Computing Coordinated Motion Plans for Robot Swarms: The CG:SHOP Challenge 2021

SÁNDOR P. FEKETE, PHILLIP KELDENICH, and DOMINIK KRUPKE, Department of Computer Science, TU Braunschweig, Germany
JOSEPH S. B. MITCHELL, Department of Applied Mathematics and Statistics, Stony Brook University, USA

We give an overview of the 2021 Computational Geometry Challenge, which targeted the problem of optimally coordinating a set of robots by computing a family of collision-free trajectories for a set S of n pixel-shaped objects from a given start configuration to a desired target configuration.

CCS Concepts: • **Theory of computation → Design and analysis of algorithms**; *Computational Geometry;*

Additional Key Words and Phrases: Computational geometry, geometric optimization, complexity, motion planning, algorithm engineering, contest

ACM Reference format:
Sándor P. Fekete, Phillip Keldenich, Dominik Krupke, and Joseph S. B. Mitchell. 2022. Computing Coordinated Motion Plans for Robot Swarms: The CG:SHOP Challenge 2021. *J. Exp. Algorithmics* 27, 3, Article 3.1 (July 2022), 12 pages.
https://doi.org/10.1145/3532773

1 INTRODUCTION

The "CG:SHOP Challenge" (Computational Geometry: Solving Hard Optimization Problems) originated as a workshop at the 2019 Computational Geometry Week (CG Week) in Portland, Oregon, in June 2019. The goal was to conduct a computational challenge competition that focused attention on a specific hard geometric optimization problem, encouraging researchers to devise and implement solution methods that could be compared scientifically based on how well they performed on a database of carefully selected and varied instances. While much of computational geometry research is theoretical, often seeking provable approximation algorithms for NP-hard optimization problems, the goal of the CG Challenge was to set the metric of success based on computational results on a specific set of benchmark geometric instances. The 2019 CG Challenge

This work was funded by DFG project "Computational Geometry: Solving Hard Optimization Problems" (CG:SHOP), FE407/21-1. J. Mitchell has been partially supported by the National Science Foundation (CCF-2007275) and the US-Israel Binational Science Foundation (BSF project 2016116).
Authors' addresses: S. P. Fekete, P. Keldenich, and D. Krupke, Department of Computer Science, TU Braunschweig, Mühlenpfordtstr. 23, 38106 Braunschweig, Germany; emails: {s.fekete, p.keldenich, d.krupke}@tu-bs.de; J. S. B. Mitchell, Department of Applied Mathematics and Statistics, Stony Brook University, Stony Brook, NY 11794, USA; email: joseph.mitchell@stonybrook.edu.
Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
© 2022 Copyright held by the owner/author(s). Publication rights licensed to ACM.
1084-6654/2022/07-ART3.1 $15.00
https://doi.org/10.1145/3532773

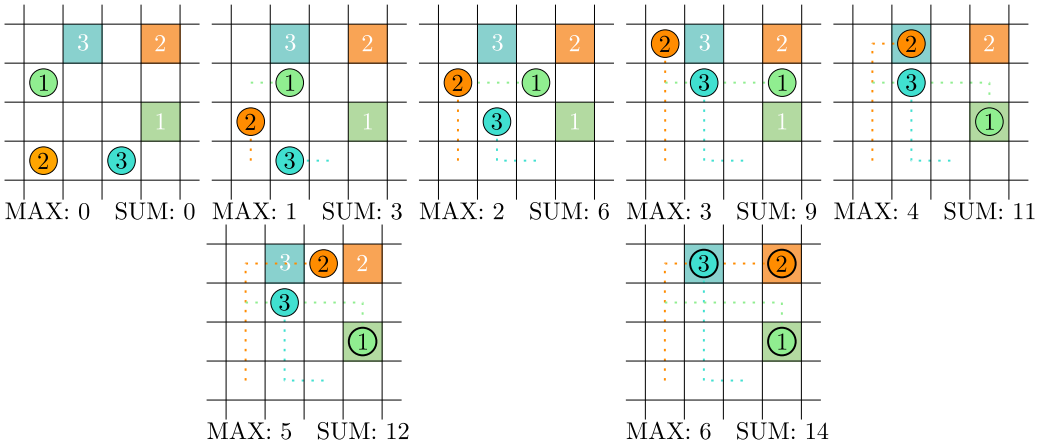

Fig. 1. A feasible sequence of positions that minimizes both makespan and total distance. Robot positions are shown by circles, target positions by squares.

focused on the problem of computing minimum-area polygons whose vertices were a given set of points in the plane. This Challenge generated a strong response from many research groups, from both the computational geometry and the combinatorial optimization communities, and resulted in a lively exchange of solution ideas.

For CG Week 2020, the second CG:SHOP Challenge became an event within the CG Week program, with the top-performing solutions reported in the Symposium on Computational Geometry proceedings. The schedule for the Challenge was advanced earlier, to give an opportunity for more participation, particularly among students, e.g., as part of course projects.

The third edition of the Challenge in 2021 continued the 2020 format, leading to contributions in the SoCG proceedings.

