A Minor-Testing Approach for Coordinated Motion Planning with Sliding Robots

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

M. S. Ramanujan
Department of Computer Science, University of Warwick, UK

Abstract

We study a variant of the Coordinated Motion Planning problem on undirected graphs, referred to herein as the COORDINATED SLIDING-MOTION PLANNING (CSMP) problem. In this variant, we are given an undirected graph G, k robots $R_1, \ldots, R_k$ positioned on distinct vertices of G, $p \leq k$ distinct destination vertices for robots $R_1, \ldots, R_p$, and $\ell \in \mathbb{N}$. The problem is to decide if there is a serial schedule of at most ℓ moves (i.e., of makespan ℓ) such that at the end of the schedule each robot with a destination reaches it, where a robot's move is a free path (unoccupied by any robots) from its current position to an unoccupied vertex. The problem is known to be NP-hard even on full grids. It has been studied in several contexts, including coin movement and reconfiguration problems, with respect to feasibility, complexity, and approximation. Geometric variants of the problem, in which congruent geometric-shape robots (e.g., unit disk/squares) slide or translate in the Euclidean plane, have also been studied extensively.

We investigate the parameterized complexity of CSMP with respect to two parameters: the number k of robots and the makespan ℓ. As our first result, we present a fixed-parameter algorithm for CSMP parameterized by k. For our second result, we present a fixed-parameter algorithm parameterized by ℓ for the special case of CSMP in which only a single robot has a destination and the graph is planar. A crucial new ingredient for both of our results is that the solution admits a succinct representation as a small labeled topological minor of the input graph.

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

Keywords and phrases coordinated motion planning on graphs, parameterized complexity, topological minor testing, planar graphs

Digital Object Identifier 10.4230/LIPIcs.SoCG.2025.44

Related Version *Full Version*: `https://arxiv.org/abs/2502.21175`

Funding *Robert Ganian*: Projects No. 10.55776/Y1329 and 10.55776/COE12 of the Austrian Science Fund (FWF), Project No. 10.47379/ICT22029 of the Vienna Science Foundation (WWTF).
M. S. Ramanujan: Supported by Engineering and Physical Sciences Research Council (EPSRC) grant EP/V044621/1.

1 Introduction

In the coordinated motion planning problem, we are given an integer $\ell \in \mathbb{N}$, an undirected graph G, k robots $R_1, \ldots, R_k$ positioned on distinct starting vertices of G and $p \leq k$ distinct destination vertices for robots $R_1, \ldots, R_p$. In each step, one robot (or a subset of robots in other variants of the problem in which robots may move in parallel) may move from its current vertex to an adjacent unoccupied vertex. The goal is to decide if there is a schedule (i.e., a sequence of moves) of length at most ℓ, at the end of which each robot $R_1, \dots, R_p$ has reached its destination vertex without colliding with other robots along the way. The problem and its variants (including the common variant where $p = k$) have been extensively studied, with respect to various motion types and graph classes, due to their ubiquitous applications in artificial intelligence (planning), robotics, computational geometry and more generally theoretical computer science [3, 6, 10, 11, 16, 17, 23, 24, 28, 31–38]. Moreover, it generalizes several well-known puzzles, including the famous $(n^2 - 1)$-puzzle and the Rush-Hour puzzle. Most of these variants are NP-hard, including the aforementioned $(n^2 - 1)$-puzzle [12, 29]. Due to its numerous applications, the coordinated motion planning problem on (full rectangular) grids was posed as the SoCG 2021 Challenge [18].

We consider a variant of the coordinated motion planning problem on undirected graphs in which robots move serially one at a time, and a robot's move consists of a move along a free/unobstructed path (i.e., a path in which no other robot is located at the time of the move) as opposed to a single edge; we refer to such a move as a *sliding* move, analogous to the type of motion in geometric variants of the coordinated motion planning problem, where robots are geometric shapes (e.g., disks or rectangles) that move by translating/sliding in the plane. These geometric variants have been extensively studied since the 1980s [24, 28, 31–33]. The goal of this variant is to decide whether there is a schedule of makespan at most ℓ, where here the makespan of the schedule is simply the number of moves in it. We denote our general problem of interest as CSMP. We will also consider the special case of CSMP where G is planar and $p = 1$ – that is, only a designated robot has a destination and the remaining robots merely act as "blockers". We refer to this special case as PLANAR-CSMP-1.

Contributions and Techniques. We present a fixed-parameter algorithm for CSMP parameterized by the number k of robots, and a fixed-parameter algorithm for PLANAR-CSMP-1 parameterized by the makespan ℓ. A crucial common ingredient for both results is showing that the solution to a problem instance admits a succinct representation, whose size is a function of the parameter, as a topological minor of the input graph. To the best of our knowledge, this is a novel approach in the context of coordinated motion planning problems.

CSMP. Showing that the solution admits a succinct representation entails first bounding the makespan in an optimal solution by a function of the number k of robots. Towards this end, we preprocess the graph by shortening (to $\mathcal{O}(k)$) sufficiently long paths comprising degree-2 vertices and focus on two cases. In one case, where the graph has bounded[1] treedepth, we show that the instance can be reduced iteratively without altering the solution. This reduction process ultimately results in an instance of bounded size, implying the existence of a solution with a bounded makespan.

In the other case, we use ideas from [10] to define a notion of *havens*, which are subgraphs of bounded size with the following property: if a robot has to pass through a haven, then with a bounded number of additional steps, we can enable it to pass through the haven regardless of the other robots in the haven (i.e., without collision), effectively "untangling" all the robots' traversals within the haven. Building on this, we construct a schedule in which each robot interacts with only a bounded number of havens, ultimately leading to a bounded-makespan schedule.

[1] In the rest of this description, we simply say *bounded* to refer to values bounded by a function of k.

After bounding the makespan in an optimal solution, we show that it essentially describes a graph H of bounded size as a topological minor in the input graph. Here, we have to work with *rooted* topological minors since all robots have specified starting points and some have specified destinations. For us, the roots are just the starting and ending vertices, as we have boundedly-many roots. Towards our goal, we partition the vertices of the graph induced by the edges participating in an optimal solution into two sets: important and unimportant vertices. We show that the number of important vertices is bounded, and that the unimportant vertices form paths whose internal vertices have degree 2. This structure enables us to show that the graph induced by the edges in the solution (with terminals as roots) is a realization of some rooted graph with a bounded number of vertices as a topological minor in the input graph. We also prove the converse, namely that from *any* realization of H as a topological minor in G, one can obtain an optimal solution for the given instance. Towards proving this, we show that if a robot follows a path that is devoid of important vertices, then it slides on that path without stopping. Finally, we leverage known results on topological minor containment to verify the existence of the correct rooted graph.

