WIRELESS COMMUNICATIONS AND MOBILE COMPUTING
Wirel. Commun. Mob. Comput. (2011)
Published online in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/wcm.1145

RESEARCH ARTICLE

Connectivity and bandwidth-aware real-time exploration in mobile robot networks

Yuanteng Pei[1]*, Matt W. Mutka[1] and Ning Xi[2]

[1] Department of Computer Science and Engineering, Michigan State University, East Lansing, MI 48824, USA
[2] Department of Electrical and Computer Engineering, Michigan State University, East Lansing, MI 48824, USA

ABSTRACT

Although there has been substantial progress for multi-robot exploration of an unknown area, little attention has been given to communication, especially bandwidth constraints in *time-sensitive* and bandwidth-consuming tasks such as search and surveillance. In such tasks, video/audio streams of a newly explored area should be sent back to the base station in a timely manner. To address this issue, we propose *connectivity and bandwidth-aware exploration* (CBAX), which is an efficient iteration based *real-time* exploration. CBAX divides the problem into frontier node placement, relay node placement with routing path selection, and matching of each robot with its target position. Moreover, we model bandwidth-constrained relay node placement into a new variant of the Steiner minimum tree problem and present our solution. While reducing the exploration time, CBAX maintains the network's connectivity and ensures the aggregated data flows are under the link capacity in transmission. Simulation shows that CBAX outperforms two recent exploration schemes *qualitatively* by demonstrating major improvement in terms of non-overflow transmission time and fully connected transmission time. With enhanced communication quality, CBAX still reduces the exploration time, on average, by 40% and 15%, respectively. In moderately dense scenarios, CBAX even decreases time by 50% and 25%.

KEYWORDS

mobile robot networks; bandwidth; real-time exploration; relay node placement; Steiner minimum tree; bottleneck assignment

***Correspondence**

Yuanteng Pei, Department of Computer Science and Engineering, Michigan State University, East Lansing, MI 48824, USA.
E-mail: peiyuant@cse.msu.edu

1. INTRODUCTION

Exploring an environment is one of the fundamental problems in mobile robotics. Recently, multi-robot exploration has received increased attention for its notable benefit for enhanced efficiency and robustness [1,2]. Many application domains of multi-robot exploration, such as surveillance, reconnaissance, search, and rescue missions in dangerous areas require robot's bandwidth-consuming video/audio information from newly explored area to be reliably and quickly sent back to an operator at the base station (BS) [3]. The reasons for this requirement are threefold. First, human operators often need to monitor the robot team's action and obtain the sensed information immediately. Second, as the current robotic sensing ability is not sufficient to accurately detect complex targets, such a process should be simultaneously augmented with human recognition [4]. Third, operators may even need to teleoperate robots promptly after target discovery or under exceptional circumstances. For instance, when a robot finds a victim partially covered by earthquake debris, the operator may teleoperate the robot to move closer to the target and control the robot arm to remove the obstacle and telediagnose the victim. Thus, limited robot intelligence may need to be augmented with an operator as a necessary and effective way to monitor and control a robot team. Reliable and smooth communication is essential in such a process. Hence, such multi-robot *real-time* exploration (MRRTX) is critical.

Multi-robot real-time exploration can be described as follows: a team of n homogenous robots sent from the *BS* to explore an unknown area. Each robot communicates with limited range by forming a multi-hop network with all robots and the BS. The question is on how to design a coordinated exploration strategy for the robot team to explore the area in a minimum amount of time under the following *two constraints*: (i) the network is always connected when data is transmitted, and (ii) the aggregated consumed bandwidth of data flows from the frontier nodes (FNs) cannot exceed the link bandwidth (capacity) when

transmitting back to the BS. In fact, many types of real-time streaming data such as video/audio have a stream floor rate [5] or a minimum rate to make the stream workable. A link should always provide adequate bandwidth for each flow to meet the floor rate requirement. Without these two constraints, disconnected nodes can evolve, and the aggregated data flows may exceed the link bandwidth, leading to considerable data loss and control disturbance.

To solve MRRTX, we present *connectivity and bandwidth-aware exploration* (CBAX). Differing from existing approaches, CBAX is an iterative exploration that divides the problem of the next round movement path generation and message routing into three subproblems, which are modeled and solved accordingly by variations of *algorithmic and graph-theoretic* problems, such as the set cover problem, the Steiner tree problem, and the linear bottleneck assignment problem (LBAP). When we solve the bandwidth-constrained relay node (RN) placement, we model it as a new variation or one with the γ-*inflow* constraint of the *Steiner minimum tree problem with minimum number of Steiner points and bounded edge length* (SMT-MSP) [6,7]. We also give an upper bound of the algorithm's approximation ratio. CBAX not only reduces the exploration time compared with recent works [1,3] but also guarantees the adequacy of link bandwidth and enhances the communication *quality*. In Figure 1, CBAX is illustrated by an exploration snapshot obtained from our simulation program with the *garden* environment given in Section 5. The figure describes that four FNs with two RNs have explored 56.3% of the area in their fourth iteration of movement, where all robots are connected with the BS and the FNs send non-overflow streams to the BS by the routing paths that CBAX generated.

The outline of the paper is as follows. Related work is described in Section 2. Section 3 introduces the system model and the problem formation. In Section 4, CBAX's general procedure is given, along with the three subproblems and optimizations on the model. Performance evaluation is in Section 5, whereas Section 6 concludes the paper and discusses the future work.

2. RELATED WORK

Communication in multi-robot systems has drawn increased attention [8–10]. Novel research for maintaining connectivity of mobile networks from a distributed control perspective is presented in [8] and [9]. However, it does not have bandwidth awareness or the specific goal of area exploration. Additionally, a networking framework for teleoperation in rescue robotics is given in [10]. In [11], four types of routing protocols are compared in an experiment of one mobile robot with several stationary RNs.

Although the problem of multi-robot exploration has been widely studied in [1–3,12–14] and various exploration strategies are compared in [15], little prior work considers the bandwidth constraints, and only some schemes assume limited communication range and attempt to maintain connectivity. Distributed local random graph, segmentation, and hill climbing-based methods in [1,12,13]

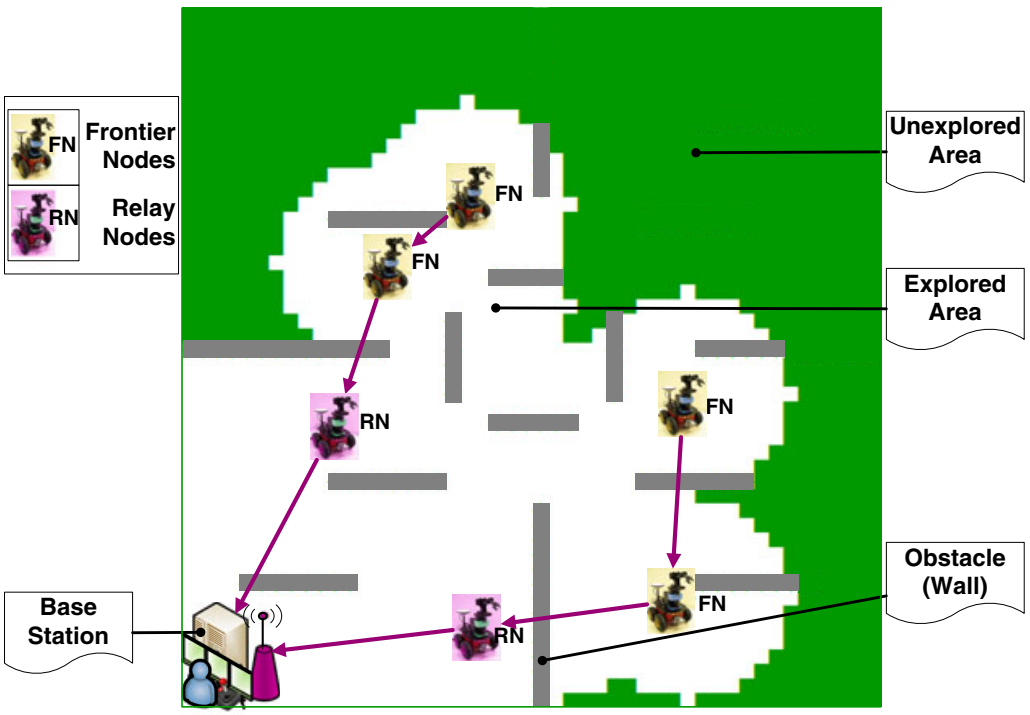


Figure 1. Exploration snapshot of the *garden* environment given in Section 5. (Nodes are scaled larger for illustration purpose.)

explore the area efficiently but provide only intermittent connectivity or assume that robots are always in range.

To maintain connectivity, *nearest measure* is applied in [2] to let robots *tend* to stay close to each other, but it cannot be guaranteed. A concept of *comfort zone* is proposed in [4,14,16], where each node monitors the distance to its neighbor and keeps it in a safe range. When the distance is beyond a safe threshold, the node will change its behavior into tracking the neighbor to retain connectivity. Such schemes lack systemic analysis on how to share relay robots for placing more frontier ones to enhance efficiency. Also, with communication latency and non-uniform velocities of robots, the *detect-and-chase* method may lead to a sequence of trace movement, and the resulting topologies need to be proved deadlock free and invariably connected.

In addition, an iterative motion planning method in [17] maintains connectivity by rejecting trajectories that break tree edges in the distributedly computed minimum spanning tree. However, its exploration model is different from ours without a BS and may not need to consider the bandwidth constraint. Moreover, a centralized scheme in [3] guarantees connectivity by solely selecting fully connected topologies with detecting and recovering from deadlock. Nevertheless, it only allows nodes to move to their immediate four-neighbor positions in each step and does not attempt to optimally place RNs to improve efficiency.

