Assignment Algorithms for Variable Robot Formations

Srinivas Akella

Department of Computer Science
University of North Carolina at Charlotte
Charlotte, NC 28223, USA

Abstract. This paper describes algorithms to perform optimal assignment of teams of robots translating in the plane from an initial formation to a variable goal formation. We consider the case when each robot is to be assigned a goal position, the individual robots are interchangeable, and the goal formation can be scaled or translated. We compute the costs for all candidate pairs of initial, goal robot assignments as functions of the parameters of the goal formation, and partition the parameter space into equivalence classes invariant to the cost order using computational geometry techniques. We compute a minimum completion time assignment for an equivalence class by formulating it as a linear bottleneck assignment problem (LBAP). To improve efficiency, we solve the LBAP problem for each equivalence class by incrementally updating the solution as the formation parameters are varied. This work is motivated by applications that include the motion of droplet formations in digital microfluidic lab-on-a-chip devices, and of robot and drone formations in the plane.

Keywords: Robot formations, linear bottleneck assignment, multiple robots

1 Introduction

The assignment and motion of teams of robots from one formation to another is a problem that arises in multiple applications ranging from robot and drone formation planning ([13, 30, 31, 34]) to the motion of droplet formations in digital microfluidic lab-on-a-chip devices ([19]). Most previous work on multiple robot assignment has either assumed fixed initial and goal formations when computing robot assignments, or has assumed a fixed assignment while varying the scale, location, or orientation of the goal formations.

In this paper, we introduce the problem of performing robot assignment while simultaneously considering variable goal formations that can be either scaled or translated. Since energy expenditure and completion time are important for many tasks, we wish to miminize the maximum distance traveled or the maximum time taken by any of the robots. We introduce the scaled goal formation problem and the translated goal formation problem, where the goal

K. Goldberg et al. (Eds.): *Algorithmic Foundations of Robotics XII*, SPAR 13, pp. 912–927, 2020.
https://doi.org/10.1007/978-3-030-43089-4_58

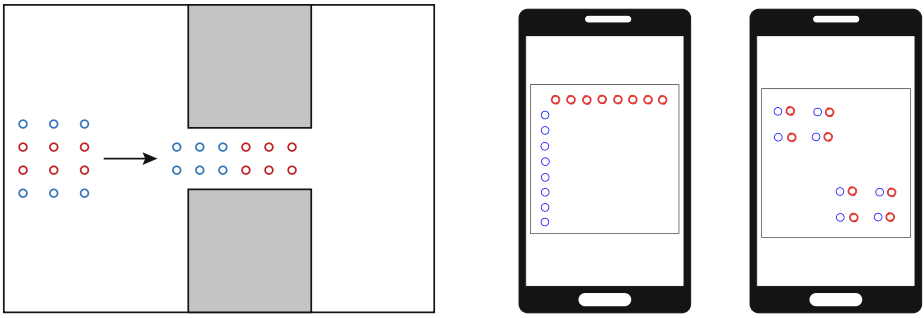


Fig. 1. Example motivating applications. (a) A robot formation on the left that has to change its shape to pass through the passage between two obstacles. Each circle represents a robot. (b) Schematic figure showing overhead view of an LADM chip (indicated by a square) on a smartphone, with circles indicating chemical droplets. (Left) Initial formation of droplets. (Right) Goal formation of droplets for mixing. LADM chip and droplets are shown enlarged and chemicals are color coded for clarity.

formation may be scaled or the goal formation may be translated in a region respectively. Given an initial formation of the robots and a specified shape formation, we wish to determine the assignment of each robot in the initial formation to a configuration in the variable goal formation that minimizes the maximum distance traveled by the robots. We formulate these assignment problems with variable goal formations as linear bottleneck assignment problems (LBAP) with geometric constraints where the size or location of the goal formation, which influences the LBAP costs, must be computed. Our approach exploits the relative cost ordering property of LBAPs and the geometric structure captured by arrangements of cost curves and surfaces. We thus partition the goal formation parameter space into equivalence classes invariant to the cost order. This enables a comprehensive evaluation to identify the globally optimum solution.

Motivating applications: The multi-robot assignment problem for variable goal formations is motivated by applications including the following examples.

1. Multiple-robot assignment: The assignment and planning of teams of robots, drones, and spacecraft often requires motion between formations. Example applications include search and rescue operations, package delivery, construction, surveillance, and sensor network monitoring [30, 31, 33, 9, 3]. These tasks can be subject to spatial constraints (e.g., the team of robots must squeeze through a passage while maintaining a minimum separation distance, as in Figure 1(a)) or communication range constraints (e.g., a team of UAVs must stay within a maximum distance of team members), which can be represented by the allowable scaling and/or translation of the goal formation.
2. Digital microfluidic lab-on-chip systems: Low-cost, portable lab-on-a-chip systems can impact a wide variety of applications including point-of-care

medical diagnostics. Recent hardware advances are enabling lab-on-a-chip devices that can be optically actuated using smartphone and tablet LCD screens (e.g., [23], [22], [25]). In such *light-actuated digital microfluidic* (LADM) chips, discrete droplets of chemicals are optically actuated using moving patterns of projected light to perform chemical reactions by repeatedly moving and mixing droplets (Figure 1(b)). A key issue is the automated planning and coordination of droplet formations on the LADM chip, which we address by modeling the droplets as robots. An important step is computing goal formations that can fit within a specified region on the chip, and identifying assignments of droplets that can efficiently travel to the goal formation.

2 Background: Assignment Problems

Assignment problems deal with how to assign n items (robots, tasks) to n other items (locations, resources). Consider a problem where there are n tasks, indexed by i, that must be assigned to n resources, indexed by j. The cost of performing task i on resource j is c_{ij}. The cost matrix C consists of entries c_{ij} corresponding to the cost of assigning task i to resource j. x_{ij} is a binary variable that is 1 if task i is assigned to resource j, and 0 otherwise.