2 THE CHALLENGE: COORDINATED MOTION PLANNING

Coordinating the motion of a set of objects is a fundamental problem that occurs in a large spectrum of theoretical contexts and practical applications. A typical task arises from relocating a large collection of agents from a given start into a desired goal configuration in an efficient manner, while avoiding collisions between objects or with obstacles.

2.1 The Problem

The specific problem that formed the basis of the 2021 CG Challenge was the following; see Figure 1 for a simple example.

Problem: Coordinated Motion Planning of unit squares in the plane.

Given: A set of n axis-aligned unit-square robots in the plane, a set $S = \{s_1, \ldots, s_n\}$ of n distinct *start* pixels (unit squares) of the integer grid, and a set $T = \{t_1, \ldots, t_n\}$ of n distinct *target* pixels of the integer grid. In addition, there may be a set of obstacles, consisting of a number of stationary, blocked pixels that cannot be used by robots at any time.

Goal: The task is to compute a *feasible* set of trajectories for all n robots, with the trajectory for robot i moving it from s_i to t_i, such that the overall schedule is optimal with respect to an objective function.

In order to be feasible, a trajectory must satisfy a number of conditions. During each unit of time, each robot can move (at unit speed) in a direction (north, south, east, or west) to an adjacent pixel, provided the robot remains disjoint from all other robots during the motions. This condition has to be satisfied at all times, not just when robots are at pixel positions. For example, if there are robots at each of the two adjacent pixels (x, y) and $(x+1, y)$, then the robot at (x, y) can move east into position $(x + 1, y)$ only if the robot at $(x + 1, y)$ moves east at the same time, so that the two robots remain in contact, during the movement, but never overlap.

The contest was run on two different objective functions, as follows.

MAX: Minimize the makespan, i.e., the time until all robots have reached their destinations.
SUM: Minimize the total distance traveled by all robots.

2.2 Related Work

Coordinating the motion of many agents plays a central role when dealing with large numbers of moving robots, vehicles, aircraft, or people. How can each agent choose an efficient route that avoids collisions with other agents as they simultaneously move to their destinations? These basic questions arise in many applications, such as ground swarm robotics [17, 18], aerial swarm robotics [3, 26], air traffic control [5], and vehicular traffic networks [10, 19].

Multi-robot coordination dates back to the early days of robotics and computational geometry. The seminal work by Schwartz and Sharir [20] from the 1980s considers coordinating the motion of disk-shaped objects among obstacles. Their algorithms are polynomial in the complexity of the obstacles, but exponential in the number of disks. Hopcroft et al. [12] and Hopcroft and Wilfong [13] proved PSPACE-completeness of moving multiple robots to a target configuration, showing the significant challenge of coordinating many robots.

There is a vast body of other related work dealing with multi-robot motion planning, from both theory and practice. For a more extensive overview, see [7]. In both discrete and geometric variants of the problem, the objects can be *labeled*, *colored*, or *unlabeled*. In the *labeled* case, the objects are all distinguishable and each object has its own, uniquely defined target position. In the *colored* case, the objects are partitioned into k groups and each target position can only be covered by an object with the right color. This was considered by Solovey and Halperin [21], who present and evaluate a practical sampling-based algorithm. In the *unlabeled* case, objects are indistinguishable and target positions can be covered by any object.

This scenario was first considered by Kloder and Hutchinson [14], who presented a practical sampling-based algorithm. Turpin et al. [25] give an algorithm for finding a solution in polynomial time, if one exists. This is optimal with respect to the longest distance traveled by any one robot but only holds for disk-shaped robots under additional restrictive assumptions on the free space. For unit disks and simple polygons, Adler et al. [1] provide a polynomial-time algorithm under the additional assumption that the start and target positions have some minimal distance from each other. Under similar separability assumptions, Solovey et al. [23] provide a polynomial-time algorithm that produces a set of paths that is no longer than OPT + $4m$, where m is the number of robots and OPT is the total length of the paths in an optimal solution. However, they do not consider the makespan, only the total path length. On the negative side, Solovey and Halperin [22] prove that the unlabeled multiple-object motion planning problem is PSPACE-hard, even when restricted to unit square objects in a polygonal environment.

There is also a wide range of practical related work. Self-configuration of robots as active agents was studied by Naz et al. [16]. A basic model in which robots are used as building material was introduced by Derakhshandeh et al. [8, 9]. This resembles Claytronics robots like Catoms; see

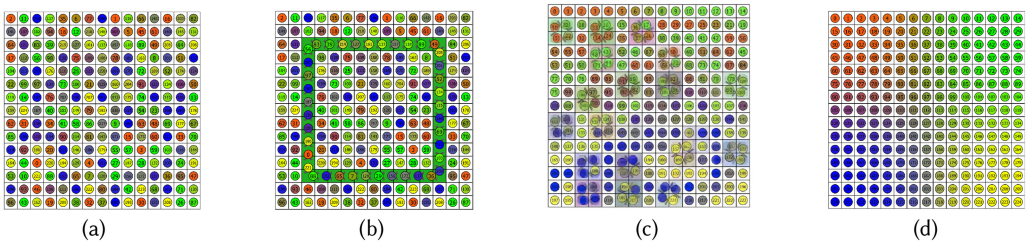

Fig. 2. Parallel reconfiguration, established by [6, 7]: (a) Start configuration. (b) A feasible, parallel reconfiguration move. (c) Parallel reconfiguration moves. (d) Target configuration. See https://www.ibr.cs.tu-bs.de/users/fekete/Videos/CoordinatedMotionPlanning.mp4 for a video [2].

