Halo: Managing Node Rendezvous in Opportunistic Sensor Networks

Shane B. Eisenman[1], Hong Lu[2], and Andrew T. Campbell[2]

[1] Columbia University, New York NY 10027, USA
shane@ee.columbia.edu
[2] Dartmouth College, Hanover NH 03755, USA
campbell@cs.dartmouth.edu

Abstract. One vision of an opportunistic sensor network (OSN) uses sensor access points (SAPs) to assign mobile sensors with sensing tasks submitted by applications that could be running anywhere. Tasked mobile sensors might upload sensed data back to these applications via subsequent encounters with this SAP tier. In a people-centric OSN, node mobility is uncontrolled and the architecture relies on opportunistic rendezvous between human-carried sensors and SAPs to provide tasking/uploading opportunities. However, in many reasonable scenarios application queries have a degree of time sensitivity such that the sensing target must be sampled and/or the resulting sensed data must be uploaded within a certain time window to be of greatest value. *Halo* efficiently, in terms of packet overhead and mobile sensor energy, provides improved delay performance in OSNs by: (i) managing tasking/uploading opportunity, and (ii) using mobility-informed scheduling at the SAP.

1 Introduction

The initial application focus of wireless sensor networking has been on *in situ* monitoring of ecological processes, or on industrial processes and equipment. Recently we and other researchers in the field have begun to consider the urban domain with a focus on *people-centric* [14] [16] [4] [22] sensing and application development. Architectures in this new domain assume mobile smart phones and embedded sensing devices equipped with a short-range radio (*e.g.*, ZigBee, Bluetooth, WiFi) are carried by humans or mounted on vehicles, leading to a network of sensors with mobility uncontrolled by the sensing architecture [11] [5]. Such architectures often employ a multi-tiered hierarchical structure where sensor tasking (*i.e.*, application query assignment) and data collection occur via mobility-enabled interactions between people-centric *mobile sensors (MSs)*, and edge wireless access nodes [10] we call *sensor access points (SAPs)*. In this context, we treat the question of how we may best task MSs and collect data from MSs in support of delay-aware applications.

Tasking and collection operations can occur when MSs enter the "spheres of interaction" of the SAPs. Generally, by sphere of interaction we mean the region (*i.e.*, the physical volume) within which services offered by a node are available to its neighbors. For the SAP case, to which we limit our discussion in this paper, these services include tasking and uploading. In practice, the adaptation of the sphere of interaction is implemented by both transmit power control, and

multihop signaling between a SAP and MSs that happen to dwell for a time near a SAP. During their stay these may be used to funnel packets from data collecting MSs to the SAP, and to relay tasking messages from the SAP to MSs.

While applications that use opportunistic sensor networks should be delay tolerant, we draw a distinction between those that are delay-aware and those that are not. Delay-aware applications do not warrant real-time treatment, but may issue queries that have a degree of time sensitivity such that the sensing target must be sampled, and/or the resulting data must be uploaded, within a specified time window to be of greatest value. Examples are myriad, and include personal applications that seek to answer questions like, "Where can I find a quiet place to study for the next hour", and public utility applications that say, "Give me my local weather spotter data in time for the next newscast". In support of delay-aware applications, we investigate a number of fundamental performance issues in OSNs, including the interplay between resource consumption and the timeliness of tasking and data collection.

To increase the frequency and duration of the sensors' travel through the sphere of interaction of a given SAP, the SAP might enlarge its sphere of interaction by increasing its transmission power and/or by building a multi-hop sphere of interaction. However, a multi-hop sphere of interaction requires increased signaling (*e.g.*, to set up and maintain routes), requiring more energy expenditure. In a setting with uniform per-link loss probabilities and fixed-power transceivers, multi-hop communication also implies a higher end-to-end loss probability compared to that possible; in a wireless environment with a link packet loss rate p the probability of success across n hops is $(1-p)^n$. Further, increasing the transmission power of the SAP implies an increased energy drain on the energy-limited MSs since these must match the higher transmission power of the SAP for bidirectional communications. Finally, a larger SAP sphere of interaction disrupts local peer-to-peer sensor communication in a larger part of the field, a problem of increasing relevance given the recent interest in mobile peer-to-peer services using localized communication [12] [20] [15].