In the linear sum assignment problem (LSAP) [7], which is the standard form of the assignment problem, the objective function to be minimized is the sum of the assigned costs $\sum_{1\leq i,j\leq n} c_{ij}x_{ij}$. LSAP is the dominant form of assignment that has been considered in the robotics literature (for example, [30, 17]), and can be solved using the Hungarian algorithm [21, 7].

2.1 Linear Bottleneck Assignment Problem

Since we want to minimize the maximum completion time or the maximum distance for any robot, we consider the linear bottleneck assignment problem [7]. The linear bottleneck assignment problem (LBAP) formulation is:

$\min_{1\leq i,j\leq n} \max c_{ij}x_{ij}$
$\sum_{j=1}^{n} x_{ij} = 1 \ \ \forall i = 1, \ldots, n$
$\sum_{i=1}^{n} x_{ij} = 1 \ \ \forall j = 1, \ldots, n$
$x_{ij} \in \{0, 1\} \ \ \forall i, j = 1, \ldots, n$

LBAPs occur in connection with assigning jobs to parallel machines to minimize the latest completion time. The LBAP minimizes the maximum cost of completing any task, whereas the LSAP minimizes the total cost of completing all the tasks. When the objective function is the time taken or distance traveled, the LBAP minimizes the maximum completion time or maximum distance traveled by a robot. We use the *threshold algorithm* [7], outlined in Section 4, for solving LBAPs.

3 Finding Optimal Assignments and Scaled Goal Formations: An LBAP Formulation

We consider the problem of simultaneously computing both the optimal robot assignment and scaled goal formation while minimizing the maximum robot travel distance. The *Scaled Goal Formation problem* is: Given n robots in an initial formation $P = \{p_i\}$ where $p_i \in \mathbb{R}^2$ is the initial configuration of robot i, and a specified shape formation $S = \{s_j\}$, where $s_j \in \mathbb{R}^2$, assign the robots to configurations in the goal formation $Q = \{q_j\}$, where $q_j \in \mathbb{R}^2$ is the jth scaled goal configuration, such that the maximum distance traveled by any robot is minimized and Q is equivalent up to a scale factor $\alpha \in \mathbb{R}_+$ of S. Here $q_j = \alpha s_j + d_0$, where $d_0 \in \mathbb{R}^2$ is a user specified translation. We assume that only scale is varied and that all robots move in straight lines to their goals with an equal and constant speed.

We formulate this as a linear bottleneck assignment problem (LBAP) [7]. For a given scale value α, the cost c_{ij} of assigning robot i in the initial formation to a location j in the goal formation is the distance from p_i to q_j, yielding cost matrix $C = [c_{ij}]$. x_{ij} is a binary variable that is 1 if robot i in the initial formation is assigned to robot j in the goal formation, and 0 otherwise.

The *scaled goal formation LBAP formulation* is:

$$\min \max c_{ij} x_{ij}$$
$$\sum_{j=1}^{n} x_{ij} = 1 \quad \forall i = 1, \ldots, n$$
$$\sum_{i=1}^{n} x_{ij} = 1 \quad \forall j = 1, \ldots, n$$
$$x_{ij} \in \{0, 1\} \quad \forall i, j = 1, \ldots, n$$

where cost c_{ij} is a function of scale α.

$c_{ij} = ||p_i - q_j||_2 = [a_i + b_{ij}\alpha + e_j\alpha^2]^{1/2}$ where
$a_i = (p_{ix} - d_{0x})^2 + (p_{iy} - d_{0y})^2$, $b_{ij} = -2((p_{ix} - d_{0x})s_{jx} + (p_{iy} - d_{0y})s_{jy})$,
and $e_j = s_{jx}^2 + s_{jy}^2$.

We must find the value of α that minimizes the objective function. An interesting property of LBAP is that the optimal assignment depends only on the relative order of the costs c_{ij} and not on their actual values (Lemma 6.1 [7]). The key idea is to exploit this property and identify intervals of α over which the relative order is invariant. Over each such interval, the value of the optimal LBAP assignment is determined by the cost of one particular initial and goal position combination. We call this the *critical cost*. The value of this cost varies as α is varied over the interval, but the cost element that determines the value of the optimal assignment does not change.

The cost c_{ij} of each assignment is a function of α. So we have a set of curves c_{ij} and the relative order of the costs changes only at the intersection points of the curves. See Figure 2. If we can compute all the intersection points as α is varied, for each interval we can compute the LBAP assignment at a specified value (say at the midpoint of the interval α_{mid}) and the associated critical cost.

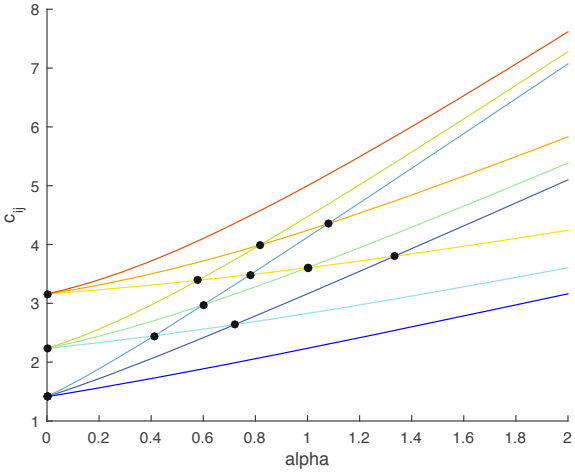
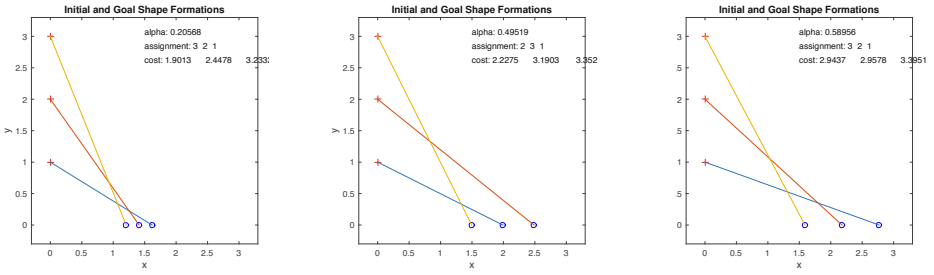


Fig. 2. Cost curves, showing cost c_{ij} as a function of scale factor α, with curve intersections indicated by dark points. There is an invariant cost order between every pair of consecutive intersections.

