Efficient Path Rescheduling of Heterogeneous Mobile Data Collectors for Dynamic Events in Shanty Town Emergency Response

Ranga Raj*, Sarath Babu†, Kyle Benson*, Gaurav Jain†, B. S. Manoj†, and Nalini Venkatasubramanian*
*University of California Irvine, CA 92697
Email: ranga.raj@uci.edu, {kebenson, nalini}@ics.uci.edu
†Indian Institute of Space Science and Technology, Thiruvananthapuram, India 695547
Email: sarath.babu.2014@ieee.org, gaurav.jain14@hotmail.com, bsmanoj@ieee.org

***Abstract*—To investigate reported emergency incidents and provide better situational awareness during an emergency response effort in a shanty town, we envision the use of volunteers with networked sensing devices employed as Mobile Data Collectors (MDCs). These MDCs are heterogeneous depending upon the type of roads they can access. They gather information about events reported dynamically at random and relay it to a central command center. We consider the problem of minimizing the Travel Time of such heterogeneous volunteer MDCs and maximizing the gathering of event data before its expiry time. We model this problem as a Dynamic Vehicle Routing Problem with Time Windows (DVRPTW), which reduces to a Combinatorial Optimization Problem and is NP-Hard to solve. In this paper, we developed two algorithms, Minimum Deviated Walk and Ortho Walk, to dynamically route or reroute the path of these MDCs to capture the data efficiently. We tested the effectiveness of these algorithms with three different classes of MDCs on simulated non-deterministic random sets of events applied to a real road map of Dharavi, a shanty town in Mumbai, India. We show that both these algorithms are capable of capturing 20% more data than a naive algorithm as well as more than 90% of the events generated within a specified time.**

I. Introduction

Shanty towns are results of rapid unplanned urbanization in less economically developed countries. The size of the vehicles that can navigate the streets in the town can be limited by the width of the roads or overhead overhanging obstructions [1]. The lack of uniform road access across the region makes the job of attending to emergencies or providing disaster relief a major challenge. Early gathering of information on the ground conditions plays an important role in the effectiveness of any emergency response that would follow. A quick-response system using volunteers, locals and emergency relief groups directed by a central Control Center (CC) will create a better opportunity to provide relief to the inhabitants.

We propose a shanty town emergency response by training a team of local volunteers who know the terrain well enough to act as Mobile Data Collectors (MDCs) and gathering information from event locations on-demand. This idea has been proposed in a previous work on participatory sensing [2]. Each MDC is networked to the CC which provides the path to reach each location. The MDCs collect information at that event location and relay it to the CC in the shortest possible time. The MDC and the CC need to have a common understanding of the road network. Temporary blockages due to construction, fire spread, or other reasons may not be discovered until an MDC attempts to cross them. Therefore, solutions must have the ability to quickly adapt to such dynamic events.

Previous work focuses on the computer networks and data transfer aspects by deploying sensor networks to capture such information and using multi-hop networking to deliver the data to a remote sink [3], [4], [5], [6]. [7] describes mobile sinks going to predetermined anchor points to opportunistically collect data potentially delaying capture of dynamic events. [8] describes mobile-element path planning as a time-constrained mobile element scheduling problem and discuss solutions based on two heuristics, tour cutting and tour packing.

Traditional wireless sensor network architectures consist of static nodes which are densely deployed over a sensing area. A detailed survey of the different sensing options is described in [9]. This work describes that deploying a dense sensor farm can be costly and perhaps require a lot of energy, both of which are in short supply in a shanty town. Using mobile sinks to visit anchor points introduces an inherent delay in data gathering. Due to these practical characteristics, we do not assume an in-situ deployment of sensors in the shanty town terrain, nor do we use anchor points. Instead, we propose equipping the volunteer MDCs with packaged sensing devices that communicate with the CC. We use the concept of mobile elements to capture the sensor data for our solution. This technique as described in [9], addresses the problem of gaps in coverage with a static sensor farm.

In this paper, we focus on addressing the path planning needs for a set of these heterogeneous MDCs that can navigate through the terrain for data acquisition on-demand. Depending upon the event location, more than one MDC can be routed to collect data. Our challenge is to identify that specific MDC that can capture the data as well as handle the data collection needs of all pending events. In addition, we recognize that in an emergency response scenario, event arrival rates can be random. We show how we can handle the dynamic nature of these events with a given finite set of resources. Our solution

978-1-4799-5952-5/15/$31.00 ©2015 IEEE

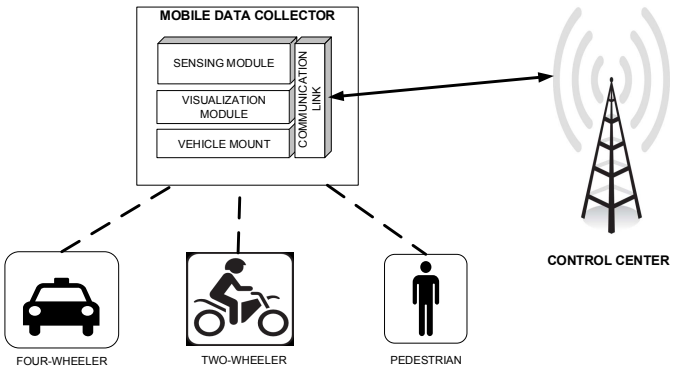


Fig. 1. Conceptual view of heterogeneous MDCs

is simple and can be applied in many real-world applications.

