2007 IEEE International Conference on
Robotics and Automation
Roma, Italy, 10-14 April 2007

The Frugal Feeding Problem: Energy-efficient, multi-robot, multi-place rendezvous

Yaroslav Litus, Richard T. Vaughan, Pawel Zebrowski
Autonomy Lab, School of Computing Science, Simon Fraser University, Canada
{ylitus, vaughan, pzebrows}@sfu.ca

***Abstract*— We consider the problem of finding an energy-efficient route for a service robot to rendezvous with every member of a heterogeneous team of mobile worker robots. We analyze the general and special cases of the problem, finding it to be at least as hard as the travelling salesman problem. Decomposing the problem into two components: (i) an ordering of robot meetings; and (ii) finding an optimal set of meeting places given an ordering, we present useful solutions to part (ii) only. We propose and compare a discrete algorithm for the restricted meeting location case and two numerical algorithms for the continuous case with weighted Euclidean distance energy cost functions. Anticipating future work, we speculate briefly on suitable ordering heuristics and the need for an integrated method.**

I. Problem description and characterization

Assume a team of robots is working in some environment. For prolonged operation, the worker robots can recharge by docking with a dedicated refueling (or equivalently, recharging) robot called a *tanker*, as described in [1]. If fuel is a precious resource, as is common in real world applications, then an important measure of system efficiency is the ratio of fuel expended by the workers in doing work to the total energy expended by all robots. In this paper we seek to minimize the total amount of fuel spent driving robots to refueling rendezvous.

As a generalization, the following natural problem can be stated: given a set of original locations of worker robots and tanker robot, find the set of meeting points such that the tanker meets every worker and that minimizes the total energy spent on locomotion. By analogy to a mother animal attending her offspring we dub this problem the "Frugal Feeding Problem". It can be stated formally as follows:

Definition 1 (Frugal Feeding Problem): Given tanker location $p_0 \in \mathbb{R}^d$, workers locations $r_i \in \mathbb{R}^d, i = 1..k$ and locomotion cost functions $C_i : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}, i = 0..k$ find

$$\min_{\pi, p_1, p_2, \ldots p_k} \sum_{i=1}^{k} \left(C_0(p_{i-1}, p_i) + C_{\pi(i)}(r_{\pi(i)}, p_i) \right) \quad (1)$$

Here $C_0(x, y)$ gives the cost of tanker relocation from x to y, $C_i(x, y)$ gives the corresponding cost for worker i, and $\pi : \{1..k\} \to \{1..k\}$ permutes the workers according to the order in which they are attended by tanker.

Definition (1) could be amended to require the tanker to return to its original location after attending all workers, perhaps to refuel itself. This modification does not change the following analysis.

The problem has two components. One is combinatorial (finding the order in which robots should be attended), another is analytical (finding the meeting points for the given order).

We will denote the solution points as p_i^* and corresponding permutation as π^*. The possibility of several robots being attended in one place is permitted, and captured by the possible coincidence of some meeting points p_i^*.

The Fermat-Torricelli problem (also called the Steiner-Weber problem) asks for the unique point x minimizing the sum of distances to arbitrarily given points $x_1, \ldots, x_n$ in Euclidean d-dimensional space. Elaborating on the conventions in the Fermat-Torricelli problem literature [2], we name the location of solution points as follows.

Definition 2 (Special cases for solution): If $p_i^* = r_j$ for some j, we will call p_i^* *worker absorbed*. In this case worker i should either remain still and wait for the tanker to come to it, or it should move to the location of another worker. If $p_i^* = p_0$, we call p_i^* *tanker absorbed*. In this case the tanker should not move, but worker i should move to the tanker's original location. Otherwise, p_i^* is not coincident with the starting point of any robot, and we call it *floating*. We show below that all of these are plausible contingencies.

The problem definition does not commit to any particular locomotion cost function. Cost functions could be complex to account for the presence of obstacles or other heterogeneities of environment or robots. Unfortunately the nature of the cost function can make the complete problem very difficult to solve. Here we use the straightforward cost function

$$C_i(x, y) = w_i ||y - x|| \quad (2)$$

So the cost of relocating a robot is simply the weighted Euclidean distance between the origin and destination. This serves as an approximation of the energy losses due to friction if w_i is set proportional to the robot weight. The analysis that follows assumes a cost function based on the l^2-norm, though similar arguments apply to more complex cost functions.