Planar-CSMP-1. It is easy to see here that the solution admits a succinct representation since the parameter itself is the makespan. The difficulty is that the number of robots in the instance can be very large compared to the parameter ℓ, and we cannot find a realization of the succinct solution which avoids the "blocker" robots that do not contribute to the realization. To cope with this hurdle, we exploit the planarity of the graph to reduce its diameter, thus reducing the instance to an instance on a graph with bounded treewidth. The fixed-parameter tractability of the problem will then follow from Courcelle's Theorem [2, 8].

The heart of our reduction method is finding an "irrelevant edge" in the graph that can be safely contracted to produce an equivalent instance of the problem. While the method of finding an irrelevant edge and contracting it has been employed in several results in the literature to exclude large grids, thus reducing the treewidth of the graph, in our setting the grid approach seems unworkable. Instead, we show that a sufficiently-large component C of free vertices (i.e., vertices with no blockers on them) must contain an irrelevant edge. We reduce all the cases to one where C is formed by a skeleton consisting of a "long" path of degree-2 vertices (the degree is taken in C), plus a "small" number of additional vertices. We then show that if the instance is a YES-instance of the problem, then it admits a solution that interacts nicely with this skeleton, in the sense that we can identify a subgraph of the representation of the solution, i.e., a topological minor that is essentially a roadmap of the solution, separated from the rest of the solution by a small cut. We can enumerate each possible representation of the part of the solution that interacts with the skeleton, and for each possible representation, test whether it exists near the skeleton; if it does, we mark the vertices in the graph that realize this representation. Since the skeleton is long, the total number of marked vertices on the skeleton is small, and an edge with both endpoints unmarked – and hence can be contracted – must exist.

Related Work. The CSMP problem, with $p = k$, has been studied in [7] on several graph classes, and with respect to various settings, based on whether or not the destinations of the robots are distinguished (referred to as labeled or unlabeled). The graph classes that were considered include trees, grids, planar graphs, and general graphs, and it was shown that the problem is NP-hard even for (full rectangular) grids. Upper and lower bounds on the makespan of a solution, as well as computational complexity and approximation results, were derived for the various settings considered in [7].

A related problem to CSMP is that of moving unit disks (or objects [16]) in the plane, which has been studied in the context of reconfiguring/moving coins in the plane [1,4], and was shown to be NP-hard [1]. We also mention the related work on coordinated pebble motion on graphs (and permutation groups), which was studied in [25] motivated by the work on the (n^2-1)-puzzle. The variant of coordinated motion planning in which only one designated robot is required to reach its destination, that is $p = 1$, and each move is along a single edge, was studied as Graph Motion Planning with 1 Robot in [27].

Despite the tremendous amount of work on coordinated motion planning problems, not much work has been done from the parameterized complexity perspective. The work in [10] studied the parameterized complexity of the non-sliding version of CSMP, that is, where each motion step is along a single edge, parameterized by the makespan in the serial setting. Another recent work [17] studied the parameterized complexity of the classical coordinated motion planning problem on grids, presenting parameterized algorithms for various objective measures (makespan, travel length). The work in [20] established the W[1]-hardness of the classical coordinated motion planning problem parameterized by the makespan in the parallel motion setting, and also showed the NP-hardness of this problem on trees, among other results. Finally, the parameterized complexity of a geometric variant of the PSPACE-complete Rush-Hour problem, which itself was shown to be PSPACE-complete [21], was studied [19]; in particular, that problem was shown to be FPT when parameterized by either the number of robots (i.e., cars) or the number of moves.

2 Terminology and Problem Definition

We assume basic familiarity with parameterized complexity theory, including *fixed-parameter tractability*, *parameterized reductions*, and the class W[1] [9,15]. We also assume knowledge of standard graph-theoretic notions such as planarity, graph diameter and vertex degree [13]. We write $[n]$, where $n \in \mathbb{N}$, for $\{1, \ldots, n\}$.

Treewidth is a fundamental graph parameter which can be seen as a measure of how similar a graph is to a tree; trees have treewidth 1, while the complete n-vertex graph has treewidth $n-1$. A formal definition of treewidth will not be necessary to obtain our results; we will make use of the fact that every planar graph with diameter r has treewidth at most $3r+1$ [5,30] and of Courcelle's Theorem [2,8], which in particular provides a linear-time procedure to identify vertex subsets satisfying properties definable in *Monadic Second Order Logic* (MSO) on graphs of constant treewidth. We also use the notion of *treedepth* [26].

Rooted graphs. A rooted graph is a graph where some vertices are labeled – formally:

▶ **Definition 1** (Rooted graphs). *A* rooted graph *is a graph G with a set $\partial(G) \subseteq V(G)$ of distinguished vertices called* roots *and a labeling $\lambda_G : \partial(G) \to \mathbb{N}$. The set $\partial(G)$ is the* root set *of G, and the* label set *of G is $\Lambda(G) = \{\lambda_G(v) \mid v \in \partial(G)\}$.*

▶ **Definition 2** (Topological minors of rooted graphs [22]). *Let G and H be two rooted graphs with labelings λ_H and λ_G for their root sets. We say that H is a* topological minor *of G if there exist injective functions $\phi : V(H) \to V(G)$ and $\varphi : E(H) \to \mathsf{paths}(G)$ such that*

- *for every $e = \{h, h'\} \in E(H)$, the endpoints of $\varphi(e)$ are $\phi(h)$ and $\phi(h')$,*
- *for every distinct $e, e' \in E(H)$, the paths $\varphi(e)$ and $\varphi(e')$ are internally vertex-disjoint,*
- *there does not exist a vertex v in the image of ϕ and an edge $e \in E(H)$ such that v is an internal vertex on $\varphi(e)$, and*
- *for every $v \in \partial(H)$, $\lambda_H(v) = \lambda_G(\phi(v))$.*

We say that (ϕ, φ) is a realization *of H as a topological minor in G. Moreover, the subgraph of G induced by the union of the edges in the paths in $\varphi(e)$ for $e \in E(H)$ and the vertices in $\phi(h)$ for $h \in V(H)$ is called a* topological minor model *of H.*

Note that for $\partial(G) = \emptyset$ the above coincides with the usual definition of topological minors.

