The Parameterized Complexity of Coordinated Motion Planning

Eduard Eiben
Department of Computer Science, Royal Holloway, University of London, Egham, UK

Robert Ganian
Algorithms and Complexity Group, TU Wien, Austria

Iyad Kanj
School of Computing, DePaul University, Chicago, IL, USA

Abstract

In Coordinated Motion Planning (CMP), we are given a rectangular-grid on which k robots occupy k distinct starting gridpoints and need to reach k distinct destination gridpoints. In each time step, any robot may move to a neighboring gridpoint or stay in its current gridpoint, provided that it does not collide with other robots. The goal is to compute a schedule for moving the k robots to their destinations which minimizes a certain objective target – prominently the number of time steps in the schedule, i.e., the makespan, or the total length traveled by the robots. We refer to the problem arising from minimizing the former objective target as CMP-M and the latter as CMP-L. Both CMP-M and CMP-L are fundamental problems that were posed as the computational geometry challenge of SoCG 2021, and CMP also embodies the famous $(n^2 - 1)$-puzzle as a special case.

In this paper, we settle the parameterized complexity of CMP-M and CMP-L with respect to their two most fundamental parameters: the number of robots, and the objective target. We develop a new approach to establish the fixed-parameter tractability of both problems under the former parameterization that relies on novel structural insights into optimal solutions to the problem. When parameterized by the objective target, we show that CMP-L remains fixed-parameter tractable while CMP-M becomes para-NP-hard. The latter result is noteworthy, not only because it improves the previously-known boundaries of intractability for the problem, but also because the underlying reduction allows us to establish – as a simpler case – the NP-hardness of the classical Vertex Disjoint and Edge Disjoint Paths problems with constant path-lengths on grids.

2012 ACM Subject Classification Theory of computation → Parameterized complexity and exact algorithms

Keywords and phrases coordinated motion planning, multi-agent path finding, parameterized complexity, disjoint paths on grids

Digital Object Identifier 10.4230/LIPIcs.SoCG.2023.28

Funding *Robert Ganian*: Project No. Y1329 of the Austrian Science Fund (FWF), Project No. ICT22-029 of the Vienna Science Foundation (WWTF).
Iyad Kanj: DePaul URC Grants 606601 and 350130.

1 Introduction

Who among us has not struggled through solving the 15-puzzle? Given a small square board, tiled with 15 tiles numbered $1, \ldots, 15$, and a single hole in the board, the goal of the puzzle is to slide the tiles in order to reach the final configuration in which the tiles appear in (sorted) order; see Figure 1 for an illustration. The 15-puzzle has been generalized to an $n \times n$ square-board, with tiles numbered $1, \ldots, n^2 - 1$. Unsurprisingly, this generalization is called the $(n^2 - 1)$-puzzle. Whereas deciding whether a solution to an instance of the $(n^2 - 1)$-puzzle exists (i.e., whether it is possible to sort the tiles starting from an initial configuration) is in P [20], determining whether there is a solution that requires at most $\ell \in \mathbb{N}$ tile moves has been shown to be NP-hard [9, 26].

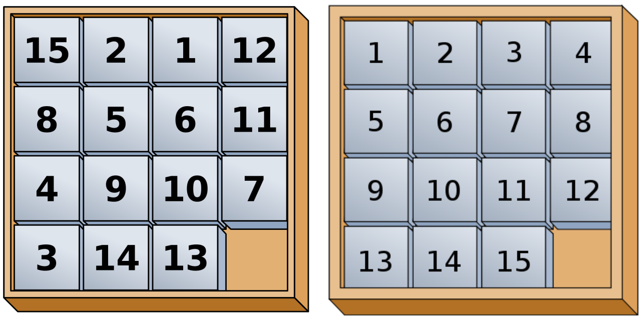
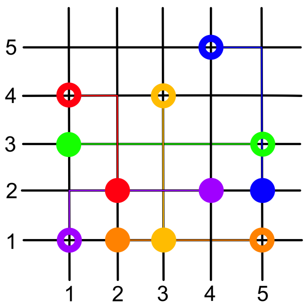


Figure 1 The left figure shows an initial configuration of the 15-puzzle and the right figure shows the desirable final configuration. Source: `https://en.wikipedia.org/wiki/15_puzzle`.

Deciding whether an $(n^2 - 1)$-puzzle admits a solution is a special case of Coordinated Motion Planning (CMP), a prominent task originating from robotics which has been extensively studied in the fields of Computational Geometry and Artificial Intelligence (where it is often referred to as Multi-Agent Path Finding). In CMP, we are given an $n \times m$ rectangular-grid on which k robots occupy k distinct starting gridpoints and need to reach k distinct destination gridpoints. Robots may move simultaneously at each time step, and at each time step, a robot may move to a neighboring gridpoint or stay in its current gridpoint provided that (in either case) it does not collide with any other robots; two robots collide if they are occupying the same gridpoint at the end of a time step, or if they are traveling along the same grid-edge (in opposite directions) during the same time step. We are also given an objective target, and the goal is to compute a schedule for moving the k robots to their destination gridpoints which satisfies the specified target. The two objective targets we consider here are (1) the number of time steps used by the schedule (i.e., the makespan), and (2) the total length traveled by all the robots (also called the "total energy", e.g., in the SoCG 2021 Challenge [1]); the former gives rise to a problem that we refer to as CMP-M, while we refer to the latter as CMP-L. An illustration is provided in Figure 2.



Time	robots					
	Green	Red	Blue	Purple	Orange	Yellow
0	(1,3)	(2,2)	(5,2)	(4,2)	(2,1)	(3,1)
1	(2,3)	(2,2)	(5,3)	(3,2)	(2,1)	(3,1)
2	(3,3)	(2,3)	(5,4)	(2,2)	(3,1)	(3,2)
3	(4,3)	(2,4)	(5,5)	(1,2)	(4,1)	(3,3)
4	(5,3)	(1,4)	(4,5)	(1,1)	(5,1)	(3,4)

Figure 2 An illustration (left) of an instance of CMP-M with six robots, indicated using distinct colors (blue, green, yellow, red, orange, purple), and a makespan $\ell = 4$. The starting points are marked using a disk shape (filled circle) and destination points using an annular shape. A schedule indicating each of the robot's position at each of the four time steps is shown in the table (right).

In this paper, we settle the parameterized complexity of CMP-M and CMP-L with respect to their two most fundamental parameters: the number k of robots, and the objective target. In particular, we obtain fixed-parameter algorithms for both problems when parameterized by k and for CMP-L when parameterized by the target, but show that CMP-M remains NP-hard even for fixed values of the target. Given how extensively CMP has been studied in the literature (see the related work below), we consider it rather surprising that fundamental questions about the problem's complexity have remained unresolved. We believe that one aspect contributing to this gap in our knowledge was the fact that, even though the problems seem deceptively easy, it was far from obvious how to obtain exact and provably optimal algorithms in the parameterized setting. Furthermore, en route to the aforementioned intractability result, we establish the NP-hardness of the classical Vertex Disjoint Paths and Edge Disjoint Paths problems on grids when restricted to bounded-length paths.

1.1 Related Work

CMP has been extensively studied by researchers in the fields of computational geometry, AI/Robotics, and theoretical computer science in general. In particular, CMP-M and CMP-L were posed as the Third Computational Geometry Challenge of SoCG 2021, which took place during the Computational Geometry Week in 2021 [1]. The CMP problem generalizes the (n^2-1)-puzzle, which was shown to be NP-hard as early as 1990 by Ratner and Warmuth [26]. A simpler NP-hardness proof was given more recently by Demaine et al. [9]. Several recent papers studied the complexity of CMP with respect to optimizing various objective targets, such as: the makespan, the total length traveled, the maximum length traveled (over all robots), and the total arrival time [2, 8, 15, 32]. The continuous geometric variants of CMP, in which the robots are modeled as geometric shapes (e.g., disks) in a Euclidean environment, have also been extensively studied [3, 8, 12, 25, 27]. Finally, we mention that there is a plethora of works in the AI and Robotics communities dedicated to variants of the CMP problem, for both the continuous and the discrete settings [4, 17, 28, 29, 31, 33, 34].