By minimizing the critical cost in the interval, we find the α value in the interval that gives the optimal assignment and cost for the interval. We compare the optimum costs over all intervals to find the globally optimal assignment and cost. See the example in Figure 3.

Algorithm for scaled goal formation LBAP:

1. From the given initial formation P and specified shape formation S, and range of permitted α values, compute the c_{ij} cost curves.
2. Compute all intersections of the cost curves by a line sweep algorithm. For each interval, compute the ordering of the costs using α_{mid}, the alpha value at the midpoint of the interval. See Figure 2.
3. Compute the cost matrix at α_{mid} in the first α interval, and compute an LBAP solution for this cost matrix.
4. Compute the optimal LBAP solution for each α interval using the optimal LBAP solution from the previous interval (to avoid solving the problem from scratch). Using the known optimal assignment for the previous cost ordering, the new optimal assignment when two consecutive elements of the ordering swap places can be efficiently computed (as will be described in Section 4).
5. From the computed LBAP solution, identify the critical cost c_{ij}, the c_{ij} that determines the objective function value, and optimal assignment for the interval. Then compute the minimum value of the critical cost c_{ij} over the interval to obtain the optimal objective value and corresponding optimal alpha value α_{opt}.
6. Compare the LBAP solutions for all intervals at their respective α_{opt}, and select the minimum among them as the global optimal assignment and opti-



(a) (b) (c)

Fig. 3. A three robot example using the cost curves of Figure 2. (a) The optimal cost assignment, at $\alpha = 0.2057$. (c) An example assignment for $\alpha = 0.49519$. (d) A different assignment for $\alpha = 0.58956$. The start positions are indicated by plus symbols, and the goal positions are indicated by circles.

mal scale factor. (This global optimal solution can be updated as the optimal assignment for each interval is computed.)

There are $O(n^2)$ costs c_{ij}. Finding all pairwise intersections of c_{ij} curves by the sweep line algorithm will yield $O(n^4)$ intersections in $O(n^4 \log n)$ time [11]. There are $O(n^4)$ intervals, and explicitly writing out the cost order for an interval takes $O(n^2)$ time. As we will discuss in Section 4, solving a new instance of the LBAP problem takes $O(n^{2.5}/\sqrt{\log n})$ time [7] while using a solution to a closely related LBAP takes $O(n^2)$ time. So the overall running time is $O(n^6)$.

4 Incremental Updates to Compute the Optimal LBAP

We now present an approach to update the optimal solution to an LBAP from the optimal solution of a closely related LBAP. We use the *threshold algorithm* [7] for solving an LBAP instance. The threshold algorithm alternates between two phases. In the first phase, a cost element c^*, the threshold value, is chosen and a threshold matrix $\overline{C}$ is defined by $\overline{c}_{ij} = 1$ if $c_{ij} > c^*$ and 0 otherwise. In the second phase, we check whether there exists an assignment with total cost 0 for the cost matrix $\overline{C}$. For this, we construct a bipartite graph $\overline{G} = (U, V; E)$ with $|U| = |V| = n$ and edges $[i, j] \in E$ if and only if $\overline{c}_{ij} = 0$. In other words, we check whether the bipartite graph with threshold matrix $\overline{C}$ contains a perfect matching or not. A perfect matching exists for a bipartite graph G with n vertices if there exists a set of n edges in G such that each vertex is incident on exactly one edge. Let the *threshold graph* $\overline{G}(c)$ be the bipartite graph such that an edge $e_{ij} = [i, j]$ exists if and only if $c_{ij} \leq c$; the edge has weight 0. The edge e_{ij} has a cost c_{ij} associated with it in the cost matrix C. The smallest value c^* for which the

threshold graph $\overline{G}(c^*)$ contains a perfect matching is the optimal value of the LBAP; this c^* is the *critical cost*. For the case of dense graphs, there is a version of the threshold algorithm with a total time complexity of $O(n^{2.5}/\sqrt{\log n})$ [7].

To explain and analyze our incremental LBAP update algorithm, we use the LBAP property that the optimal solution of an LBAP depends only on the relative order of the cost coefficients and not on their numerical value [7]. This enables us to simplify the cost structure of the cost matrix so it consists of consecutive integers starting at 0. In the remainder of this section, c_{ij} will be a representative integer cost (and not the cost from the actual matrix C). Now the questions are: If two entries in the cost matrix swap their cost values, is there an efficient way of telling whether the optimal assignment has changed? If the optimal assignment has changed, is there an efficient way of computing it from the previously computed optimal assignment?