We design, implement and evaluate Halo, a framework providing algorithmic and protocol support for managing rendezvous between SAPs and human-carried and vehicle-mounted mobile sensors in the urban domain. Halo manages opportunities for tasking and uploading operations via deadline-driven adaptation of SAP sphere of interaction. When multiple simultaneous operations are possible, Halo takes a snapshot of the system (*i.e.*, the sensors available for tasking and uploading in its sphere of interaction, the pending tasking operations, and the applications waiting for data upload), and incorporates sensor-driven mobility prediction of the available MSs to generate a schedule of the tasking and uploading operations. This novel scheduling approach integrates a traditional shortest-job-first approach, and a mobility-based approach tailored for OSNs.

2 Managing Rendezvous Opportunity

There are competing pressures in managing tasking and collection opportunities. We wish to expand the SAP sphere of interaction to increase the number of MSs available to task, reduce tasking delay, increase the amount and utility

of sensed data to delay-aware applications, reduce collection delay, and reduce the likelihood of mobile sensor storage overflow. On the other hand, we wish to contract sphere of interaction to reduce required energy expenditure of mobile sensors when transmitting to a SAP, reduce disruption to communications ongoing between MSs in the vicinity of a SAP, and increase the security of the system by probabilistically reducing overhearing, and explicitly limiting the number of nodes offering (authenticated) proxy service on behalf of the SAP.

We use a simple model of a single SAP to illustrate the impact of sphere of interaction radius on data transfer opportunity between MS and SAP, required MS transmit energy, and the SAP interference area. Here, the radius of the "sphere" (our 2D analysis can directly be extended to 3D) is a real valued abstraction of the range extension due to power control only. The trigonometry is straightforward and is omitted due to space constraints, but details are available in the technical report [8, Appx I]. Figure 1 shows: (i) the data transfer opportunity (in abstract time units) of a single MS by plotting the average straight line trajectory length through the SAP sphere of interaction, assuming the MS maintains a constant unit velocity along the trajectory; (ii) the average transmit energy a MS must use to communicate with the SAP, assuming a symmetric link and a simplified Friis model with a loss exponent of 4, as it traverses the sphere of interaction along the average length chord; and (iii) the lower bound on the area disrupted by SAP-MS communications as the MS travels along the average length chord. Data transfer opportunity grows linearly with the sphere of interaction radius, while energy cost and SAP interference experience super-quadratic growth. The tradeoff between data transfer opportunity, and MS energy and SAP interference impact, motivates a managed SAP sphere of interaction radius. This illustrative analysis necessarily ignores important realities of wireless networks: the relationship between transmission power and physical distance between the SAP and MS is complex due to antenna characteristics, the attenuation by the body [13] and other environmental factors, and humans typically do not walk along chords. However, we reasonably assume that increasing SAP transmission power increases the probability of interaction between SAPs and MSs, and insofar as this happens the disrupted area and energy costs to the network also increase. Section 4 provides simulation results bolstering these numeric arguments.

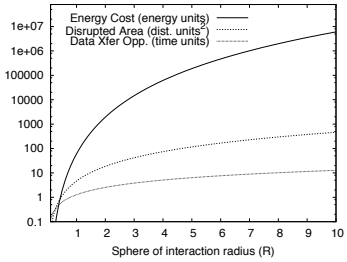


Fig. 1: Impact of SAP sphere of interaction radius.

While there are a number of possible triggers for increasing or decreasing the SAP sphere of interaction, we believe the fundamental driver for sphere of interaction adaptation should be fulfilling application requests (*i.e.*, tasking and collection operations) since this is the metric that mostly closely reflects the user experience. Generally, we wish to expand the sphere of interaction when application demands require it, and contract the sphere of interaction at all other times to reduce energy consumption and channel access contention in the vicinity

of the SAP. In an OSN architecture, the baseline is the lazy approach where we passively wait on mobility to bring suitable sensors to task, or previously tasked sensors carrying back sensed data within the radio range of the SAP when the transmit power is fixed. However, if some sensed data are most valuable if sensed and/or delivered within a particular time window, an improvement over the performance of this lazy approach is required.