The fundamental vertex and edge disjoint paths problems have also been thoroughly studied, among others due to their connections to graph minors theory. The complexity of both problems on grids was studied as early as in the 1970's motivated by its applications in VLSI design [13, 21, 24, 30], with more recent results focusing on approximation [5, 6].

1.2 High-Level Overview of Our Results and Contributions

As our first set of results, we show that CMP-M and CMP-L are fixed-parameter tractable (FPT) parameterized by the number k of robots, i.e., can be solved in time $f(k) \cdot n^{\mathcal{O}(1)}$ for some computable function f and input size n. Both results follow a two-step approach for solving each of these problems. In the first step, we obtain a structural result revealing that every YES-instance of the problem has a *canonical* solution in which the number of "turns" (i.e., changes in direction) made by any robot-route is upper bounded by a function of the parameter k; this structural result is important in its own right, and we believe that its applications extend beyond this paper. This first step of the proof is fairly involved and revolves around introducing the notion of "slack" to partition the robots into two types, and then exploiting this notion to reroute the robots so that their routes form a canonical solution. In the second step, we show that it is possible to find such a canonical solution (or determine that none exists) via a combination of delicate branching and solving subinstances of Integer Linear Programming (ILP) in which the number of variables is upper bounded by a function of the parameter k; fixed-parameter tractability then follows since the latter can be solved in FPT-time thanks to Lenstra's result [14, 16, 19].

Next, we consider the other natural parameterization of the problem: the objective target. For CMP-L, this means parameterizing by the total length traveled, and there we establish fixed-parameter tractability via exhaustive branching. The situation becomes much more intriguing for CMP-M, where we show that the problem remains NP-hard even when the target makespan is a fixed constant. As a by-product of our reduction, we also establish the NP-hardness of the classical Vertex and Edge Disjoint Paths problems on grids when restricted to bounded-length paths.

The contribution of our intractability results are twofold. First, the NP-hardness of CMP with constant makespan is the first result showing its NP-hardness in the case where one of the parameters is a fixed constant. As such, it refines and strengthens several existing NP-hardness results for CMP [2, 8, 15]. It also answers the open questions in [15] about the complexity of the problem in restricted settings where the optimal path of each robot passes through a constant number of starting/destination points, or where the overlap between any two optimal paths is upper bounded by a constant, by directly implying their NP-hardness. Second, the NP-hardness results for the bounded-length vertex and edge disjoint paths problems on grids also refine and deepen several intractability results for these problems. All previous NP-hardness (and APX-hardness) results for the vertex and edge disjoint paths problems on grids [2, 6, 8, 9, 15, 21, 24, 26] yield instances in which the path length is unbounded. Last but not least, we believe that the NP-hardness results we derive are of independent interest, and have the potential of serving as a building block in NP-hardness proofs for problems in geometric and topological settings, where it is very common to start from a natural problem whose restriction to instances embedded on a grid remains NP-hard.

2 Preliminaries and Problem Definition

We use standard terminology for graph theory [10] and assume basic familiarity with the parameterized complexity paradigm including, in particular, the notions of *fixed-parameter tractability* and *para-NP-hardness* [7, 11]. For $n \in \mathbb{N}$, we write $[n]$ for the set $\{1, \ldots, n\}$.

Let G be an $n \times m$ rectangular grid, where $n, m \in \mathbb{N}$. Let $\{R_i \mid i \in [k]\}$, $k \in \mathbb{N}$, be a set of robots that will move on G. Each R_i, $i \in [k]$, is associated with a starting gridpoint s_i and a destination gridpoint t_i in $V(G)$, and hence can be specified as the pair $R_i = (s_i, t_i)$; we assume that all the s_i's are pairwise distinct and that all the t_i's are pairwise distinct, and we denote by $\mathcal{R} = \{(s_i, t_i) \mid i \in [k]\}$ the set of all robots. At each time step, a robot may either stay at the gridpoint it is currently on, or move to an adjacent gridpoint, and robots may move simultaneously. We reference the sequence of moves of the robots using a time frame $[0, t]$, $t \in \mathbb{N}$, and where in time step $x \in [0, t]$ each robot remains stationary or moves.

Let a *route* for R_i be a tuple $W_i = (u_0, \ldots, u_t)$ of vertices in G such that (i) $u_0 = s_i$ and $u_t = t_i$ and (ii) $\forall j \in [t]$, either $u_{j-1} = u_j$ or $u_{j-1}u_j \in E(G)$. Intuitively, W_i corresponds to a "walk" in G, with the exception that consecutive vertices in W_i may be identical (representing waiting time steps), in which R_i begins at its starting point at time step 0, and is at its destination point at time step t. Two routes $W_i = (u_0, \ldots, u_t)$ and $W_j = (v_0, \ldots, v_t)$, where $i \neq j \in [k]$, are *non-conflicting* if (i) $\forall r \in \{0, \ldots, t\}$, $u_r \neq v_r$, and (ii) $\nexists r \in \{0, \ldots, t-1\}$ such that $v_{r+1} = u_r$ and $u_{r+1} = v_r$. Otherwise, we say that W_i and W_j *conflict*. Intuitively, two routes conflict if the corresponding robots are at the same vertex at the end of a time step, or go through the same edge (in opposite directions) during the same time step.

A *schedule* $\mathcal{S}$ for $\mathcal{R}$ is a set of routes $W_i, i \in [k]$, during a time interval $[0, t]$, that are pairwise non-conflicting. The integer t is called the *makespan* of $\mathcal{S}$. The (*traveled*) *length* of a route (or its associated robot) within $\mathcal{S}$ is the number of time steps j such that $u_j \neq u_{j+1}$, and the *total traveled length* of a schedule is the sum of the lengths of its routes.

We are now ready to define the problems under consideration.

Coordinated Motion Planning with Makespan Minimization (CMP-M)
Given: An $n \times m$ rectangular grid G, where $n, m \in \mathbb{N}$, and a set $\mathcal{R} = \{(s_i, t_i) \mid i \in [k]\}$ of pairs of gridpoints of G where the s_i's are distinct and the t_i's are distinct; $k, \ell \in \mathbb{N}$.
Question: Is there a schedule for $\mathcal{R}$ of makespan at most ℓ?

The Coordinated Motion Planning with Length Minimization problem (CMP-L) is defined analogously but with the distinction being that, instead of ℓ, we are given an integer λ and are asked for a schedule of total traveled length at most λ. For an instance $\mathcal{I}$ of CMP-M or CMP-L, we say that a schedule is *valid* if it has makespan at most ℓ or has total traveled length at most λ, respectively. We remark that even though both CMP-M and CMP-L are stated as decision problems, all the algorithms provided in this paper are constructive and can output a valid schedule (when it exists) as a witness.

We will assume throughout the paper that $k \geq 2$; otherwise, both problems can be solved in linear time. Furthermore, we remark that the inputs can be specified in $\mathcal{O}(k \cdot (\log n + \log m) + \log \ell)$ (or $+ \log \lambda$) bits, and our fixed-parameter algorithms work seamlessly even if the inputs are provided in such concise manner. On the other hand, the lower-bound result establishes "strong" NP-hardness of the problem (i.e., also applies to cases where the input contains a standard encoding of G as a graph).