When two entries in the cost matrix swap their cost values due to the intersection of their cost curves, they are represented by consecutive integers (we assume here that the costs are all distinct). Let the two edges that are swapping costs be e_{ij} and e_{kl} with costs c_{ij} and c_{kl} respectively before the swap (where c_{ij} and c_{kl} are consecutive integers). Then edge e_{ij} will have cost c_{kl} and e_{kl} will have cost c_{ij} after the swap. Let the optimum objective value of the LBAP prior to the swap be c_{opt}. Therefore there is a perfect matching in the threshold graph $\overline{G}(c_{opt})$. Using the threshold algorithm for solving the LBAP after the swap, the four possible cases that can occur are:

1. If $c_{opt} > \max(c_{ij}, c_{kl})$, then both edges e_{ij} and e_{kl} are present in the threshold graph $\overline{G}(c_{opt})$ before the swap. The threshold graph after the swap $\overline{G}_{swap}(c_{opt})$ will be unchanged because the only edges affected are e_{ij} and e_{kl}, and they will both be present. Therefore the optimum value and the perfect matching will be unchanged after the swap.
2. If $c_{opt} < \min(c_{ij}, c_{kl})$, then both edges e_{ij} and e_{kl} are not present in the threshold graph $\overline{G}(c_{opt})$ before the swap. The threshold graph after the swap $\overline{G}_{swap}(c_{opt})$ will be unchanged because the only edges affected are e_{ij} and e_{kl}, and they will both still be absent. Therefore the optimum value and the perfect matching will be unchanged after the swap.
3. If $c_{opt} = \max(c_{ij}, c_{kl})$, assume without loss of generality that $c_{opt} = c_{ij}$, that is, $c_{ij} > c_{kl}$ before the swap. Therefore there is a perfect matching in the threshold graph $\overline{G}(c_{ij})$. We have two cases to consider since the LBAP optimum after the swap must be greater than or equal to c_{kl}[1].
   (a) If we consider the threshold graph $\overline{G}_{swap}(c_{ij})$ after the swap of c_{ij} and c_{kl}, it will be identical to $\overline{G}(c_{ij})$ before the swap. Therefore a perfect matching is present, and the objective function value c_{ij} and the matching will be unchanged after the swap.

[1] Proof that optimum value of the LBAP after the swap, c_{opt}^{swap}, cannot be less than c_{kl}: Assume the LBAP after the swap has an optimum value $c_{opt}^{swap} = c_* < c_{kl} < c_{ij}$. This implies that $\overline{G}_{swap}(c_*)$ has a perfect matching without edges e_{ij} and e_{kl}. Since $\overline{G}_{swap}(c_*) = \overline{G}(c_*)$, this implies the optimum LBAP value before the swap is c_*, which leads to a contradiction.

(b) The optimum value after the swap may decrease at most to c_{kl}. So we need to additionally evaluate $\overline{G}_{swap}(c_{kl})$, which is a subgraph of $\overline{G}(c_{ij})$. Here edge e_{ij} is present, but edge e_{kl} is not present. A new matching must be computed, and if the graph $\overline{G}_{swap}(c_{kl})$ contains a perfect matching, the optimum objective will be c_{kl} $(= c_{ij} - 1)$. We can avoid recomputing the matching from scratch in this graph $\overline{G}_{swap}(c_{kl})$ by using the matching in $\overline{G}(c_{ij})$ before the swap to efficiently compute the new matching. To do this, we use the standard reduction of the maximum cardinality matching problem to the maximum flow problem [21], and use a procedure linear in the size of the graph to compute the new maximum flow for $\overline{G}_{swap}(c_{kl})$ with one edge (e_{kl}) whose capacity has decreased to 0 from 1 in $\overline{G}(c_{ij})$, for which we have the maximum flow from its perfect matching. If this corresponds to a perfect matching, it represents the optimum value of the LBAP.

4. If $c_{opt} = \min(c_{ij}, c_{kl})$, assume without loss of generality that $c_{opt} = c_{kl}$, that is, $c_{kl} < c_{ij}$ before the swap. Therefore there is a perfect matching in the threshold graph $\overline{G}(c_{kl})$. We have two cases to consider.
   (a) If we consider the threshold graph $\overline{G}_{swap}(c_{ij})$ after the swap, it will include the edge e_{ij} in addition to all edges in $\overline{G}(c_{kl})$ before the swap. Therefore a perfect matching is present, with an optimum value of c_{ij}.
   (b) We need to additionally evaluate $\overline{G}_{swap}(c_{kl})$. Here edge e_{ij} is present, but edge e_{kl} is not present. A new matching must be computed, and if this graph contains a perfect matching, the optimum objective will be c_{kl}. We can use the matching before the swap to efficiently compute the new matching. We use the reduction from the maximum cardinality matching problem to the maximum flow problem, and use a procedure linear in the size of the graph to compute the new maximum flow. To do this, we use the perfect matching from $\overline{G}(c_{kl})$ in $\overline{G}_{swap}(c_{ij})$, and then reduce the capacity of edge e_{kl} from 1 to 0 in the flow problem corresponding to $\overline{G}_{swap}(c_{kl})$. If this corresponds to a perfect matching, it represents the optimum value of the LBAP.

The threshold graph $\overline{G}$ has $2n$ vertices and up to n^2 edges, where n is the number of robots. Computing the new maximum flow takes time linear in the size of this graph, and so takes $O(n^2)$ time. Therefore computing the optimal LBAP by an incremental update takes $O(n^2)$ time.

5 Finding Optimal Translated Goal Formations

Selecting an optimal location for the goal formation Q can minimize the maximum distance traveled by any robot (or maximum completion time). We therefore introduce the *Translated Goal Formation problem*: Given an initial formation P with n robots and a specified shape formation S, assign the robots to configurations in the goal formation Q such that the maximum distance traveled by any robot is minimized and Q is equivalent to S up to translation d, which is to be

computed. As before, $P = \{p_i\}$, $S = \{s_j\}$, $Q = \{q_j\}$, and $p_i, s_j, q_j, d \in \mathbb{R}^2$. Here we assume that the scale is fixed, and that the origin of the shape formation can be translated by $d = (d_x, d_y)$ in the plane. So $q_j = s_j + d$, where d is measured with respect to the same reference frame as P.

We formulate this problem as an LBAP. We now need to compute the relative order of the costs c_{ij} as functions of two variables: d_x, d_y. Our goal is to partition the $d_x d_y$-plane into cells with invariant cost orderings. We can then apply the algorithm of Section 3, appropriately modified, to identify the optimal (d_x, d_y) value.