II. Analysis

The first observation we make is that solution points p_i^* can not lie outside the convex hull of $\{p_0, r_1, r_2, ...r_k\}$. This can be seen by considering a candidate meeting point outside the hull: replacing the candidate point with the closest point on the convex hull will unambiguously decrease the value of the goal function. Also, it is easy to prove the convexity of the objective function in (1) (see, e.g. [3, p. 239] for the idea of the proof).

A. Special cases

Under certain conditions we can quickly find solutions without solving the general problem. We show first the sufficient condition for the case where the meeting point is absorbed at the worker's own location:

Lemma 1: If $w_i \geq 2w_0$ then $p_i^* = r_i$. If $w_i > 2w_0$, then $p_i^* = r_i$ is the unique solution for p_i.

Proof: The components of the objective function in (1), which depend on p_i are

$$g_i(p_i) = w_0||p_i - p_{i-1}|| + w_0||p_{i+1} - p_i|| + w_i||p_i - r_i|| \quad (3)$$

We can consider g_i in isolation. Point $p_i^* = \operatorname{argmin} g_i(p_i)$ is the solution to the weighted Fermat-Torricelli problem formed by points p_{i-1}, p_{i+1}, r_i with corresponding weights w_0, w_0, w_i. As shown by Kupitz [2] there is a sufficient condition for the solution to be exactly at point r_i (the *absorbed case*) if points are non-collinear:

$$||w_0(\vec{u}(r_i, p_{i+1}) + \vec{u}(r_i, p_{i-1}))|| \leq w_i, \quad (4)$$

where $\vec{u}(x, y) = (y - x)/||y - x||$ gives the unit vector in direction from point x to point y. Now we find the upper bound of the left side of (4) to solve for the condition, sufficient for any pair of p_{i-1}, p_{i+1} which satisfies the non-collinearity condition. The left side can achieve value $2w_0$ only when $p_{i-1} = p_{i+1}$, thus for the non-collinear case the condition is $w_i \geq 2w_0$. We complete the proof by considering the collinear case. [3, p. 251] gives the sufficient condition for the collinear case of the Fermat-Torricelli problem. Point $p_{\min}$ is unique, if $W^- < W/2$ and $W^+ < W/2$, where W^- is the total weight of the points to the one side of $p_{\min}$, W^+ is the total weight of the points to the other side, and W is the weight of the minimum point itself. If $W^- = W^+ = W/2$, then $p_{\min}$ is one of the many minimum points which comprise a closed segment and are defined by these equalities. Interpreting these conditions for our case we obtain the desired result. ■

Now we show the sufficient conditions for the case where the tanker stands still and all workers are charged at the tanker location.

Lemma 2: If $\sum_{i=1}^{k} w_i \leq w_0$ then $p_i^* = p_0$ for all i. If inequality is strict, then this solution is unique.

Proof: Let $C_a = \sum_{i=1}^{k} w_i||r_i - p_0||$ be the cost of a complete tanker absorbed solution, and $C(p_1, p_2, ...p_k) = \sum_{i=1}^{k} (w_i||r_i - p_i|| + w_0||p_i - p_{i-1}||)$ denote the cost of an alternative solution. We will show that for all values of $p_i, i = 1..k$ inequality $C_a - C(p_1, ...p_k) \leq 0$ holds.

$$C_a - C(p_1, ...p_k) = \sum_{i=1}^{k} w_i(||r_i - p_0|| - ||r_i - p_i||) -$$

$$-w_0 \sum_{i=1}^{k} ||p_i - p_{i-1}|| \leq$$

(triangle inequality)

$$\leq \sum_{i=1}^{k} w_i||p_i - p_0|| - w_0 \sum_{i=1}^{k} ||p_i - p_{i-1}|| \leq$$

(let $||p_j - p_0|| = \max_i\{||p_i - p_0||\}$)

$$\leq \sum_{i=1}^{k} w_i||p_j - p_0|| - w_0 \sum_{i=1}^{j} ||p_i - p_{i-1}|| \leq$$

(series of triangle inequalities)

$$\leq \sum_{i=1}^{k} w_i||p_j - p_0|| - w_0||p_j - p_0|| \leq 0$$

The case with strict inequality is argued similarly. ■

B. Complexity

Lemma 1 can be used to show that the frugal feeding problem is NP-hard because of its combinatorial component which finds the best order in which robots are visited. To do this we reduce the geometric salesman problem to the frugal feeding problem.