▶ **Proposition 3.** ([22]) *There is an algorithm that, given two rooted graphs G and H, runs in time $f(|H|) \cdot \mathcal{O}(|G|^3)$ for some computable function f and decides whether H is a topological minor of G.*

Problem Formalizations. In our problems of interest, we are given an undirected graph G and a set $\mathcal{R} = \{R_1, R_2, \ldots, R_k\}$ of k robots where $\mathcal{R}$ is partitioned into two sets $\mathcal{M}$ and $\mathcal{F}$. Each $R_i \in \mathcal{M}$, has a starting vertex s_i and a destination vertex t_i in $V(G)$ and each $R_i \in \mathcal{F}$ is associated only with a starting vertex $s_i \in V(G)$. We refer to the elements in the set $\{s_i \mid i \in [k]\} \cup \{t_i \mid R_i \in \mathcal{M}\}$ as *terminals*. We assume that all the s_i are pairwise distinct and that all the t_i are pairwise distinct. We use a discrete time frame $[0, q]$, $q \in \mathbb{N}$, to reference the sequence of moves of the robots. In each time step $x \in [0, q]$, exactly one robot moves and the rest remain stationary.

In Coordinated Sliding-Motion Planning (CSMP), we are given a tuple $(G, \mathcal{R} = (\mathcal{M}, \mathcal{F}), k, \ell)$, where $G, \mathcal{R}$ and k are as described in the last paragraph and $\ell \in \mathbb{N}$. The goal is to decide whether there is a sequence of $q \leq \ell$ moves such that robots move serially one at a time, where a robot's move consists of a move along a free/unobstructed path (i.e., a path in which no other robot is located at the time of the move), and such that each $R_i \in \mathcal{R}$ starts in s_i at time step 0 and each $R_i \in \mathcal{M}$ is positioned on t_i at the end of time step q. The second problem that we consider is a restriction of CSMP to the case where G is planar and $\mathcal{M}$ contains exactly one robot, referred to as the *main* robot. We refer to this restriction as Planar-CSMP-1, and for simplicity represent its instances as tuples of the form $(G, s, t, \ell, \mathcal{R})$, where s and t are the starting and destination vertices of the main robot, respectively, and $\mathcal{R}$ is the set of all robots.

We next introduce some notation that will be used in our technical sections. A *route* for robot R_i is a tuple $W_i = (u^i_0, P^i_1, u^i_1, \ldots, P^i_q, u^i_q)$ where each u^i_j is a vertex in G and each P^i_j is a simple path in G, such that (i) $u^i_0 = s_i$ and $u^i_q = t_i$ if $R_i \in \mathcal{M}$ and (ii) $\forall j \in [q]$, P^i_j is a u^i_{j-1}-u^i_j path in G (if $u^i_{j-1} = u^i_j$, then P^i_j is the singleton path comprising the same vertex).

Put simply, each W_i corresponds to a "walk" in G. If vertices of W_i at two consecutive time steps $j-1$ and j are identical, then j is a *waiting time step* for the robot R_i. Moreover, each R_i begins at its starting vertex at time step 0, and if $R_i \in \mathcal{M}$ then R_i is at its destination vertex at time step q. Though we work with undirected graphs and the paths described above are undirected paths, each path P^i_j has a natural *starting vertex* and *ending vertex* designated by the time step j.

▶ **Definition 4.** Two routes $W_i = (u^i_0, P^i_1, \ldots, P^i_q, u^i_q)$ and $W_j = (u^j_0, P^j_1, \ldots, P^j_q, u^j_q)$, where $i \neq j \in [k]$, are *non-conflicting* if $\forall r \in \{0, \ldots, q\}$, the path P^i_r is disjoint from u^j_{r-1} and the path P^j_r is disjoint from u^i_{r-1}. Otherwise, we say that W_i and W_j *conflict*.

In particular, in the above definition, for every $i \neq j \in [k]$ and $\forall r \in \{0, \ldots, q\}$, $u^i_r \neq u^j_r$.

A *schedule* S for $\mathcal{R}$ is a set of pairwise non-conflicting routes $W_i, i \in [k]$, during a time interval $[0, q]$, where exactly one robot moves in each time step. The *number of moves* in a route $W_i = (u^i_0, P^i_1, \ldots, P^i_q, u^i_q)$ or the number of moves of its associated robot is the number of time steps j such that $u^i_j \neq u^i_{j+1}$. The *total number of moves* (or *makespan*) in a schedule is the sum of the number of moves in its routes. A schedule of minimum total moves is called an *optimal schedule*.

A vertex v is *occupied* at time step i if some robot is located at v at time step i; otherwise, v is said to be *free* (or *unoccupied*). If the time step is not specified, it is assumed to be time step 0 (i.e., in the initial state). At each time step, a robot may either move to a different vertex, or stay at its current vertex. Finally, we always assume that the input graph is connected since otherwise, we can work on individual connected components.

Remarks on Feasibility and Constructiveness. When only considering feasibility, CSMP is equivalent to the version where each move is along a single edge instead of an unobstructed path. Hence, feasibility testing is linear-time solvable [10, 39] and we can work under the assumption that every instance of CSMP is feasible (otherwise, we can reject the instance).

3 FPT Algorithm for CSMP Parameterized by the Number of Robots

In this section, we show that CSMP is FPT parameterized by the number k of robots in the input. Note that we do not assume any restrictions on the input graph.

Our strategy has three high-level steps: (i) We first show that if a solution exists, then there is an optimal schedule whose makespan is bounded[2]. (ii) We then show that such a solution is essentially a realization of a rooted graph H of bounded size as a topological minor in a rooted graph G' obtained by assigning unique labels to the terminals in G. We also prove the converse, that is, from any realization of H as a topological minor in G', one can obtain an optimal solution for the given instance. (iii) Finally, we leverage known results on topological minor containment to verify the existence of the correct rooted graph. Let us now give more detail on Steps (i) and (ii).

Step (i): To prove the existence of a solution with boundedly many moves, we preprocess the graph by applying a reduction rule that shortens degree-2 paths to a bounded length. This simplification allows us to focus on two cases for the graph (assuming it is connected):

Case (a): Bounded Treedepth. If the graph has bounded treedepth, we show that instances with bounded treedepth and a bounded number of robots can be reduced iteratively without altering the solution. This reduction process results in an instance of bounded size, implying the existence of a solution with a bounded makespan.