The *translated goal formation LBAP formulation* is:

min max $c_{ij}x_{ij}$
$\sum_{j=1}^{n} x_{ij} = 1 \ \ \forall i = 1, \ldots, n$
$\sum_{i=1}^{n} x_{ij} = 1 \ \ \forall j = 1, \ldots, n$
$x_{ij} \in \{0, 1\} \ \ \forall i, j = 1, \ldots, n$
where cost c_{ij} can be written as a function of translation (d_x, d_y).
$c_{ij} = ||p_i - q_j||_2 = [(p_{ix} - (s_{jx} + d_x))^2 + (p_{iy} - (s_{jy} + d_y))^2]^{1/2}$
$= [(d_x - (p_{ix} - s_{jx}))^2 + (d_y - (p_{iy} - s_{jy}))^2]^{1/2}$

Each $c_{ij}(d_x, d_y)$ describes a surface, and the obvious approach is to determine the intersections of all the surfaces for all pairs i, j, and use this to determine the d_x, d_y regions over which the cost ordering is invariant. We instead characterize the d_x, d_y regions with invariant cost order using a different approach. It will be helpful to first introduce new variables $r_{ij} = p_i - s_j$ and rewrite the cost equation as: $c_{ij} = [(d_x - r_{ijx})^2 + (d_y - r_{ijy})^2]^{1/2}$.

Each c_{ij} can now be viewed as describing the distance of the point (d_x, d_y) from a fixed site r_{ij}. We have a set $R = \{r_{ij}\}$ of n^2 sites in the plane, and we use the reformulated costs to compute the regions of $d_x d_y$-space that are invariant in their cost ordering. For this, we use some beautiful connections between Voronoi diagrams, unit paraboloids, and arrangements from computational geometry [6]. Consider the unit paraboloid $z = d_x^2 + d_y^2$. For each site r_{ij}, lift it by projecting it vertically upwards to the paraboloid, to the point $(r_{ijx}, r_{ijy}, r_{ijz})$ where $r_{ijz} = r_{ijx}^2 + r_{ijy}^2$. Next construct the plane H_{ij} tangent to the paraboloid at the point $(r_{ijx}, r_{ijy}, r_{ijz})$. The equation of this plane is $z = 2r_{ijx}d_x + 2r_{ijy}d_y - (r_{ijx}^2 + r_{ijy}^2)$. Given any point (d_x, d_y), the upward vertical ray from it intersects the set of planes $H = \{H_{ij}\}$. The order in which the ray intersects these planes corresponds to their decreasing cost order, where the cost associated with a plane H_{ij} is the distance of its site r_{ij} to the point d. The highest plane identifies the site closest to the point[2], and the lowest plane identifies the site farthest from the point,

[2] To understand the connection between the height ordering of the planes and the point's distances to the sites, consider two planes H_{ij} and H_{kl}. Let the equation of plane H_{ij} be $z_{ij} = 2r_{ijx}d_x + 2r_{ijy}d_y - (r_{ijx}^2 + r_{ijy}^2)$ and the equation of plane H_{kl} be $z_{kl} = 2r_{klx}d_x + 2r_{kly}d_y - (r_{klx}^2 + r_{kly}^2)$. When plane H_{ij} is higher than plane H_{kl}, $z_{ij} > z_{kl}$, which implies that $2r_{ijx}d_x + 2r_{ijy}d_y - (r_{ijx}^2 + r_{ijy}^2) > 2r_{klx}d_x +$

and the planes from highest to lowest are ordered in increasing order of distance of their sites to the point (d_x, d_y).

An *arrangement* $A(H)$ induced by the set of planes H is the convex subdivision of space defined by the set of planes H [6]. A *cell* lies at *depth* i if there are exactly i planes above the cell. *Level* i refers to the boundary of the union of cells at depths zero, one, up to $i-1$. Here a level is a piecewise-linear surface formed by pieces of the planes H_{ij}.

The arrangement $A(H)$ of the planes H provides useful geometric structure. The projection of the upper envelope of these planes (i.e., level 1 of $A(H)$) onto the $d_x d_y$-plane gives the Voronoi diagram of the set of sites R [6]. This is in fact the order-1 Voronoi diagram, where each cell contains the points closest to one site. More generally, we can partition the space according to the k closest sites of N sites, for some $1 \leq k \leq N-1$. The resulting diagrams are called higher-order Voronoi diagrams, and for a given k, the diagram is called the order-k Voronoi diagram. An order-2 Voronoi diagram is one where each cell contains the points closest to an unordered pair of sites. This order-2 Voronoi diagram can be computed by projecting level 2 of $A(H)$. We can similarly compute the order-3 through order-$(N-1)$ Voronoi diagrams. Conceptually, the overlay of these order-1 through order-$(N-1)$ Voronoi diagrams on the $d_x d_y$-plane partitions the plane into convex cells, and each cell has an invariant ordering of sites based on the distances of points in the cell to the sites. In other words, within each cell, the cost order is invariant.

There are $N = n^2$ sites in R, which implies $O(n^2)$ planes that intersect at $O(n^4)$ intersection curves. When projected onto the $d_x d_y$-plane, we obtain a planar arrangement with at most $O(n^8)$ faces, edges, and vertices [11, 10]. This arrangement can be computed in $O(n^8)$ time, and the cost order for each cell can be enumerated in $O(n^2)$ time.