For two gridpoints $p = (x_p, y_p)$ and $q = (x_q, y_q)$, the Manhattan distance between p and q, denoted $\Delta(p, q)$, is $\Delta(p, q) = |x_p - x_q| + |y_p - y_q|$. For two robots $R_i, R_j \in \mathcal{R}$ and a time step $x \in \mathbb{N}$, denote by $\Delta_x(R_i, R_j)$ the Manhattan distance between the grid points at which R_i and R_j are located at time step x. The following notion will be used in several of our algorithms:

▶ **Definition 1.** Let $(G, \mathcal{R}, k, \bullet)$ be an instance of CMP-M or CMP-L and let $T = [t_1, t_2]$ for $t_1, t_2 \in \mathbb{N}$. For a robot R_i with corresponding route W_i, let u_p and u_q be the gridpoints in W_i at time steps t_1 and t_2, respectively. Define the *slack* of R_i w.r.t. T, denoted $\texttt{slack}_T(R_i)$, as $(t_2 - t_1) - \Delta(u_p, u_q)$ (alternatively, $(q - p) - \Delta(u_p, u_q)$).

Observe that the slack measures the amount of time (i.e., number of time steps) that robot R_i "wastes" when going from u_p to u_q relative to the shortest time needed to get from u_p to u_q. For a robot R_i with route W_i, for convenience we write $\texttt{slack}_T(W_i)$ for $\texttt{slack}_T(R_i)$. When dealing with CMP-M, we write $\texttt{slack}(R_i)$ as shorthand for $\texttt{slack}_{[0,\ell]}(R_i)$, and when dealing with CMP-L, we write $\texttt{slack}(R_i)$ as shorthand for $\texttt{slack}_{[0,\lambda]}(R_i)$.

3 CMP Parameterized by the Number of Robots

In this section, we establish the fixed-parameter tractability of CMP-M and CMP-L parameterized by the number k of robots. Both results follow the two-step approach outlined in Subsection 1.2: showing the existence of a canonical solution, and then reducing the problem via branching to a tractable fragment of Integer Linear Programming. These two steps are described for CMP-M in Subsections 3.1 and 3.2, while Subsection 3.3 shows how the same technique is used to establish the fixed-parameter tractability of CMP-L.

3.1 Canonical Solutions for CMP-M

We begin with a few definitions that formalize some intuitive notions such as "turns".

Let $W = (u_0, \ldots, u_\ell)$, where $\ell > 2$, be a route in an $n \times m$ grid G, where $n, m \in \mathbb{N}$. We say that W makes a *turn* at $u_i = (x_i, y_i)$, where $i \in \{1, \ldots, \ell - 1\}$, if the two vectors $\overrightarrow{u_{i-1}u_i}$ and $\overrightarrow{u_i u_{i+1}}$ have different orientations (i.e., either one is horizontal and the other is vertical, or they are parallel but have opposite directions). We write $\langle u_{i-1}, u_i, u_{i+1}\rangle$ for the turn at u_i. A turn $\langle u_{i-1}, u_i, u_{i+1}\rangle$ is a *U-turn* if $\overrightarrow{u_{i-1}u_i} = -\overrightarrow{u_i u_{i+1}}$; otherwise, it is a *non U-turn*. The *number of turns* in W, denoted $\nu(W)$, is the number of vertices in W at which it makes turns. A sequence $M = [u_i, \dots, u_j]$ of consecutive turns is said to be *monotone* if all the turns in each of the two alternating sequences $[u_i, u_{i+2}, u_{i+4}, \dots]$ and $[u_{i+1}, u_{i+3}, u_{i+5}, \dots]$, in which M can be partitioned, have the same direction (see Figure 3).

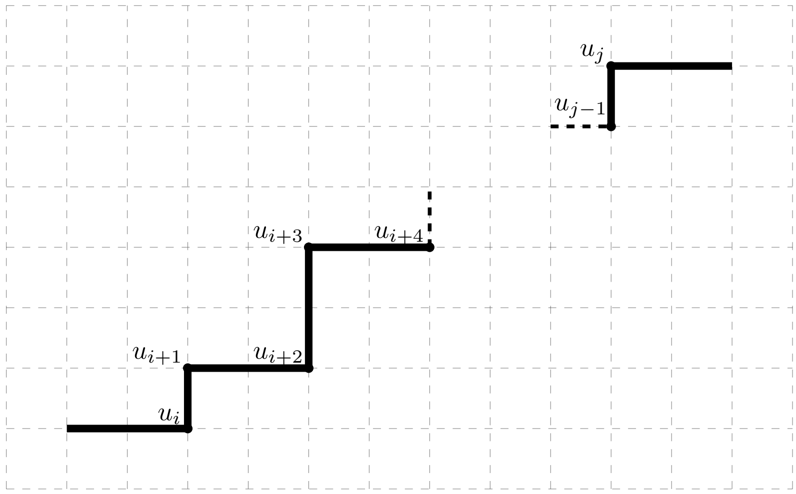


Figure 3 Illustration of a monotone sequence of consecutive turns.

Let $T = [t_1, t_2] \subseteq [0, \ell]$. We say that a route W_i for R_i has *no slack* in T if $\texttt{slack}_T(R_i) = 0$; that is, robot R_i does not "waste" any time and always progresses towards its destination during T. The following observation is straightforward:

▶ **Observation 2.** *Let W_i be a route for R_i and $T \subseteq [0, \ell]$ be a time interval such that $\texttt{slack}_T(R_i) = 0$. The sequence of turns that W_i makes during T is a monotone sequence (and in particular does not include any U-turns).*

Let $W_i = (s_i = u_0, \dots, u_t = t_i)$ be a route for R_i in a valid schedule $\mathcal{S}$ of a YES-instance of CMP-M or CMP-L, and let $W = (u_q, u_{q+1}, \dots, u_r)$ be the subroute of W_i during a time interval $T \subseteq [0, t]$. We say that a route $W' = (v_q, \dots, v_r)$ is *equivalent to* W if: (i) $v_q = u_q$ and $v_r = u_r$ (i.e., both routes have the same starting and ending points); (ii) $|W| = |W'|$; and (iii) replacing W_i in $\mathcal{S}$ with the route $(s_i = u_0, \dots, u_{q-1}, v_q, \dots, v_r, u_{r+1}, \dots, u_t = t_i)$ still yields a valid schedule of the instance.

▶ **Definition 3.** Let $\mathcal{I} = (G, \mathcal{R}, k, \bullet)$ be a YES-instance of CMP-M or CMP-L. A valid schedule $\mathcal{S}$ for $(G, \mathcal{R}, k, \bullet)$ is *minimal* if the sum of the number of turns made by all the routes in $\mathcal{S}$ is minimum over all valid schedules of $\mathcal{I}$.

The following lemma is the building block for the crucial Lemma 5, which will establish the existence of a canonical solution (for a YES-instance) in which the number of turns made by "small-slack" robots is upper bounded by a function of the parameter. This is achieved by a careful application of a "cell flattening" operation depicted in Figure 4.

More specifically, we show that if in a solution a robot has no slack during a time interval but its route makes a "large" number of turns, then there exists a "cell" corresponding to a turn in its route that can be flattened, resulting in another (valid) solution with fewer turns.

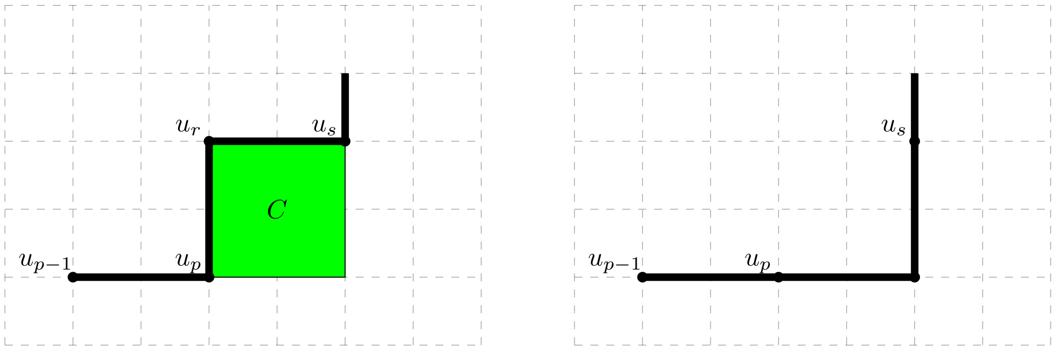


Figure 4 Illustration of a *cell* in a route (left) and its *flattening* (right).