Case (b): Long Paths. Suppose every vertex is the endpoint of a sufficiently long path. We use ideas from [10] to define a suitable notion of *havens* for our setting; in particular, havens are subgraphs of bounded size that can efficiently handle collisions. Here, by the "efficient handling" of collisions, we mean that given any pair of "configurations" of robots in a haven, we can move the robots from one configuration to another without any conflicts using only boundedly many moves. Thus, with a small number of time steps, a robot can pass through a haven without collision, regardless of other robots in the haven. Building on this fact, we construct a schedule where each robot interacts with only a bounded number of havens. Within each haven, the number of moves used by a robot is bounded and we show that outside the havens, the robot only makes boundedly many moves as well. This yields a bound on the makespan of the schedule.

Step (ii): Consider the graph induced by the edges in the solution. We partition the vertices of this graph into two sets: important and unimportant vertices. The important vertices are those with degree at least three, the terminals, and any vertex where a robot waits at any time step. We show that the number of important vertices is bounded by a function of k (say, $\zeta(k)$). Moreover, by definition, the unimportant vertices form paths with only degree-2 vertices as internal vertices. Additionally, every robot that visits a vertex in such a path

[2] We recall that by bounded, we mean "bounded by a function of the parameter" – in this case, k.

slides without stopping on any vertex of this path. This structure enables us to show that the graph induced by the edges in the solution (with terminals as roots) is a realization of some rooted graph with at most $\zeta(k)$ vertices as a topological minor in the input graph.

3.1 Bounding the Makespan in an Optimal Schedule

Our first task is to apply a reduction rule to the given instance that enforces useful structural properties on the graph without affecting the existence of a solution.

▶ **Reduction Rule 1.** *Let $I = (G, \mathcal{R} = (\mathcal{M}, \mathcal{F}), k, \ell)$ be an instance of* CSMP*. If there is a path P in G such that every internal vertex of P is disjoint from the set of terminals and has degree exactly two in G, then shorten P to length $min\{|P|, 2k + 1\}$.*

We say that an instance I is *irreducible* if Reduction Rule 1 cannot be applied. In the rest of this section, we work with such instances. Our goal now is to show that every YES-instance has a schedule in which the makespan is bounded by a function of the number of robots.

▶ **Lemma 5.** *Let $I = (G, \mathcal{R} = (\mathcal{M}, \mathcal{F}), k, \ell)$ be an instance of* CSMP*. If I is a* YES*-instance, then I has an optimal schedule with $2^{\mathcal{O}(k^4)}$ moves.*

The proof strategy of Lemma 5 is as follows. We show that for a YES-instance, either the treedepth of the input graph is bounded or a certain locality property holds (roughly speaking, every vertex is sufficiently close to a vertex of degree at least three). We then show that in either case, there is a solution with boundedly-many moves. Specifically, when the treedepth is bounded, we show how to reduce the graph to a bounded-size graph without affecting the existence of a solution (thus implicitly bounding the makespan of an optimal solution) and otherwise, we show how the aforementioned locality property can be used along with a greedy strategy to produce a schedule with boundedly many moves (again, bounding the makespan of an optimal solution).

3.2 Succinct Representation of a Solution

In what follows, fix an optimal schedule $S = \{W_i \mid i \in [k]\}$ for an instance $I = (G, \mathcal{R} = (\mathcal{M}, \mathcal{F}), k, \ell)$. By Lemma 5, we may assume, without loss of generality, that S has length $2^{\mathcal{O}(k^4)}$. We also assume that no robot moves in two consecutive time steps since otherwise, the movements of such a robot could be done in a single time step while the remaining robots wait. Moreover, we say that an edge e of G is *traversed by S* if there is an $i \in [k]$ and a time step j such that the path P^i_j contains the edge e. That is, some robot R_i uses the edge e in some time step j. We denote by G_S, the subgraph of G induced by the edges traversed by S. For a time step j, we denote by $G_{S,j}$ the subgraph of G_S induced by the edges traversed by S up to time step j. A vertex v in G_S is a *waiting vertex* within time interval $[0, j]$ if there is a robot R_i and a time step $j' \leq j$ such that R_i is at v at time steps $j' - 1$ and j. We say that v is a waiting vertex if it is a waiting vertex within some time interval. A vertex v in G_S is an *intersection* vertex within time interval $[0, j]$ if its degree in $G_{S,j}$ is at least three. We say that v is an intersection vertex if it is an intersection vertex within some time interval. A vertex v in $G_{S,j}$ is an *important* vertex within time interval $[0, j]$ if it is either a terminal vertex, or within time interval $[0, j]$ it is a waiting vertex or an intersection vertex. Otherwise, v is *unimportant* within time interval $[0, j]$. We say that v is an important (unimportant) vertex if it is important (resp. unimportant) within some time interval.

▶ **Observation 6.** *The number of waiting vertices within time interval $[0, j]$ is at most $k + j$.*

▶ **Observation 7.** *The unimportant vertices within any time interval $[0, j]$ induce a disjoint union of paths and any endpoint of any of these paths has degree 2 in $G_{S,j}$.*

For a time step j, we denote by $\mathcal{P}_{S,j}$, the set of all x-y paths P of non-zero length such that x and y are distinct important vertices within time interval $[0, j]$ and every internal vertex of P is an unimportant vertex within time interval $[0, j]$.

For two paths P and P' in G, their *crossing points* are those vertices that have degree at least three in the graph induced by the edges in $E(P) \cup E(P')$.

We say that S is a *special schedule up to time step j* if, for every robot R_i and $j > 0$ such that R_i moves in time step j, the path P^i_j has at most four crossing points with any path in $\mathcal{P}_{S,j'}$ where $j' \leq j - 1$. Note that every schedule is trivially special at time step 1 since only a single path has been taken at this point. Moreover, $\mathcal{P}_{S,1}$ has size at least one because a move has been made and so, $\mathcal{P}_{S,j}$ has size at least one for every j. We say that S is a *special schedule* if it is a special schedule up to the final time step.

We say that two schedules are *equivalent at time step j* if the positions of all the robots at the end of time j is the same in both schedules, that is, robot R_i is on vertex v after time step j in one schedule if and only if the same happens in the other schedule. We say that two schedules are equivalent if they are equivalent at every time step. In particular, two equivalent schedules take the same number of time steps.

▶ **Lemma 8.** *There is a solution* S' *that is a special schedule.*

Henceforth, we assume that the solution S is a special schedule.

▶ **Lemma 9.** *The number of important vertices in* G_{S} *is bounded by* $2^{2^{\mathcal{O}(k^4)}}$.