The problem of RN placement studied in [6] and [18] is also pertinent to maintaining connectivity. The authors in [18] employ bipartite graph to cover the maximum number of mobile nodes using a fixed number of RNs in one tier. In [6], the relay sensor placement problem is modeled as a variant of the Steiner tree problem, and improved approximation ratio algorithms are provided for this NP-hard problem without bandwidth consideration.

3. SYSTEM MODEL AND PROBLEM FORMULATION

Before presenting our solution, the system model and the problem formulation are introduced as follows. Notations introduced in the system model are also presented in Table I.

3.1. System model

3.1.1. Robot model.

Each robot has traveling, communication, and sensing capability. It scans the environment using cameras and acoustic sensors with sensing range r_s and communicates with other nodes with communication range r_c by a 802.11 radio, where $r_c > r_s$.

3.1.2. Environment and exploration model with a base station.

The exploration begins with the operator selecting a targeting area, all of whose borders are reachable when robots form a straight line from the BS. The area is modeled by the 2D occupancy map composed of grid cells. The cell size equals that of a robot. The states of a cell are *unvisited*, *exploring*, and *explored*. An explored cell can be either *free* or *obstructed*. An obstacle does not affect communication but blocks the sensing range. An unvisited cell denotes the area that has not been reached by the sensing range of robots.

Arriving at their target positions, all nodes are synchronized and start simultaneously to sense the area and transmit streams back to the BS for a fixed *sensing and transmitting interval* (STI). In STI, the unvisited grid cells within the range of r_s of a node and not blocked by obstacles are changed to *exploring* status. After STI, these cells change to *explored* status because the operator has observed and detected the targets in STI. The cells that are explored but have at least an unvisited neighbor cell are called *frontier cells* and are denoted as $\mathbb{FC}$. We mainly focus on how to efficiently explore the area, and the issues after the target discovery, that is, potential teleoperation tasks, are not discussed here.

Coordinated exploration is executed in an iterative and synchronized way controlled by the BS. At iteration t, Map^t denotes the explored map, and T^t gives the *total time* elapsed. Each robot i has a position P_i^t and is dynamically classified as a *RN* or an *FN*. An FN explores a new area while an RN maintains connectivity for the FNs with the BS. The *robot team size* is n with n_{fn} number of FNs and n_{rn} number of RNs. We define the *information gain* of node i in iteration t, or IG_i^t, as the number of unvisited cells that can be explored with a new position P_i^t.

A *position configuration* is the array of positions of the n robots. Specifically, pCG^t, the position configuration at iteration t is defined as $pCG^t = \{P_1^t, P_2^t, \ldots, P_n^t\}$. The set of all possible pCG^t is denoted as ${}_P\mathbb{CG}^t$. Each pCG^t is associated with a *total information gain*, or tIG^t, which is the total number of non-overlapped unvisited cells that can be explored with a new position P_i^t for each node i. A *valid* position configuration is the one that satisfies *Constraint I* defined in the following paragraphs. Besides, $Path_i^t$ is the migrating path of node i from iteration $t-1$ to t, and $Path^t$ is the set of $Path_i^t$ for each node i. Similar to [8], we define a *dynamic graph* $G(t) = \{V, E(t)\}$ where V denotes the sets of vertices indexed by the set of robots with BS and $E(t)$ is the edge set representing the time varying set of bi-directional communication links of vertices and $(i, j) \in E(t)$ iff $dist(P_i^t, P_j^t) \leq r_c$, where $dist(i, j)$ denotes the straight-line distance between i and j.

In addition, we define a *routing configuration* by the path array from all FNs back to BS where the streams traverse. Specifically, rCG^t, the routing configuration at iteration t is defined as $rCG^t = \{\mathbb{RL}_1^t, \mathbb{RL}_2^t, \ldots, \mathbb{RL}_n^t\}$, where $\mathbb{RL}_i^t$ is the *route list* or the sequence of nodes on the routing path from FN i to BS in iteration t. A *valid* routing configuration is the one satisfying *Constraint II* or Equation 2 defined in the following section. Combining both position and routing configurations, we define the *configuration* at iteration t as $CG^t = \{pCG^t, rCG^t\}$. A *valid* configuration is the one that satisfies *Constraints I and II* given in the

succeeding sections and $\mathbb{CG}_v^t$ denotes the set of all valid configurations in iteration t.

3.1.3. Wireless communication model.

We analyze the wireless communication in area exploration with the following properties: (i) the unit-disk communication model is adopted, and two nodes i, j, can communicate as long as $dist(i,j) \leq r_c$; (ii) all links have a uniform *link capacity* R_{capacity}, and an FN has a uniform *flow sending rate* R_s for its video/audio streams not less than their stream floor rate in STI, where a *flow* is the combined video and audio streams from an FN to the BS; (iii) streams are sent in a single selected path rather than multiple paths; and (iv) interference between nodes will be analyzed in future work.

3.1.4. Constraints I and II.

In the STI of each iteration t,

(1) *Constraint I.* The robot team with BS's network $G(t)$ is connected.
(2) *Constraint II.* The aggregated consumed bandwidth cannot exceed the link capacity on each link of the path from FNs to BS, which can also be expressed as follows

$$R^e_{\text{capacity}} \geq \sum_{f \in F_e} R^e_{\text{consumed}(f)} \tag{1}$$

f represents a flow and F_e denotes all the flows passing through link e. With the properties in our wireless communication model, Equation 1 is rewritten as follows:

$$R^e_{\text{capacity}} \geq \gamma \cdot R_s \tag{2}$$

An integer γ, or the *bandwidth ratio*, denotes the maximum number of flows that can pass through link e without overflowing the link, for example, $\gamma = 3$ when $R_s = 450$ KB/s with 11 Mbps link; $\gamma = 5$ when $R_s = 128$ KB/s with 5 Mbps link.

3.1.5. Illustration.

When iteration t begins, BS computes CG^t and sends P_i^t and $Path_i^t$ to each node i. Upon receiving them, i will migrate to the targeting position P_i^t by path $Path_i^t$ from P_i^{t-1} and send the acknowledgement back to the BS when it arrives. mT^t or the *migration time* in iteration t is determined by the bottleneck or the longest moving time among all nodes. After the BS receives the acknowledgements from all nodes, it informs all FNs to enter STI and starts the stream transmission to the BS by the routes defined in rCG^t. Then, the human operator can start to identify targets in STI by monitoring the video/audio sent from the FNs. An iteration adjourns with the completion of STI. Figure 2 also illustrates the robot's states in exploration.

3.2. Problem formulation

Based on the system model, the major problem is on how to explore the whole area for a robot team with minimum time *under the communication constraints I and II*. Although the problem can be difficult to solve as a whole, our solution leverages a heuristic answer to a subproblem. The subproblem becomes more tractable but remains nontrivial because it is further divided into variants of two NP-hard problems and a polynomially solvable problem. This subproblem is to find a *local optimal configuration* in iteration t, or CG^t_{lopt}, which has maximal total number of cells explored in unit migration time. Formally, the subproblem is as follows: given pCG^{t-1}, find CG^t_{lopt} such that

$$CG^t_{\text{lopt}} = \underset{CG^t \in \mathbb{CG}_v^t}{\text{argmax}} \left(\frac{tIG^t}{mT^t} \right) \tag{3}$$

4. CONNECTIVITY AND BANDWIDTH-AWARE EXPLORATION

4.1. CBAX overview

Generally, CBAX attempts to enhance efficiency by maximizing the explored area by placing more robots in frontier areas while reserving less robots as RNs. Additionally, we endeavor to reduce the bottleneck longest distance that robots travel, as it also directly impacts the exploration time. The problem of finding CG^t_{lopt} in Section 3.2 is largely divided into three subproblems:

- Frontier node placement: Where to place n_{fn} number of FNs in $\mathbb{FC}$ to cover maximum amount of unexplored area. The distances from current positions to

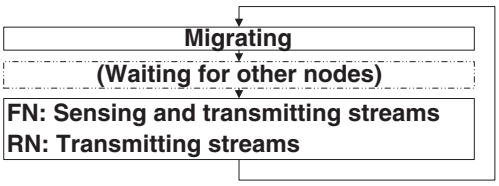


Figure 2. Robot status in one iteration.

Table I. Notations in connectivity and bandwidth-aware exploration.

Terms	Definitions
BS	Base station
FN, RN	Frontier node and relay node; $\mathbb{FN}, \mathbb{RN}$ represent their set
n, n_{fn}, n_{rn}	Number of all robots, number of FN, and number of RN
STI	Sensing and transmitting interval
Flow	Combined video and audio streams from an FN to the BS
r_c,r_s	Communication range and sensing range
$\mathbb{FC}$	Set of Frontier cells
P_i^t	Position of node i in iteration t
$G(t)$	Dynamic graph, $G(t) = \{V, E(t)\}$
Map^t	Explored map in iteration t
$Path_i^t$, $Path^t$	Moving path of node i and set of all robots moving path from iteration $t-1$ to t
pCG^t	Position configuration, array of node positions in iteration t
rCG^t	Routing configuration, path set of data flows from an FN to the BS
CG^t	Configuration, $CG^t = \{pCG^t, rCG^t\}$
$\mathbb{CG}_v^t$	Set of all *valid* configurations in iteration t
IG_i^t	Information gain of node i in iteration t: number of unvisited cells to be explored
tIG^t	Total information gain
T^t, mT^t	Total time elapsed after iteration t; migration time in iteration t
γ in Equation 2	Bandwidth ratio
R_{capacity}	A uniform maximum link rate
R_s	A uniform flow sending rate from FN

new ones are also considered and modeled in the utility function.

- Relay node placement with routing path selection: How to place minimum number of RNs in explored area to satisfy the *Constraints I and II* and select which paths to route flows from FNs to BS.
- Position assignment and path generation: How to assign each robot with its target position and generate movement paths to minimize the bottleneck time mT^t.

