Efficient and Reliable Breadcrumb Systems via Coordination among Multiple First Responders

Hengchang Liu, Zhiheng Xie, Jingyuan Li, Kamin Whitehouse, and John Stankovic
Computer Science Department
University of Virginia
Charlottesville, VA, USA 22904
Email: {hl4d, zx3n, jl3sz, whitehouse, stankovic}@cs.virginia.edu

Shan Lin
Computer Science Department
Temple University
Philadelphia, PA, USA
Email: slin@temple.edu

David Siu
Science and Technology Division
OCEANIT
Honolulu, HI, USA
Email: dsiu@oceanit.com

***Abstract*—Breadcrumb systems (BCS) aid first responders by communicating their physiological parameters to remotely located base stations. However, state-of-the-art research only focuses on deploying breadcrumb systems on the assumption of uncoordinated users, which is inefficient. In this paper, we present the *first* design, implementation, and evaluation of reliable multi-user breadcrumb systems (MUBCS) which exploits efficient and automatic coordination among system users to achieve better utilization of limited breadcrumbs. We propose *UF*, a distributed cooperative deployment algorithm, to achieve longer breadcrumb chain length while maintaining fairness and high system reliability via selecting appropriate benefit and cost functions. *UF* also requires no prior assumptions about users' mobility models, making the design practical for real applications. We deployed and evaluated our system in real buildings with several different first responder mobility patterns. Experimental results indicate that this approach can maintain connectivity for up to 87% longer distances than baseline greedy coordination approach while maintaining 96% packet delivery ratio.**

I. Introduction

First responders are at an increasing risk of cardio-vascular problems especially during mission execution [3], [4], [5]. For instance, 118 emergency workers died in America in 2008 according to the U.S. Fire Administration [4], and 45 of them died from heart attacks during actual firefighting, not from the physical dangers of their work. Therefore, continuous physiological monitoring during missions can be effective in reducing the number of fatalities. However, existing solutions using one-hop communications [7] to connect first responders to a remotely located base station suffer from limited transmission range due to complex indoors infrastructures. The recently emerged breadcrumb systems [1], [2] allow each first responder to carry a small dispenser filled with sensor nodes and deploy them one-by-one in a manner that guarantees reliable communication and extends transmission range. To date, all related work focused on uncoordinated first responders, resulting in inefficient performance in multi-user breadcrumb system (MUBCS) scenarios. In this paper, we address the challenge associated with efficient and reliable automatic coordination among the dispenser systems carried by multiple first responders.

Our work is motivated by the fact that first responders are organized into small groups to execute different tasks and geographically close to each other in each group. Previous systems and algorithms do not fit into this situation and lead to suboptimal system resource (breadcrumbs) utilization as a result of inefficient breadcrumb deployments. One example is that a group of first responders are running along a hallway, in an uncoordinated scenario, the one at the head of the group drops breadcrumbs all the time because his system always detects decreased link quality first. Later, when this first responder takes another separate route by himself, he finds himself running out of breadcrumbs. On the other hand, simple coordination algorithms may not help increase deployment efficiency. For example, for fairness reasons, one simple algorithm requires the one with the most number of breadcrumbs in a group to deploy. In this case, breadcrumbs deployed by the one at the end of the group may not improve communication for the one at the very head of the group due to the distance between them, resulting in another request for breadcrumb deployment by the leading first responder very soon.

In this paper, we present the first sophisticated study to exploit efficient design for automatic MUBCS. The driving idea behind our design is *efficient and automatic coordination* to determine which user should deploy a new breadcrumb. More concretely, given a limited number of breadcrumbs available for each user (due to physical size and cost reasons), we address the problem of finding an optimized coordination scheme that maximizes the transmission range while maintaining high end-to-end reliability.