Theorem 1: GEOMETRIC TRAVELING SALESMAN $\leq$ FRUGAL FEEDING

Proof: Given an instance of the geometric traveling salesman problem (set of points $a_i, i = 0..n$ where a_0 is the original location of the salesman) we create the following frugal feeding problem. Assign $p_0 = a_0; w_0 = 1; r_i = a_i, w_i > 2$ for $i = 1..n$. According to Lemma 1, for any considered order of meetings (given by permutation π in (1)) optimal locations $p_i = r_{\pi(i)}$. This means that the set of solution points coincides with the set of robot locations, $\{p_i^*\} = \{r_i\}$. Thus the solution for the frugal feeding problem will be the minimum traveling path between all points r_i starting at location p_0 which is exactly the solution to the original problem. ■

Theorem 1 proves that the combinatorial component of the frugal feeding problem is not easier than the traveling salesman problem. Thus, a search for efficient domain-specific heuristics looks more promising than attempts to find an exact solution. An interesting opportunity for future work is to find an algorithm which will solve the combinatorial component (meeting order) and analytical component (rendezvous locations) simultaneously. Literature on the traveling salesman problem is abundant, so depending on the particular initial conditions, a suitable heuristic could be selected. For example, if the number of robots is large and time is critical, heuristics presented in [4] could be used. A cheap and simple alternative ordering could be a nearest-neighbor series, starting at p_o. We conjecture that the order given by the solution to the geometric traveling salesman problem set on the points $p_0, r_1, ...r_n$ may be a reasonable approximation to the optimal order π^*. A possible heuristic is to use this order and avoid the expansion of the permuted orderings, but we do not yet have results to evaluate this idea.

In the remaining analysis we will assume that the order π is given. For the practical experiments we use brute force by iterating through all possible orderings, solving the analytical component for each of the orderings and using the ordering which resulted in the best solution.

The analytical component of the problem is a specific case of the facility location problem [5], where points $p_0, r_1, ...r_n$ serve as the existing facilities, points $p_1, ..., p_n$ are new facilities and the costs are set appropriately. Because the

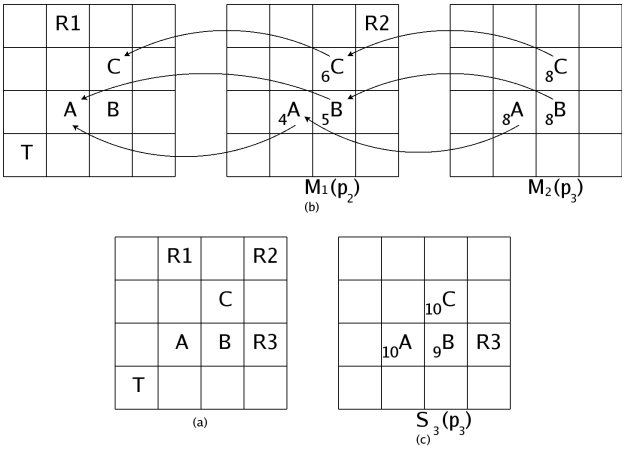


Fig. 1. Illustrative example. (a) Grid world with rectilinear distances. Tanker is denoted as T, workers as $R1, R2, R3$, possible meeting locations as A, B, C. (b) Construction of maps for problem solution. (c) Function, used in the selection of the minimum cost value.

gradient and Hessian of the cost function are not defined at the locations of existing facilities, defining gradient-based numerical methods is not trivial. Moreover the instance of the facility location problem we consider could be degenerate at the optimal solution, since more then one new facility may coincide with the same existing facility [6]. This makes finding a good numerical algorithm even more difficult.

Below we present three methods to solve the analytical component of the frugal feeding problem. We start from the case where locations of p_i are restricted to some finite set and proceed with two custom numerical methods for the general case with cost functions, described by (2).

III. Restricted locations case

Assume that the locations where the tanker could attend workers are not arbitrary, but are limited to a fixed set of places. This models, for example, a list of coffee shops where a consultant can meet clients, or a list of air strips where a robot reconnaissance airplane could meet a ground-based tanker truck. The finite set of meeting places could also be a division of continuous space into a regular grid.