Goldstein and Mowry [11]. In more recent work, Thalamy et al. [24] consider using scaffolding structures for asynchronous reconfiguration.

For an instance of parallel reconfiguration, a lower bound for the time required for *all* robots to reach their destinations is the time it takes to move just *one* robot to its destination in the absence of other robots, i.e., by the maximum distance between a robot's origin and destination. Moving a dense arrangement of robots to their destinations while avoiding collisions may require substantially more time than this lower bound. This motivates the *stretch factor*, which is defined to be the ratio of the time taken by a parallel motion plan divided by the simple lower bound.

In recent work, Demaine et al. [6, 7] provide several fundamental insights into these problems of coordinated motion planning for the scenario with labeled robots. They develop algorithms that (under relatively mild assumptions on the separation between robots, slightly more generous than provided in the Challenge) achieve *constant* stretch factors that are independent of the number of robots. Thus, these algorithms provide an absolute performance guarantee on the makespan of the parallel motion schedule, which implies that the schedule is a constant-factor approximation of the best possible schedule. For densely packed arrangements of robots (without separation assumptions), they proved that a constant stretch factor is no longer possible, and gave upper and lower bounds on the worst-case stretch factor. See Figure 2 for an illustration.

2.3 Instances

An important part of any challenge is the creation of suitable instances. If the instances are easy to solve to optimality, the challenge becomes trivial. If instances require a huge amount of computation to find any decent solutions, the challenge may heavily favor teams that can afford better computation equipment. The same is true if the set of instances becomes too large to manage with a single (or few) computers.

A priori, it is difficult to tell how hard finding a good solution to an instance is and which parameters influence the difficulty of solving an instance (and in what way). Therefore, it is important to create a set of instances that are diverse with respect to parameters that are likely to influence their difficulty.

To create interesting and challenging instances, we thus developed an instance generator that can be tuned by adjusting several parameters to create diverse instances. Basic parameters control the size of the map, the density to which the map is filled with robots, and the distribution functions for start and target positions. For the distribution functions, we used uniform distributions and distributions based on various images, such as microscope images of bacteria colonies. The image-based instances often have interesting, natural patterns; their accumulations of robots in some areas were expected to yield challenging instances. Additionally, we created artificial

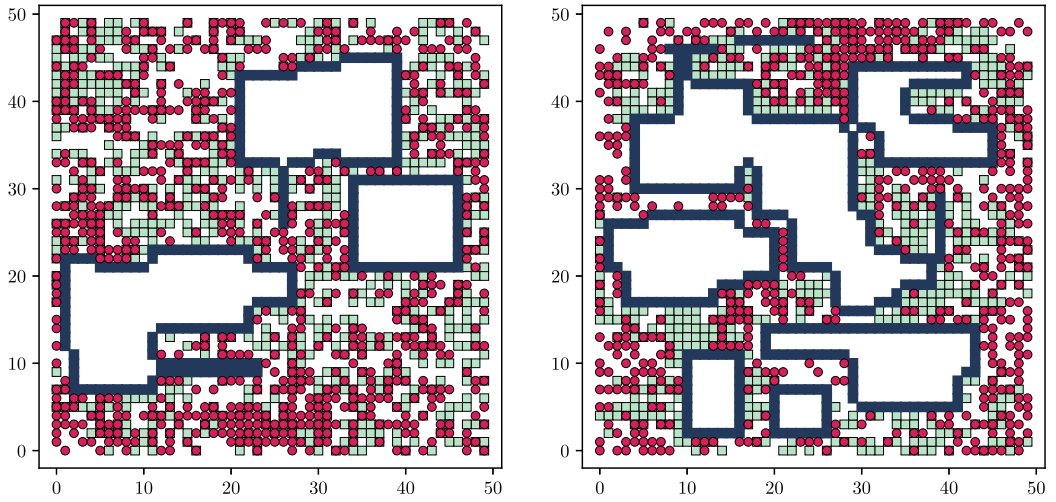

Fig. 3. A visualization of two instances with obstacles (dark pixels). Start and target positions of robots are indicated by red disks and green squares.

bottlenecks and accumulations of robots by inserting obstacles and clusters of robots to increase the difficulty.

Obstacles are created by randomly placing rectangles with (truncated) normally distributed sizes. Any area that is completely surrounded by obstacles also becomes an obstacle to ensure that instances remain feasible. Despite its simplicity, this procedure is able to create various interesting obstacles after some parameter tuning; see Figure 3 for an example.

The idea of robot clusters is to have groups of robots that are close to each other in the start and target configurations but have to travel a significant distance as a group. During its motion, such a robot cluster may have to pass through narrow obstacles or may clash with other clusters, yielding interesting conflicts. Robot clusters are created by specifying the desired number of clusters and the parameters of a normal distribution controlling their size. For each cluster, a start and a target is randomly selected according to the distribution functions controlling the robots' start and target positions. A corresponding number of robots is then distributed within size-dependent windows around the start and target positions of the cluster. These windows may be too small to contain the start and target positions due to obstacles or robots that have already been placed; in that case, we retry with a larger window.