In this paper, we discuss efficient approaches to assigning mobility to the MDCs in such terrains. Key contributions of this paper include:

- Mathematical formulation of this dynamic scenario to define the optimized path rescheduling problem leading to its NP-hardness (Sec. III).
- Design of a family of novel heuristics (using naive, local and greedy approaches) for path rescheduling of heterogeneous MDCs and complexity analysis of the proposed heuristics (Sec. IV).
- Evaluation of the proposed algorithms using Dharavi, a shanty town in Greater Mumbai, India, to provide spatial layouts and mobility scenarios responding to random emergency events (Sec. V).

II. Approach

Service organizations are usually the first point of contact for the residents of a shanty town seeking relief from urban emergencies like fire, smoke, gas, medical etc. They register the request in their system and coordinate [10] with local agencies who provide relief to the requestor. Their effectiveness is usually measured in terms of response time which is the amount of time that it takes for the emergency responders to arrive at the scene of the incident. We call the emergency requests received by a system as an *event*. Providing a reliable response requires the event data to be captured within a finite amount of time window that we call *event expiration*. With the development so unplanned in a shanty town, some event locations are barely accessible by a four-wheeler while others can only be navigated by a two-wheeler and some others only by foot.

We propose that using heterogeneous MDCs with different capabilities can navigate different types of access paths as shown in Fig. 1. For such a scheme to be effective, our objective is to reach these locations quickly to gather accurate data about these events. Therefore, we propose packaging relevant sensor devices into a small size that can be carried by a human or mounted on a bicycle, scooter, car or a bus. These MDC devices are then carried by volunteers to the vicinity of the event location to gather information about the situation.

The gathered data is then transmitted to the CC for further analysis. For an effective response, the MDCs must reach the event locations within the shortest possible time and relay the information to the CC. In this paper we concentrate on an efficient mechanism for such an MDC to get to the event locations. The volunteer MDC operators can be distributed anywhere in the shanty town. Over the course of time, additional volunteers can be signed up in specific hotspots in the shanty town to improve their availability. We route or reroute the MDCs on-demand dynamically to event locations and hence call this *dynamic event-based routing*.

A. Path Planning for Dynamic Events using Heterogeneous MDCs

We model the event requests to be non-deterministically random in time and space. We translate its location to the nearest navigable junction on the road network. We next identify the best available MDC that can reach that spot. As and when new event requests are received existing paths may need to be recomputed with an objective to capture the data from every event before it expires. To assess the effectiveness of our schemes we measure the percentage data collected, the average time to capture an event and the average distance traveled by the MDCs during an observation window.

Section IV describes two path planning algorithms that we developed to efficiently determine the mobility model for each of the heterogeneous MDCs.

III. Modeling and Problem Statement

A. Shanty Town Road Network as a Graph

In order to plan a path for each MDC, we need an accurate representation of the shanty town and the different artifacts used to formally describe the access paths. We model the entire space as a graph with junctions and road-ends modeled as vertices on the graph and the accesses linking these junctions to one another as edges. Each edge weight is proportional to the length of the corresponding access link. We export the map data from *openstreetmap.org* as an XML file in OpenstreetMap (OSM) format. The graph is generated using the road characteristics given in the XML tags. We therefore have a graph $G(V, E)$ as a generic representation of the road network.

The shanty town can have roads of different type. For the sake of a generic model, we consider three different road networks using the set of edges: *(i) H-Type* (E_H) with edges where a four-wheeler can reach, *(ii) T-Type* (E_T) with edges where a two-wheeler (e.g. scooter or moped) can access, and *(iii) P-Type* (E_P) with edges where pedestrians can walk. A pedestrian-mounted MDC can traverse an edge of any type while the two-wheeler MDC can traverse just the T-Type and H-Type edges. Thus, we have three different graphs each having the same set of vertices for convenience but different sets of edges such that:

$$E_H \subseteq E_T \subseteq E_P$$

B. Emergency Request as an Event

We model a request for emergency assistance received by the CC as an event. This is on similar lines as in [11]. Detecting the event in such a setting has been addressed to sufficient depth in a related work [12]. We do not go into this aspect here in this paper. An event could apply to any location in the shanty town. For simplicity, in order to pinpoint the location for emergency response, we translate the event to the nearest vertex location. Thus an *event* is represented as $e(v_e, ts_e, exp_e)$ and is described in terms of the following attributes:

- v_e: The nearest junction on the road network, which is considered as the event location.
- ts_e: The timestamp when the event has occurred.
- exp_e: The expiration time of the event.