▶ **Lemma 4.** *Let $\mathcal{S}$ be a minimal (valid) schedule for a* YES*-instance of* CMP-M. *Let W_i be a route in $\mathcal{S}$ and $T_i \subseteq [0, \ell]$ be a time interval during which W_i has no slack. Then there is an equivalent route, W'_i, to W_i such that the number of turns that W'_i makes during T_i, $\nu_{T_i}(W'_i)$, satisfies $\nu_{T_i}(W'_i) \leq 3k^k$.*

By carefully subdividing a time interval into roughly $\sigma(k)$ subintervals, for a function $\sigma(k)$ that upper bounds the slack of a robot, and applying Lemma 4 to each of these subintervals, we can extend the result in Lemma 4 to robots whose slack is upper bounded by $\sigma(k)$:

▶ **Lemma 5.** *Let $(G, \mathcal{R}, k, \ell)$ be a* YES*-instance of* CMP-M, *and let $T_i \subseteq [0, \ell]$. Then $(G, \mathcal{R}, k, \ell)$ has a minimal schedule such that, for each R_i, $i \in [k]$, satisfying $\texttt{slack}_{T_i}(W_i) \leq \sigma(k)$ for an arbitrary function σ, its route W_i satisfies $\nu_{T_i}(W_i) \leq \tau(k)$, where $\tau(k) = 3k^k(\sigma(k) + 1) + \sigma(k)$.*

Lemma 5 already provides us with the property we need for "small-slack" robots: their number of turns can be upper-bounded by a function of the parameter. We still need to deal with the more complicated situation of "large-slack" robots. Our next course of action will be establishing the existence of a sufficiently large time interval during which the "large-slack" robots are far from the "small-slack" ones. We begin with an observation linking the slack of two robots that "travel together".

▶ **Observation 6.** *Let $R, R' \in \mathcal{R}$ and let $T = [t_1, t_2] \subseteq [0, \ell]$. Let u, u' be the gridpoints at which R and R' are located at time step t_1, respectively, and v, v' those at which R and R' are located at time t_2, respectively. Suppose that $\Delta(u, u') \leq d(k)$ and $\Delta(v, v') \leq d(k)$, for some function $d(k)$. Then $\texttt{slack}_T(R') \leq \texttt{slack}_T(R) + 2d(k)$.*

Intuitively speaking, the above observation implies that a robot with a large slack in some time interval cannot be close to a robot with a small slack for the whole interval (otherwise, both robots would be moving at "comparable speeds", which would contradict that one of them has a small slack and the other a large-slack).

Next, we observe that either the slack of all the robots can be upper-bounded by a function h, or there is a sufficiently large multiplicative gap between the slack of some robots. This will allow us to partition the set of robots into those with small or large slack. For any function h, let $h^{(j)} = \underbrace{h \circ \cdots \circ h}_{j \text{ times}}$ denote the composition of h with itself j times.

▶ **Lemma 7.** *Let $(G, \mathcal{R}, k, \ell)$ be an instance of* CMP-M *and let $T \subseteq [0, \ell]$. Let $h(k)$ be any computable function satisfying $h^{(p)}(k) \leq h^{(q)}(k)$ for $p \leq q \in [k]$. Then either $\texttt{slack}_T(R_i) \leq h^{(k)}(k)$ for every $i \in [k]$, or there exists $j \in \mathbb{N}$ with $2 \leq j \leq k$, such that $\mathcal{R}$ can be partitioned into $(\mathcal{R}_S, \mathcal{R}_L)$ where $\mathcal{R}_L \neq \emptyset$, $\texttt{slack}_T(R) \leq h^{(j-1)}(k)$ for every $R \in \mathcal{R}_S$, and $\texttt{slack}_T(R') > h^{(j)}(k)$ for every $R' \in \mathcal{R}_L$.*

The next definition yields a time interval with the property that small-slack robots are sufficiently far from large-slack ones during that interval. Such an interval will be useful, since within it we will be able to re-route the large-slack robots (which are somewhat flexible) to reduce the number of turns they make, while avoiding collision with small-slack robots.

▶ **Definition 8.** Let $\sigma(k), \gamma(k), d(k)$ be functions such that $\sigma(k) < \gamma(k)$. An interval $T = [t_1, t_2] \subseteq [0, \ell]$ is a $[\sigma, \gamma]$-*good interval* w.r.t. $d(k)$ if $\mathcal{R}$ can be partitioned into $\mathcal{R}_S$ and $\mathcal{R}_L$ such that: (i) every $R \in \mathcal{R}_S$ satisfies $\texttt{slack}_T(R) \leq \sigma(k)$ and every $R' \in \mathcal{R}_L$ satisfies $\texttt{slack}_T(R') \geq \gamma(k)$; (ii) for every time step $t \in T$, $\Delta_t(R, R') \geq d(k)$ for every $R \in \mathcal{R}_S$ and every $R' \in \mathcal{R}_L$; and (iii) there exists a robot $R_i \in \mathcal{R}_L$ such that $\nu_T(W_i) > 3k^k(\sigma(k)+1)+\sigma(k)$. If the function $d(k)$ is specified or clear from the context, we will simply say that T is a $[\sigma, \gamma]$-good interval (and thus omit writing "w.r.t. $d(k)$").

The following key lemma asserts the existence of a good interval assuming the solution contains a robot that makes a large number of turns:

▶ **Lemma 9.** *Let $(G, \mathcal{R}, k, \ell)$ be a* YES*-instance of* CMP-M *and let $\mathcal{S}$ be a minimal schedule for $(G, \mathcal{R}, k, \ell)$. If there exists $R' \in \mathcal{R}$ with route W' such that $\nu(W') > 3^{k^3+1} \cdot (3k^k \cdot (3^{13^{2k^2-2k}} \cdot k^{13^{2k^2-2k}} + 1) + 3^{13^{2k^2-2k}} \cdot k^{13^{2k^2-2k}})$, then there exists a $[\sigma, \gamma]$-good interval $T \subseteq [0, \ell]$ w.r.t. a function $d(k)$ such that $k^{13^{k-1}} \leq \sigma(k) \leq 3^{13^{2k^2-2k}} \cdot k^{13^{2k^2-2k}}$, and $d(k) = \gamma(k) = \sigma^{13}(k)$.*

Once we fix a good interval T, we can finally formalize/specify what it means for a robot to have small or large slack within T:

▶ **Definition 10.** Let $T = [t_1, t_2] \subseteq [0, \ell]$ be a $[\sigma, \gamma]$-good interval with respect to some function $d(k)$, where $\sigma(k) < \gamma(k)$ are two functions, and let $R_i \in \mathcal{R}$. We say that R_i is a T-*large slack robot* if $\texttt{slack}_T(R_i) \geq \gamma(k)$; otherwise, $\texttt{slack}_T(R_i) \leq \sigma(k)$ and we say that R_i is a T-*small slack robot*.

At this point, we are finally ready to prove Lemma 11, which is the core tool that establishes the existence of a solution with a bounded number of turns (w.r.t. the parameter), even in the presence of large-slack robots: for each solution with too many turns, we can produce a different one with strictly less turns. Note that if one simply replaces the routes of large-slack robots so as to reduce their number of turns, then the new routes may bring the large-slack robots much closer to the small-sack robots and hence may lead to collisions. Therefore, the desired rerouting scheme needs to be carefully designed, and it exploits the properties of a good interval: property (i) is used to reorganize and properly reroute these robots, while property (ii) is used to avoid collisions.