As described above, certain parameters (such as the density) can be controlled directly by our generator. To ensure that our instances are diverse even with respect to parameters for which this is not possible, we generated a large set of instances by selecting a diverse set of possible values for each parameter of our generator and then generating several instances for each possible combination of these parameter values.

This resulted in a set of more than 10,000 candidate instances from which we then selected the challenge instances as follows. We measured several properties for each of the generated instances, including the number of robots, the density, the number of robot clusters, the number of robots that were part of a robot cluster, and the volume and free area of the map. Based on these properties, we then defined and tuned a distance function between the instances and used a greedy dispersion algorithm to select a total of 200 diverse instances from the candidates; see Figure 4 for an overview of the distribution of some important properties across the selected instances.

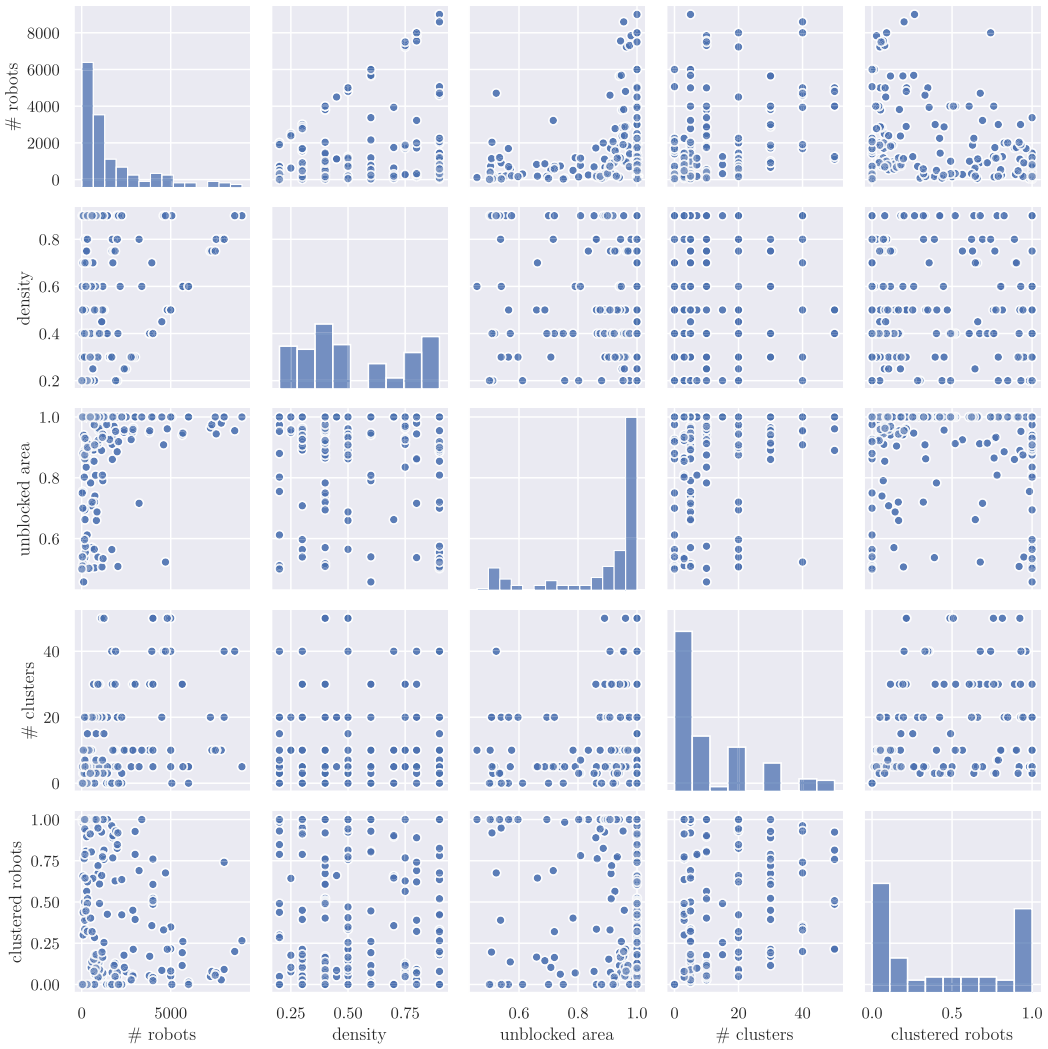

Fig. 4. A pair plot of the distribution of our instances according to several important instance properties.

2.4 Evaluation

The contest was run on a total of 203 instances, 3 of which were small, manually generated instances. For either objective function, a team's score for an instance I is the ratio L/V between L, the objective value of the best submitted solution for I, and V, the best objective value of any valid solution for I submitted by the team. The score for each instance is thus a number between 0 and 1, with 1 being the score awarded to the teams with the best submitted solution and 0 being a default score. As a consequence, the total score is a number between 0 and 203 for either contest, where 203 is best possible. For each contest (MAX and SUM), participants were compared based on their total scores; the winner for each contest was the one with the highest score.