Algorithm 1 illustrates the general procedure of CBAX for each iteration. Initially, n_{fn} is set as the team size because more FNs normally can cover more area. Given FN positions computed in line 3, RNs are placed accordingly with routing to relay FNs with the BS, and n_{rn} is decided. Line 5 checks whether the required number of robots $n_{\text{fn}} + n_{\text{rn}}$ exceeds the fixed team size. If yes, the algorithm reduces n_{fn} and places FNs and RNs again until it finds that the required number of robots can be satisfied. If $n_{\text{fn}} + n_{\text{rn}} < n$, then the remaining nodes may be able to

Algorithm 1: General procedure of connectivity and bandwidth-aware exploration.

Input: pCG^{t-1}, Map^{t-1}, T^{t-1}, Robot team size n

Output: CG^t, $Path^t$, mT^t, T^t, Map^t

1 $n_{\text{fn}} \leftarrow n$ *// "**tmp**" means "temporary" throughout this paper*

2 **while** $n_{\text{fn}} > 0$ **do**

3 $\quad \mathbb{FN}^t \leftarrow placeFN(Map^{t-1}, n_{\text{fn}})$ *// in Algorithm 2*

4 $\quad rCG^t, \mathbb{RN}^t, n_{\text{rn}} \leftarrow placeRN\_selectRoutingPath(\mathbb{FN}^t)$ *// in Algorithm 3*

5 $\quad$ **if** $n_{\text{fn}} + n_{\text{rn}} \leq n$ **then**

6 $\quad\quad$ **if** $n_{\text{fn}} + n_{\text{rn}} < n$ **then**

7 $\quad\quad\quad pCG^t \leftarrow placeRemainNode(pCG^{t-1}, pCG^t)$ *// in Algorithm 8*

8 $\quad\quad$ **break**

9 $\quad$ **else**

10 $\quad\quad$ reset $\mathbb{FN}^t, \mathbb{RN}^t, rCG^t, n_{\text{rn}}$

11 $\quad\quad n_{\text{fn}} \leftarrow n_{\text{fn}} - 1$

12 Generate migration paths with time: $mT^t, T^t, Path^t \leftarrow genPath(pCG^{t-1}, T^{t-1})$ *// in Algorithm 7*

13 $Map^t \leftarrow search(Map^{t-1}, CG^t)$

move to frontier areas to explore extra areas where RNs can support the additional flows. In line 12, function genPath is called to assure that bottleneck distance is minimized when generating movement paths for each node to reach its target. The map will be updated after the nodes arrive at their new positions and enter STI.

Connectivity and bandwidth-aware exploration is a centralized scheme. Although it takes time to synchronize and transmit control messages to the BS, it fits our model as global topology information is requested by the operator at the BS anyway for remote monitoring, control, and even teleoperation. More notably, it facilitates graph-theoretic modeling and methodically solves the problem. Furthermore, a distributed approach usually suffers from local suboptimal results, large amount of local data exchange [17], and non-trivial waiting time for other nodes to broadcast better results [2]. Therefore, CBAX computes the robot's trajectories at the powerful BS in a safe area to ensure effective control and monitoring.

4.2. Frontier robot placement

Algorithm 2: Frontier node placement *placeFN*().

Input: Map as a copy of Map^{t-1}; i as a copy of n_{fn}

Output: $\mathbb{FN}^t$

```
1 while i > 0 do
2     q* ← argmax_{q∈FC} U(q)
3     FN^t.add(q*)
4     Map ← updateLocalMap(q*, Map)
5     FC ← updateFrontierPosSet(Map, FC)
6     i ← i − 1
```

Our utility function is defined as follows:

$$\begin{aligned} U(q) &= IG_q^t \cdot e^{-\frac{d(q)}{\theta}} \\ \text{where } d(q) &= \min_{i \in pCG^{t-1}} d(q,i) \\ \theta &= \max\{\theta_0 \cdot (1-\delta), \theta_{\text{th}}\} \\ &\left(\delta = \frac{size(Map^{t-1})}{size(Map_{\text{total}})}\right) \end{aligned} \tag{4}$$

where δ is the *exploration ratio*, or explored area over total area. Notations relevant to Algorithm 1-8 are also presented in Table II. Empirically, θ_0 is set as 20, and θ_{th} (the threshold) is set as 12. When $\theta \to \infty$, distance is not considered, and FNs are greedily placed as the positions that cover most. On the other hand, when $\theta \to 0$, the distance is weighed heavily, so the frontier cell closest to the current positions pCG^{t-1} is selected. Thus, θ defined in Equation 4 enhances performance with initial emphasis on coverage while focusing on migrating time in the end to avoid unnecessary fluctuating moves. This approach resembles the candidate evaluation method in [19] but shows disparities in setting θ dynamically to adjust the distance's weight and defining distance as the minimum from one point to a set of positions.

This problem can also be viewed as a variant of the well-known NP-hard *set cover* problem [20] when $\theta \to \infty$. In the set cover problem, the minimum number of sets are selected so that all elements are contained, whereas in our problem, a fixed number of sets are selected with the goal that the maximum of elements are covered when assuming a *set* is the unvisited cell set by each frontier cell and an *element* as an unvisited cell. It is not difficult to prove that our problem is *no easier than* the set cover (thus is also NP-hard) because a polynomial solution to ours, if it ever exists, will lead to a polynomial solution to the set cover. To sum up, our solution inherits a greedy approximation of set cover, along with dynamic cost modeling of migrating distance.

4.3. Relay robot placement with routing path selection

Now that FNs are placed; we need to determine the following: (i) whether the rest $(n - n_{\text{fn}})$ of the nodes are adequate to relay the BS and the FNs and how to place them, and (ii) if the nodes are abundant, how to route the streams from each FN to the BS. We present two solutions with respect to situations considering different bandwidth ratio γ defined in Equation 2.

- When γ is sufficiently large, the first problem can also be viewed as a decision version of SMT-MSP where FNs and BS are the terminal points and RNs are the Steiner points. The routing problem can be easily solved afterwards.
- When γ is small and the bandwidth of aggregated flows may exceed link capacity, we model the problem as a decision version of SMT-MSP with the γ-*inflow* constraint and present our solution.

4.3.1. Bandwidth-sufficient relay node placement and routing.

The problem is modeled as the NP-hard problem of SMT-MSP, which was introduced and investigated in [6,7]. First we build a *complete graph*[†] G_{comp} of the BS and the FNs. In the next step, our solution is built on the well-known approximation algorithm called *Steinerized minimum spanning tree*(S-MST) for SMT-MSP, with an approximation ratio of 4 [7]. Interested readers may try more sophisticated schemes in [6] with ratios of 3 and 2.5. Because the ratio is only an upper bound compared with the optimal, the improvement in our application may not be that significant. When the RN is placed at an obstructed cell, *A* search* [21] is adopted to compute the obstacle-aware shortest path from one end (starting point) to the

[†]A complete graph is a simple graph in which every pair of distinct vertices is connected by an edge.

Algorithm 3: Relay node placement with routing selection *placeRN_RoutingPathSelection()*.

Input: pCG^t

Output: CG^t, n_{rn}

if isBandwidthSufficient **then**
  $n_{\text{rn}}, pCG^t \leftarrow$ *SteinerizedMST*($\mathbb{FN}^t$) // in *Algorithm 4*
  $rCG^t \leftarrow$ *BFS_RoutingPath*(pCG^t)
else
  $CG^t, n_{\text{rn}} \leftarrow$ *ConstrainedSteinerTree_Routing*($\mathbb{FN}^t$) // in *Algorithm 5*

Algorithm 4: Bandwidth-sufficient relay node placement: Steinerized minimum spanning tree *SteinerizedMST*().

Input: $\mathbb{FN}^t$

Output: pCG^t

1 Construct a complete graph G_{cmp} from $\{\mathbb{FN}^t \cup BS\}$
2 Construct a MST T_{mst} from G_{cmp}, $T \leftarrow T_{\text{mst}}$
3 $\mathbb{RN}^t.add($ non-leaf nodes in $T)$
4 **foreach** *edge* e *in* T *whose length* $l(e) > r_c$ **do**
5 Cut long edge e into $\lceil l(e)/r_c \rceil$ shorter ones of equal length by placing $\lceil l(e)/r_c \rceil - 1$ Steiner points $\{sp\}_e$ on e
6 **if** $\exists i(i \in \{sp\}_e \wedge P_i$=*obstructed)* **then**
7 Update e by: $e \leftarrow A^*Search(e)$
8 Place Steiner points again on e and update $\{sp\}_e$
9 $\mathbb{RN}^t.add(\{sp\}_e)$
10 Replace edge e with $|\{sp\}_e|$+1 shorter edges
11 $n_{\text{rn}} \leftarrow |\mathbb{RN}^t|$
12 $pCG^t \leftarrow \mathbb{FN} + \mathbb{RN}$

other end (goal) of the edge (line 7 in Algorithm 4), and the RN is placed again on the new path. We use the straight-line distance to the goal as the *admissible*[‡] and *consistent* heuristic function $h(n)$ that never overestimates the distance to goal. A* search guarantees the same optimal result as Dijkstra while reducing the searching space. With the RNs placed and the tree constructed, the shortest routing path (in terms of number of hops) from each FN to BS can be obtained by breadth-first search (line 3 in Algorithm 3). We are able to use breadth-first search because only the number of intermediate nodes counts, whereas the weight of each edge is not considered.

4.3.2. Bandwidth-constrained relay node placement and routing.

When the bandwidth ratio γ is small, the problem is modeled as SMT-MSP with the constraint that each node can admit no more than γ flows from FNs (FNs may also relay flows). Alternatively, if we define a node as *saturated* when it carries γ flows and as *over-saturated* when it carries more than γ ones, the problem becomes SMT-MSP without *over-saturated* nodes when flows transmit from $n-1$ terminal points to one terminal point (BS). The problem is *as hard as* SMT-MSP (thus is also NP hard) because if we can find a polynomial algorithm for it and we set $\gamma \to \infty$, SMT-MSP would be polynomially solvable.