3.3 Leveraging Succinct Representations and Topological Minor Containment

In what follows, fix a schedule $S = \{W_i \mid i \in [k]\}$ for an instance $I = (G, \mathcal{R} = (\mathcal{M}, \mathcal{F}), k, \ell)$.

We define the *representation* of S to be the rooted graph (denoted by H_S) whose vertices are the important vertices of G_S, and in which there is an edge between two vertices u and v if and only if there is a u-v path in G_S whose internal vertices have degree 2 in G_S and are unimportant. Moreover, the terminals receive unique labels (assume that the terminals are labeled with $[p]$ for some $p \leq 2k$) and the remaining vertices receive the same label which is distinct from those of the terminals, say $p + 1$. An illustration is provided in Figure 1. From Lemma 9, we directly obtain:

▶ **Lemma 10.** *If S is a special schedule, then its representation has* $2^{2^{\mathcal{O}(k^4)}}$ *vertices.*

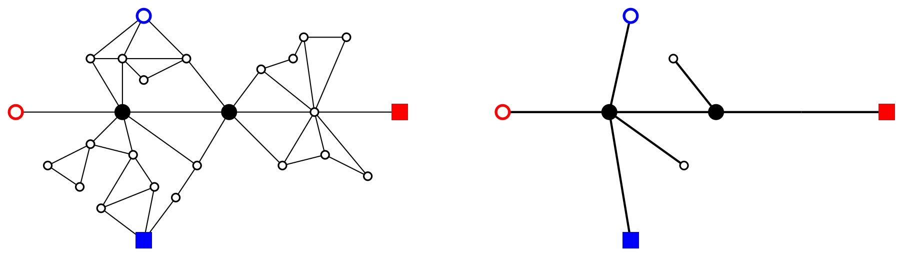

Figure 1 Left: an instance of CSMP with two robots with destinations (red and blue, with destinations marked as squares) and two blocker robots (both marked as solid black circles). Right: A representation of one particular optimal schedule for this instance, where we first move the two blockers and then the remaining two robots.

For the graph G, we let G' denote the rooted graph where the terminals get their unique labels and the rest of the vertices get a specific label $p+1$, where p is the number of terminals.

▶ **Lemma 11.** *H_{S} is a topological minor of G' and moreover, any realization of H_{S} as a topological minor in G' implies a schedule with length at most that of* S.

▶ **Theorem 12.** CSMP *is fixed-parameter tractable parameterized by the number of robots.*

Proof. For a hypothetical optimal solution S, we guess (i.e., use brute-force branching to determine) the representation H_S. By Lemmas 8 and 10, there are at most $2^{2^{2^{\mathcal{O}(k^4)}}}$ choices and these can be enumerated in FPT time. For each H_S, we do the following:

1. Decide whether the guess is feasible by brute-forcing over it. More precisely, we guess the configurations of the robots at each time step and verify that this guess corresponds to a set of non-conflicting routes with at most ℓ moves in total. Reject guesses that do not pass this check. Clearly, this step takes time $f(k)$ for some computable function f.
2. If H_S is not rejected, then invoke Proposition 3 to decide whether H_S is a topological minor of G' and return "yes" if and only if at least one of the invocations returns "yes".

Fixed-parameter tractability follows from the fact that Proposition 3 is used at most $f(k)$ times. The correctness follows from Lemma 10 and Lemma 11. ◀

4 An FPT Algorithm for Planar-CSMP-1 Parameterized by the Makespan

We start by giving a high-level description of the result. Our aim is to prove that Planar-CSMP-1 is fixed-parameter tractable parameterized by ℓ. The crux of our technique is showing that a sufficiently large component of free vertices must contain an edge that can be safely contracted. These contractions eventually result in bounding the size of each free component in the graph, and consequently the diameter and hence the treewidth of the graph (due to planarity). The fixed-parameter tractability of the problem will then be established using Courcelle's Theorem [2, 8].

To show that each large component C of free vertices contains an edge that can be safely contracted, we first show that this is the case unless C is formed by a skeleton consisting of a sufficiently long path of degree-2 vertices (the degree is taken in C), plus a sufficiently small number of additional vertices. We then focus on the skeleton of this component. We show that if the instance is a YES-instance of the problem, then it admits a solution S that interacts "nicely" with this skeleton. Recall from the previous section that a solution can be succinctly represented as a rooted topological minor of size $\sigma(\ell)$ for some computable function σ. This is because at most ℓ robots move in the solution. Now, we can identify a subgraph S of the representation of the solution (i.e., a topological minor that is essentially a roadmap of the solution), that is separated from the rest of the solution by a small cut. The realization of this subgraph uses most of the skeleton and the remaining part of this realization consists of small subgraphs that are close to the skeleton. Therefore, we can assume that all edges in these parts of the subgraphs are realized by edges in the graph.

Thus, it suffices to enumerate each possible representation of the part of the solution that interacts with the skeleton (which can be done in FPT-time), and test whether it exists near the skeleton; if it does, we mark the vertices in the graph that realize this representation. Since the skeleton is long, the total number of marked vertices on the skeleton is small, and an edge with both endpoints unmarked that can be contracted must exist. We remark that in many of the arguments that we make, we rely on the crucial fact that, except for the main robot, the robots are "indistinguishable" from each other. We now proceed to the details.

Refer to a vertex on which a robot other than the main robot is located as a *blocker* vertex. Let the *blocker-distance* between a vertex v and a vertex w, denoted *blockd*(v, w), be the minimum integer p such that there is a v-w path in G containing p blocker vertices. The reason why the blocker-distance is relevant is that we show that vertices which have high blocker-distance from s and t can be removed from the graph. Observe that, for any two vertices v, w, $\mathit{blockd}(v, w)$ can be computed in polynomial time via, e.g., Dijkstra's algorithm.