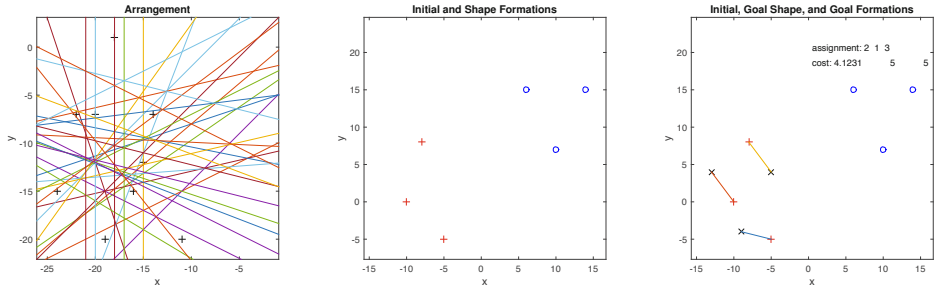
At each cell of the planar arrangement, we compute an LBAP at an interior point to identify the critical cost, say c_{ij}. Since each cell is a convex polygon, we can compute the optimal d_x, d_y values that minimize the LBAP by formulating a convex quadratic program (QP) that minimizes c_{ij}^2 given the linear boundary constraints of the cell. The QP formulation is:

$$\begin{aligned}
&\text{minimize } c_{ij}^2 = (d_x - r_{ijx})^2 + (d_y - r_{ijy})^2 \text{ subject to} \\
&2(r_{vwx} - r_{tux})d_x + 2(r_{vwy} - r_{tuy})d_y + (r_{tux}^2 + r_{tuy}^2) - (r_{vwx}^2 + r_{vwy}^2) \leq 0 \quad \forall r_{tu}, r_{vw} \\
&l_x \leq d_x \leq u_x \\
&l_y \leq d_y \leq u_y
\end{aligned}$$

where sites $r_{tu}, r_{vw} \in R$ correspond to consecutive distances in the cost order in the cell, and l_x, l_y and u_x, u_y are lower and upper bounds for d_x and d_y.

Since this is a convex QP in just two variables, it can be solved in time that is linear in the number of edges of the cell. Each cell has an invariant cost ordering

$2r_{kly}d_y - (r_{klx}^2 + r_{kly}^2)$. Adding $d_x^2 + d_y^2$ to both sides and rearranging terms shows that $(d_x - r_{klx})^2 + (d_x - r_{kly})^2 > (d_x - r_{ijx})^2 + (d_x - r_{ijy})^2$, thus establishing that point (d_x, d_y) is closer to site r_{ij} than site r_{kl}.



(a) (b) (c)

Fig. 4. A translation LBAP example for 3 robots. (a) Arrangement that results from overlay of order-1 through order-$(N-1)$ Voronoi diagrams, for $N = 9$ sites indicated by black plus symbols. (b) Initial formation and specified shape formation. (c) Initial formation, shape formation, and optimal goal formation, with the lines indicating the optimal assignment. The initial positions are indicated by red plus symbols, the shape positions are indicated by blue circles, and the goal positions are indicated by black x symbols.

of N sites, with each edge corresponding to a bisector of two sites that defines the relative ordering of a successive pair of distance costs. At each cell edge, we can interchange only consecutive cost elements. Since there is a maximum of $N-1$ swaps between consecutive elements, this implies a maximum of $N-1$ edges for each cell. A cell has less than $N-1$ edges if it is completely on one side of a bisector corresponding to two of its consecutive distances (i.e., it is completely contained in the corresponding halfplane). Therefore the QP can be solved in $O(N)$ time, that is, $O(n^2)$ time.

We find the globally optimal solution by comparing the optimal LBAP solutions over all the cells in the $d_x d_y$-plane. See Figure 4 for an example. The overall running time of the algorithm is $O(n^{10})$, since there are $O(n^8)$ cells with $O(n^2)$ computation at each cell.

6 Related Work

Multi-robot Assignment and Path Planning: Coordinating multiple robots in a shared workspace has attracted much attention in the robotics community [16], [26], [2], [33]. A typical goal of coordination is to achieve collision-free and time-optimal robot motions. There has been extensive research on motion planning for multiple robots; see [14, 15, 8] for overviews and [16, 26, 24, 33] for example approaches related to our work. There have been many exciting recent

developments in multiple robot motion planning and formation planning, some of which are summarized below.

Derenick and Spletzer[9] coordinated a large-scale robot team to change the shape of formation, for a fixed robot assignment, by modifying scale, translation, and rotation through second-order cone programming techniques. The solution is explored by minimizing the total distance or minimizing the maximum distance robots travel while geometric shape constraints are satisfied.

Kloder and Hutchinson [13] developed a representation for collision-free path planning of multiple unlabeled robots translating in the plane from one formation to another formation. They represent a formation by the coefficients of a complex polynomial whose roots represent the robot configurations.

The CAPT algorithm [30] developed by Turpin et al. performs concurrent assignment and planning of trajectories for unlabeled robots when there are N robots and M goals, and each goal is visited by one robot. They used the Hungarian assignment algorithm to minimize the sum of individual robot-goal costs. They developed synchronized trajectories that are collision-free when the start and goal locations are at least $2\sqrt{2}R$ from each other. They also present an online decentralized version of the algorithm.

Yu and LaValle [34, 35] demonstrated that multi-robot path planning on unit distance graphs can be modelled as multi-commodity dynamic network flow problems. They provide fast and complete algorithms to efficiently solve permutation-invariant versions of these problems. They also studied time and distance optimality of the feasible solutions. They [35] use integer linear program formulations to solve time and distance problems. Katsev, Yu, and LaValle [12] perform path planning for large robot formations of indistinguishable robots using a hierarchical approach. The algorithm provides paths with total distance within a constant multiple of the optimal total distance.

Solovey and Halperin [27] developed a sampling-based algorithm for the k-color multi-robot motion planning problem, where the robots are partitioned into groups such that the robots are interchangeable within each group. The algorithm reduces the k-color problem to several discrete pebble problems. Adler et al. [1] present an efficient algorithm for multi-robot motion planning for unlabeled discs in simple polygons. They transform a continuous problem into a discrete pebble motion on a graph problem.

Solovey, Yu, Zamir, Halperin [29] present a polynomial-time complete algorithm for unlabeled disc robots in the plane. It minimizes the total path length for the set of robots, shown to be at most $4m$ longer than the optimal solution, where m is the number of robots, assuming certain robot-obstacle and start-goal separation constraints are satisfied. Solovey and Halperin [28] show that the problem of unlabeled unit-square robots translating among obstacles is PSPACE-hard.