We assume that the events are discrete and non-deterministic. In our analysis, we assume M events during an observation window. Therefore, we model the distribution of the *Event Arrival Time* for the events as uniformly random, thereby, generating the requisite number of events.

C. Modeling the Mobility of Heterogeneous MDCs

In our approach, the Control Center (CC) provides the sequence of event locations and the fastest traversal path based on global knowledge of the road network, its condition, and of other events occurring in the shanty town. We first describe the concepts that help to define our model.

An MDC Walk: A sequence of event locations visited by (or planned for) an MDC is an *MDC Walk* or simply a *walk*. We write:

$$W_x = \{v_1, v_2, \ldots, v_n\}, \text{where } v_i \in V,\ \forall i \in [1, n]$$

Representation of an MDC: An MDC x (Fig. 1), is represented as $M_x(g_x, v_{x0}, W_x)$ where:

- g_x is the graph network that the MDC x can traverse which can be an instance of any of the three graphs $G_H(V, E_H)$, $G_T(V, E_T)$ and $G_P(V, E_P)$.
- v_{x0} is the vertex location where changes to the trajectory can be specified for the MDC ($v_{x0} \in V$).
- W_x is a walk assigned to the MDC x starting from the reference location v_{x0}.

When the CC receives a new event, it computes an efficient walk to handle this new event as well as the previously scheduled set of events. The CC then updates the corresponding MDC with a new walk W_x.

Generic Cost Model Design: We consider two types of cost here. *Travel-related costs* are typically in terms of distance traveled or time elapsed during the travel or for completing a service. *Data Capture Constraints* are associated with missing the capture of an event as it represents an opportunity missed to serve a customer.

Travel Related Costs:

Travel Time: We write the Travel Time of an MDC x between two locations v_a and v_b as:

$$TT_x(v_a, v_b, t) = \frac{d_x(v_a, v_b)}{s_x(v_a, v_b, t)} + (h_x(v_a, v_b) * \alpha) + \beta \quad (1)$$

Here $d_x(v_a, v_b)$ is the shortest distance between v_a and v_b on the graph that the MDC x is associated with, $s_x(v_a, v_b, t)$ is the average speed along the edges from v_a to v_b that the MDC x travels starting at v_a at time t, $h_x(v_a, v_b)$ is the number of hops along the shortest distance on the graph of MDC x between the vertices v_a and v_b, α is a delay at each junction along the path that accounts for time negotiating an intersection, and β is the service time at an event location for gathering data. This generic representation allows us to model that the travel time can change depending upon the traffic conditions at t. Traffic is assumed to be constant and therefore t is omitted.

Expected Time of Arrival: The ETA of an MDC x at v_b, starting from v_a, at time t is expressed as:

$$ETA_{x,v_b} = ETA_{x,v_a} + TT_x(v_a, v_b) \quad (2)$$

$v_a, v_b \in W_x$.

Total Distance: The Total Distance traveled during an observation by K MDCs operating in the neighborhood is computed as:

$$TD = \sum_{j=1}^{K} \sum_{i=1}^{m(j)-1} d(v_i, v_{i+1}) \quad (3)$$

Here *m(j)* is the number of events successfully visited by an MDC j and $v_i \in W_j$.

Data Capture Constraints:

Miss Penalty: For an event at location v_a we assess a penalty when none of the K available MDCs arrives within the observation window to collect the event data. We refer to this as $MP(v_a)$. For that event location when MP is evaluated as 1, the event has been missed.

Total Missed: A measure of the total number of the missed events TM gives an indication of how efficient the path plan is in terms of data capture.

Percentage of Event Data Captured can be computed as:

$$\frac{(M - TM)}{M} \times 100 \quad (4)$$

Over a period of time, for a set of M events, we hereby measure the effectiveness of any scheme of data capture.

Average Time of Event Data Capture shows how responsive a scheme is. The time elapsed between the event request and the event capture is averaged over a period of time across the set of $M - TM$ events and using K MDCs as:

$$\frac{\sum_{j=1}^{K} \sum_{v_i \in W_j} (ARR_j(v_i) - e_i)}{M - TM} \quad (5)$$

$ARR_j(v_i)$ is the arrival time of the MDC j at event location v_i such that $v_i \in V$. e_i is the time the event request was received for location v_i.