Let $\mathcal{I} = (G, s, t, \ell, \mathcal{R})$ be an instance and let $F \subseteq V(G)$ be the set of free vertices in G. A *free component* is a connected component of $G[F]$. Free components will be central to our proof – in particular, observe that the blocker-distance between s and t is at most $\ell - 1$ and we can safely remove all vertices from G at blocker distance at least $\ell + 1$ from s. Hence, if there are no large free components, then the diameter of G is bounded. As G is planar, also its treewidth is bounded and we can use Courcelle's Theorem [2, 8] to prove the following:

▶ **Lemma 13.** Planar-CSMP-1 *can be solved in time $f(\ell, \lambda) \cdot |\mathcal{I}|$, where f is a computable function and λ is the size of the largest free component in the input instance $\mathcal{I}$.*

The Structure of Free Components. The following observation is straightforward:

▶ **Observation 14.** *Let Q be a path of free vertices and let x, y be arbitrary vertices on Q such that the subpath of Q between x and y has length at least ℓ. If there exists an x-y path P_{xy} whose internal vertices are disjoint from Q such that the number of blockers on P_{xy} is at most $\ell - \mathrm{blockd}(s, x) - \mathrm{blockd}(y, t) - 1$, then the instance $\mathcal{I}$ is a* YES*-instance and we can output a solution for $\mathcal{I}$ in polynomial time.*

We refer to a path where the above observation does not result in solving the instance as a *resilient* path. In other words, a path Q is resilient if and only if for every pair of vertices x, y in $V(Q)$ whose distance on Q is at least ℓ, there is no x-y path Z in $G - Q$ with at most $\ell - \mathit{blockd}(s, x) - \mathit{blockd}(y, t) - 1$ blockers. Our goal is to show that every "sufficiently large" free component contains a "still sufficiently long" path with certain special properties that, intuitively, allow the path to be "cleanly separated" from a hypothetical solution. This is formalized in the following definition and lemma, where the property of being "still sufficiently long" is given by the function $g(\ell) = 32\ell^6 \cdot 2^{14\ell^2} + 1$.

▶ **Definition 15.** *Let Q be a path in a free component C with endpoints u', v'. We say that Q is ℓ-*clean *if it satisfies the following conditions:*

1. *every vertex in Q has degree precisely 2 in C;*
2. *$|V(Q)| > g(\ell)$;*
3. *if $\mathcal{I}$ admits a solution that intersects $V(Q)$, then there are two vertices $u, v \in V(Q)$, each at distance at most $\ell^2 + 1$ from u' and v', respectively, and a solution S for $\mathcal{I}$ such that: $\{u, v\}$ is a cut in G_{S} between $\{s, t\}$ and the u-v subpath of Q; we denote by Q^{S}_{uv} the connected subgraph of G_{S} that contains u, v and all the connected components of $G_{\mathrm{S}} - \{u, v\}$ that do not contain s or t; and*
4. *Q is resilient.*

▶ **Lemma 16.** *There is a polynomial-time procedure which takes as input a free component C in an instance $\mathcal{I} = (G, s, t, \ell, \mathcal{R})$ such that $|V(C)| \geq (g(\ell) + 1) \cdot (\ell - 1) + 3(\ell + 2)$, and either solves the instance, or outputs an instance equivalent to $\mathcal{I}$ obtained by contracting an edge in C, or outputs an ℓ-clean path Q in C.*

4.1 Finding Irrelevant Edges

For the following considerations, it will be useful to proceed with a fixed choice of C and an ℓ-clean path Q. Next, we will show that we can make an even stronger assumption about the hypothetical solutions "passing through" Q.

▶ **Lemma 17.** *Assume that $\mathcal{I}$ admits a solution that intersects $V(Q)$. Then there exist vertices u, v each at distance at most $\ell^2 + 1$ from u' and v', respectively, and a solution S^* for $\mathcal{I}$ satisfying properties 3-4 in Definition 15, and furthermore satisfying the following property 5: $|V(Q^{S^*}_{uv}) - V(Q)| \leq \ell$.*

For the following, we will refer to a solution satisfying properties 3-5 as (Q, u, v)-*canonical*, or simply Q-canonical in brief when the identities of u and v do not matter. From the properties of canonical solutions, we can directly establish the following:

▶ **Lemma 18.** *Let S^* be a Q-canonical solution. Then there is a $(2\ell^2)$-vertex rooted graph $U^{S^*}_{uv}$ (with roots u, v and* `occupied`*), such that $U^{S^*}_{uv}$ is a rooted topological minor of G (with roots u, v and* `occupied`*) and there is a realization (ϕ, ψ) of $U^{S^*}_{uv}$ that maps $U^{S^*}_{uv}$ to $Q^{S^*}_{uv}$. Moreover, waiting vertices are in $\phi(U^{S^*}_{uv})$ and for every edge $e \in E(U^{S^*}_{uv})$ either $\psi(e)$ is a single edge or a subpath of Q.*

Crucially, below we provide a procedure that can efficiently test for $U^{S^*}_{uv}$ as defined above.

▶ **Lemma 19.** *Let Q be an ℓ-clean u'-v' path and let u, v be the vertices specified in property 3 of Definition 15. Let $p \leq 2\ell$ and U be a connected p-vertex rooted graph with roots u, v and "*`occupied`*" such that U contains at most $\ell - \text{blockd}(s, x) - \text{blockd}(y, t) - 1$ occupied vertices. There is a procedure which runs in time $2^{\mathcal{O}(\ell^2)} \cdot |V(G)||V(Q)|$ and either outputs a realization (ϕ, ψ) of U which results in a subgraph $\Gamma \subseteq G$ such that:*

- *u, v separate Γ from u' and v', and*
- *for every edge $e \in E(U)$ either $\psi(e)$ is a single edge or a subpath of Q, or*

correctly determines that no such $(\Gamma, (\phi, \psi))$ exists.

Proof Sketch. As the first step, we use the ℓ-cleanness of Q to establish that if Γ exists, then it must have a "necklace-like" structure. In particular, such a Γ must consist of the u-v subpath of Q along with at most p connected subgraphs $\Gamma_1, \ldots, \Gamma_j$, each of size $p + p\ell$, each intersecting the u-v subpath of Q but otherwise being pairwise disjoint, and having pairwise distance at least $\ell + 1$ on $G[Q]$. We further may assume that $\Gamma_1, \ldots, \Gamma_j$ are ordered based on their distance to u.

With this knowledge, we exhaustively branch to consider every possible choice of rooted subgraphs $\Gamma_1, \ldots, \Gamma_j$ (up to isomorphism) to construct a set of *signatures* of a hypothetical tuple $(\Gamma, (\phi, \psi))$; crucially, the number of such signatures can be upper-bounded by $\ell^{\mathcal{O}(\ell)}$. We can now show that to test whether there exists some tuple $(\Gamma, (\phi, \psi))$, it suffices to consider each signature individually and see if there is a "leftmost" Γ matching that signature, i.e., a Γ where the subgraphs $\Gamma_1, \ldots, \Gamma_j$ are pushed as close to u as possible (while adhering to the properties of the given signature). This then allows us to process the u-v subpath of Q from u to v and test, at each vertex x, whether it could be the "first vertex" of Γ_1. Once we find the leftmost such x, we can jump ahead by $\ell + 1$ and proceed to Γ_2, and so forth.