Our contributions in this paper are three-fold. First, Our work makes a further step from the traditional study on reliable single-user breadcrumb systems into more practical and efficient MUBCS design. Conceptually, our design principle is general enough to be applied to any sensor network applications that require opportunistic sensor deployment, such as firefighting, underground mining, and wilderness area exploration applications. Second, we present *UF*, a utility function based algorithm that provides an efficient and distributed

coordination process via selecting appropriate benefit and cost functions. This novel algorithm results in a reliable and longer breadcrumb chain compared to a baseline greedy algorithm in which the user with most breadcrumbs always deploys. In addition, *UF* requires no a priori user mobility models, making the design practical. Third, we have built a prototype of MUBCS using 2.4 GHz based hardware. We deployed and evaluated our system in real buildings with multiple first responder mobility patterns. Experimental results indicate that this approach can maintain connectivity for up to 87.3% longer distances than baseline greedy approach while maintaining 96% packet delivery ratio. This implies that our proposed MUBCS design is reliable and efficient.

The remainder of this paper is organized as follows. In Section II, we describe the MUBCS model and present the functionalities of our proposed breadcrumb system. Then the *UF* algorithm is presented in detail in Section III. The prototype and evaluation for our proposed system are discussed in Sections IV. Finally, we conclude the paper in Section V.

II. MUBCS Model

This section presents the main design of the MUBCS model, which is a classical finite state machine. A dispenser under the MUBCS model is in one of four states at any time: (1) maintenance, (2) requester, (3) coordinator, and (4) ghost. Transitions between the states are triggered by events.

After the MUBCS model is initiated, the dispenser enters the maintenance state, in which it has reliable communication with the breadcrumb chain. Here, the link between users is maintained by periodically exchanging ping messages and the link quality of these transmissions is measured. Whenever the dispenser receives a HelpMsg from other dispenser, the dispenser enters the coordinator state, sends back response packets and involves itself into the coordination process. When a new breadcrumb is deployed, the dispenser goes back to the maintenance state. When the link monitoring algorithm determines that the link between the dispenser and breadcrumb chain is getting out of range, it enters the requester state and asks its neighbor dispensers for help. When the coordination is finished, it goes back to the maintenance state; otherwise, when all dispensers run out of breadcrumbs, the connection is completely lost and the dispenser goes to the ghost state. Finally, the dispenser can go from the ghost state to the maintenance state if the user walks backwards to the coverage of breadcrumb trail.

Before describing the coordination algorithms used in MUBCS, we first review some other system components in our breadcrumb system design. The combination of these components provides a practical and reliable breadcrumb system for a single first responder. For more details and related work, please refer to [1], [**?**].

Each first responder carries m breadcrumbs in his breadcrumb dispenser and deploys one whenever connection to the breadcrumb chain is getting weak. As they run into the building, breadcrumbs are deployed automatically on the fly. Due to the harsh environment in which breadcrumbs may break or burn up, our deployment policy requires that each breadcrumb keeps "good communication" with at least $n+1$ other breadcrumbs at any time in order to have redundancies to tolerate physical failures. Here, n represents the *redundancy degree* of each breadcrumb.

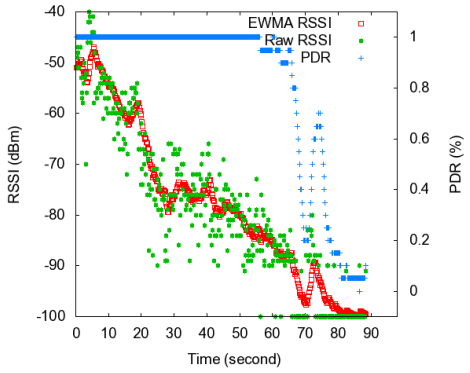


Fig. 1. PDR and $RSSI$ as time changes in an example trace.

As the first responder moves on for rescue work, the link quality between the dispenser on the user and the breadcrumbs becomes weaker. The *link monitoring algorithm* is used to estimate the link quality and make optimal decisions on when to deploy a new breadcrumb. Note that the design of the link monitoring algorithm should take both independent and non-independent breadcrumb failures into account.

Another key factor that needs to be taken into account while deciding when to deploy new breadcrumbs is the height effect. Since the dispenser (and the link estimator inside the dispenser) is normally placed at the waist of the first responder, thus there is a gap between the estimated link quality and the actual link quality after the new breadcrumb is deployed on ground. Solutions proposed to eliminate this height effect are called *height effect solvers*.

After the new breadcrumb is deployed and joins the crumb chain, the link quality between this new crumb and its n neighbors may vary due to the dynamic impact from the environment. *Adaptive power control* enables it to adaptively adjust its transmission power according to real-time link quality estimation.