In case of ties, the tiebreaker was set to be the time a specific score was obtained. This turned out not to be necessary.

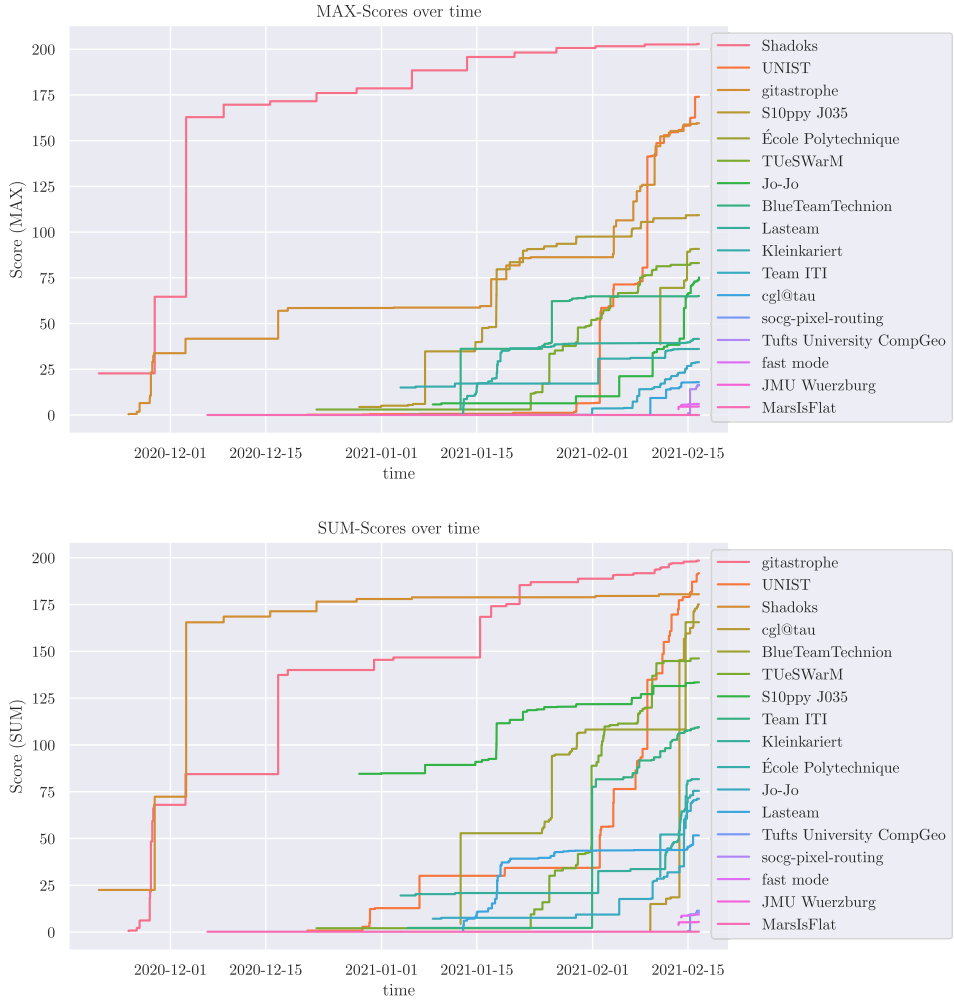

Fig. 5. Progress of the teams over time. Team Shadoks built its lead in the MAX-objective very early and only did minor improvements over the rest of the time. For team gitastrophe, a continuous progress is visible. Team UNIST, on the other hand, started relatively late. Most teams seem to have only focused on the SUM-objective, as these scores show the largest jumps, with submissions picking up in the last month of the competition.

2.5 Categories

The contest was run in an *Open Class*, in which participants could use any computing device, any amount of computing time (within the duration of the contest), and any team composition. In the *Junior Class*, a team was required to consist exclusively of participants who were eligible according to the rules of CG:YRF (the *Young Researchers Forum* of CG Week), defined as not having defended a formal doctorate before 2019.

2.6 Server and Timeline

The contest itself was run through a dedicated server at TU Braunschweig, hosted at https://cgshop.ibr.cs.tu-bs.de/competition/cg-shop-2021/. It opened at 00:00 AoE on November 20, 2020, and closed at 24:00 (midnight, AoE), February 15, 2021.