The solution's central theme is to reduce unnecessary RNs by *aggregating flows*, and the following definitions relevant to flow aggregation are introduced. First, on top of the dynamic graph $G(t)$ of FNs and BS, a *hybrid flow graph* $G_f(t) = \{V, E(t) \cup \vec{E}(t)\}$ is defined where an undirected edge e denotes nodes in communication range, whereas a directed edge $\vec{e}$ represents not only connected vertices but also directed flow(s) from the FNs to the BS. Second, the *vector of carried flow source* for each $\vec{e}$, or $\mathbb{FS}_{\vec{e}}$, denotes the vector of sources of the flows passing through this edge. For each node i, $\mathbb{FS}_i$ gives the *carried and to-be relayed flows source vector* (sources here are the FNs that produce flows). Third, L_{cur} gives the *current layer* of nodes and L_{new} gives the *new layer*. Fourth, a *cluster head* (CH) is defined as the node that aggregates flows in each layer, and $\mathbb{CH}_{\text{cur}}$ denotes the CH vector of current layer nodes. Last, but not the least, $path(i, j)$ denotes the shortest obstacle-aware migrating path from i to j and $\mathbb{N}_i$ is node i's neighbor node set excluding L_{new} (j is a neighbor of i when $dist(i,j) \le r_c$).

Additionally, we define $\mathbb{CF}_i$ as the *vector of covered flow* of i, in which each element is a linked list that begins

[‡]Most admissible $h(n)$ are also consistent. Details of the proof of consistency is in [21].

Table II. Notations in Algorithms 1–8.

Terms	Definitions
δ	Exploration ratio; explored area compared with total area
$G_f(t)$	Hybrid flow graph, $G_f(t) = \{V, E(t) \cup \vec{E}(t)\}$
$dist(i, j)$	Straight-line distance from i to j
$path(i, j)$	Shortest obstacle-aware migrating path from i to j
L_{cur}, L_{new}	The current layer and the new layer
$\mathbb{N}_i$	Node i's neighbor node set excluding L_{new}, j is a neighbor of i when $dist(i, j) \leq r_c$
CH, $\mathbb{CH}_{\text{cur}}$, i_{ch}	Cluster head, vector of CH of current layer, and cluster head node i
$\mathbb{FS}_{\vec{e}}$, $\mathbb{FS}_i$	Vector of carried and to-be-relayed flow source passing through edge $\vec{e}$ and node i
$\mathbb{CF}_i$, $\mathbb{FCN}_i$	Covered flows (CF) and fully covered nodes (FCN) of node i; $\lvert\mathbb{CF}_i\rvert \leq \gamma$
$\mathbb{RL}_i^t$	Route list; sequence of nodes on routing path from FN i to BS in iteration t
$\mathbb{N}_{\text{BS}}^-$	Unsaturated one-hop neighbors to the BS
n_{remain}	Number of remaining nodes

with the node ID $j \in \mathbb{N}_i \cup i$ as the *head* and continues with sources from $\mathbb{FS}_j$ (An example has been provided in Figure 3. Note that $\mathbb{CF}_i$ depicts the potential flows that may be carried where $\mathbb{FS}_i$ describes the flows that have already been carried. A node $j \in \mathbb{N}_i \cup i$ is *fully covered* by node i when *all* of j's flows can be aggregated by i. $\mathbb{FCN}_i$ denotes the vector of i's *fully covered nodes* ($j \in \mathbb{FCN}_i$ iff $j \in \mathbb{N}_i \cup i \wedge \mathbb{FS}_j \subseteq \mathbb{CF}_i$). Note that $\lvert\mathbb{CF}_i\rvert \leq \gamma$ and $\lvert\mathbb{CF}_i\rvert$ only counts the source nodes. Each node must first carry its own flow if it is in L_{cur} or its parent's flow if it is an RN in L_{new}. Then, $\lvert\mathbb{FCN}_i\rvert - 1$ denotes that the number of extra RNs can be saved with such aggregation using the remaining capacity.

4.3.2.1. Flow aggregation. To minimize RN size, flow aggregation is executed in an order with maximal $\lvert\mathbb{FCN}_i\rvert$ first. The problem of finding maximal $\lvert\mathbb{FCN}_i\rvert$ is solved optimally by greedily covering minimal $\lvert\mathbb{FS}_i\rvert$ node first. For example, in Figure 3, new layer RN node n can maximally cover two extra nodes or save two RNs immediately otherwise needed by nodes 3 and 4. n can aggregate all flows from them: n's $\mathbb{FCN}_n = \{3, 4\}$ with the remaining capacity $9 - 4 = 5$ after it obtains its parent node 1's four flows.

4.3.2.2. Major strategy with illustration. Relay node placement and routing path selection are solved jointly in a layer-based approach, which is also described in Algorithm 5 and illustrated in an example in Figure 4. First, FNs $\mathbb{FN}^t$ are set as L_{cur}, and covered nodes/flows are computed for each node $i \in L_{\text{cur}}$ based on $\mathbb{FS}_i$, which is initialized as itself being the only element because a node carries its own flow (line 3). With L_{cur}, the algorithm is to compute L_{new} as the new RN until all current layer nodes are in range of BS, which is the terminating condition given in line 4.

Computing L_{new} is executed in two steps: (i) clustering nodes and aggregating flows in L_{cur} with CH selection and RN placement and (ii) aggregating flows into L_{new} with L_{new} RN filtering, which are given in lines 5–12 and 13–22, respectively.

In the first step, we repeatedly select CHs in L_{cur} and insert them into $\mathbb{CH}_{\text{cur}}$ to aggregate flow in the cluster in the order of 'maximal $\lvert\mathbb{FCN}_i\rvert$ first' (choose minimum distance to BS if $\lvert\mathbb{FCN}_i\rvert$ are equal because shorter distance indicates fewer needed RNs). Moreover, we place RNs when needed to form L_{new} in the paths from $\mathbb{CH}_{\text{cur}}$ to BS, and A* search is used similarly as in Algorithm 4 to avoid obstacles. Note that i_{ch} is set as the new RN's *parent*. Function *updateFlow_Route* (i) (line 15–19 in Algorithm 6) describes how a node i aggregates flows by moving elements in neighbor node j's $\mathbb{FS}_j$ to $\mathbb{FS}_i$ following i's covered flow $\mathbb{CF}_i$. Moreover, source s's $\mathbb{RL}_s^t$ is updated with the new RN appended. For instance, in Figure 4(a), FN c is chosen as CH because it is the closest to the BS among the ones with maximal $\lvert\mathbb{FCN}_i\rvert = 3$. $\mathbb{FS}_c$ is updated as

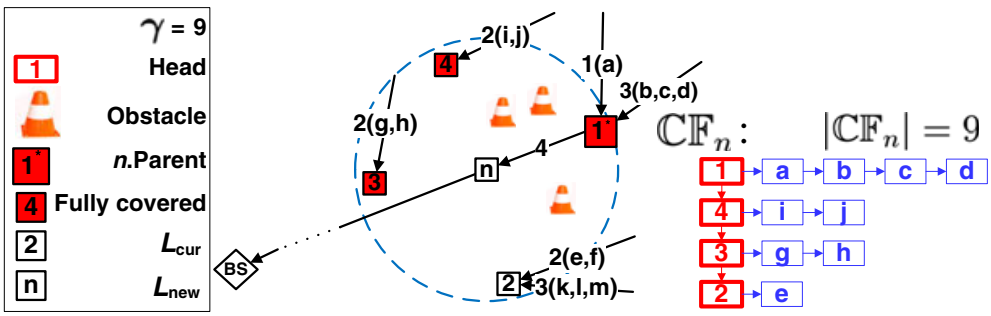


Figure 3. Aggregate flows to nodes with maximal number of fully covered neighbors first to conserve relay nodes.

Algorithm 5: Bandwidth-constrained relay node placement with routing path selection *ConstrainedSteinerTree_Routing*().

Input: $\mathbb{FN}^t$

Output: CG^t, n_{rn}

1 $L_{\text{cur}} \leftarrow \mathbb{FN}^t$, $L_{\text{new}} \leftarrow \emptyset$ // FNs as current layer

2 **foreach** $i \in L_{\text{cur}}$ **do**

3 $\quad$ *initialize*($\mathbb{FS}_i$), *setCoveredFlow_Node*($i, \mathbb{N}_i \cup i$, false)

4 **while** $\exists$ *node* $i (i \in L_{\text{cur}} \wedge dist(i, BS) > r_{\text{c}})$ **do**

5 $\quad L_{\text{cur}}^{\text{tmp}} \leftarrow L_{\text{cur}}$

6 $\quad$ **while** $L_{\text{cur}}^{\text{tmp}} \neq \emptyset$ **do**

7 $\quad\quad i_{\text{ch}} \leftarrow$ max $|\mathbb{FCN}_i|$ node $i \in L_{\text{cur}}^{\text{tmp}}$ with min $dist(i, BS)$ // Select cluster head

8 $\quad\quad \mathbb{CH}_{\text{cur}}$.*add*(i_{ch}), $L_{\text{cur}}^{\text{tmp}}$.*remove*($\mathbb{FCN}_{i_{\text{ch}}}$)

9 $\quad\quad$ *updateFlow_Route* (i_{ch}) // in Algorithm 6

10 $\quad\quad$ *placeRN_along_path* (i_{ch}) // Generate L_{new} in Algorithm 6

11 $\quad\quad$ **foreach** $i \in L_{\text{cur}}^{\text{tmp}}$ **do**

12 $\quad\quad\quad$ *setCoveredFlow_Node*($i, \mathbb{N}_i \cup i$, false) // in Algorithm 6