In this discrete version, the optimization problem (1) is extended with a constraint $p_i^* \in L$, where L is the set of possible meeting places. L could be a superset of the original robot locations $\{r_i\} \cup \{p_0\}$. The naive brute-force algorithm for solving the analytical part of the problem will take $O(|L|^k k)$ time ($|L|^k$ possible meeting points arrangements and $O(k)$ to calculate the cost value). However, it is possible to exploit the structure of the objective function and to provide the algorithm with running time $O(|L|^2 k)$. This algorithm is based on the ideas of dynamic programming [7]. First we consider an example which will explain the approach, then we describe the algorithm formally.

A. Illustrative example

Consider the setting described in Fig.1a. The tanker is located at point T, and three workers are located at $R1$, $R2$, and $R3$. All robots can move into the 4-connected adjacent cells only, and the locomotion cost for tanker and workers is the rectilinear (Manhattan) distance between the points, measured in number of cells moved. The set of possible meeting places is restricted to three points $L = \{A, B, C\}$. We will solve the problem by building a sequence of maps illustrated in Fig.1(b),(c).

The first map $M_1(p_2)$ gives the answer to the question: "for the given location p_2 of meeting with robot $R2$, what is the best place p_1 to meet robot $R1$ in terms of the cost of moving tanker from T to p_1 and then to p_2 plus the costs of moving $R1$ to p_1?" The map also provides corresponding minimal costs. For example, if robot $R2$ is to be met at point B, then the best place to meet $R1$ is A and associated costs are 5 (Fig. 1b). If there are several locations which yield the same minimum cost, any of them could be used without affecting the quality of the solution.

Using $R1$ we can build the second map $M_2(p_3)$ which again answers the question "for the given location p_3 of meeting with robot r_3 what is the cheapest place p_2 to meet $R2$ in terms of the accumulated minimum cost of the tanker arriving at p_2 (this is given by $M_1(p_2)$), relocating to point p_3 plus the costs of moving $R2$ to p_2."

Finally, using M_2 we can build the function $S_3(p_3)$ which shows the minimum total cost for each possible point for charging p_3 (Fig. 1c). This cost is the accumulated minimum cost of the tanker getting to p_3 which is given by $M_2(p_3)$ plus the cost of moving $R3$ to p_3. By minimizing S_3 we conclude, that the best place for charging r_3 is B. Now we can roll back the maps and find the rest of the locations. The best place to charge $R2$ is given by M_2. Thus, $p_2^* = M_2(p_3^*) = M_2(B) = B$. Similar, $p_1^* = M_1(p_2^*) = A$. The solution to the problem is (A, B, B) and the minimum cost is 9.

B. Formal presentation

Now we formally state the procedure described above. Consider the following reformulation of the problem (for a given ordering of robots):

$$\begin{aligned} &\min_{p_k}[C_k(r_k, p_k) + \\ &\min_{p_{k-1}}[C_0(p_{k-1}, p_k) + C_{k-1}(r_{k-1}, p_{k-1}) + \\ &\min_{p_{k-2}}[C_0(p_{k-2}, p_{k-1}) + C(r_{k-2}, p_{k-2}) + ... + \\ &\min_{p_1}[C_0(p_1, p_2) + C_1(r_1, p_1) + C_0(p_0, p_1)]...]]] \end{aligned} \tag{5}$$

The equivalence of the analytical part of (1) and (5) is the corollary of the obvious equality $\min_{x,y} f(x, y) = \min_x[\min_y f(x, y)]$.

Now consider the following sequence of functions:

$$\begin{aligned} S_1(p_1, p_2) &= C_0(p_1, p_2) + C_1(r_1, p_1) + C_0(p_0, p_1); \\ S_i(p_i, p_{i+1}) &= C_0(p_i, p_{i+1}) + C_i(r_i, p_i) + S_{i-1}(p_i), \\ &\quad \text{for } i = 2..k-1; \\ S_k(p_k) &= C_k(r_k, p_k) + S_{k-1}(p_k) \end{aligned} \tag{6}$$

Problem (5) could be solved by sequentially building the

mappings

$$M_1(p_2) = (\min_{p_1} S_1, \arg\min_{p_1} S_1); \quad (7)$$
$$M_i(p_{i+1}) = (\min_{p_i} S_i, \arg\min_{p_i} S_i), \text{ for } i = 2..k-1;$$
$$M_k = (\min_{p_k} S_k, \arg\min_{p_k} S_k);$$

Each mapping M_i is built by considering every value in L as a parameter and for each value of parameter every value in L is considered as a possible solution to the minimization problem. If minimization does not yield a unique solution, we arbitrarily select one of the minimizing points. Thus, it takes $O(k|L|^2)$ steps to build all maps. M_{k1} gives the minimum value for (5) and M_{k2} gives point p_k^*. The rest of the points could be found as $p_i^* = M_i(p_{i+1}^*)_2$.