Luna and Bekris [18] developed a method for cooperative path-finding that is polynomial in the number of robots and is complete for all problem instances with at least two empty vertices in the graph. It uses push-and-swap primitives to restrict unnecessary exploration of search space. Liu and Shell [17] devel-

oped a hierarchical dynamic partitioning and distribution scheme for large-scale multi-robot task allocation. Nam and Shell [20] analyzed cost uncertainties in multi-robot task allocation problems, and the sensitivity of optimal assignments to variations in the cost matrix. Wagner and Choset [32] develop an efficient multiple robots path planning algorithm, where paths for individual robots are initially generated and coordination among robots is performed when needed due to collisions.

van den Berg et al. [4] presented an efficient method for reciprocal n-body collision avoidance that provides a sufficient condition for multiple robots to select an action that avoids collisions with other robots, though each acts independently without communication with others. van den Berg et al. [5] study path planning for multiple robots in the presence of obstacles. They apply optimal decoupling techniques to problems with low degrees of coupling. They decompose a multi-robot problem into a sequence of subproblems with a minimum degree of coupled control. The arrival times of the robots are not optimal as the plans are executed sequentially.

Light-actuated digital microfluidics: Lab-on-a-chip technology scales down multiple laboratory processes to miniature chips capable of performing automated chemical analyses.

Light-actuated digital microfluidics (LADM) systems use moving patterns of projected light on a continuous photoconductive surface to move droplets [22],[23]. The projected pattern of light effectively creates virtual electrodes on the lower substrate. By moving the virtual electrodes, droplets can be moved in parallel on the microfluidic chips to perform multiple biochemical reactions (Figure 1(a)).

By modeling droplets as robots, we can achieve collision-free motions optimized to reduce completion time. We have explored the problem of coordinating multiple droplets in light-actuated digital microfluidic systems intended for use as lab-on-a-chip systems. We focused primarily on creating matrix formations of droplets for biological applications. To achieve collision-free droplet coordination while optimizing completion times, we applied multiple robot coordination techniques. We used a mixed integer linear programming (MILP) approach to schedule coordination of both individual droplets and batches of droplets given their paths [19]. We also developed a linear time coordination algorithm for batch coordination of droplet matrix layouts.

7 Conclusion

This paper described algorithms to perform optimal assignment of teams of robots translating in the plane from an initial formation to a variable goal formation. We considered the case when each robot is to be assigned a goal position, the individual robots are interchangeable, and the goal formation can be either scaled or translated. We computed the costs for all pairs of initial, goal robot assignments as functions of the parameters of the goal formation, and partitioned the parameter space into equivalence classes using computational geometry techniques. We compute a minimum completion time assignment for an equivalence

class by formulating it as a linear bottleneck assignment problem (LBAP). To improve efficiency, we solved the LBAP problem for each equivalence class by incrementally updating the solution as the formation parameters are varied. This work is motivated by applications that include the motion of droplet formations in digital microfluidic lab-on-a-chip devices, and of robot and drone formations in the plane.

Our emphasis has been on robot assignment with variable goal formations. In the event that two or more robots have assigned paths that lead to collisions, we must take additional steps to avoid collisions. These could include coordination using MILP solutions [2], path coordination [26], or collision avoidance [4].

With the LBAP solution, once the critical edge that determines the objective function value has been determined, some of the lower cost edges can have their assignments swapped without affecting the LBAP cost. This can potentially result in multiple optimal assignments. One way to enforce a consistent assignment for each cell of the arrangement is to solve the Lexicographic LBAP [7] instead of the standard LBAP. The advantage of using the Lexicographic LBAP (LexLBAP) is that for the optimum LBAP value, it minimizes the distance traveled by each robot. We need to evaluate the LexLBAP only once, for the optimal LBAP found.

This paper describes initial steps towards addressing the problem of simultaneously optimizing robot assignments and variable goal formations. There are many aspects to be explored in future work. We are working to solve the assignment problem with combined scaling and translation. The relatively high computational complexity of the approach will make application to very large formations (with hundreds of robots) challenging. We are therefore interested in approaches to reduce the effective complexity. For example, methods to prune the set of cells at which an LBAP has to be solved will be useful. Another approach is to partition the set of robots into smaller subsets. Since UAVs and spacecraft operate in a 3D workspace, another useful direction is to extend the approach to 3D formations.

Acknowledgments: This work benefited from discussions with Dan Halperin, Zhiqiang Ma, and Erik Saule. This work was supported in part by NSF Award IIS-1547175.

References

1. Adler, A., de Berg, M., Halperin, D., Solovey, K.: Efficient multi-robot motion planning for unlabeled discs in simple polygons. In: 11th International Workshop on the Algorithmic Foundations of Robotics (WAFR) (2014)
2. Akella, S., Hutchinson, S.: Coordinating the motions of multiple robots with specified trajectories. In: IEEE International Conference on Robotics and Automation. pp. 624–631. Washington, DC (May 2002)
3. Alonso-Mora, J., Baker, S., Rus, D.: Multi-robot navigation in formation via sequential convex programming. In: 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 4634–4641 (2015)