13 $\quad L_{\text{new}}^{\text{tmp}} \leftarrow L_{\text{new}}$

14 $\quad$ **foreach** $i \in L_{\text{new}}^{\text{tmp}}$ **do** *setCoveredFlow_Node*($i, \mathbb{N}_i$, true)

15 $\quad$ **while** $L_{\text{new}}^{\text{tmp}} \neq \emptyset$ **do**

16 $\quad\quad i^{\dagger} \leftarrow$ max $|\mathbb{FCN}_i|$ node $i \in L_{\text{new}}^{\text{tmp}}$ with max $|\mathbb{CF}_i|$

17 $\quad\quad$ **if** $|\mathbb{FS}_{i^{\dagger}\,\text{parent}}| = 0$ **then** L_{new}.*remove*($i^{\dagger}$) // Filtered

18 $\quad\quad L_{\text{new}}^{\text{tmp}}$.*remove*($i^{\dagger}$)

19 $\quad\quad$ *updateFlow_Route* ($i^{\dagger}$)

20 $\quad\quad$ **foreach** $i \in L_{\text{new}}^{\text{tmp}}$ **do** *setCoveredFlow_Node*($i, \mathbb{N}_i$, true)

21 $\quad$ Add L_{new} to $\mathbb{RN}^t$

22 $\quad$ Set L_{new} as L_{cur} ; Clear L_{new}

23 $n_{\text{rn}} \leftarrow |\mathbb{RN}^t|$, $pCG^t \leftarrow \mathbb{FN}^t + \mathbb{RN}^t$

24 **foreach** i **do** rCG^t.*add*($\mathbb{RL}_i^t$)

25 $CG^t \leftarrow pCG^t \cup rCG^t$

$\{c, b, d\}$, and c is added in $\mathbb{RL}_b^t$ and $\mathbb{RL}_d^t$ for sources b and d, respectively.

In the second step, CBAX executes flow aggregation with the same order of 'maximal $|\mathbb{FCN}_i|$ first' (choose the maximum size of covered flows if $|\mathbb{FCN}_i|$ are equal) for L_{new} and attempts to remove unnecessary nodes by filtering the new layer nodes whose parent's flows have all been carried by others. To illustrate in Figure 4(c), new layer RN_8 is first to aggregate flows with $|\mathbb{FCN}_{\text{RN\_8}}| = 2$, and then, RN_7 has $|\mathbb{FS}_{\text{RN\_3}}| = 0$ and is consequently filtered.

A layer placement ends when $L_{\text{new}}^{\text{tmp}}$ is empty (line 15). Then, L_{new} becomes L_{cur}, and a new round starts. To conclude, the flows, denoted by $\mathbb{FS}_i$ for node i , are aggregated in the CH in L_{cur} and then in L_{new} with best endeavor to reduce the number of RNs.

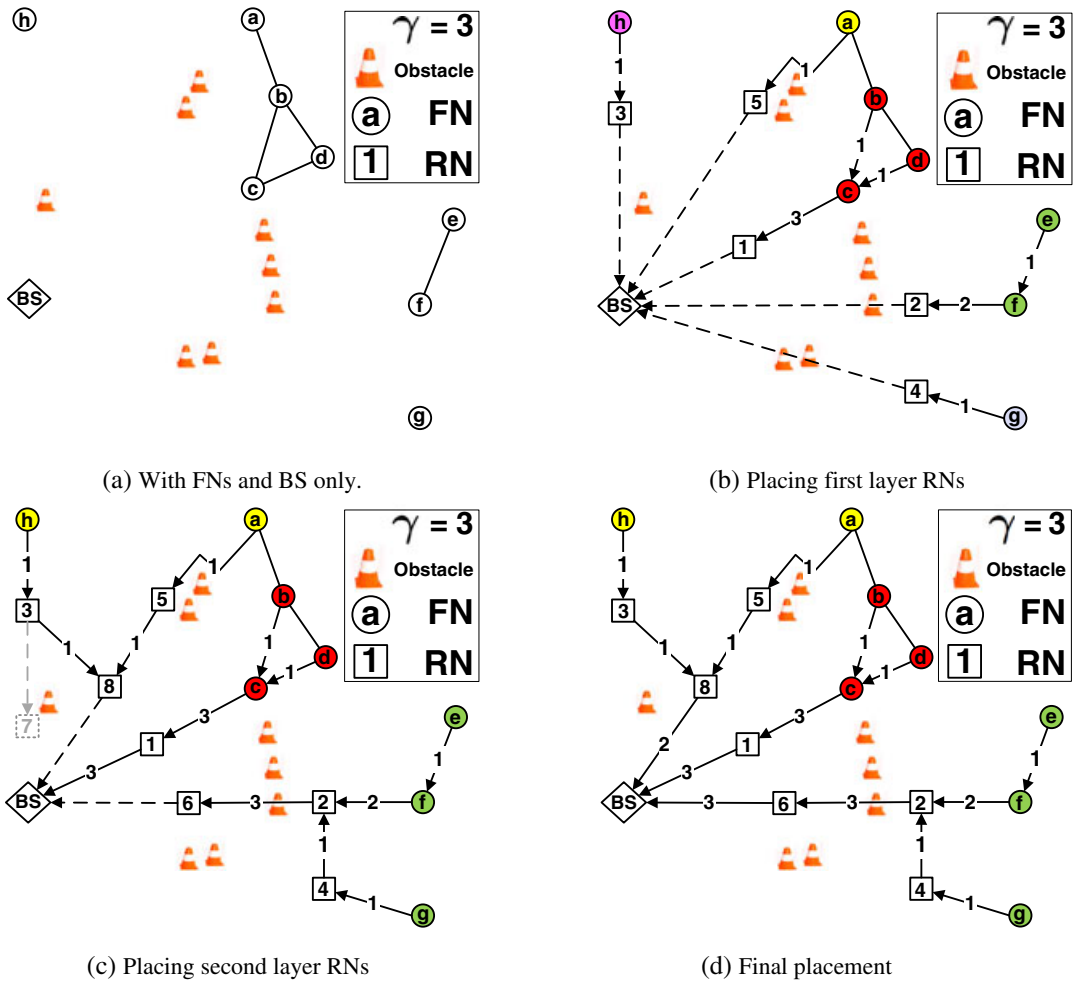
4.3.2.3. *Update flow information.*

In the beginning of the algorithm and after both flow aggregation in the two steps, function *setCoveredFlow_Node* (lines 1–14 in Algorithm 6) is called to update the covered flows and fully covered node information. Furthermore, the rest of the nodes still in $L_{\text{new}}^{\text{tmp}}$ are updated in line 20, and these updates are necessary for the next round computation. With $\gamma \rightarrow \infty$, this flow aggregation-based relay placement algorithm also fits the sufficient bandwidth situation.

4.3.2.4. *Upper bound of approximation ratio.*

We derive the upper bound of the approximation ratio of the bandwidth-constrained relay placement algorithm.

First, we define some notations. A maximum degree in a graph G is: $\Delta(G) = \max\{deg_G(v) | v \in V(G)\}$. We denote n, n_{s}, n_{d}, and n_{opt} are the respective relay size obtained by the following: (i) flow constrained relay placement algorithm; (ii) S-MST without the flow constraint; (iii) direct tree (DT) or a simple aggregation tree (star graph with center BS) where all flows from the FNs are directly relayed to the BS without aggregation; and (iv) T_{opt}, an optimal



(a) With FNs and BS only.

(b) Placing first layer RNs

(c) Placing second layer RNs

(d) Final placement

Figure 4. Example of bandwidth-constrained relay node placement and routing. ($\gamma = 3$) Edge weight denotes $|\mathbb{FS}_{\vec{e}}|$. BS, base station; FN, frontier node; RN, relay node.

Steiner tree with flow constraint and a minimum number of RN.

In addition, MST denotes the minimum spanning tree for FNs and BS. DT and MST are two very simple tree structures easily obtained when given the input of FNs and BS. We define $\kappa = n_d/n_s = \sum_{e\in E(\mathrm{DT})}\lfloor e/r_c\rfloor / \sum_{e\in E(\mathrm{MST})}\lfloor e/r_c\rfloor$, where κ is the ratio of the relay size for DT over that of MST. Hence, κ is related to $\mathbb{FN}$ and can be directly obtained from input.

Lemma 1. *Every Steiner tree satisfies the flow constraint must have its maximum degree* $\Delta(T) \le 1 + \gamma$.

Proof. For a vertex, its incoming degree is constrained by γ, the number of incoming flows for aggregation. With only one out-coming link, the total degree is at most $\Delta(T) \le 1 + \gamma$. □

Lemma 2. *There exists a shortest optimal Steiner tree* T_{opt} *whose maximum degree is less than or equal to five, that is,* $\Delta(T_{\mathrm{opt}}) \le 5$, *in an obstacle-free area or obstructive area where triangular inequality holds.*

Proof. When the obstacle distribution does not impact the triangular inequality, with A* obstacle-aware paths, the shortest side is still across from the smallest angle. Then, we can prove that two edges meeting at a vertex in a T_{opt} form an angle of at least 60°, by the proof similar to that in [7] on an Euclidean plane. Besides, authors in [7] also proved that an optimal tree with maximum vertex degree of six can lead to another optimal tree with maximum vertex degree of five. Therefore, $\Delta(T_{\mathrm{opt}}) \le 5$. □

Theorem 1. *The approximation ratio of bandwidth-constrained relay placement algorithm is no greater than* $\kappa \cdot (1 + \gamma)$. *In an environment where obstacle distribution does not affect the triangular inequality, the ratio is no greater than* $\kappa \cdot \min\{5, 1 + \gamma\}$.

Algorithm 6: Supporting functions for Algorithm 5.