D. Problem Statement

The above formal definitions lead us to the following overall objective:

Assuming that K MDCs are available in a shanty town emergency response effort and M independent events occur

random non-deterministically distributed in time and space in the shanty town, our ideal objective is to generate a path plan for the K MDCs to handle all the M random events without exceeding their event expiration times. Informally we try to primarily maximize the Percentage of Events Captured before expiration and secondarily minimize the Average Capture Time of an Event.

Stated more formally,

GIVEN:

An MDC x defined as $M_x(g_x, v_{x0}, W_x)$.

$$W_x = \{v_1, v_2, \ldots, v_n\}, \text{where } v_i \in V$$
$$EXP_x = \{exp_1, exp_2, \ldots, exp_n\}$$

EXP_x is the set of expiration times $EXP(exp_i)$ of each of the event locations in W_x. There are *K* MDCs and *M* events to capture. As before, $ARR_x(v_{xn})$ is the arrival time of the MDC x at v_{xn}.

Find an ideal set of K walks W^{Ideal} such that

$$W^{Ideal} = \{W_1, W_2, \ldots, \ldots, W_K\} \text{ and} \tag{6}$$

$$\begin{aligned} &\text{minimize} && (5) \\ &\text{subject to} && ARR_x(v_{xj}) + TT_x(v_j, v_{j+1}) \leq exp_{j+1}, \quad (7) \\ & && \forall j \in [0, n-1] \text{ and } \forall x \in [1, k] \end{aligned}$$

Here, v_{x0} is the current location of the MDC x and $v_{x0} = v_0$. In reality, Equation 7 may at best lead us to some events being missed due to a lack of available MDCs. Therefore, we approximate this ideal solution with heuristic algorithms described in Section IV.

NP-Hardness: This problem falls into the broad category of Vehicle Routing Problem with Time Window (VRPTW). Traditional Vehicle Routing Problems (VRP) involve a deterministic set of customer locations and a set of vehicles with finite capacity to serve them. However, the MDC should accommodate new emergency requests as and when they come in. We thus have a Dynamic VRPTW problem. This is known to be a subclass of Combinatorial Optimization Problems, which are NP-hard, and requires searching among $O(M!)$ combinations. Like many such problems, heuristic-based solutions can be quite effective.

Multiple-TSP with Time Windows (mTSPTW) [13] and Vehicle Routing Problems [14], [15] have been studied extensively in the operations research community. In [16], Laporte describes the exact algorithms to solve relatively small problems. When the demand changes over time, i.e., handling non-deterministic events as we are, the classic solutions become even more expensive. Spliet, in [17], describes that one of the solutions is to come up with a master schedule and apply minimal changes to accommodate new demand. This might cause the resulting schedule to be sub-optimal but it may yield acceptable results, i.e., customers are served within their time window constraints but the overall distance covered by the service may not be optimal. Moreover, TSP or mTSP, by definition, are constrained to avoid visiting the same location more than once on a tour and they typically do not handle dynamic events. Finally, techniques proposed in [18], [19] are not directly applicable as we need path rescheduling for heterogeneous vehicles responding to dynamic requests.

IV. Path Re-Scheduling Algorithms

We studied route planning techniques in VRP applications to find an optimal route between two points on a road map [20], [21], [22]. In the previous section we discussed the NP-Hardness of our problem and therefore, solve the same using heuristics we describe here. We call our best solution, W^{eff} (a subset of all the vertices to be visited by the walks assigned to each of the k MDCs) and, therefore, only a subset of W^{Ideal} (that will be hard to find), i.e.,

$$\{v_i \in W_k, \ \forall \ W_k \in W^{eff}\} \subseteq$$
$$\{v_j \in W_x^{Ideal}, \ \forall \ W_x^{Ideal} \in W^{Ideal}\} \tag{8}$$

Some event locations are therefore discarded by the algorithm.

In this section we propose two different heuristic algorithms, the *Minimum Deviated Walk* (Section IV-A) and the *Ortho Walk* (Section IV-B) to handle such dynamic events and provide walk for each MDC. We compare the performance to a naive approach, we call the *Nearest Walk* where we append the new event at the end of the walk of one of the MDCs that reaches the location soonest.

We make the following overall assumptions:

- The CC keeps track of the position of each MDC and therefore knows the next vertex from where a new path can be assigned to it.
- The CC knows the time the MDC will be at that location.
- There is a one-time cost to obtain the TravelTime between all vertex pairs by the shortest path available for each MDC based on its type. The Floyd-Warshall (or similar) algorithm can yield this in $\mathcal{O}(v^3)$. If we keep this information in persistent storage we can provide a constant time look up for travel time between one node and another. This technique has been proposed as an optimization for such problems in [23].
- The CC handles dynamic events by applying the algorithms on all pending events based on the MDCs' next routable position.