III. *UF* Algorithm

In this section, we describe the proposed *UF* algorithm, which introduces the calculation of a utility function as the criteria of deploying new breadcrumbs. The utility function based algorithm works as follows: the requester initiates the algorithm by broadcasting a help message; then all of his neighbors send their information to the requester; after a predefined timeout, the requester calculates the value of utility functions for each of its neighbor and sends a drop message to the one with the highest value to deploy a new breadcrumb.

The essential part of the algorithm is to define an appropriate utility function. A good utility function must precisely represent the tradeoffs between the gain of communication range extension to the requester/group and the cost of the

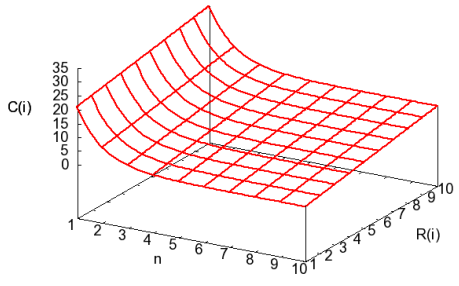


Fig. 2. $C(i)$ as $R(i)$ and $P(i)$ change.

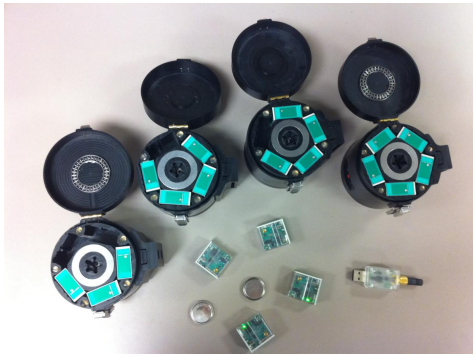


Fig. 3. Breadcrumb system prototype with four dispensers, twenty breadcrumbs, and one USB-ported base station.

breadcrumb counts to the deployer. The utility function for User i, denoted by $U(i)$, is defined as the weighted difference between a benefit function and a cost function, i.e.,

$$U(i) = \alpha \cdot B(i) - \beta \cdot C(i) \tag{1}$$

in which α and β and coefficients for the benefit function $B(i)$ and cost function $C(i)$, respectively. Without changing the final decisions, we use another variable $\gamma = \alpha/\beta$ and rewrite the formula as follows:

$$U'(i) = \gamma \cdot B(i) - C(i) \tag{2}$$

$B(i)$ represents the gain of the communication link if the first responder i deploys a new breadcrumb. This gain can be measured by either the $RSSI$ value for the requester himself or the average $RSSI$ value for both the requester and his neighbors. Thus it can be represented as:

$$B(i) = \frac{1}{n} \sum_{k=1}^{n} RSSI(i,k) \tag{3}$$

or

$$B'(i) = RSSI(i, req) \tag{4}$$

The tradeoff between these two functions $B(i)$ and $B'(i)$ is as follows. $B(i)$ considers all communication links in the group and takes the global optimal gain, but it requires each group to broadcast within the group, which increases both the communication overhead and the coordination delay. Moreover, the coordination delay may become even worse, since all group members are trying to broadcast simultaneously and thus schemes like random backoff timers have to be used. On the other hand, $B'(i)$ takes only a local optimal gain upon the requester, but it leads to much less communication overhead and shorter coordination delay. Based on these reasons, we choose the metric $B'(i)$ during the implementation and evaluation of our system.

To convert the $RSSI$ into a value for the calculation of $U(i)$, we propose the following approach: First, the timestamp when a predefined threshold of PDR is reached is set to be the minimum 0, and the timestamp when the experiment starts is set to be the maximum K. Then we record the fraction of time μ when the exponentially weighted moving average ($EWMA$) value of $RSSI$ first reaches $RSSI(i, req)$ and calculate the corresponding value $k = \mu \cdot K$. The final value for the benefit function is then represented as the average of k for all 20 trials. We collected experimental data in traces in which the dispenser is moved far away from a breadcrumb on the floor until the connection is lost. To eliminate the effect of noise, we repeat the same experiment for 20 times, 5 in each different environments including hallway, corner, upstairs, and downstairs. Figure 1 shows the results of PDR and $EWMA$ $RSSI$ in an example trace. Since the mapping between $RSSI$ and PRR is different from one building to another, a linear approximation model can be used to obtain the corresponding value $k = \frac{RSSI - (RSSI_{min})}{(RSSI_{max}) - (RSSI_{min})} \cdot K$, here $RSSI_{min}$ stands for the minimum possible RSSI value, which for instance is -92 dBm for $CC2430$ radio [9].