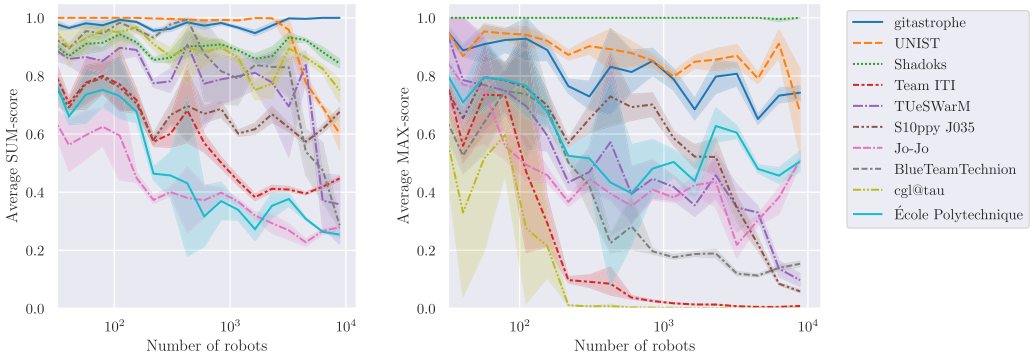

Fig. 6. The scores over the instance sizes of the 10 best teams. We can see that team Shadoks is nearly unbeaten on the MAX objective. Team UNIST is very strong for smaller instances and SUM, but the approach of team gitastrophe seems to scale better and wins on the larger instances by a significant margin.

Table 1. The Top of the Final Leaderboard for MAX

Position	Team	Score	# Best Solutions
1	Shadoks	202.9375	202
2	UNIST	174.0180	14
3	gitastrophe^J	159.5472	24
4	S10ppy J035^J	109.2778	3
5	École Polytechnique^J	90.8213	3
6	TUeSWarM^J	83.0897	7
7	JoJo^J	75.2449	6
8	BlueTeamTechnion^J	65.0906	7
9	Lasteam^J	41.6065	1
10	Kleinkariert^J	36.0898	0

"Best solutions" are the best found by any participating team, which does not exclude the possibility of better solutions, but provides a score of 1 for that instance; scores are truncated to four decimal places. Junior teams are indicated by ^{J}.

3 OUTCOMES

A total of 17 teams submitted solutions. In the end, the leaderboard for the top 10 teams in both categories looked as shown in Tables 1 and 2; note that according to the scoring function, a higher score is better.

The progress over time of each team's score can be seen in Figure 5; the best solutions for all instances (displayed by score) can be seen in Figure 6. For a breakdown of individual difficulty of instances, see Figure 7 (for SUM) and Figure 8 (for MAX).

There were three clear frontrunners, with Team Shadoks placing first and third in MAX and SUM, respectively; Team UNIST came in second in both categories, while Team gitastrophe placed third and first in MAX and SUM. A closer look at the scores reveals that Team Shadoks achieved almost perfect overall in MAX, which was sufficient to place first in the sum of both objective functions, at 401.4318. Team UNIST managed a balanced outcome for both objective functions, for a combined score of 365.8073. Conversely, Team gitastrophe achieved the best score for SUM, but a slightly inferior result for MAX, resulting in a combined score of 358.0415; at the same time, they also placed first in all three events in the Junior Class.