// Obtain (pre)knowledge on covered flows ($\mathbb{CF}$) and fully covered nodes ($\mathbb{FCN}$); not to transmit flows yet.

setCoveredFlow_Node (node i, node set S, bool *isNewRN*):
$\mathbb{CF}_i.clear(), \mathbb{FCN}_i.clear()$
if *isNewRN =true* **then**
  $S.remove(i.parent)$
  $\mathbb{CF}_i.add(i.parent \rightarrow \mathbb{FS}_{i.parent}), \mathbb{FCN}_i.add(i.parent)$ *// Obtain all of its parent's flows*
else
  $S.remove(i)$
  $\mathbb{CF}_i.add(i \rightarrow \mathbb{FS}_i), \mathbb{FCN}_i.add(i)$ *// Obtain all of its own flows*
while $|\mathbb{CF}_i| < \gamma \wedge (\exists$ *node* $k \in S$ *whose* $|\mathbb{FS}| \neq 0)$ **do**
  $j \leftarrow$ such node k with min $|\mathbb{FS}|$
  $S.remove(j)$
  while $|\mathbb{CF}_i| < \gamma$ **do**
    $\mathbb{CF}_i.add(j \rightarrow$ distinct source $k \in \mathbb{FS}_j)$ *// $\mathbb{CF}_i.j \rightarrow \{k\}$*
  if $\mathbb{FS}_j \subseteq \mathbb{CF}_i$ **then** $\mathbb{FCN}_i.add(j)$ *// Fully covered node*

// To transmit flows.

updateFlow_Route (node i): *// Function begins*
foreach *neighbor (head in* $\mathbb{CF}$*)* $j \in \mathbb{CF}_i.* \wedge j \neq i$ **do**
  foreach *FN* $s \in$ *source nodes for flows carried by* j **do**
    Add i as a relay for FN s in $\mathbb{RL}_s$
  Transmit j's flows from j to i by updating their $\mathbb{FS}$

placeRN_along_path (i): *// Function begins*
if *dist(i,BS)*$> r_c$ **then** *// Place RN if needed*
  Place relay rn at $dist(rn, i) = r_c$ on the obstacle aware path from i to BS generated by A* search(i, BS)
  Set rn's parent as i; Add rn to L_{new}.

Proof. With an optimal tree T_{opt}, we duplicate each edge and build an Eulerian tour such that every Steiner point appears at most $\Delta(T_{\text{opt}})$ times in the tour, whereas each terminal point appears exactly once. Removing one (or more) edge in the tour will give a spanning tree T' where Steiner points have 2° at most. Its Steiner point size $n(T')$ is no more than $\Delta(T_{\text{opt}})$ times in T_{opt}. Because n_s is from S-MST, a *minimum* spanning tree, we have

$$n_s \leq n(T') \leq \Delta(T_{\text{opt}}) \cdot n_{\text{opt}} \tag{5}$$

With $\kappa = n_d/n_s$ and the fact that flow aggregation guarantees $n \leq n_d$, we have

$$n \leq n_d \leq \kappa \cdot \Delta(T_{\text{opt}}) \cdot n_{\text{opt}} \tag{6}$$

$$\frac{n}{n_{\text{opt}}} \leq \kappa \cdot \Delta(T_{\text{opt}}) \tag{7}$$

With Lemmas 1 and 2, $\Delta(T_{\text{opt}}) = \min\{5, 1 + \gamma\}$ in obstacle-free area or obstructive area where triangular inequality holds. Otherwise, $\Delta(T_{\text{opt}}) = 1 + \gamma$. Replacing $\Delta(T_{\text{opt}})$ concludes the proof. □

4.4. Position assignment and path generation

With current positions pCG^{t-1} from the execution of the last iteration of n robots and the computed n next positions pCG^t, there is a problem with how to assign each robot to go to which newly computed position so that mT^t, which is decided by the bottleneck longest distance of all pairs, is minimum. We formally model this problem as the LBAP [22]. LBAP can also be viewed as finding a perfect match in a weighted bipartite graph that minimizes the maximum weight of all matched edges [23]. Mathematically, LBAP is

$$mT^t = \min_{x_{ij} \in \{0,1\}} \max_{i,j} \quad c_{ij} x_{ij} \tag{8}$$

$$\text{subject to} \sum_{i=1}^{n} x_{ij} = 1, \quad j = 1, \ldots, n$$

$$\sum_{j=1}^{n} x_{ij} = 1, \quad i = 1, \ldots, n$$

$$x_{ij} \in \{0, 1\}, \quad i, j = 1, \ldots, n$$

where $\{c_{ij}\}_{n\times n}$ is the cost matrix and c_{ij} is the cost as the length of shortest obstacle-aware path by A* from node i to position j. Moreover, $\{x_{ij}\}_{n\times n}$ is the resulting binary matrix where $x_{ij} = 1$ iff the node i is assigned to position j. We adopt the algorithm in [24] with expected running time $O(n^2)$, which generates the *optimal* result. Its general procedure is as follows: (i) from the original bipartite graph, a subgraph with $2n \ln n$ minimum-cost edges is selected; (ii) in this subgraph, LBAP is solved by scheme in [25]; and (iii) if a perfect matching is not found with a low probability, then the matching is done in the original graph. According to [23] and [26], it is one of the most efficient algorithms for LBAP.

Because the paths obtained may not be collision free, we adopt a simple scheduling to avoid collision (lines 6–9 in Algorithm 7). First, we check whether the intersection points of the paths exist. If true, then we will check whether the intersection is reached by different nodes simultaneously. If true again, collision may occur and different nodes may wait different amounts of time; for example, node i waits i second(s) before reaching the intersection place to avoid such collision.

4.5. CBAX enhancement

Besides the aforementioned major steps in CBAX, we improve the control message transmission and use remaining nodes for placement to further enhance CBAX's performance.

4.5.1. Filter-based control message transmission.

In each iteration, a node needs to notify the BS by sending an 'arrived' control message when arriving at the target position. Because a node in movement indicates that it will send to the BS its 'arrived' message in the future, the communication overhead can be reduced by not forwarding packets to the BS if a moving node j receives the 'arrived' message from a stopped (arrived) node i. When j also arrives at the targeting position, j can append i's information in j's 'arrived' message. The completion of the *last* mobile node's movement indicates the completion of all nodes in one iteration. Besides, it is possible that when an arrived node sends the message, the intermediate nodes on the path back to the BS are still moving. They may be out of reach of the sender and therefore not able to relay the message. Under such occasions, the arrived node will retransmit the packet on timeout until the BS sends an acknowledgement to confirm that it has received the message.

4.5.2. Remaining node placement.

When $n_{\text{fn}} + n_{\text{rn}} < n$ (lines 6–7 in Algorithm 1), we need to identify the remaining node positions to maximize the exploration efficiency. Generally, we attempt to place nodes as FNs using the unsaturated paths to carry flows to the BS. First, we identify the unsaturated one-hop neighboring nodes of BS $\mathbb{N}^{-}_{\text{BS}}$, which have extra capacity not yet used. A node i is unsaturated when $|\mathbb{FS}_i| < \gamma$. Second, to place the remaining nodes as FNs, we collect the cells in $\mathbb{FC}$, which are in range of the flow source nodes of $\mathbb{N}^{-}_{\text{BS}}$, to form $\mathbb{FC}^*$. The remaining node positions are selected as the cells with the maximum gain in $\mathbb{FC}^*$. When there are no unsaturated paths, we place the remaining node using the previous iteration position in pCG^{t-1}, which is closest to the nearest node in pCG^t. Such placement avoids the remaining node relocation to be the bottleneck in assignment. Algorithm 8 shows the procedure in detail.

5. PERFORMANCE EVALUATION

We evaluated CBAX's performance using a multi-robot simulator in C++ and compared CBAX with two recent

Algorithm 7: Linear bottleneck assignment problem-based position assignment and path generation with collision avoidance *genPath*().

Input: $pCG^{t-1}, pCG^t, T^{t-1}$

Output: $mT^t, T^t, Path^t$

1 Construct complete bipartite graph $G_{cb}= (pCG^{t-1} + pCG^t, E)$

// update edge weight as obstacle-aware ones

2 **foreach** $e \in E_{G_{cb}}$ **do** $e \leftarrow$ A* search(e)

// Call bipartite graph and perfect matching based LBAP() in [24]

3 $mT^t, Path^t \leftarrow$ *graphBasedLBAP*(G_{cb})

4 $mT^t, Path^t \leftarrow$ *avoidCollision*($mT^t, Path^t$)

5 $T^t \leftarrow T^{t-1} + mT^t$ *// Algorithm ends*

6 *avoidCollision*($mT^t, Path^t$): *// Function begins*

7 **if** *IntersectionExists*($Path^t$) **then**

8 **if** *SameTimeIntersectionExists*($Path^t$) **then**

9 $mT^t \leftarrow$ *RescheduleCollisionNodes*(mT^t)

schemes that consider limited communication range: (i) a distributed 'sensor-based random graph' (SRG) scheme in [1], which is more efficient but does not guarantee connectivity and (ii) a centralized 'possible moves sampling' (PMS) approach in [3] that assures connectivity but is less efficient. The metrics for comparison include exploration efficiency and communication quality. We used the total exploration time as the primary criteria for efficiency. If not stated otherwise, the time is measured from a clustered start at the left–bottom corner to a state with over 95% of explored area. Also, the constant STI in each iteration is not included in the measured time for the three schemes. To evaluate the communication quality, the *non-overflow transmission time ratio* (NO_TT), or the percentage of transmission time for not over-saturated flows divided by the total transmission time, is used. Besides, the percentage of iterations in which all robots maintain connected with BS and within the robot team, or *fully Connected time ratio* with the BS (**CT_BS**) and within the robot team (**CT_RT**), are also measured. Two environments, the *open* and *garden* in Figure 5(a), are used to compare the schemes.