The last thing to mention before continuing with the general case is that the algorithm presented in this section does not depend on the particular form of the movement cost functions.

IV. Continuous case

From now on we will use the energy cost functions, described by (2). We introduce some notation to describe the algorithms below. The desired convergence precision is ϵ. The objective function is $F(p_1, p_2, ...p_k)$. Define $f((s_i),(c_i),x) = \sum_i c_i||x - s_i||$, $g((s_i),(c_i),x) = \sum_{x \neq s_i} c_i \nabla||x - s_i||$, $G((s_i)(c_i),x) = \sum_{x \neq s_i} c_i \nabla^2||x - s_i||$, $\lambda((s_i),(c_i),x) = \sum_{x \neq s_i} c_i/||x - s_i||$.

At the time of writing we do not have a formal analysis of the convergence properties of these algorithms. However, both methods are based on the generalizations of algorithms for the Fermat-Torricelli problem which have proofs of convergence (see [8], [9] cited by [10]).

A. First order method

We describe one step $s(x)$ of this iterative method. It is used recursively to obtain a sequence of solution approximations $x_{t+1} = s(x_t)$ until $||x_t - x_{t-1}|| < \epsilon$.

At iteration t given a current approximation to the solution $p_i^t, i = 1..k$

1) Let $S(i) = (p_{i-1}, p_{i+1}, r_i), W(i) = (w_0, w_0, w_i)$ for $i = 1..k-1$; $S(k) = (p_{k-1}, r_k), W(k) = (w_0, w_k)$
2) Next approximation $p_i^{t+1} = P(S(i), W(i), p_i)$

Here $P(S, W, p)$ denotes the step of the first-order Wang acceleration of the Weiszfeld algorithm for the Fermat-Torricelli problem [10]. Given the set of points s_i with weights c_i, $i = 1..m$ and current approximation x we can calculate $F((s_i),(c_i),x)$ as follows:

1) Compute $g' = g((s_i),(c_i),x)$. Compute $\sigma = ||x - s_j|| = \min_i\{||x - s_i||\}$. If $\sigma > 0$, goto step 2, else goto step 3
2) If $g' = 0$ then return x, else return $x - g'/\lambda((s_i),(c_i),x)$
3) If $||g'|| \leq c_j$ then return x, otherwise return $x - [g'/\lambda((s_i),(c_i),x))][(||g'|| - c_j)/||g'||]$

B. Second order method

The second order method we propose is a Newton minimization procedure. We exploit the fact that the Hessian of the objective function has a block-diagonal structure. Because of this it is possible to calculate the Newton direction separately for each of solution point approximations p_i. If the Hessian for some point is singular, or does not exist because p_i is absorbed, then the first order step is used to calculate the direction. After all directions are calculated, the line search is performed to calculate the length of the step. This method is based on a second order Xue method for the Fermat-Torricelli problem [10].

Initially set $\delta_i = \delta$, where δ is a cutoff parameter. At iteration t given current approximation to the solution $p^t = (p_i^t), i = 1..k$ and current values of cutoff parameters δ_i^t

1) Let $S(i) = (p_{i-1}, p_{i+1}, r_i), W(i) = (w_0, w_0, w_i)$ for $i = 1..k-1$, $S(k) = (p_{k-1}, r_k), W(k) = (w_0, w_k)$
2) For $i = 1..k$ calculate direction and new value of cutoff parameter $(\delta_i^{t+1}, d_i) = Q(S(i), W(i), \delta_i^t, p_i)$
3) Next approximation $p^{t+1} = p^t + \alpha d$,where α is a largest number in $\{1, 1/2, 1/4, ...\}$ such that $F(p^t + \alpha d) \leq F(p^t)$

Here $Q(S, W, \delta_i^t, p_i)$ calculates the direction for a particular component of the solution approximation and a new cutoff parameter value. Given a set of points s_i with weights c_i, $i = 1..m$ and current approximation x we can calculate $Q((s_i),(c_i),\delta,x)$ as follows:

1) Compute $g' = g((s_i),(c_i),x)$. Compute $\sigma = ||x - s_j|| = \min_i\{||x - s_i||\}$. If $\sigma = 0$, then set $\delta^{+1} = \delta$, goto step 4. If $\sigma \geq \delta$, then set $\delta^{+1} = \delta$, goto step 3.
2) (Cutoff step) If $f((s_i),(c_i),s_j) \leq f((s_i)(c_i),x)$ then return $(\delta^{+1}, s_j - x)$
3) Compute $G' = G((s_i),(c_i),x)$ If G' is singular, goto step 4. Otherwise solve system $G'd = -g'$ for d and return (δ^{+1}, d)
4) If $||g'|| \leq c_j$ then return $(\delta^{+1}, 0)$, otherwise return $(\delta^{+1}, -[g'/\lambda((s_i),(c_i),x))][(||g'|| - c_j)/||g'||])$

V. Experiment

A. Experimental setting

We performed a series of experiments to empirically evaluate the performance of the methods described above. Three different settings are considered, each in a 2-dimensional continuous world with a tanker and five robots placed as shown in Fig.3. The standard locomotion function (2) is used. The three settings differ only in the weights in the cost functions. In Fig.3 the robot start positions are indicated by solid black dots, with the size of the dot representing the relative locomotion cost. The first setting (Map 1) has the weight of the tanker $w_0 = 20$, the worker robots have weights $w_i = 1$. In the second setting (Map 2) $w_0 = w_i = 1$. In the last setting (Map 3) $w_0 = 1, w_i = 2$. Map 1 should have solutions which are tanker absorbed, since the weights satisfy the conditions of Lemma 2. Map 2 could possibly have some floating solution points, and Map 3 satisfies the sufficient conditions in Lemma 1, thus all of the solution

points should be worker absorbed. Therefore, these three settings cover three major classes of solutions. Though we can use the sufficient conditions provided in Section III to find the solutions for Map 1 and Map 3 at once, we use these maps to see how well the methods will perform in capturing the absorbed cases.

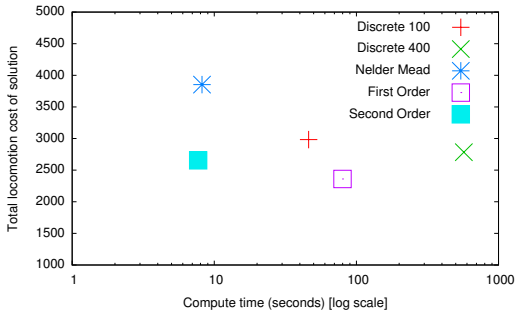
As explained in Section II-B the combinatorial component of the problem is tackled by iterating through all possible orders of visits. Investigating faster methods is left as future work. Here we compare 5 different ways to solve the analytical part of the problem. These are:

1) *Discrete100* covers the embedding square of all robots with a 100 point (10 by 10) uniform rectangle grid and applies the discrete method described in Section III.
2) *Discrete400* uses the discrete method on a 400 point (20 by 20) grid.
3) *Nelder-Mead* is a simplex optimization method which works by querying the value of the objective function in the vertexes of the simplex and updates that simplex accordingly [11, Ch. 8]. Thus, it does not require any special properties of the objective function to operate. For this method the desired precision of optimization is set to 10^{-4} and maximum number of iteration is 2000.
4) *First Order* is the method described in Section IV-A with the desired precision set to 1.
5) *Second Order* is the method described in Section IV-B with the desired precision set to 10^{-1} and cutoff parameter $\delta = 10$.

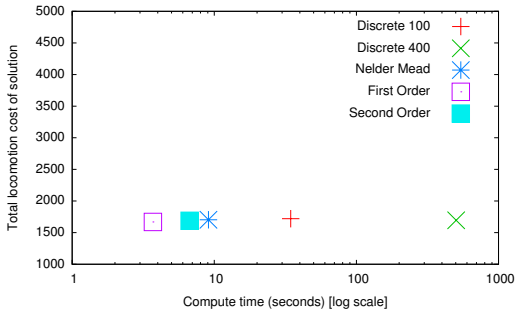
These first and second order precision parameter values are selected experimentally to give comparable running times for the numerical methods in all three settings, since we are interested in finding a method which will find good solutions quickly in all three representative settings. All three numerical methods start from the same solution approximation, which is $p_i^0 = r_{k+1-i}$. An alternative approximation could be to set each meeting place to the weighted average between original tanker location and original worker location, $p_i^0 = (w_0 p_0 + w_i p_i)/(w_0 + w_i)$. We plan to explore the influence of the starting point on the convergence in our future work.