A. Minimum Deviated Walk - A Local Search Approach

The motivation of this algorithm is to preserve the relative order of the events in the existing schedule intact from one iteration to the next. When a new event is inserted in an existing walk, the capture of the last event in that walk is delayed. *Minimum Deviated Walk* aims to delay the capture of such events the least, hence the name. If there are m unexpired events across all the MDCs, the algorithm tests $\mathcal{O}(m)$ insertion points in the k MDCs' walks. Therefore, this algorithm requires $\mathcal{O}(m + k)$ computations to find out the appropriate position for the new event.

When the inclusion of a new event in any walk causes a pre-scheduled event to expire, the new event is deferred

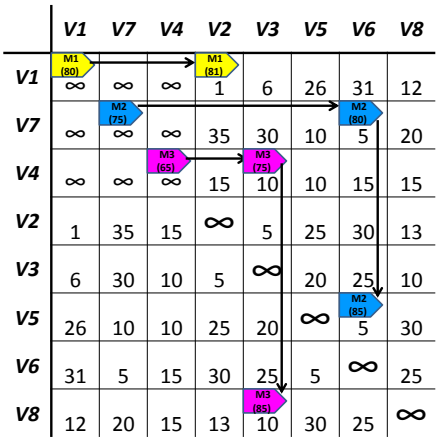


Fig. 2. Ortho Walk Run (For a sample input of 5 events at V2, V3, V5, V6, V8 and 3 MDCs at V1, V7 and V4)

and included in a subsequent iteration. If we assume the expected number of additional executions due to this is α, the complexity for handling a new event can be approximated to $\mathcal{O}((m+k)(1+\alpha))$.

B. Ortho Walk - A Greedy Approach

We now describe a greedy approach for determining an efficient set of walks. The Ortho Walk considers all m pending events including the new event location to be visited by the k MDCs. As the algorithm progresses, it tracks the ETA at each scheduled location. It incrementally identifies the next event to schedule until all events have been scheduled or the events cannot be accommodated in any valid schedule.

Each iteration starts with a simple pre-processing step. Otho Walk uses a 2-dimensional array of pre-computed *Travel Time* values between any two of m vertex locations. It then adds the corresponding entries for the times from the next routable positions for each available k MDCs. The combinatoric explosion of the problem is efficiently handled with such a matrix and using it to make a greedy transition for one of the MDCs to the next event location.

The algorithm ends with the iteration where (1) there are no more event locations to visit or (2) there was no transitions of an MDC possible without expiring the event. Such events are likely to get missed unless a new event gets added causing a re-shuffling of schedules.

We show a sample run (Fig. 2) generating the walks for three MDCs with five events to be visited. The walks and *ETA* at that event location of each MDC is indicated. For simplicity, we assume that all MDCs are of the same type. The path traced by each MDC on the matrix changes in right angles as shown, which was our inspiration for naming the algorithm *Ortho Walk*.

With m pending events to be serviced using k MDCs, the pre-processing step takes just $((m+k)(m+k+1))/2$ constant time lookups. Choosing the lowest ETA for each event would add an additional mk comparisons. The running time for the algorithm to supply a schedule for all the m event locations, is at most $\mathcal{O}(km^2)$. Both m and k are typically much smaller in comparison to the number of vertices and edges in the graph network making this a very efficient solution.

V. Experimental Validation

A. Our Setup and Experimental Strategy

We chose to run our tests on a graph representation of a real shanty town. We converted the map of Dharavi (Fig. 3a), which is near Mumbai, India to a road network as shown in Fig. 3b. We used an XML file, extracted from *openstreetmap.org*, as the basis and annotated the roads, based on their width, as supporting pedestrian, two-wheeler and highway traffic. We assumed different fixed speeds for the three MDC classes to match these road types.

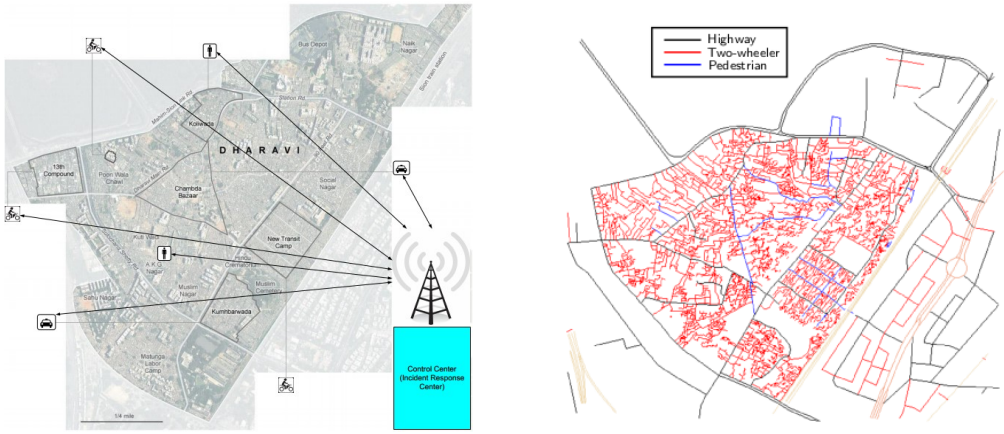
Event occurrences are modeled as spatially and temporally random. We adjusted parameters to simulate a predictable total number of events within an observation window as an average. We assumed that the MDC needs a service time of 10 seconds at the event location to capture the data before setting out to the next event location. We tested with different numbers of MDCs and varying numbers of events per hour. For the sake of comparison, all our events expire after 5 mins. In reality, this can change with the type of event.