The parameters are set to make the comparisons fair. We obtained the results and parameters from SRG's sample video in [27] and simulated CBAX using the same set of parameters (with proper scaling) and the same *garden* environment. Although teams with only four, six, and eight robots are tested in SRG, we simulated PMS and compare with CBAX on 6–12 robots to obtain results for more complex topologies using the *open* environment. Specifically, the grid cell and robot size are 1 m, the robot's moving velocity is 1 m/s, and the accelerating/decelerating time is ignored. Map sizes for the *open* and *garden* environments are 45 m × 45 m and 48 m × 48 m, respectively, and (r_c, r_s) are (10 m,7 m) and (21 m,7 m), respectively. All experiments are run five times to obtain the average. For CBAX, two relay placement strategies were evaluated: CBAX with bandwidth-sufficient relay placement (BS-CBAX) and CBAX with bandwidth-constrained relay placement (BC-CBAX) ($\gamma = 3$).

5.1. Exploration efficiency

Compared with PMS, BS-CBAX and BC-CBAX reduce explore time on average by 40.9% and 39.7%, respectively, as shown in Figure 6(a). The improvement is more significant when team size increases. With 12 robots, the improvements are 54.7% and 45.3%. The major reason is that CBAX expends nodes promptly and systematically places nodes to reduce relay robots while in PMS, nodes are expended slowly especially when the number of nodes increases. In addition, Figure 5(b) shows that CBAX is more efficient than PMS with different percentages of explored area. Compared with SRG, CBAX remains the more efficient approach, even though SRG is a distributed and efficient scheme maintaining neither connectivity nor bandwidth adequacy. BS-CBAX and BC-CBAX decrease exploration time on average by 15.9% and 13.0%, respectively, as shown in Figure 6(b). With eight robots, the improvements are 27.4% and 22.2%. In most cases, BC-CBAX is slightly more efficient than BS-CBAX when n is small but is less efficient when more robots are in the team. The reasons are as follows: (i) our flow aggregation-based

Algorithm 8: Remaining node placement *placeRemainNode*().

Input: pCG^{t-1}, pCG^t

Output: pCG^t

$n_{\text{remain}} \leftarrow n - (n_{\text{fn}} + n_{\text{rn}})$

// Put unsaturated and close to BS's nodes in pCG^t *to* $\mathbb{N}^-_{\text{BS}}$

foreach $i \in pCG^t \wedge dist(i, BS) \leq r_c \wedge |\mathbb{FS}_i| < \gamma$ **do** $\mathbb{N}^-_{\text{BS}}.add\ (i)$

while $|\mathbb{N}^-_{\text{BS}}| \neq 0 \wedge n_{\text{remain}} > 0$ **do**

  $\mathbb{FC}^* \leftarrow$ Frontier cells that are in range of flow sources of $\mathbb{N}^-_{\text{BS}}$

  foreach $q \in \mathbb{FC}$ **do** Store corresponding node pair (i, j) of BS's neighbor i and flow source j: $\{i, j | i \in \mathbb{N}^-_{\text{BS}}, j \in \mathbb{FS}_i \wedge dist(q, j) \leq r_c\}$

  $q^* \leftarrow \text{argmax}_{q \in \mathbb{FC}^*} U(q)$; add q^* to pCG^t.

  $i^*, j^* \leftarrow q^*$'s corresponding i, j

  $Map \leftarrow$ *updateLocalMap*(q^*, Map)

  $\mathbb{FC} \leftarrow$ *updateFrontierPosSet*$(Map, \mathbb{FC})$

  Add $j^* \cup \mathbb{RL}^t_{j^*}$ to q^*'s route list

  for $k \in \mathbb{RL}^t_{q^*}$ **do** update $\mathbb{FS}_k$

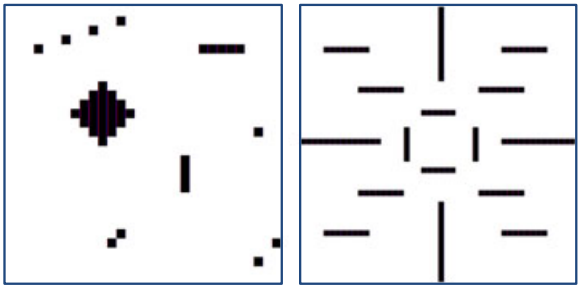
  if $|\mathbb{FS}_{i^*}| = \gamma$ **then** $\mathbb{N}^-_{\text{BS}}.remove(i^*)$

  $n_{\text{remain}} \leftarrow n_{\text{remain}} - 1$

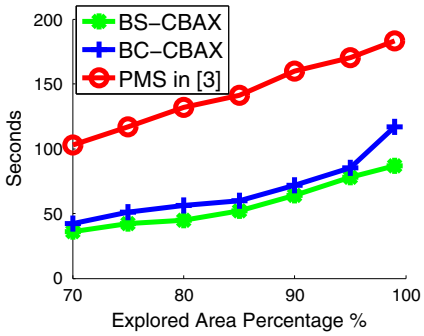
while $n_{\text{remain}} > 0$ **do** *// No unsaturated paths*

  $q^* \leftarrow$ node $q \in pCG^{t-1}$ with min distance to the nearest node in pCG^t

  $pCG^t.add\ (q^*)$

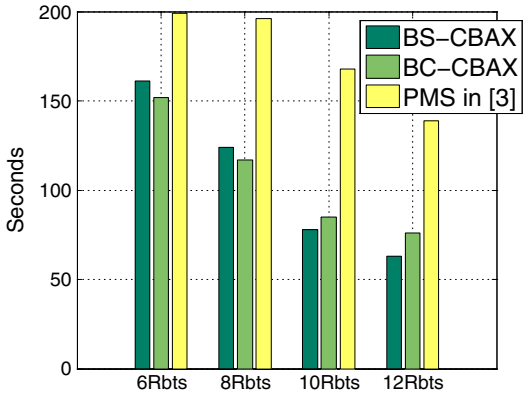


(a) The *Open* and *Garden* environments where black parts denote obstacles.

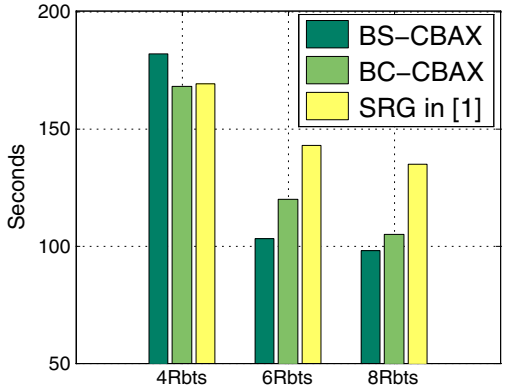


(b) Exp. time: CBAX versus PMS in [3] with varying percentages of explored area (10 Robots).

Figure 5. Environments and exploration time. BC-CBAX, connectivity and bandwidth-aware exploration with bandwidth-constrained relay placement; BS-CBAX, CBAX with bandwidth-sufficient relay placement; PMS, 'possible moves sampling' approach.



(a) Exp. time: CBAX versus PMS in [3] with 6, 8, 10, and 12 of robots in the *open* environment.



(b) Exp. time: CBAX versus SRG in [1] with 4, 6, and 8 robots in the *garden* environment.

Figure 6. Simulation results on exploration time. BC-CBAX, connectivity and bandwidth-aware exploration with bandwidth-constrained relay placement; BS-CBAX, CBAX with bandwidth-sufficient relay placement; PMS, 'possible moves sampling' approach; SRG, sensor-based random graph.

solution is more efficient than the general SMT-based approximation algorithm, and (ii) the flows are less likely to be overflowed ($\gamma = 3$) when node number is small. With more robots, however, the impact of the constraint is more noticeable, resulting in longer exploration time with more RNs placed.

5.2. Communication quality

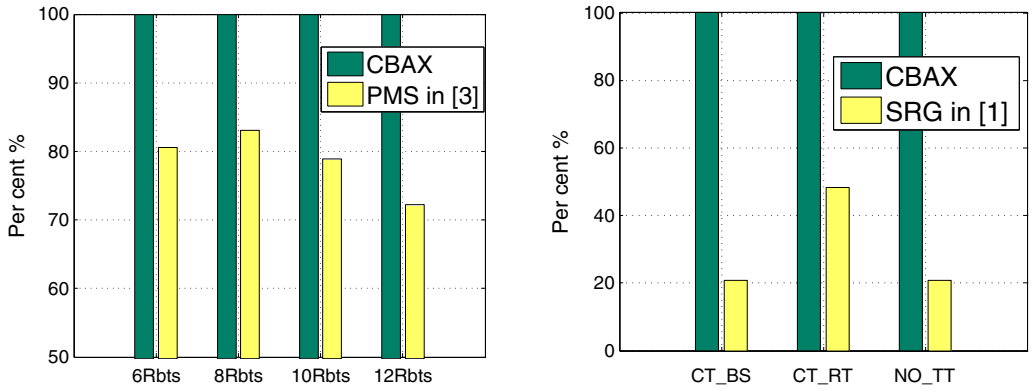
Compared with PMS that only guarantees connectivity, CBAX further assures the real-time data flow of being under link capacity. In PMS, overflow transmissions occur in 21.5% of the iterations on average, in which 31.5% of data is lost. Although CBAX always maintains connectivity and ensures no overflow occur, SRG with four robots[§] shows that only in 20.7% of iterations, all nodes are (through multi-hops) connected to the BS and 48.3% of iterations all nodes are connected within the robot team. Besides, in 79.3%[¶] of iterations, overflow transmission occurs, and 83.7% of data is lost in such overflow sessions. Figure 7 also depicts the results.

5.3. Discussion

A mobile multi-hop wireless network's quality of service (QoS) is a complex issue affected by various factors such

[§]Video available online is only with four robots. However, the results with six out of eight robots are similar because SRG is not aware of connectivity and bandwidth.

[¶]We define a disconnected link has link capacity 0 bit/s.