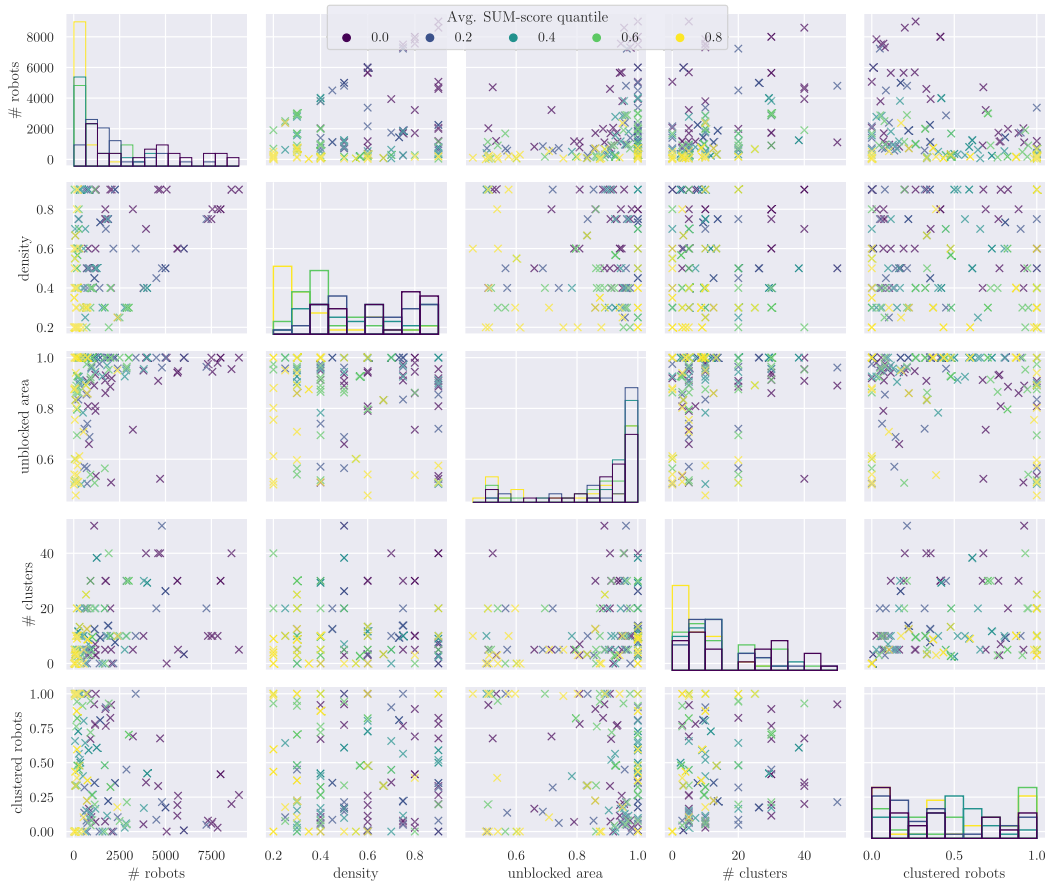

Fig. 7. This plot provides some clues for the practical difficulty of instances, indicated by low average scores. Because the scores are based on the best submitted solution, this implies that some teams with special approaches were able to obtain significantly better results than the "basic" approach. In the plots, we split the instances into five quantiles, such that the instances with high average score are shown in yellow, while darker colors indicate lower average score, i.e., instances considered to be difficult. The most dominant factor appears to be the number of robots. Because it correlates with many other parameters and the distribution is not fully homogeneous, considering individual parameters would be highly skewed. We therefore also show combinations of parameters, especially with the number of robots to counteract the effect that some parameter ranges have on average significantly fewer or more robots. The diagonals show the distribution of individual parameters; we can see that the easiest instances appear to be the small ones, because the better quantiles are nearly full. Scatter plots show combinations of two parameters, with every instance shown as one point in the plot. We can also see that increasing the density without changing the number of robots makes the instances more difficult. Moreover, instances without obstacles seem to be easier to solve. Due to lack of data, not much can be said for clusters, with difficulty mostly correlating with the number of robots in the available data points.

These top 3 finishers were invited for contributions in the 2021 SoCG proceedings, as follows:

(1) Team Shadoks: Loïc Crombez, Guilherme D. da Fonseca, Yan Gerard, Aldo Gonzalez-Lorenzo, Pascal Lafourcade, Luc Libralesso [4]
(2) Team UNIST: Hyeyun Yang, Antoine Vigneron [27]
(3) Team gitastrophe: Jack Spalding-Jamieson, Paul Liu, Brandon Zhang, Da Wei Zheng [15]

Fig. 8. A detailed analysis for MAX instead of SUM; refer to Figure 7 for the breakdown into subfigures.

Table 2. The Top of the Final Leaderboard for SUM

Position	Team	Score	# Best Solutions
1	gitastrophe^J	198.4943	57
2	UNIST	191.7893	120
3	Shadoks	180.4952	0
4	cgi@tau	175.0754	6
5	BlueTeamTechnion^J	165.5633	34
6	TUeSWarM^J	146.2244	1
7	S10ppy J035^J	133.450	1
8	Team ITI^J	109.4631	0
9	Kleinkariert^J	81.7086	0
10	École Polytechnique^J	75.4734	0

"Best solutions" are the best found by any participating team, which does not exclude the possibility of better solutions, but provides a score of 1 for that instance; scores are truncated to four decimal places. Junior teams are indicated by ^{J}.

All three teams engineered their solutions based on a spectrum of heuristics for generating start configurations, in combinations of a variety of local search methods, including simulated annealing, k-optimization, and more refined, tailor-made approaches. Details of their methods and the engineering decisions they made are given in their respective papers.

4 CONCLUSIONS

The 2021 CG:SHOP Challenge motivated a considerable number of teams to engage in intensive optimization studies. The outcomes promise further insight into the underlying, important optimization problem. Moreover, the considerable participation of junior teams indicates that the Challenge itself motivates a considerable number students and young researchers to work on practical algorithmic problems.

REFERENCES

[1] Aviv Adler, Mark de Berg, Dan Halperin, and Kiril Solovey. 2015. Efficient multi-robot motion planning for unlabeled discs in simple polygons. *IEEE Transactions on Automation Science and Engineering* 12, 4 (2015), 1309–1317.

[2] Aaron T. Becker, Sándor P. Fekete, Phillip Keldenich, Matthias Konitzny, Lillian Lin, and Christian Scheffer. 2018. Coordinated motion planning: The video. In *Symposium on Computational Geometry (SoCG'18)*. 74:1–74:6. https://doi.org/10.4230/LIPIcs.SoCG.2018.74. Video at https://www.ibr.cs.tu-bs.de/users/fekete/Videos/CoordinatedMotionPlanning.mp4.

[3] Soon-Jo Chung, Aditya Avinash Paranjape, Philip Dames, Shaojie Shen, and Vijay Kumar. 2018. A survey on aerial swarm robotics. *IEEE Transactions on Robotics* 34, 4 (August 2018), 837–855.

[4] Loïc Crombez, Guilherme Dias da Fonseca, Yan Gerard, Aldo Gonzalez-Lorenzo, Pascal Lafourcade, and Luc Libralesso. 2021. Shadoks approach to low-makespan coordinated motion planning. In *Symposium on Computational Geometry (SoCG'21)*, Vol. 189. 63:1–63:9. https://doi.org/10.4230/LIPIcs.SoCG.2021.63

[5] Daniel Delahaye, Stéphane Puechmorel, Panagiotis Tsiotras, and Eric Féron. 2014. Mathematical models for aircraft trajectory design: A survey. In *Air Traffic Management and Systems*. Springer, 205–247.