▶ **Lemma 11.** *Let $(G, \mathcal{R}, k, \ell)$ be a* YES*-instance of* CMP-M *and let $\mathcal{S}$ be a minimal schedule for $(G, \mathcal{R}, k, \ell)$. Let $T = [t_1, t_2] \subseteq [0, \ell]$ be a $[\sigma, \gamma]$-good interval with respect to $d(k)$, where $k^{13^{k-1}} \leq \sigma(k) \leq 3^{13^{2k^2-2k}} \cdot k^{13^{2k^2-2k}}$, and $d(k) = \gamma(k) = \sigma^{13}(k)$. For every T-large-slack robot R_i, there is a route W'_i that is equivalent to W_i and such that $\nu_T(W'_i)$ is at most $3k^3$ and W'_i is identical to W_i in $[0, \ell] \setminus T$.*

We now establish the canonical-solution result that forms the culmination of this section.

▶ **Theorem 12.** *Let $(G, \mathcal{R}, k, \ell)$ be an instance of* CMP-M *such that at least one dimension of the grid G is lower bounded by $2k \cdot (3k^k(3^{13^{2k^2-2k}} \cdot k^{13^{2k^2-2k}} + 1) + 3^{13^{2k^2-2k}} \cdot k^{13^{2k^2-2k}}) + 4k$. If $(G, \mathcal{R}, k, \ell)$ is a* YES*-instance, then it has a valid schedule $\mathcal{S}$ in which each route makes at most $\rho(k) = 3^{k^3+1} \cdot (3k^k(3^{13^{2k^2-2k}} \cdot k^{13^{2k^2-2k}} + 1) + 3^{13^{2k^2-2k}} \cdot k^{13^{2k^2-2k}})$ turns.*

Proof. Suppose that $(G, \mathcal{R}, k, \ell)$ is a YES-instance of CMP-M. We proceed by contradiction. Let $\mathcal{S}$ be a minimal schedule for $(G, \mathcal{R}, k, \ell)$ and assume that S has a route W_i for R_i that makes more than $\rho(k)$ turns. By Lemma 9, there exists a $[\sigma, \gamma]$-good interval $T \subseteq [0, \ell]$ such that $\nu_T(W_i) > 3k^k(\sigma(k) + 1) + \sigma(k)$, where σ and γ are the function specified in Lemma 9. By Lemma 11, there is an equivalent route W_i' to W_i that agrees with W_i outside of T and such that $\nu_T(W_i') \leq 3k^3 < 3k^k(\sigma(k) + 1) + \sigma(k)$, which contradicts the minimality of $\mathcal{S}$. ◀

3.2 Finding Canonical Solutions

Having established the existence of canonical solutions with a bounded number of turns, we can proceed to describe the proof of the FPT result. In the proof, we identify a "combinatorial snapshot" of a solution whose size is upper-bounded by a function of the parameter k. We then branch over all possible combinatorial snapshots and, for each such snapshot, we reduce the problem of determining whether there exists a corresponding solution to an instance of Integer Linear Programming in which the number of variables is upper-bounded by a function of the parameter, which can be solved in FPT-time by existing algorithms [14, 16, 19].

In particular, the aforementioned combinatorial snapshot will be a tuple (G_{snap}, $\mathcal{R}_{snap}$, W_{snap}, ι) where G_{snap} is a bounded-size subgrid, $\mathcal{R}_{snap}$ is a tuple of k pairs of starting and ending vertices in G_{snap}, W_{snap} specifies a set of routes connecting the individual starting and ending vertices, and ι contains information about the order in which vertices are visited by the routes in W_{snap}. For each snapshot, we construct an ILP instance with variables that capture (1) the amount of "expansion" necessary to go from the snapshot to the full input grid, and (2) the amount of waiting a robot performs at certain "critical" junctions in the route. Constraints are then used to ensure that each robot arrives in time, that the routes correspond to the information in ι and do not lead to conflicts, and finally that the amount of expansion needed matches the size of the input grid.

▶ **Theorem 13.** CMP-M *is* FPT *parameterized by the number of robots.*

3.3 Minimizing the Total Traveled Length

In this subsection, we discuss how the strategy for establishing the fixed-parameter tractability of CMP-M parameterized by the number k of robots can be used for CMP-L.

The main difference between the two problems can be intuitively stated as follows: for CMP-M "time matters" but travel length could be lax, whereas for CMP-L "travel length matters" but time can be lax. The key tool we use to handle the complications arising in CMP-L when showing the existence of a canonical solution is a result that exhibits a schedule for any instance of CMP-L whose travel length is within a quadratic additive factor in k from any length-optimal solution. Denote by $\mathtt{dist}_{min}$ the sum of the Manhattan distances, over all the robots, between the starting point of the robot and its destination point. We have:

▶ **Theorem 14.** *Let $\mathcal{I} = (G, \mathcal{R}, k, \lambda)$ be a* YES*-instance of* CMP-L*. There is a schedule $\mathcal{S}$ for $\mathcal{I}$ satisfying that the total travel length of $\mathcal{S}$ is at most $\mathtt{dist}_{min} + c(k)$, where $c(k) = \mathcal{O}(k^2)$ is a computable function, and in which the number of turns made by each robot is $\mathcal{O}(k)$.*

The above theorem is then exploited for showing that if a robot makes a large number of turns, then we can find a time interval and a large rectangle of the grid such that, during that time interval, all the robots that are present in that rectangle behave "nicely". We formalize these notions in the following definitions:

Let $M = [u_i, \ldots, u_j]$ be a monotone sequence of turns made by a robot $R \in \mathcal{R}$ during some time interval. The *rectangle* of M, denoted $\texttt{rectangle}(M)$, is the rectangle with diagonally-opposite vertices u_i and u_j. We refer to Figure 5 for illustration.

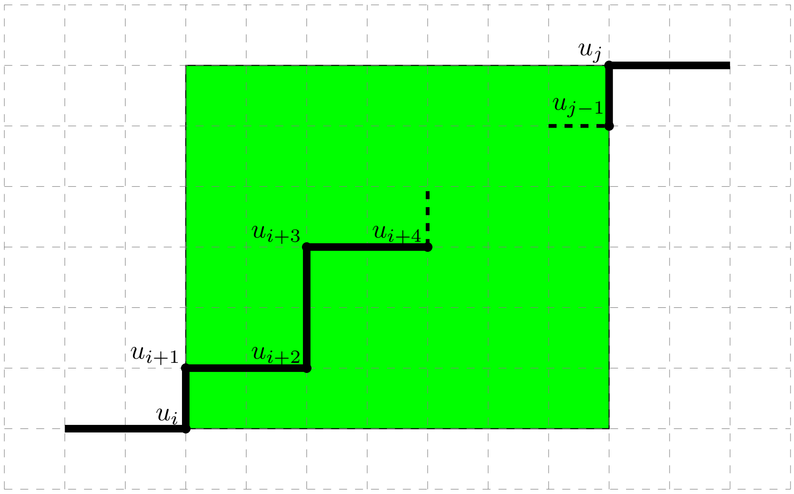


Figure 5 $\texttt{rectangle}(M)$ (the green-shaded area) for a monotone sequence $M = [u_1, \ldots, u_j]$.