(a) **NO_TT**: CBAX versus PMS in [3] with 6,8,10,12 of robots in the *open* environment.

(b) **CT_BS**, **CT_RT**, and **NO_TT**: CBAX versus SRG in [1] with 4 robots in the *garden* environment.

Figure 7. Simulation results on communication quality. CBAX, connectivity and bandwidth-aware exploration; PMS, 'possible moves sampling' approach; SRG, sensor-based random graph.

as underlying MAC protocol, interference, channel quality, scheduling, congestion control, and routing path selection. Besides, the contention-based carrier sense multiple access with collision avoidance MAC protocol in 802.11 may not guarantee an absolute link rate and throughput. However, topology and node movement significantly influence the performance of the network protocol in mobile *ad hoc* networks [28] and with our QoS-aware mobility model, we provide the most fundamental assurance of bandwidth adequacy. To further enhance rate control, we can employ the time division multiple access-based scheduling to guarantee the required bandwidth for each flow on a certain link.

6. CONCLUSION AND FUTURE WORK

Mobility and communication are two critical and interdependent topics, and we believe that jointly considering the two aspects is essential in many research fields. From the mobile robotics viewpoint, we leverage algorithmic and graph-theoretic tools to systematically solve the connectivity and bandwidth-aware multi-robot exploration problem. CBAX achieves both improved efficiency in exploration time and enhanced quality for communication. From the multi-hop mobile network perspective, CBAX demonstrates how a pragmatic coordinated mobility model can ensure the network connectivity and enhance its QoS. Future work may include the following: (i) implementing CBAX in a real-robot test bed and (ii) considering heterogeneous communication ranges and data rates along with the impact of interference.

ACKNOWLEDGEMENT

This work is supported in part by Natural Sciences Foundation under Grant Nos. OCI-0753362 and CNS-0721441. The preliminary version of this work appeared in the *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA) 2010* at Anchorage, Alaska, USA.

REFERENCES

1. Franchi A, Freda L, Oriolo G, Vendittelli M. The sensor-based random graph method for cooperative robot exploration. *IEEE/ASME Transactions on Mechatronics* 2009; **14**(2): 163–175.
2. Sheng W, Yang Q, Tan J, Xi N. Distributed multi-robot coordination in area exploration. *Robotics and Autonomous Systems* 2006; **54**(12): 945–955.
3. Rooker MN, Birk A. Multi-robot exploration under the constraints of wireless networking. *Control Engineering Practice* 2007; **15**(4): 435–445.
4. Sugiyama H, Tsujioka T, Murata M. Coordination of rescue robots for real-time exploration over disaster areas. In *Proceedings of the 11th IEEE International Symposium on Object Oriented Real-time Distributed Computing (ISORC '08),* Orlando, FL, USA, 05–07 May 2008. IEEE Computer Society: Washington, DC, 2008; 170–177.
5. Cen Z, Mutka M, Liu Y, Goradia A, Xi N. QoS management of supermedia enhanced teleoperation via overlay networks. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '05),* Edmonton, Alberta, 2–6 Aug 2005. IEEE Computer Society: Washington, DC, 2005; 1630–1635.
6. Cheng X, Du D, Wang L, Xu B. Relay sensor placement in wireless sensor networks. *Wireless Networks* 2008; **14**(3): 347–355.

7. Du D, Hu X. Steiner tree problem for minimal steiner points, chapter 5. *Steiner Tree Problems in Computer Communication Networks*. World Scientific Publishing Co.: Singapore, 2008; 177–193.
8. Zavlanos M, Pappas G. Distributed connectivity control of mobile networks. *IEEE Transactions on Robotics* 2008; **24**(6).
9. Michael N, Zavlanos MM, Kumar V, Pappas GJ. Maintaining connectivity in mobile robot networks. In *International Symposium on Experimental Robotics (ISER '08),* Athens, Greece, 13–16 July 2008. Springer-Verlag: Berlin, Heidelberg, 2008; 117–126.
10. Birk A, Schwertfeger S, Pathak K. A networking framework for teleoperation in safety, security, and rescue robotics. *IEEE Wireless Communications, Special Issue on Wireless Communications in Networked Robotics* 2009; **16**(1): 6–13.
11. Zeiger F, Kraemer N, Schilling K. Commanding mobile robots via wireless ad-hoc networks—a comparison of four ad-hoc routing protocol implementations. In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA '08),* Pasadena, CA, 19–23 May 2008. IEEE Computer Society: Washington, DC, 2008; 590–595.
12. Wurm K, Stachniss C, Burgard W. Coordinated multi-robot exploration using a segmentation of the environment. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '08),* Nice, France, 22–26 September 2008. IEEE Computer Society: Washington, DC, 2008; 1160–1165.
13. Rocha R, Ferreira F, Dias J. Multi-robot complete exploration using hill climbing and topological recovery. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '08),* Nice, France, 22–26 September 2008. IEEE Computer Society: Washington, DC, 2008; 1884–1889.
14. Vazquez J, Malcolm C. Distributed multirobot exploration maintaining a mobile network. In *Proceedings of the Second International IEEE Conference on Intelligent Systems,* Varna, Bulgaria, 22–24 June 2004, Vol. 3. IEEE Computer Society: Washington, DC, 2004; 113–118.
15. Amigoni F. Experimental evaluation of some exploration strategies for mobile robots. In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA '08),* Pasadena, CA, 19–23 May 2008. IEEE Computer Society: Washington, DC, 2008; 2818–2823.
16. Mosteo A, Montano L, Lagoudakis M. Multi-robot routing under limited communication range. In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA '08),* Pasadena, CA, 19–23 May 2008. IEEE Computer Society: Washington, DC, 2008; 1531–1536.
17. Bekris KE, Tsianos KI, Kavraki LE. A distributed protocol for safe real-time planning of communicating vehicles with second-order dynamics. In *Proceedings of the First International Conference on Robot Communication and Coordination (RoboComm '07),* Odense, Denmark, 31 March–2 April 2009. IEEE Press: Piscataway, NJ, 2007; 15–17.
18. Guo W, Huang X. Mobility model and relay management for disaster area wireless networks. In *Proceedings of the Third International Conference on Wireless Algorithms, Systems, and Applications (WASA '08), Dallas, TX, USA, 26–28 October 2008*. Springer-Verlag: Berlin, Heidelberg, 2008; 274–285.
19. Gonzáles-Baños HH, Latombe J-C. Navigation strategies for exploring indoor environments. *International Journal of Robotics Research* 2002; **21**(10–11): 829–848.
20. Cormen TH, Leiserson CE, Rivest RL, Stein C. Approximation algorithms, chapter 35. *Introduction to Algorithms, 2nd Edition*. The MIT Press: Cambridge, MA, 2001; 1056–1061.
21. Russell S, Norvig P. Informed search methods, chapter 4. *Artificial Intelligence: a Modern Approach, 2nd Edition*. Prentice-Hall: Upper Saddle River, NJ, 2003; 97–101.
22. Burkard R, Dell'Amico M, Martello S. Other types of linear assignment problems, chapter 6. *Assignment Problems*. SIAM: Philadelphia, PA, 2009; 172–191.
23. Krokhmal P, PM P. Random assignment problems. *European Journal of Operational Research* 2009; **194**(1): 1–17.
24. Pferschy U. The random linear bottleneck assignment problem. *RAIRO-Operations Research* 1996; **30**(2): 145–156.
25. Gabow H, Tarjan R. Algorithms for two bottleneck assignment problems. *Journal of Algorithms* 1988; **9**(3): 411–417.
26. Burkard R, Cela E. Linear assignment problems and extensions. In *Handbook of Combinatorial Optimization*, Vol. A (Suppl.) Kluwer Academic Publishers: Dordrecht, 1999; 75–149.
27. Franchi A, Freda L, Oriolo G, Vendittelli M. SRG-based video clips from a cluster start with 4 robots. http://www.dis.uniroma1.it~labrob/research/multiSRG.html.2007. [Accessed on June 2009].
28. Jardosh A, Belding-Royer EM, Almeroth KC, Suri S. Towards realistic mobility models for mobile ad hoc networks. In *Proceedings of the Ninth ACM International conference on Mobile Computing and networking (MobiCom '03),* San Diego, CA, 14–19 September 2003. ACM: New York, NY, 2003; 217–229.

AUTHORS' BIOGRAPHIES

Yuanteng Pei received the B.Eng. degree in software engineering from Wuhan University, China, in 2007. He has been an exchange student in computer science at City University of Hong Kong in 2005. He is currently a Ph.D. candidate in computer science at Michigan State University. His research interests include wireless networking and mobile computing. He is currently working on QoS support, remote sensing and control, motion and path planning on wireless mobile networks. He is a student member of the IEEE.

Matt Mutka received the B.S. degree in electrical engineering from the University of Missouri–Rolla, the M.S. degree in electrical engineering from Stanford University, and the Ph.D. degree in computer science from the University of Wisconsin–Madison. He is on the faulty of the Department of Computer Science and Engineering, Michigan State University, East Lansing, MI, where he is currently professor and department chairperson. He has been a visiting scholar at the University of Helsinki, Helsinki, Finland, and a member of technical staff at Bell Laboratories in Denver, CO. His current research interests include mobile computing, wireless networking, and multimedia networking.

Ning Xi received his D.Sc. degree from Washington University in St. Louis, MO, in December, 1993. Currently, he is the John D. Ryder professor of Electrical and Computer Engineering at Michigan State University. Xi received The Best Paper Award of IEEE Transactions on Automation Science and Engineering in 2007. He was also awarded SPIE Nano Engineering Award in 2007. In addition, he is also a recipient of US National Science Foundation CAREER Award. Dr. Xi is a fellow of IEEE. Currently, his research interests include robotics, manufacturing automation, micro/nano manufacturing, nano sensors and devices, and intelligent control and systems.