4. van den Berg, J., Guy, S.J., Lin, M., Manocha, D.: Reciprocal n-body collision avoidance. In: Pradalier, C., Siegwart, R., Hirzinger, G. (eds.) Robotics Research: The 14th International Symposium ISRR, pp. 3–19. Springer, Berlin, Heidelberg (2011)
5. van den Berg, J., Snoeyink, J., Lin, M., Manocha, D.: Centralized path planning for multiple robots: Optimal decoupling into sequential plans. In: Robotics: Science and Systems. pp. 137–144 (2010)
6. de Berg, M., Cheong, O., van Kreveld, M., Overmars, M.: Computational Geometry: Algorithms and Applications. Springer-Verlag, Berlin, third edn. (2008)
7. Burkard, R., Dell'Amico, M., Martello, S.: Assignment Problems. SIAM, revised reprint edn. (2012)
8. Choset, H., Lynch, K.M., Hutchinson, S., Kantor, G.A., Burgard, W., Kavraki, L.E., Thrun, S.: Principles of Robot Motion: Theory, Algorithms, and Implementations. MIT Press (2005)
9. Derenick, J.C., Spletzer, J.R.: Convex optimization strategies for coordinating large-scale robot formations. IEEE Transactions on Robotics 23(6), 1252–1259 (2007)
10. Halperin, D.: Personal communication (2016)
11. Halperin, D., Sharir, M.: Arrangements. In: Goodman, J.E., O'Rourke, J., Tóth, C.D. (eds.) Handbook of Discrete and Computational Geometry. CRC Press, Boca Raton, FL, third edn. (2017), to appear
12. Katsev, M., Yu, J., LaValle, S.M.: Efficient formation path planning on large graphs. In: 2013 IEEE International Conference on Robotics and Automation (ICRA) (2013)
13. Kloder, S., Hutchinson, S.: Path planning for permutation-invariant multirobot formations. IEEE Transactions on Robotics 22(4), 650–665 (2006)
14. Latombe, J.C.: Robot Motion Planning. Kluwer Academic Publishers, Norwell, MA (1991)
15. LaValle, S.M.: Planning Algorithms. Cambridge University Press, Cambridge, U.K. (2006), available at http://planning.cs.uiuc.edu/
16. LaValle, S.M., Hutchinson, S.A.: Optimal motion planning for multiple robots having independent goals. IEEE Transactions on Robotics and Automation 14(6), 912–925 (Dec 1998)
17. Liu, L., Shell, D.: Large-scale multi-robot task allocation via dynamic partitioning and distribution. Autonomous Robots 33(3), 291–307 (2012)
18. Luna, R., Bekris, K.E.: Efficient and complete centralized multi-robot path planning. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS-11). San Francisco, CA (25-30 Sept 2011)
19. Ma, Z., Akella, S.: Coordination of droplets on light-actuated digital microfluidic systems. In: IEEE International Conference on Robotics and Automation. pp. 2510–2516. St. Paul, MN (May 2012)
20. Nam, C., Shell, D.A.: When to do your own thing: Analysis of cost uncertainties in multi-robot task allocation at run-time. In: 2015 IEEE International Conference on Robotics and Automation (ICRA). Seattle, Washington (May 2015)
21. Papadimitriou, C.H., Steiglitz, K.: Combinatorial Optimization: Algorithms and Complexity. Prentice-Hall, Englewood Cliffs, New Jersey (1982)
22. Park, S.Y., Teitell, M.A., Chiou, E.P.Y.: Single-sided continuous optoelectrowetting (SCOEW) for droplet manipulation with light patterns. Lab Chip 10, 1655–1661 (2010), http://dx.doi.org/10.1039/C001324B

23. Pei, S.N., Valley, J.K., Neale, S.L., Jamshidi, A., Hsu, H., Wu, M.C.: Light-actuated digital microfluidics for large-scale, parallel manipulation of arbitrarily sized droplets. In: 23rd IEEE International Conference on Micro Electro Mechanical Systems. pp. 252–255. Wanchai, Hong Kong (Jan 2010)
24. Peng, J., Akella, S.: Coordinating multiple robots with kinodynamic constraints along specified paths. International Journal of Robotics Research 24(4), 295–310 (Apr 2005)
25. Shekar, V., Campbell, M., Akella, S.: Towards automated optoelectrowetting on dielectric devices for multi-axis droplet manipulation. In: IEEE International Conference on Robotics and Automation. pp. 1431–1437. Karlsruhe, Germany (May 2013)
26. Simeon, T., Leroy, S., Laumond, J.P.: Path coordination for multiple mobile robots: A resolution-complete algorithm. IEEE Transactions on Robotics and Automation 18(1), 42–49 (Feb 2002)
27. Solovey, K., Halperin, D.: k-color multi-robot motion planning. International Journal of Robotics Research 33(1), 82–97 (2014)
28. Solovey, K., Halperin, D.: On the hardness of unlabeled multi-robot motion planning. In: Robotics: Science and Systems. Rome, Italy (Jul 2015)
29. Solovey, K., Yu, J., Zamir, O., Halperin, D.: Motion planning for unlabeled discs with optimality guarantees. In: Robotics: Science and Systems. Rome, Italy (Jul 2015)
30. Turpin, M., Michael, N., Kumar, V.: CAPT: Concurrent assignment and planning of trajectories for multiple robots. The International Journal of Robotics Research 33(1), 98–112 (2014)
31. Turpin, M., Mohta, K., Michael, N., Kumar, V.: Goal assignment and trajectory planning for large teams of interchangeable robots. Autonomous Robots 37(4), 401–415 (Dec 2014)
32. Wagner, G., Choset, H.: Subdimensional expansion for multirobot path planning. Artificial Intelligence 219, 1–24 (Feb 2015)
33. Wurman, P., D'Andrea, R., Mountz, M.: Coordinating hundreds of cooperative, autonomous vehicles in warehouses. AI Magazine 29(1), 9–19 (2008)
34. Yu, J., LaValle, S.M.: Multi-agent path planning and network flow. In: Frazzoli, E., Lozano-Perez, T., Roy, N., Rus, D. (eds.) Algorithmic Foundations of Robotics X, Springer Tracts in Advanced Robotics, vol. 86, pp. 157–173. Springer Berlin Heidelberg (2013)
35. Yu, J., LaValle, S.M.: Planning optimal paths for multiple robots on graphs. In: 2013 IEEE International Conference on Robotics and Automation (ICRA) (2013)