▶ **Definition 15.** Let W be a subroute of a robot $R \in \mathcal{R}$ during some time interval T such that the sequence M of turns in W is monotone. Let $\sigma(k)$ be a function to be specified later. We say that $\texttt{rectangle}(M)$ is *good* w.r.t. $\sigma(k)$ and a time subinterval $T' \subseteq T$ if: (i) the set of robots present in $\texttt{rectangle}(M)$ is the same during each time step of T'; (ii) each robot R_i present in $\texttt{rectangle}(M)$ during T' satisfies $\texttt{slack}_{T'}(R_i) \geq \sigma(k)$; (iii) each robot R_i present in $\texttt{rectangle}(M)$ during T' is traveling in the same direction as (the directions of the turns in) M; and (iv) each robot R_i present in $\texttt{rectangle}(M)$ during T' satisfies $\nu_{T'}(W_i) \geq \sigma(k)$.

Next, we show that if a robot makes a large number of turns, then a good rectangle exists:

▶ **Lemma 16.** *Let $\mathcal{I} = (G, \mathcal{R}, k, \lambda)$ be a* YES*-instance of* CMP-L*, let $\mathcal{S}$ be a valid schedule for $\mathcal{I}$, and assume that $\lambda < \texttt{dist}_{min} + c(k)$, where $c(k) = \mathcal{O}(k^2)$ is the computable function in Theorem 14. Let $\sigma(k) = 4k^2$ and $\tau(k) = 3k^k(\sigma(k) + 1) + \sigma(k)$. Let R be a robot such that the walk W of R during the time interval T spanning $\mathcal{S}$ satisfies $\nu(W) = \Omega(\tau(k)^{2k+1})$. Then there exists a subwalk W' for R and a time interval $T' \subseteq T$ such that the sequence of turns M' in W' corresponding to T' is monotone and $\texttt{rectangle}(M')$ is good w.r.t. $\sigma(k)$ and T'.*

Using Theorem 14 and Lemma 16, we can prove that, given a good rectangle, we can reroute the robots that are present in that rectangle during a certain time interval so as to reduce the number of turns they make, which leads to the existence of a canonical solution:

▶ **Theorem 17.** *If $\mathcal{I} = (G, \mathcal{R}, k, \lambda)$ is a* YES*-instance of* CMP-L*, then $\mathcal{I}$ has a valid schedule $\mathcal{S}$ in which each route makes at most $\mathcal{O}(\tau(k)^{2k+1})$ turns, where $\tau(k) = 3k^k(\sigma(k) + 1) + \sigma(k)$, and $\sigma(k) = 4k^2$.*

At this point, we can turn to the second step of our approach, notably checking whether an instance of CMP-L admits a solution in which the number of turns is upper-bounded by a function of the parameter. Luckily, here the proof of Theorem 13 can be reused almost as-is, with only a single change in the ILP encoding at the end.

▶ **Theorem 18.** CMP-L *is* FPT *parameterized by the number of robots.*

4 CMP Parameterized by the Objective Target

Having resolved the parameterization by the number k of robots, we now turn our attention to the second fundamental measure in CMP problems, notably the objective target. Unlike the case where we parameterize by the number k of robots, here the complexity of the problem strongly depends on the considered variant. We begin by establishing the fixed-parameter tractability of CMP-L parameterized by λ via an exhaustive branching algorithm. The rest of this section then deals with the significantly more complicated task of establishing the intractability of CMP-L parameterized by ℓ.

▶ **Theorem 19.** CMP-L *is* FPT *parameterized by the objective target* λ*.*

4.1 Intractability of CMP-M with Small Makespans

The aim of this subsection is to establish that CMP-M is NP-hard even when the makespan ℓ is upper bounded by a constant. Before we proceed to show this NP-hardness result for CMP-M, we will establish the NP-hardness of d-Bounded Length Vertex Disjoint Paths on grids, as well as its edge variant d-Bounded Length Edge Disjoint Paths, which can be seen as a stepping stone for the para-NP-hardness proof for CMP-M. In fact, the NP-hardness result for these two classical disjoint paths problems on grids with constant path lengths is significant in its own right, as discussed earlier in the paper.

All our reductions start from 4-Bounded Planar 3-SAT, a problem which is known to be NP-complete [18, 22]. The *incidence graph* of a CNF formula is the graph whose vertices are the variables and clauses of the formula, and in which two vertices are adjacent if and only if one is a variable, the other is a clause, and the variable-vertex occurs either as a positive or a negative literal in the clause-vertex. In 4-Bounded Planar 3-SAT, we are asked to evaluate a CNF formula whose incidence graph is planar and in which each clause contains exactly 3 distinct literals and each variable occurs in at most 4 clauses. On the other hand, in the aforementioned d-Bounded Length Vertex (resp. Edge) Disjoint Paths problems, we are given a graph with a set of vertex-pairs (called *requests*), and are asked to determine if there is a set of vertex (resp. edge) disjoint paths containing an s-t path of length at most $d \in \mathbb{N}$ for every $(s, t) \in R$.

For all three reductions, consider an instance φ of 4-Bounded Planar 3-SAT and let G_φ be its incidence graph. We start with an orthogonal drawing Ω of G_φ in a polynomial-size grid. Our first goal is to show how to encode the satisfiability of φ as an instance of d-Bounded Length Vertex Disjoint Paths on grids; the reduction for d-Bounded Length Edge Disjoint Paths is almost the same, and both can be seen as a stepping stone towards CMP-M. We encode variable assignment and clause satisfaction using bounded-length path requests that conform to the drawing Ω. To model a variable-assignment, we create a variable gadget with a single request between two vertices, s and t, on this gadget such that this request can be fulfilled by selecting one of the two s-t paths in this gadget, each of length 27. Selecting one of the two paths corresponds to assigning the variable a truth value; an illustration is provided in Figure 6. We model clause-satisfaction by creating, for each clause, a clause-gadget, where a clause-gadget for a clause C contains two vertices, s_C and t_C, with a request between them that can be fulfilled in one of three ways, each corresponding to choosing a length-27 path between s_C and t_C in the gadget (see Figure 7).

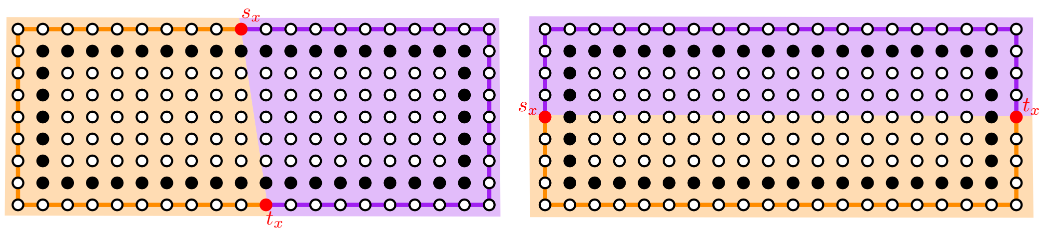


Figure 6 Variable Gadget examples. In both cases, there is a request (s_x, t_x). Each of the full black circles is a request (v, v) forcing only two different paths of length at most 27 between s_x and t_x. Examples of a left-right variable gadget (left) and a top-bottom variable gadget (right).

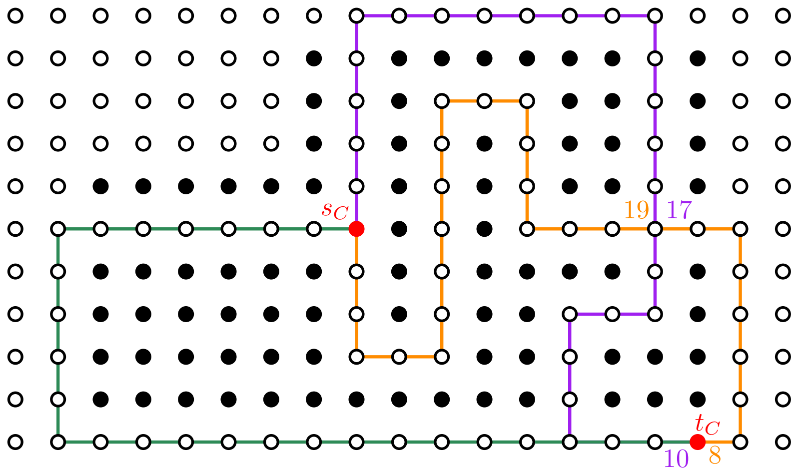


Figure 7 Clause gadget example. There is a request (s_C, t_C). Each of the full black circles is a request (v, v). There are three possible ways to leave s_C. Choosing to go left forces us to take the green path of length 27. The orange path going down reaches the intersection point with the purple path (going up) after 19 steps on the orange path, but only 17 on the purple. Hence, the purple can choose between going down and taking 10 steps to reach t_C, or going right and taking 8 more steps, but the orange is forced to go right and reach t_C in 8 steps from the intersection point.