$C(i)$ is the penalty for the number of remaining breadcrumbs for first responder i. A good cost function must take both relative ranking in the group and the absolute counts into consideration. The relative ranking is necessary since global information can provide important support for making decisions. The function of absolute counts, which grow exponentially as the counts decrease, is especially useful when most group members have only a few breadcrumbs left after the system is running for a long time. Thus, taking both factors into account, we have

$$C(i) = R(i) + P(i) \tag{5}$$

in which $R(i) \in \{1, 2, 3, \ldots, n\}$ represents the ranking of group member i in the group in terms of breadcrumb counts and $P(i) = e^{n_0 - n}$. Here n_0 is a predefined threshold to indicate that this user has very few breadcrumbs and will be at high risk of running out of breadcrumbs soon. Figure 2 shows how the value of $C(i)$ changes with the relative ranking and absolute counts. The number of users in the group is set to be 10, the initial number of breadcrumbs is 10 for each user, and the threshold for absolute counts is 4. We can observe from the graph that, relative ranking and absolute counts each affect the $C(i)$ value in a different way. $R(i)$ provides a constant

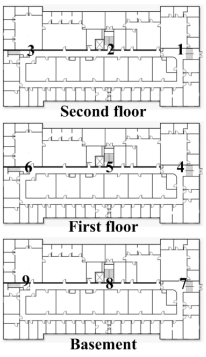


Fig. 4. Nine sampling points in the Computer Science building.

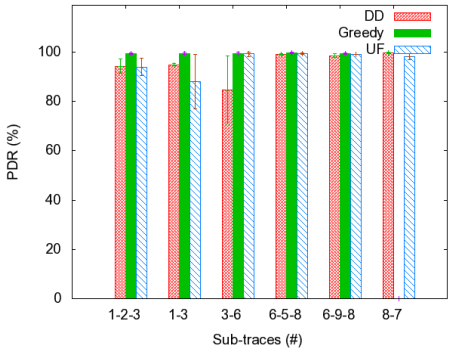


Fig. 5. Comparison of PDR between *DD*, *Greedy* and *UF* in the *BM* trace.

increasing factor to $C(i)$, $P(i)$ does not change much when there are plenty of remaining breadcrumbs, but as the counts decrease to the threshold, $P(i)$ starts to be the dominating factor for the total value of $C(i)$.

In summary, the utility function can be written as follows:

$$U(i) = \frac{RSSI - (RSSI_{min})}{(RSSI_{max}) - (RSSI_{min})} \cdot K - R(i) - e^{n_0 - n} \quad (6)$$

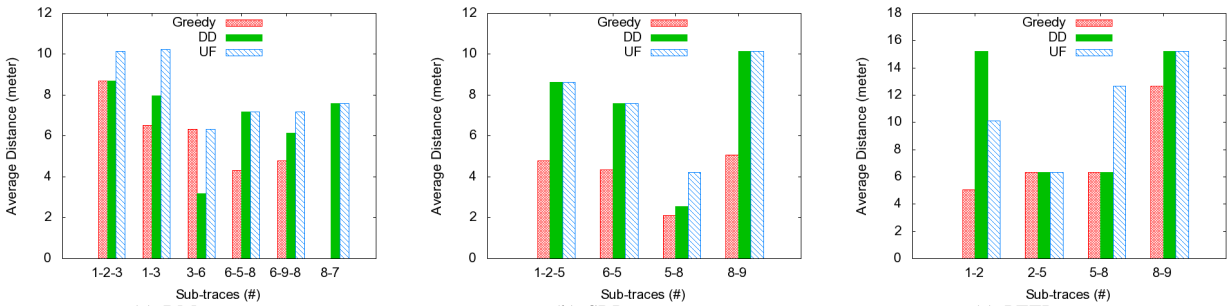
IV. Evaluation

We have built several new-version custom hardware modules, as shown in Figure 3, to accommodate the specific needs of the breadcrumb system. All modules are designed to be compatible with CC2430 series hardware. The key difference to the last version [1] lies in that the transmission range for both dispenser and breadcrumb increases to 40 meters. For more details on hardware configuration, please refer to [**?**].