B. Experimental results

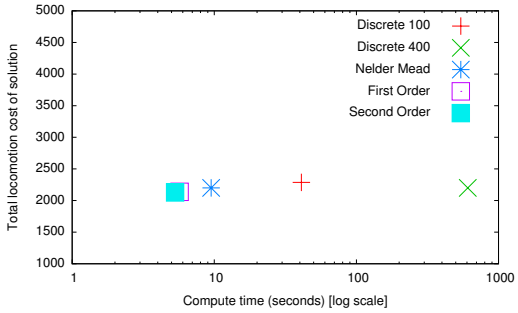
Figure 2 shows the results of running each of the five methods in three experimental settings. The first thing to note is the increase in run-time between the 100 point and 400 point grids for the Discrete method, consistent with the complexity estimates presented in Section III. Run-time grows quadratically in the size of the grid. Thus, for practical purposes the discrete method can be recommended only if the number of *a priori* allowed meeting locations is small, or, similarly, the required precision is low so that the search space can be covered by a coarse grid. The fact that the discrete method failed to find the absorbed solutions for Map 1 and 3 is evident from the cost of the solution not being minimal between all methods. The reason for this is that the grid nodes failed to capture the appropriate points. Thus,



(a) Map 1: $w_0 = 20, w_i = 1 | i > 0$ (Tanker absorbed)



(b) Map 2:$w_0 = w_i = 1$



(c) Map 3: $w_0 = 1, w_i = 2 | i > 0$ (Worker absorbed)

Fig. 2. Experimental results. Solution quality is plotted against compute time for each solution method, each of three settings (maps). Maps 1-3 differ only in locomotion cost weights on the single tanker w_0 and five worker $w_i | i > 0$ robots.

if a discretization is used to find the solution, the original locations of workers and robots should be included in the set of possible locations L even if the search grid does not capture them.

Fig.3a,b,c shows the solutions found by the numerical methods on Map 1, which is constructed to be a tanker absorbed case. The first order method finds the best-quality solution but takes a relatively long time for the method to converge. The second order method converges faster, but the solution it finds is worse. However, a change of the required convergence precision of second order method to

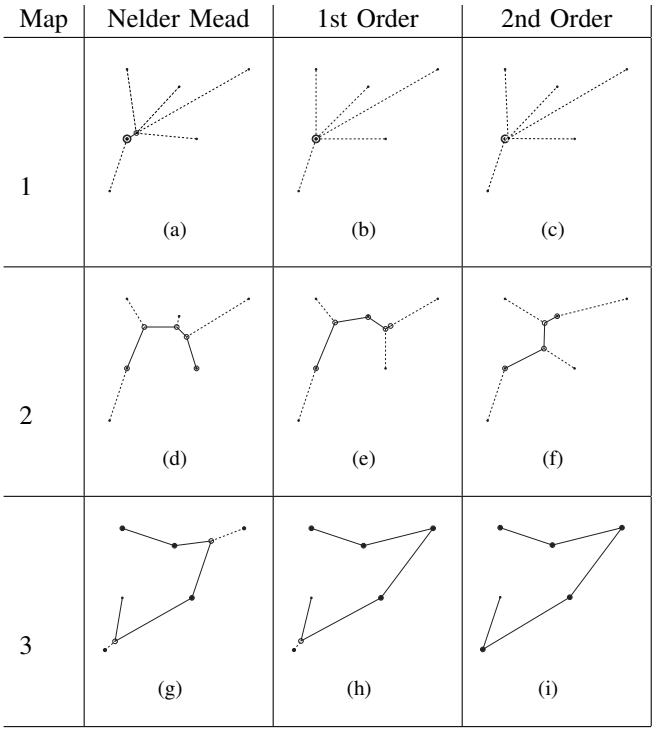


Fig. 3. Rendezvous solutions discovered. Tanker is at the solid spot on figure (a), other points show workers. Circles denote meeting places. Tanker route is shown with solid lines, workers routes are shown with dashed lines.