To implement the above idea, we needed to overcome several issues. First, the position of a variable in the embedding could be very far from the position of the clauses that it is incident to, hence prohibiting us from using bounded-length requests to encode the variable-clause incidences. Second, due to planarity constraints, embedding the three paths corresponding to a clause-gadget such that each intersects a different variable gadget, is only possible if two of the clause-paths intersect, which could create shortcuts (i.e., paths that do not intersect the variable gadgets). Third, requests may use grid paths that are not part of the embedding.

To handle the first issue, instead of using a single variable-gadget per variable, we use a "cycle" of copies of variable gadgets such that a variable assignment in any gadget of this cycle forces the same variable assignment in all copies, thus ensuring assignment consistency. The clause gadget for C is placed around the position of the vertex corresponding to C in Ω, whereas the cycle corresponding to a variable x is placed around the edges of Ω joining the position of x in Ω to that of C; see Figure 8. To fit all the variable cycles around a clause gadget in the embedding, we use a connection gadget, which is a path of copies of variable gadgets propagating the same variable assignment as in the corresponding variable cycle.

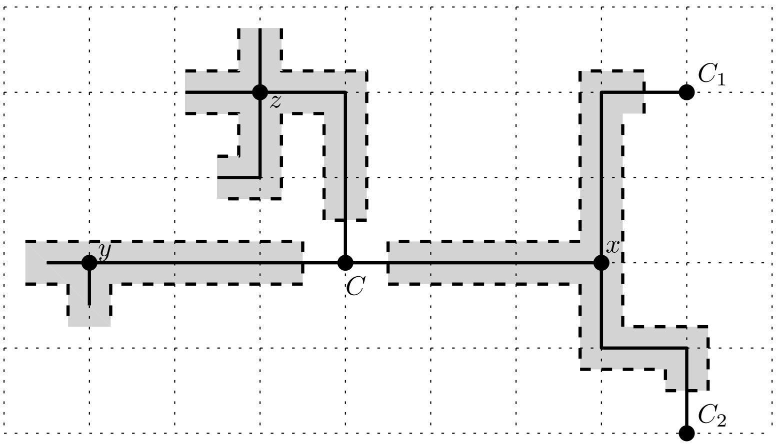


Figure 8 Part of an orthogonal drawing of G_φ. Clause C contains variables x, y, z. The variable x is also in clauses C_1 and C_2. The dashed lines represent the variable cycles.

To model clause-satisfaction for a clause C, each of the three s_C-t_C paths in the clause gadget of C overlaps with a copy of a variable gadget corresponding to one of the variables whose literal occurs in C. If an assignment to variable x whose literal occurs in C does not satisfy C, then the path corresponding to this assignment in the copies of the variable gadgets for x intersects the s_C-t_C path corresponding to x in the clause-gadget of C, thus prohibiting the simultaneous choice of these clause-path and variable path.

To handle the second issue, we prevent any shortcuts from being taken by making each created shortcut longer than the prescribed upper bound on the path length (i.e., 27).

Finally, to handle the third issue, when dealing with vertex disjoint paths we can artificially place an obstacle on a vertex v in the grid to "block" that vertex (i.e., to prevent it from being used by any path other than (v, v)) by adding the request (v, v), thus forcing the set of possible paths between s and t for every request (s, t), where $s \neq t$, to be chosen from the paths prescribed by the encoding of the instance of 4-Bounded Planar 3-SAT. A slight extension of this idea also works for edge disjoint paths. This allows us to establish:

▶ **Theorem 20.** *d-Bounded Length Vertex Disjoint Paths and d-Bounded Length Edge Disjoint Paths are* NP*-hard even when restricted to instances where $d = 27$ and G is a grid-graph.*

These high-level ideas are then used to obtain the targeted NP-hardness proof of CMP-M for a fixed makespan of 26, by having an (s, t) path request correspond to routing some robot from its starting gridpoint s to its destination gridpoint t. However, the way we force robots to follow the prescribed paths here is completely different and presents the main difficulty when going from d-Bounded Length Vertex Disjoint Paths on grids to CMP-M; in particular, it is no longer possible to block certain points on the grid by creating "dummy requests". To ensure that the prescribed paths are followed, we block certain regions of the embedding by adding a large number of auxiliary non-stationary robots, and coordinating their motion so that they block the desired regions while still allowing the original robots to follow the set of paths prescribed by the encoding; this task turns out to be highly technical.

We introduce a set of new gadgets whose role is to force the main robots in the reduction to follow the paths prescribed by the embedding. Those gadgets are dynamic, as opposed to the "static blocker gridpoints" used in the d-Bounded Length Vertex Disjoint Paths on grids reduction. The two main new gadgets employed are a gadget simulating a "stream" of robots and a gadget simulating an "arrow" of robots.

The stream gadget consists of a relatively large number of robots, all moving along the same line, such that each needs to move precisely the makespan many steps in the same direction, and hence cannot afford to waste a single time step. The robots in the stream gadget will be used to either push the main robots in a certain direction, or to prevent them from taking shorter paths than the prescribed ones. See Figure 9 for an illustration. In the figure, the main red robot is pushed right by the green stream and forced to move right along the same horizontal line by the two blue streams sandwiching it.

Figure 10 shows an example of an arrow gadget. In this gadget, there is an orange robot whose destination is 26 steps somewhere down and to the left. The gadget is again a "stream" of robots that force the orange robot to select one of the two directions towards its destination in the first step, and then to stick to this selection for a number of steps that depends on the size of the arrow. For example, in Figure 10, there is a "right arrow" of green robots that all want to go 26 steps right. Since the orange robot has a slack of 0, the right arrow forces it to either take the first 5 steps all to the left, or the first 7 steps all down.

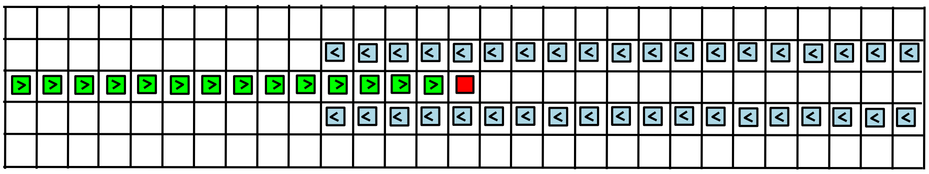


Figure 9 Example of streams.

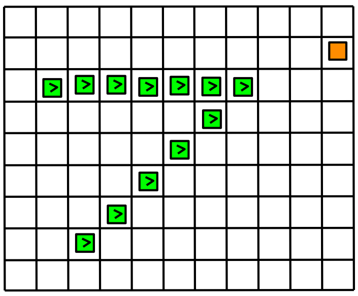


Figure 10 An example of an arrow.

Other gadgets are needed to ensure that robots in the stream and arrow gadgets do not collide with anything. Using such enforcement gadgets, we can simulate the gadgets constructed in the reduction for d-Bounded Length Vertex Disjoint Paths on grids, thus encoding the instance of 4-Bounded Planar 3-SAT as an instance of CMP-M.