We built a test harness to compare the proposed techniques, *Nearest Walk* (NW), *Minimum Deviated Walk* (DW) and *Ortho Walk* (OW). The harness takes in parameters to simulate the path planning conditions for the MDCs. This was written in Python (version 2.7.6) using utility modules such as *networkx*, *numpy* and *scipy*. Our setup was tested on Intel i5/i7-based computers running Ubuntu 14.04. Each simulation was run 20 times and aggregated (averaged within a group) to obtain a set of results for the input parameters. We compare the heuristics using three different metrics: *Percentage of Event Data Captured*, *Average Time of Event Data Capture*, and *Average Distance Traveled by an MDC*.

B. Results of the Experiments

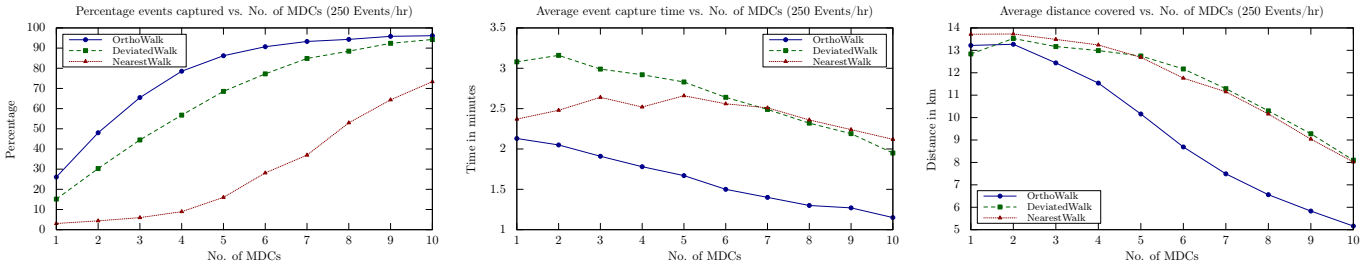
1) Basic Comparison of Algorithms: Fig. 4a presents the *Percentage of Event Data Captured* for a simulated load of 250 events/hr with varying numbers of MDCs during our observation window. We show that both, DW and OW are effective in data capture even though DW takes more MDCs to achieve that. The *Percentage of Data Capture* does not account for the duration within which the event data is captured. We therefore consider the *Average Event Capture Time* in Fig. 4b which shows that OW captures the events faster and is hence far more responsive. Our test showed that OW consistently performed 40% better than DW on this aspect. The *Average Total Distance* covered by the MDCs, presented in Fig. 4c, directly contributes to the cost of operating the MDCs. Again OW consistently shows less travel for the MDCs.

2) Detailed Comparison of OW and DW: Fig. 5 shows a more detailed comparison of OW and DW for the same metrics we discussed. For non-deterministically random event requests, it is possible to achieve more than 90% event data capture using either heuristic (Fig. 5a). With fewer MDCs, the constraint placed on DW to preserve the relative sequence of



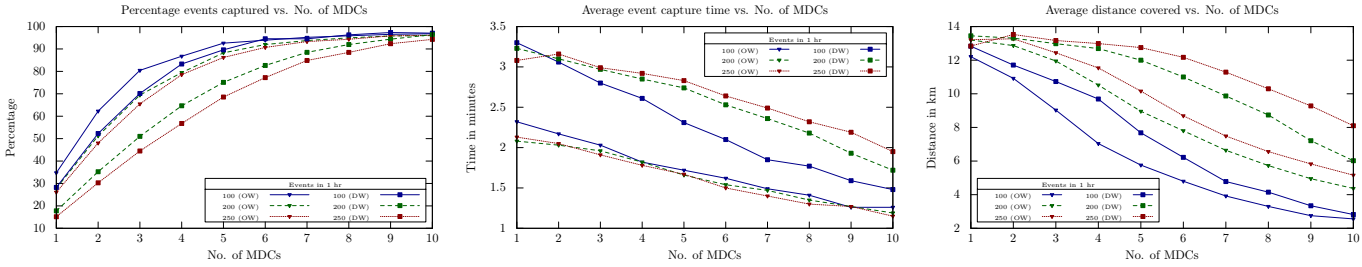
(a) Map of Dharavi (b) Graph Network for Dharavi

Fig. 3. Shanty Town Map to Graph Transformation: the graph has 2221 nodes and 2642 edges of 3 kinds: pedestrian, two-wheeler and four-wheeler.