To evaluate our proposed *UF* algorithm, we conducted groups of experiments in the Computer Science Building at the University of Virginia. We compare UF to two baseline algorithms: the greedy algorithm (Greedy) in which the user with the most breadcrumbs deploys immediately, and the delay dropping approach (DD) in which the user with the most breadcrumbs deploys, but the deployment is delayed to where his own link quality monitor indicates a new drop. Nine predefined branch points are used to record the experiment traces as shown in Figure 4 (Points 1, 4, 7, and 9 are entrances/exits due to hilly environments), and users divide into subgroups or merge to a larger group at some of these branch points. The experiments involve four users in total, denoted by A, B, C, and D. Users walk in a zigzag style at normal walking speed. The three traces that they walked through are listed below.

1) *Branching and Merging (BM):* This is to simulate the searching application. $ABCD$ all enter the building from branch point 1, and divide into two groups: $AB : 1 \rightarrow 2 \rightarrow 3$ and $CD : 1 \rightarrow 3$. $ABCD$ merge at 3 and walk through $3 \rightarrow 6$, and divide again: $AC : 6 \rightarrow 5 \rightarrow 8$ and $BD : 6 \rightarrow 9 \rightarrow 8$. Then $ABCD$ merge at 8, walk through $8 \rightarrow 7$, and leave the building.
2) *Single Rescue Point (SRP):* This is to simulate the rescuing application with a single rescue point. $ABCD$ enter the building from two different entrances 1 and 6, and the rescue point is 8. First, they walk through the trace: $AB : 1 \rightarrow 2 \rightarrow 5$ and $CD : 6 \rightarrow 5$. Then $ABCD$ merge at 5 and walk through $5 \rightarrow 8 \rightarrow 9$ and leave the building.
3) *Peeling Off One By One (PEEL):* This is to simulate the rescuing application with multiple rescue points. $ABCD$ all enter the building from branch point 1, and walk through $1 \rightarrow 2 \rightarrow 5 \rightarrow 8 \rightarrow 9$. At each branch point $2, 5$, and 8, one user will leave the team and search via a different route.

During the experiments, users walked along the predefined traces and breadcrumb trails were automatically established. Multi-hop communication is then applied to transmit useful data packets to the base station. The redundancy degree is set to 1 and exponentially weighted moving average was adopted to guide when a new breadcrumb is needed, and the parameters are set to be the optimal value that results in the least probability of dropping late while maintain a low Least Square value [1]: the weight β is 0.0313 and dropping threshold is -81.8 dBm. The timer used for waiting for responses from neighbors in *UF* is set to 1 second. For simplicity, the height effect is solved by a fixed offset 10 dBm. Along the trace, the dispenser sends out request messages periodically at the rate of 5 packets per second in order to get responses from "active" breadcrumbs. Link quality information is then recorded according to the identity of breadcrumbs. Physiological data are sent from the dispensers to active breadcrumbs at the rate of 2 packets per second. Due to the spatial locality, the synchronization message for group management is sent once per 2 seconds. For performance analysis purposes, in each data packet we included information such as timestamp and source node ID. Upon receiving the data packet, the intermediate breadcrumbs recorded this information in its own flash memory. Zigbee techniques [8] are used for the networking layer protocol during the experiments. To eliminate the effect of random noise, experiments were repeated five times when evaluating the reliability and the coordination delay of candidate algorithms and we found that the results have little variations. Unless stated otherwise, we used the above default values in all the experiments.



(a) BM trace. (b) SRP trace. (c) PEEL trace.

Fig. 6. Average distances between breadcrumbs for sub-traces.

A. Reliability

To investigate whether the candidate coordination algorithm leads to a high PDR breadcrumb system, a group of indoor experiments were conducted. We attached sequence numbers to data packets for statistical purposes and recorded the PDR when running *Greedy*, *DD*, and *UF*. Due to page limitations, we selected the most complex BM trace as the experiment environment and recorded the PDR for each subtrace. The PDR for each user is recorded separately so as to see the variance.