10^{-4} increases the time to 16.2s while beating the first order solution quality with the value 2346.71. This confirms our expectation that the second order method should be able to find a high-quality solution faster than the first order method. The Nelder Mead method converges quickly on this map, however the quality of the solution it found is relatively poor.

Map 2 reveals an interesting result. The first order method manages to find a better solution (Fig.3(e)) in less time than the second order method takes to find an inferior solution (see Fig.3(f)). Though the difference between the quality of solutions is only 1.2%, it shows, that there is potential for improvement of the second order procedure. Inspection of the sequence of approximations that the second order method produced for this Map shows that many steps had a singular Hessian, thus the method resorts to the first order direction calculation. One of the potential improvements is to take special care of these points. Nelder-Mead performed better on Map 2 than on Map 1, still taking a relatively short time to run (Fig.3(d)).

Map 3 shows that all methods find the same order of tanker/worker meetings. Due to the cutoff procedure used in the second order method, it is able to converge precisely to the locations of workers (Fig.3(i)), providing the best solution. The first order method fails to put one of the meeting places at the location of the worker. (Fig.3(h)). The Nelder-Mead method fails to place two of the meeting places at the best location (Fig.3(g)).

Between-map comparison of the numerical methods shows that the second order method seems to be the safest choice for practical purposes. However, since the first order procedure performs well for the floating case, it could be of practical use when absorbed cases are not likely (based on consideration of the initial conditions). This issue requires more investigation and is left as future work. Finally, the Nelder-Mead method (provided by a multitude of numerical method code libraries) could be used if the quality of the solution is not critical, and convenient implementation is preferred.

VI. Conclusions and Future Work

We have defined and analyzed the frugal feeding problem, which is concerned with finding the set of meeting places for a tanker robot to rendezvous with multiple worker robots, such that the total locomotion cost is minimized.

This paper has provided and compared a variety of *partial* solutions to the problem, dealing only with finding an optimal set of meeting places given an ordering of meetings. The discrete methods produce optimal results in polynomial time; a useful improvement over exponential time naive brute force approach. Our numerical algorithms for the general problem are shown empirically to converge to good solutions.

There are many opportunities for future work, including the search for algorithms which will solve both components of the problem simultaneously. Our preferred class of solutions are distributed online heuristics which will achieve near-optimal solutions without precomputing a complete solution in advance, preferably coping with uncertainty in information about robot locations.

Acknowledgements

This work was supported by NSERC in Canada. The authors gratefully acknowledge the advice of Greg Mori and three anonymous reviewers. Yaroslav Litus thanks Bob Hadley for his generous support.

References

[1] P. Zebrowski and R. Vaughan, "Recharging robot teams: A tanker approach," in *Proceedings of the International Conference on Advanced Robotics (ICAR)*, Seattle, Washington, July 2005.

[2] Y. Kupitz and H. Martini, "Geometric aspects of the generalized fermat-torricelli problem," *Bolyai Society Mathematical Studies*, vol. 6, pp. 55–129, 1997.

[3] H. M. V. Boltyanski and V. Soltan, *Geometric methods and optimization problems*. Kluwer Academic Publishers, 1999.

[4] G. Reinelt, "Fast heuristics for large geometric travelling salesman problems," *ORSA Journal on Computing*, vol. 4, no. 2, 1992.

[5] Z. Drezner, Ed., *Facility location. A survey of application and methods*. Springer-Verlag, 1995.

[6] Y. Li, "A newton acceleration of the weiszfeld algorithm for minimizing the sum of euclidean distances," Ithaca, NY, USA, Tech. Rep., 1995.

[7] R. Bellman, *Dynamic programming*. Dover Publications, 2003.

[8] C. Wang, "On the convergence and rate of convergence of an iterative algorithm for the plant location problem," *Qufu Shiyun Xuebao*, vol. 2, 1975.

[9] G.-L. Xue, "A fast convergent algorithm for $\min \sum_{t=1}^{m} c_t||x - a_t||$ on a closed convex set," *Journal of Qufu Normal University*, vol. 13, no. 3, pp. 15–20, 1987.

[10] J. Rosen and G.-L. Xue, "Computational comparison of two algorithms for the euclidean single facility location problem," *ORSA Journal on Computing*, vol. 3, no. 3, 1991.

[11] J. H. Mathews and K. K. Fink, *Numerical Methods Using Matlab*, 4th ed. Prentice-Hall Inc., 2004.