(a) Effectiveness of Event Capture (b) Average Event Capture Time (c) Average Distance Traveled by MDC

Fig. 4. Basic comparison of our three proposed algorithms



(a) Effectiveness of Event Capture (b) Average Event Capture Time (c) Average Distance Traveled by MDC

Fig. 5. More detailed comparison of Ortho Walk and Minimum Deviated Walk algorithms with varying event rates

pre-planned events results in more deferrals of new events. A deferred event is less likely to be serviced as it will have an effectively shorter expiration time when it is again considered. Fewer events are deferred with more MDCs, thereby improving the effective data capture.

The comparison of the *Average Event Capture Time* (Fig. 5b) between DW and OW shows that, at higher loads, OW can be $30-40\%$ faster than DW to capture the events and trends even higher with a larger frequency of events. We see OW consistently performing better in this regard than DW because the former explicitly minimizes this in a greedy fashion by the way it chooses the next event to service. This metric shows little variation, when considering different event rates, for OW because of this. However, for DW, this value increases steadily, which further indicates the effects of the constraint. Intuitively, we see that using more MDCs increases the speed of capture for the same frequency of events.

The *Average Distance Traveled* (Fig. 5c) decreases monotonically with more MDCs but increases with the frequency of event requests. We attribute this to having more events to capture, thereby necessitating more travel. With increasing event rate, there will also be more walks that get rescheduled or adjusted while the MDCs are in transit, which can result in back-tracking and wasted effort. OW consistently travels less distance than DW because the former minimizes the *TravelTime*, which translates to the Distance Traveled. Over an observation period we find that *Average Distance Traveled* using OW was typically $20-40\%$ less than DW for a similar load, and OW gets even better at larger request rates.

Overall, OW appears more optimal than DW. The effective-

ness of DW being as high as OW (i.e., $> 90\%$) in most cases gives us a good reason to keep it in consideration. Having said that, DW has some aspects that make it some areas better than OW. The complexity of the DW algorithm is lower as its search space is small and therefore very fast. OW is slightly more complex but yields better results and lower travel time costs due to its minimal constraints.

TABLE I
AVERAGE DISTANCE TRAVELED BY EACH TYPE OF MDCS (SPEED: TW-15KMPH, FW-20KMPH)

Algorithm	**Nearest Walk**		**Deviated Walk**		**Ortho Walk**	
Events/hr	**FW**	**TW**	**FW**	**TW**	**FW**	**TW**
100	0.48	6.31	1.15	7.63	0.68	5.60
150	1.03	9.79	1.77	11.82	1.03	7.74
200	0.99	11.61	2.87	14.61	1.17	9.52
250	1.79	13.47	3.93	16.66	1.23	10.97

3) Heterogeneous MDCs and Varying Service Times: One of the unique aspects of the proposed solution is that we handle MDCs of multiple type. These MDCs move across the shanty town with different average speeds. This means that two MDCs of different type could have two different travel times, and therefore the earlier arrival time is normally preferable. We kept the number of events at 200/hour but varied the total number of MDCs. As before, we measured the average distance traveled by each of the MDC types. See Table I.

The distances traveled by each type of MDC confirm our intuition that the Four-wheeler MDC cover less distance as a large portion of these events may be unreachable by them. Given this result, we are able to make recommendations such as recruiting 8-10 times more two-wheeler MDCs than the number of four-wheeler MDCs.

VI. CONCLUDING REMARKS

This paper considered the approach of using heterogeneous MDCs with different capabilities to navigate the streets of a shanty town and capture data from emergency event locations as and when they are detected. For such a scheme to be effective, our objective is to gather data from all the events before their expiry and promptly transmit the information gathered to the Control Center. We formulated this problem as a Dynamic VRPTW problem, which is NP-Hard. We then proposed two heuristic based solutions: *Minimum Deviated Walk* (DW) and *Ortho Walk* (OW). We conducted a number of experiments on these two algorithms and proved that they handle heterogeneity of MDCs as well as dynamic events. We have shown that our algorithms scale well, have a small footprint and can yield a data capture rate of over 90%.

We are working towards a system implementation to be deployed and evaluated in real-world testbeds. We plan to investigate a DTN-based solution combined with a multi-network topology to handle any connectivity issues. We can extend the study to consider how emergency response teams navigate to locations that are not suitable for four-wheel vehicle access and continue on foot at some point.

Acknowledgments: The authors would like to thank X. Yi for her preliminary work. This effort is part of an IndoUS Collaborative Research Project funded by NSF (#1143705) and IIST, Trivandrum, India.