▶ **Theorem 21.** CMP-M *is* NP*-hard even when restricted to instances where* $\ell = 26$.

5 Conclusion

In this work, we settled the parameterized complexity of both CMP-M and CMP-L with respect to their two most fundamental parameters: the number of robots, and the objective target. Along the way, we established the NP-hardness of the classical Vertex Disjoint Paths and the Edge Disjoint Paths problem with constant path-lengths on grids, strengthening the existing lower bounds for these problems as well. Our results reveal structural insights into the properties of optimal solutions that may also prove useful in contexts that lie outside of this work. We conclude by stating two open questions that arise from our work.

1. What is the parameterized complexity of other variants of CMP, such as the ones where the objective is to minimize the maximum length traveled or the total arrival time?
2. Can the fixed-parameter tractability of CMP-M or CMP-L parameterized by the number k of robots be lifted to grids with obstacles/holes, or more generally to planar graphs? It is worth noting that neither the structural results developed in this paper, nor other known techniques [23], seem to be applicable to these more general settings.

References

1 Computational Geometry Week: The Third Computational Geometry Challenge. `https://cse.buffalo.edu/socg21/challenge.html`.
2 Jacopo Banfi, Nicola Basilico, and Francesco Amigoni. Intractability of time-optimal multirobot path planning on 2D grid graphs with holes. *IEEE Robotics and Automation Letters*, 2(4):1941–1947, 2017.
3 Bahareh Banyassady, Mark de Berg, Karl Bringmann, Kevin Buchin, Henning Fernau, Dan Halperin, Irina Kostitsyna, Yoshio Okamoto, and Stijn Slot. Unlabeled multi-robot motion planning with tighter separation bounds. In *SoCG*, volume 224 of *LIPIcs*, pages 12:1–12:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022.
4 Eli Boyarski, Ariel Felner, Roni Stern, Guni Sharon, David Tolpin, Oded Betzalel, and Solomon Eyal Shimony. ICBS: Improved conflict-based search algorithm for multi-agent pathfinding. In *IJCAI*, pages 740–746, 2015.
5 Julia Chuzhoy and David H. K. Kim. On Approximating Node-Disjoint Paths in Grids. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2015)*, volume 40 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 187–211, 2015.
6 Julia Chuzhoy, David Hong Kyun Kim, and Rachit Nimavat. Almost polynomial hardness of node-disjoint paths in grids. *Theory of Computing*, 17(6):1–57, 2021.
7 Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015.
8 Erik D. Demaine, Sándor P. Fekete, Phillip Keldenich, Henk Meijer, and Christian Scheffer. Coordinated motion planning: Reconfiguring a swarm of labeled robots with bounded stretch. *SIAM Journal on Computing*, 48(6):1727–1762, 2019.
9 Erik D. Demaine and Mikhail Rudoy. A simple proof that the $(n^2 - 1)$-puzzle is hard. *Theoretical Computer Science*, 732:80–84, 2018.
10 Reinhard Diestel. *Graph Theory, 4th Edition*, volume 173 of *Graduate texts in mathematics*. Springer, 2012.
11 Rodney G. Downey and Michael R. Fellows. *Fundamentals of Parameterized Complexity*. Texts in Computer Science. Springer, 2013.
12 Adrian Dumitrescu. Motion planning and reconfiguration for systems of multiple objects. In Sascha Kolski, editor, *Mobile Robots*, chapter 24. IntechOpen, Rijeka, 2007.
13 András Frank. Disjoint paths in a rectilinear grid. *Combinatorica*, 2:361–371, 1982.
14 András Frank and Éva Tardos. An application of simultaneous diophantine approximation in combinatorial optimization. *Combinatorica*, 7(1):49–65, 1987.
15 Tzvika Geft and Dan Halperin. Refined hardness of distance-optimal multi-agent path finding. In *AAMAS*, pages 481–488, 2022.
16 Jr. H. W. Lenstra. Integer programming with a fixed number of variables. *Mathematics of Operations Research*, 8(4):538–548, 1983.

17 Peter E. Hart, Nils J. Nilsson, and Bertram Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968.

18 Md. Manzurul Hasan, Debajyoti Mondal, and Md. Saidur Rahman. Positive planar satisfiability problems under 3-connectivity constraints. *Theoretical Computer Science*, 917:81–93, 2022. doi:10.1016/j.tcs.2022.03.013.

19 Ravi Kannan. Minkowski's convex body theorem and integer programming. *Mathematics of Operations Research*, 12(3):415–440, 1987. doi:10.1287/moor.12.3.415.

20 D. Kornhauser, G. Miller, and P. Spirakis. Coordinating pebble motion on graphs, the diameter of permutation groups, and applications. In *FOCS*, pages 241–250, 1984.

21 Mark Kramer. The complexity of wire-routing and finding minimum area layouts for arbitrary VLSI circuits. *Advances in Computing Research*, 2:129–146, 1984.

22 Jan Kratochvíl. A special planar satisfiability problem and a consequence of its NP-completeness. *Discrete Applied Mathematics*, 52(3):233–252, 1994. doi:10.1016/0166-218X(94)90143-0.

23 Daniel Lokshtanov, Pranabendu Misra, Michal Pilipczuk, Saket Saurabh, and Meirav Zehavi. An exponential time parameterized algorithm for planar disjoint paths. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, pages 1307–1316. ACM, 2020.

24 Dániel Marx. Eulerian disjoint paths problem in grid graphs is NP-complete. *Discrete Applied Mathematics*, 143(1):336–341, 2004.

25 Geetha Ramanathan and V.S. Alagar. Algorithmic motion planning in robotics: Coordinated motion of several disks amidst polygonal obstacles. In *ICRA*, volume 2, pages 514–522, 1985.

26 Daniel Ratner and Manfred Warmuth. The (n^2-1)-puzzle and related relocation problems. *Journal of Symbolic Computation*, 10(2):111–137, 1990.

27 Jacob T. Schwartz and Micha Sharir. On the piano movers' problem: III. coordinating the motion of several independent bodies: The special case of circular bodies moving amidst polygonal barriers. *The International Journal of Robotics Research*, 2:46–75, 1983.

28 Guni Sharon, Roni Stern, Ariel Felner, and Nathan R. Sturtevant. Conflict-based search for optimal multi-agent pathfinding. *Artificial Intelligence*, 219:40–66, 2015.

29 Irving Solis, James Motes, Read Sandström, and Nancy M. Amato. Representation-optimal multi-robot motion planning using conflict-based search. *IEEE Robotics Autom. Lett.*, 6(3):4608–4615, 2021.

30 Hitoshi Suzuki, Akira Ishiguro, and Takao Nishizeki. Edge-disjoint paths in a grid bounded by two nested rectangles. *Discrete Applied Mathematics*, 27(1):157–178, 1990.

31 Glenn Wagner and Howie Choset. Subdimensional expansion for multirobot path planning. *Artificial Intelligence*, 219:1–24, 2015.

32 Jingjin Yu. Constant factor time optimal multi-robot routing on high-dimensional grids. In Hadas Kress-Gazit, Siddhartha S. Srinivasa, Tom Howard, and Nikolay Atanasov, editors, *Robotics: Science and Systems XIV, Carnegie Mellon University, Pittsburgh, Pennsylvania, USA, June 26-30, 2018*, 2018.

33 Jingjin Yu and Steven M. LaValle. Structure and intractability of optimal multi-robot path planning on graphs. In Marie desJardins and Michael L. Littman, editors, *AAAI*. AAAI Press, 2013.

34 Jingjin Yu and Steven M. LaValle. Optimal multirobot path planning on graphs: Complete algorithms and effective heuristics. *IEEE Transactions on Robotics*, 32(5):1163–1177, 2016.