[6] Erik D. Demaine, Sándor P. Fekete, Phillip Keldenich, Henk Meijer, and Christian Scheffer. 2018. Coordinated motion planning: Reconfiguring a swarm of labeled robots with bounded stretch. In *Proceedings of the 34th International Symposium on Computational Geometry (SoCG'18)*. 29:1–29:17. Full version to appear in *SIAM Journal on Computing*.

[7] Erik D. Demaine, Sándor P. Fekete, Phillip Keldenich, Henk Meijer, and Christian Schefffer. 2018. Coordinated motion planning: Reconfiguring a swarm of labeled robots with bounded stretch. *Computing Research Repository (CoRR'18)* 1801 (2018), 1–32. Available at https://arxiv.org/abs/1801.01689.

[8] Zahra Derakhshandeh, Robert Gmyr, Andréa W. Richa, Christian Scheideler, and Thim Strothmann. 2015. An algorithmic framework for shape formation problems in self-organizing particle systems. In *Nanoscale Computing and Communication (NANOCOM'15)*. 21.

[9] Zahra Derakhshandeh, Robert Gmyr, Andréa W. Richa, Christian Scheideler, and Thim Strothmann. 2016. Universal shape formation for programmable matter. In *Symposium on Parallelism in Algorithms and Architectures (SPAA'16)*. 289–299.

[10] Sándor P. Fekete, Björn Hendriks, Christopher Tessars, Axel Wegener, Horst Hellbrück, Stefan Fischer, and Sebastian Ebers. 2011. Methods for improving the flow of traffic. In *Organic Computing — A Paradigm Shift for Complex Systems*, Christian Müller-Schloer, Hartmut Schmeck, and Theo Ungerer (Eds.). Autonomic Systems, Vol. 1. Birkhäuser.

[11] Seth Copen Goldstein and Todd Mowry. 2004. Claytronics: A scalable basis for future robots. *Robosphere* November (2004).

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

[13] J. E. Hopcroft and G. T. Wilfong. 1986. Reducing multiple object motion planning to graph searching. *SIAM Journal on Computing* 15, 3 (1986), 768–785.

[14] S. Kloder and S. Hutchinson. 2006. Path planning for permutation-invariant multi-robot formations. *IEEE Transactions on Robotics and Automation* 22, 4 (2006), 650–665.

[15] Paul Liu, Jack Spalding-Jamieson, Brandon Zhang, and Da Wei Zheng. 2021. Coordinated motion planning through randomized k-opt. In *Symposium on Computational Geometry (SoCG'21)*, Vol. 189. 64:1–64:8. https://doi.org/10.4230/LIPIcs.SoCG.2021.64

[16] A. Naz, B. Piranda, J. Bourgeois, and S. C. Goldstein. 2016. A distributed self-reconfiguration algorithm for cylindrical lattice-based modular robots. In *Network Computing and Applications (NCA'16)*. 254–263. https://doi.org/10.1109/NCA.2016.7778628

[17] Michael Rubenstein, Alejandro Cornejo, and Radhika Nagpal. 2014. Programmable self-assembly in a thousand-robot swarm. *Science* 345, 6198 (2014), 795–799.

[18] Erol Šahin and Alan Winfield (eds.). 2008. Special issue on swarm robotics. *Swarm Intelligence* 2, 2–4 (December 2008), 69–72.

[19] Michael Schreckenberg and Reinhard Selten (eds.). 2004. *Human Behaviour and Traffic Networks*. Springer.

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

[21] K. Solovey and D. Halperin. 2014. k-Color multi-robot motion planning. *International Journal of Robotics Research* 33, 1 (2014), 82–97.

[22] Kiril Solovey and Dan Halperin. 2016. On the hardness of unlabeled multi-robot motion planning. *International Journal of Robotics Research* 35, 14 (2016), 1750–1759.

[23] K. Solovey, J. Yu, O. Zamir, and D. Halperin. 2015. Motion planning for unlabeled discs with optimality guarantees. In *Proceedings of the 11th Conference Robotics: Science and Systems (RSS'15)*.

[24] Pierre Thalamy, Benoît Piranda, and Julien Bourgeois. 2019. Distributed self-reconfiguration using a deterministic autonomous scaffolding structure. In *Autonomous Agents and MultiAgent Systems (AAMAS'19)*. 140–148.

[25] M. Turpin, N. Michael, and V. Kumar. 2013. Trajectory planning and assignment in multirobot systems. In *Algorithmic Foundations of Robotics X*. Springer, 175–190.

[26] M. Turpin, K. Mohta, N. Michael, and V. Kumar. 2014. Goal assignment and trajectory planning for large teams of interchangeable robots. *Autonomous Robots* 37, 4 (December 2014), 401–415.

[27] Hyeyun Yang and Antoine Vigneron. 2021. A simulated annealing approach to coordinated motion planning (CG challenge). In *Symposium on Computational Geometry (SoCG'21)*, Vol. 189. 65:1–65:9. https://doi.org/10.4230/LIPIcs.SoCG.2021.65

Received April 2022; accepted April 2022