Figure 5 compares the average PDR with error bars of each subtrace when running *Greedy*, *DD*, and *UF*. We observe that 15 out of totally 18 bars achieve more than 95% PDR, which indicates that all three coordination algorithms lead to a high PDR. Particularly, *UF* achieves an average 96.3% PDR for all users in all subtraces, more concretely, 94%, 88%, 99.4%, 99.3%, 99%, and 98%, respectively.

We also observed that sometimes people shadowing is a big factor in packet loss. This occurs when multiple users walking in a narrow environment simultaneously and one stands in between the breadcrumbs and another user. As shown in Figure 5, subtrace $3 \rightarrow 6$ (stairway from the third floor to the second floor, 4 users) when running *DD* only results in an average PDR 84.6% and the minimum PDR is only 70.7%.

B. Efficiency

To compare the efficiency of candidate coordination algorithms, precise locations of deployed breadcrumbs in the previous experiments were recorded and analyzed. As shown in Figure 6, due to the inefficient coordination mechanism, *Greedy* produces only 5.10, 4.08, and 9.82 meters in terms of average distances in three traces. *UF*, however, achieves 7.76, 7.64 and 11.09 meters, which is 52.2%, 87.3%, and 12.9% better than *Greedy*. This is because in *UF*, both breadcrumb costs and link quality gains are taken into account, situations in which the deployer is far from the requester rarely happen. *DD* has an average distance of 6.79, 7.22, and 10.77 meters, which is close to the performance of *UF*. This is reasonable since the delay dropping process allows the deployer to carry the breadcrumb for a while before dropping it, and thus extends the average distance between breadcrumbs. However, *DD* relies on the assumption that users are walking towards the same direction with the same speed all the time, and they cannot stop or go backwards during the process. *UF* does not have these impractical constraints and performs well via the well-designed utility function, i.e., considering both benefit function and cost function, and taking both relative ranking and absolute counts into account when calculating the cost function. Due to page limitations, we do not show the results of fairness, but UF results in a maximum STDEV value of 2 in all traces, which implies that the group is well balanced.

V. Conclusion

In this paper, we present the breadcrumb system design in the context of multiple users. We propose a novel coordination algorithm (*UF*) to provide efficient and reliable breadcrumb deployment processes for multiple firefighter applications. We fully implemented the MUBCS on several custom hardware modules we have built, and evaluated our system in a real building with several different firefighter mobility patterns. Experimental results indicate that this approach can maintain connectivity for up to 87% longer distances than baseline greedy approach while maintaining 96% packet delivery ratio.

Acknowledgment

This work was performed on a Phase II SBIR on Wireless Body Area Networks funded by the Science and Technology Directorate of the Department of Homeland Security. This work was also supported, in part, by the NSF grants EECS-0901686 and CSR-0720640.

References

[1] H. Liu, J. Li, Z. Xie, S. Lin, K. Whitehouse, and J. A. Stankovic. Automatic and Robust Breadcrumb System for Indoor Firefighter Applications. In MobiSys 2010.
[2] M. R. Souryal, J. Geissbuehler, L. E. Miller, and N. Moayeri. Real-time Deployment of Multihop Relays for Range Extension. In MobiSys 2007.
[3] National Fire Protection Association, 2005. URL: http://www.nfpa.org.
[4] Firefighter Fatalities in the United States in 2008. URL: http://www.usfa.dhs.gov/downloads/pdf/publications/ff_fat08.pdf.
[5] C. Stefanos, N. Kales, and E. S. Soteriades. Emergency duties and Deaths from Heart Disease among Firefighters in the United States. In the New England Journal of Medicine, 356(12), 2007.
[6] H. Liu. Design and Implementation of an Automatic, Reliable, and Efficient Breadcrumb Sensor Network. Ph.D. dissertation, UVA, 2011.
[7] P25 projects, URL: http://www.project25.org.
[8] ZigBee Alliance, URL: http://www.zigbee.org.
[9] ChipCon CC2430 datasheet. URL: http://www.alldatasheet.com/datasheet-pdf/pdf/241043/TAOS/CC2430.html.