REFERENCES

[1] "Mumbai's Shadow City," National Geographic Magazine - NGM.com, May 2007. [Online]. Available: http://ngm.nationalgeographic.com/print/2007/05/dharavi-mumbai-slum/jacobson-text

[2] K.-c. Lan, C.-M. Chou, and H.-Y. Wang, "Using vehicular sensor networks for mobile surveillance," in *Vehicular Technology Conference (VTC Fall), 2012 IEEE*. IEEE, 2012, pp. 1–5.

[3] Y.-F. Wang and L.-L. Liu, "Grey target tracking and self-healing on vehicular sensor networks," *EURASIP Journal on Wireless Communications and Networking*, vol. 2007, 2007.

[4] C. Mascolo and M. Musolesi, "Scar: context-aware adaptive routing in delay tolerant mobile sensor networks," in *Proceedings of the 2006 international conference on Wireless communications and mobile computing*. ACM, 2006, pp. 533–538.

[5] S. Kapadia, B. Krishnamachari, and L. Zhang, *Data delivery in delay tolerant networks: A survey*. INTECH Open Access Publisher, 2011.

[6] X. Yi, "Controlled mobility for event data collection within wireless sensor networks," Master's thesis, UC Irvine, 2012.

[7] M. Zhao and Y. Yang, "Optimization-based distributed algorithms for mobile data gathering in wireless sensor networks," *Mobile Computing, IEEE Transactions on*, vol. 11, no. 10, pp. 1464–1477, 2012.

[8] K. Almi'ani, A. Viglas, and L. Libman, "Mobile element path planning for time-constrained data gathering in wireless sensor networks," in *Advanced Information Networking and Applications (AINA), 2010 24th IEEE International Conference on*. IEEE, 2010, pp. 843–850.

[9] M. Di Francesco, S. K. Das, and G. Anastasi, "Data collection in wireless sensor networks with mobile elements: A survey," *ACM Transactions on Sensor Networks (TOSN)*, vol. 8, no. 1, p. 7, 2011.

[10] UDF-RCL-08-270, "Empowerment of shanty towns settlers through democratic spaces," UN Democracy Fund, July 2012.

[11] N. Bisnik, A. A. Abouzeid, and V. Isler, "Stochastic event capture using mobile sensors subject to a quality metric," *Robotics, IEEE Transactions on*, vol. 23, no. 4, pp. 676–692, 2007.

[12] T. P. Lambrou and C. G. Panayiotou, "Collaborative event detection using mobile and stationary nodes in sensor networks," in *Collaborative Computing: Networking, Applications and Worksharing, 2007. CollaborateCom 2007. International Conference on*. IEEE, 2007, pp. 106–115.

[13] T. Bektas, "The multiple traveling salesman problem: an overview of formulations and solution procedures," *Omega*, vol. 34, no. 3, 2006.

[14] J.-Y. Potvin, T. Kervahut, B.-L. Garcia, and J.-M. Rousseau, "The vehicle routing problem with time windows part i: tabu search," *INFORMS Journal on Computing*, vol. 8, no. 2, pp. 158–164, 1996.

[15] M. M. Solomon, "Algorithms for the vehicle routing and scheduling problems with time window constraints," *Operations research*, vol. 35, no. 2, pp. 254–265, 1987.

[16] G. Laporte, "The vehicle routing problem: An overview of exact and approximate algorithms," *European Journal of Operational Research*, vol. 59, no. 3, pp. 345–358, 1992.

[17] R. Spliet, A. F. Gabor, and R. Dekker, "The vehicle rescheduling problem," *Computers & Operations Research*, vol. 43, 2014.

[18] R. Dondo and J. Cerdá, "An milp framework for dynamic vehicle routing problems with time windows," *Latin American applied research*, vol. 36, no. 4, pp. 255–261, 2006.

[19] Y. Wang, M. Colledanchise *et al.*, "A distributed convergent solution to the ambulance positioning problem on a streetmap graph," *World Congress*, vol. 19, no. 1, 2014.

[20] C. Sommer, "Shortest-path queries in static networks," *ACM Computing Surveys (CSUR)*, vol. 46, no. 4, p. 45, 2014.

[21] V. T. N. Nha, S. Djahel, and J. Murphy, "A comparative study of vehicles' routing algorithms for route planning in smart cities," in *Vehicular Traffic Management for Smart Cities (VTM), 2012 First International Workshop on*. IEEE, 2012, pp. 1–6.

[22] P. Sanders and D. Schultes, "Engineering fast route planning algorithms," in *Experimental Algorithms*. Springer, 2007, pp. 23–36.

[23] D. Schultes, "Route planning in road networks." in *Ausgezeichnete Informatikdissertationen*, 2008, pp. 271–280.