A key point here is that since each of the subgraphs $\Gamma_1, \ldots, \Gamma_j$ has bounded size, it suffices to perform each such test on the $(p + p\ell)$-neighborhood of x. Since this graph is planar and has bounded diameter (by a function of ℓ), it also has bounded treewidth [5, 30], thus allowing us to employ well-known subgraph-isomorphism testing algorithms [14] to determine whether each x can indeed be part of a specified subgraph Γ_i. We remark that performing this test on each vertex individually is critical: the subgraph of G induced on the neighborhood of all of Q may have arbitrarily large treewidth. ◀

We call the subgraph Γ identified above a *host* for the *roadmap* U.

▶ **Reduction Rule 2.** *Let Q be a given ℓ-clean u'-v' path. There is a reduction rule which enumerates:*

- *the family $\mathcal{U}$ of all connected rooted graphs (with roots u, v and `occupied`) containing at most $(2\ell^2)$-many vertices, and*
- *the set Z of all vertex pairs (u, v) at distance at most $\ell^2 + 1$ from the respective endpoints of Q,*

and then for each choice of $U \in \mathcal{U}$ and $(u, v) \in Z$, applies Lemma 19 to determine whether there is a host of U w.r.t. (u, v) on Q. If there is, we mark *every vertex in $\phi(U) \cap V(Q)$ where (ϕ, ψ) is the computed realization.*

Afterwards, if there is an edge e on Q with two unmarked endpoints, we contract e.

▶ **Lemma 20.** *Reduction Rule 2 is safe. Moreover, for each ℓ-clean path Q, it can be executed in time at most $2^{\mathcal{O}(\ell^2)} \cdot |V(G)| \cdot |V(Q)|$, and is guaranteed to perform a contraction if Q contains more that $32\ell^6 \cdot 2^{14\ell^2} + 1$ many vertices.*

▶ **Theorem 21.** PLANAR-CSMP-1 *is fixed-parameter tractable parameterized by ℓ.*

Proof. We begin by applying Lemma 16 to each free component in G of size at least $\big(g(\ell) + 1) \cdot (\ell - 1)\big) + 3(\ell + 2)$. If this solves the instance, we are done; otherwise, it is guaranteed to either produce an equivalent instance obtained by an edge contraction (in which case we repeat), or identify an ℓ-clean path Q of length more than $g(\ell) = 32\ell^6 \cdot 2^{14\ell^2} + 1$. In the third case, we apply Reduction Rule 2 to Q to produce an equivalent instance obtained by an edge contraction in Q – as guaranteed by Lemma 20 – and restart.

The exhaustive application of the above procedure either correctly solves the input instance, or results in an equivalent instance such that each free component in G has size at most $\big(g(\ell)+1) \cdot (\ell-1)\big) + 3(\ell+2) - 1$. At this point, we invoke Lemma 13 to solve the instance. The overall running time is upper-bounded by $2^{\mathcal{O}(\ell^2)} \cdot |\mathcal{I}|^{\mathcal{O}(1)} + \zeta(\ell) \cdot |\mathcal{I}|$, where the former term is the running time of the preprocessing steps and ζ is a computable function that results from the application of Lemma 13 (and specifically of Courcelle's Theorem within that lemma) on an instance with the aforementioned bound on the size of free components. ◀

5 Conclusion

In this paper, we designed FPT algorithms for two important variants of the Coordinated Motion Planning problem on undirected graphs, CSMP and Planar CSMP-1, in which robots move by sliding along unobstructed paths. Our results take a different perspective and contribute new insights to the design of FPT algorithms for coordinated motion planning problems on graphs, developing and employing new techniques that combine topological graph theory, fixed-parameter tractability and computational logic. Our work opens up several interesting questions pertaining to the parameterization by the makespan ℓ.

1. Is CSMP parameterized by the makespan W[1]-hard on general graphs?
2. Can Theorem 21 be generalized to solve CSMP on planar graphs?
3. What is the parameterized complexity of CSMP and CSMP-1 on grids where each sliding-move/path is either horizontal or vertical?

References

1 Manuel Abellanas, Sergey Bereg, Ferran Hurtado, Alfredo García Olaverri, David Rappaport, and Javier Tejel. Moving coins. *Computational Geometry*, 34(1):35–48, 2006. doi:10.1016/J.COMGEO.2005.06.005.

2 Stefan Arnborg, Jens Lagergren, and Detlef Seese. Easy problems for tree-decomposable graphs. *Journal of Algorithms*, 12:308–340, 1991. doi:10.1016/0196-6774(91)90006-K.

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, pages 12:1–12:16, 2022. doi:10.4230/LIPIcs.SoCG.2022.12.

4 Sergey Bereg, Adrian Dumitrescu, and János Pach. Sliding disks in the plane. *International Journal of Computational Geometry & Applications*, 18(05):373–387, 2008. doi:10.1142/S0218195908002684.

5 Therese Biedl. On triangulating k-outerplanar graphs. *Discret. Appl. Math.*, 181:275–279, 2015. doi:10.1016/J.DAM.2014.10.017.

6 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. URL: http://ijcai.org/Abstract/15/110.

7 Gruia Călinescu, Adrian Dumitrescu, and János Pach. Reconfigurations in graphs and grids. *SIAM Journal on Discrete Mathematics*, 22(1):124–138, 2008. doi:10.1137/060652063.

8 Bruno Courcelle. The monadic second-order logic of graphs. i. recognizable sets of finite graphs. *Inf. Comput.*, 85(1):12–75, 1990. doi:10.1016/0890-5401(90)90043-H.

9 Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015. doi:10.1007/978-3-319-21275-3.

10 A. Deligkas, E. Eiben, R. Ganian, I. Kanj, and M. S. Ramanujan. Parameterized algorithms for coordinated motion planning: Minimizing energy. In Karl Bringmann, Martin Grohe, Gabriele Puppis, and Ola Svensson, editors, *51st International Colloquium on Automata, Languages, and Programming, ICALP 2024, July 8-12, 2024, Tallinn, Estonia*, volume 297 of *LIPIcs*, pages 53:1–53:18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPICS.ICALP.2024.53.

11 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. doi:10.1137/18M1194341.

12 Erik D. Demaine and Mikhail Rudoy. A simple proof that the $(n^2 - 1)$-puzzle is hard. *Theoretical Computer Science*, 732:80–84, 2018.

