] J. T. Schwartz and M. Sharir. On the piano movers' problem: III. Coordinating the motion of several independent bodies: the special case of circular bodies moving amidst polygonal barriers. *Int. J. Robotics Research*, 2(3):46–75, 1983.

[12] K. Solovey and D. Halperin. k-color multi-robot motion planning. *Int. J. Robotics Research*, 33(1):82–97, 2014.

[13] K. Solovey and D. Halperin. On the hardness of unlabeled multi-robot motion planning. In *Robotics: Science and Systems (RSS)*, 2015.

[14] K. Solovey, J. Yu, O. Zamir, and D. Halperin. Motion planning for unlabeled discs with optimality guarantees. In *Robotics: Science and Systems (RSS)*, 2015.

[15] M. Turpin, N. Michael, and V. Kumar. Trajectory planning and assignment in multirobot systems. In *Algorithmic foundations of robotics X*, pages 175–190. Springer, 2013.

[16] R. M. Wilson. Graph puzzles, homotopy, and the alternating group. *Journal of Combinatorial Theory, Series B*, 16(1):86–96, 1974.