13 Reinhard Diestel. *Graph Theory, 4th Edition*, volume 173 of *Graduate texts in mathematics*. Springer, 2012.

14 Frederic Dorn. Planar subgraph isomorphism revisited. In Jean-Yves Marion and Thomas Schwentick, editors, *27th International Symposium on Theoretical Aspects of Computer Science, STACS 2010, March 4-6, 2010, Nancy, France*, volume 5 of *LIPIcs*, pages 263–274. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2010. doi:10.4230/LIPICS.STACS.2010.2460.

15 Rodney G. Downey and Michael R. Fellows. *Fundamentals of Parameterized Complexity*. Texts in Computer Science. Springer, 2013. doi:10.1007/978-1-4471-5559-1.

16 Adrian Dumitrescu. Motion planning and reconfiguration for systems of multiple objects. In Sascha Kolski, editor, *Mobile Robots*, chapter 24. IntechOpen, Rijeka, 2007.

17 Eduard Eiben, Robert Ganian, and Iyad Kanj. The parameterized complexity of coordinated motion planning. In *Proceedings of the 39th International Symposium on Computational Geometry*, volume 258 of *LIPIcs*, pages 28:1–28:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPICS.SOCG.2023.28.

18 Sándor P. Fekete, Phillip Keldenich, Dominik Krupke, and Joseph S. B. Mitchell. Computing coordinated motion plans for robot swarms: The CG: SHOP challenge 2021. *ACM Journal on Experimental Algorithmics*, 27:3.1:1–3.1:12, 2022. doi:10.1145/3532773.

19 Henning Fernau, Torben Hagerup, Naomi Nishimura, Prabhakar Ragde, and Klaus Reinhardt. On the parameterized complexity of the generalized rush hour puzzle. In *Proceedings of the 15th Canadian Conference on Computational Geometry*, pages 6–9, 2003. URL: http://www.cccg.ca/proceedings/2003/22.pdf.

20 Foivos Fioravantes, Dusan Knop, Jan Matyás Kristan, Nikolaos Melissinos, and Michal Opler. Exact algorithms and lowerbounds for multiagent pathfinding: Power of treelike topology. *CoRR*, abs/2312.09646, 2023. doi:10.48550/arXiv.2312.09646.

21 Gary W. Flake and Eric B. Baum. Rush hour is PSPACE-complete, or "Why you should generously tip parking lot attendants". *Theoretical Computer Science*, 270(1-2):895–911, 2002. doi:10.1016/S0304-3975(01)00173-6.

22 Martin Grohe, Ken-ichi Kawarabayashi, Dániel Marx, and Paul Wollan. Finding topological subgraphs is fixed-parameter tractable. In Lance Fortnow and Salil P. Vadhan, editors, *Proceedings of the 43rd ACM Symposium on Theory of Computing, STOC 2011, San Jose, CA, USA, 6-8 June 2011*, pages 479–488. ACM, 2011. doi:10.1145/1993636.1993700.

23 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. doi:10.1109/TSSC.1968.300136.

24 Iyad Kanj and Salman Parsa. On the parameterized complexity of motion planning for rectangular robots. In Wolfgang Mulzer and Jeff M. Phillips, editors, *40th International Symposium on Computational Geometry, SoCG 2024, June 11-14, 2024, Athens, Greece*, volume 293 of *LIPIcs*, pages 65:1–65:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPICS.SOCG.2024.65.

25 Tuukka Korhonen. A single-exponential time 2-approximation algorithm for treewidth. In *62nd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2021, Denver, CO, USA, February 7-10, 2022*, pages 184–192. IEEE, 2021. doi:10.1109/FOCS52979.2021.00026.

26 Jaroslav Nešetřil and Patrice Ossona de Mendez. *Sparsity - Graphs, Structures, and Algorithms*, volume 28 of *Algorithms and combinatorics*. Springer, 2012. doi:10.1007/978-3-642-27875-4.

27 Christos H. Papadimitriou, Prabhakar Raghavan, Madhu Sudan, and Hisao Tamaki. Motion planning on a graph (extended abstract). In *STOC*, pages 511–520, 1994. doi:10.1109/SFCS.1994.365740.

28 Geetha Ramanathan and Vangalur S. Alagar. Algorithmic motion planning in robotics: Coordinated motion of several disks amidst polygonal obstacles. In *ICRA*, volume 2, pages 514–522, 1985. doi:10.1109/ROBOT.1985.1087248.

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

30 Neil Robertson and Paul D. Seymour. Graph minors. III. planar tree-width. *J. Comb. Theory B*, 36(1):49–64, 1984. doi:10.1016/0095-8956(84)90013-3.

31 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.

32 Jacob T. Schwartz and Micha Sharir. On the "piano movers'" problem I. The case of a two-dimensional rigid polygonal body moving amidst polygonal barriers. *Communications on Pure and Applied Mathematics*, 36(3):345–398, 1983.

33 Jacob T. Schwartz and Micha Sharir. On the "piano movers" problem. II. General techniques for computing topological properties of real algebraic manifolds. *Advances in Applied Mathematics*, 4(3):298–351, 1983.

34 Guni Sharon, Roni Stern, Ariel Felner, and Nathan R. Sturtevant. Conflict-based search for optimal multi-agent pathfinding. *Artificial Intelligence*, 219:40–66, 2015. doi:10.1016/J.ARTINT.2014.11.006.

35 Irving Solis, James Motes, Read Sandström, and Nancy M. Amato. Representation-optimal multi-robot motion planning using conflict-based search. *IEEE Robotics Automation Letters*, 6(3):4608–4615, 2021. doi:10.1109/LRA.2021.3068910.

36 Glenn Wagner and Howie Choset. Subdimensional expansion for multirobot path planning. *Artificial Intelligence*, 219:1–24, 2015. doi:10.1016/J.ARTINT.2014.11.001.

37 Jingjin Yu and Steven M. LaValle. Structure and intractability of optimal multi-robot path planning on graphs. In *AAAI*, pages 1443–1449, 2013. `doi:10.1609/AAAI.V27I1.8541`.

38 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. `doi:10.1109/TRO.2016.2593448`.

39 Jingjin Yu and Daniela Rus. Pebble motion on graphs with rotations: Efficient feasibility tests and planning algorithms. In *WAFR*, volume 107 of *Springer Tracts in Advanced Robotics*, pages 729–746, 2014. `doi:10.1007/978-3-319-16595-0_42`.