Assume we have an application query i with which to task the sensor network that requires data from a particular sensor type (*e.g.*, CO_2 sensor). Suppose that the data must be sensed at time $t_{s_i}(min) \leq t \leq t_{s_i}(max)$ to capture the event of interest (*e.g.*, rush hour pollution), and that a constant T_i exists that reflects the time it takes to travel from the tasking SAP (which is assumed to know its location) and the sensing target location defined in the query, using average case human speed. When an application query is inserted into the SAP task queue at time t_i^0, we calculate the time until sphere expansion $\delta_{s_i}^0 = (t_{s(max)_i} - t_i^0) - T_i$. If a MS matching the task requirements is available for tasking within the current SAP sphere of interaction, then no sphere adjustment is necessary and the tasking operation can proceed. Otherwise, at any time t^j a SAP calculates its sensing-driven sphere adjustment multiplier ξ_s for query i as $\xi_{s_i} = (1 - \delta_{s_i}^j / \delta_{s_i}^0)$.

Similarly, assume for an application query i, an MS was previously tasked and was able to sample the requested target. Suppose the data must be delivered back to a SAP by time $t \leq t_u$ in order for the data to have the greatest utility, and T_i is defined as before. Then at any time t^j a SAP calculates its upload-driven sphere adjustment multiplier ξ_u for query i as $\xi_{u_i} = (1 - \delta_{u_i}^j / \delta_{u_i}^{s(max)})$, where $\delta_{u_i}^{s(max)} = (t_{u_i} - t_i^{s(max)}) - T_i$. Queries not specifying sensing target locations set T_i=0. Queries not having sensing or uploading deadlines, set $t_s(max) = \infty$ and $t_u = \infty$, respectively.

We use a small set of power settings at both the MSs and SAPs, and limit the maximum number of hops of sphere expansion to keep the cost and complexity of interactions low. The choice of the supported power levels can be arbitrary or based on historical information kept at the SAP about the number of MSs found at a given sphere radius. Let $P = \{p_1, ..., p_K\}$ be the supported power levels at each node and let M be the maximum allowed number of hops. Then there are $M \cdot K$ possible sphere extension settings, and we write the set of settings as $S = \{s_1, ..., s_{M \cdot K}\}$, where for s_j, the number of hops $m = 1 + \lfloor \frac{j-1}{K} \rfloor$ and the power setting of the last hop $k = j \ mod(K + 1)$ (hops prior to the last hop are at power setting p_K).

For a set of tasks Q at a particular SAP, the sphere of interaction is set according to

$$s = \max \left(\left\lceil \max_Q(\xi_{s_i}) \cdot M \cdot K \right\rceil, \left\lceil \max_Q(\xi_{u_i}) \cdot M \cdot K \right\rceil \right) \quad (1)$$

Following this rule, the sphere of interaction setting adapts to the current set of pending deadlines. Taking tasking as an example, as the time t^j gets closer to $t_{s(max)_i} - T_i$, then $\delta_{s_i}^j$ goes to 0 and the sphere setting grows to $M \cdot K$. On the other hand, if all pending deadlines are far enough in the future, then $\delta_{s_i}^j$ is still close to $\delta_{s_i}^0$ and the sphere setting shrinks close to the minimum. While a

number of variations on this scheme are possible, the rule in Equation 1 has the advantage of encouraging early submission of application queries to the system. Though outside the scope of our current work, early submission might allow for query load balancing among SAPs, and smart assignment to particular SAPs.

3 Scheduling Operations

We focus on two scheduling design choices that impact both the efficiency of communications between the SAPs and MSs, the average operation turnaround time and the average operation throughput. First, we discuss reasons for serving a single MS until its operation is completed (or it leaves the SAP range) rather than switching between multiple MS sessions. Second we discuss how the SAP determines the order in which it will serve the MSs in its current sphere of interaction. To support scheduling, the SAP takes a snapshot of the system at particular points in time. Within this freeze frame, the SAP knows: the set of application tasks to complete, the set of MSs in the sphere of interaction of a SAP available for tasking, the set of MSs in the sphere of interaction of a SAP offering data to upload, the set of applications waiting for particular data, and an estimation of each node's proximity and mobility.

Atomic vs. Interleaved Prior experimentation [4] with a testbed of Moteiv Tmote Invent motes (which use IEEE 802.15.4 radios) shows that at typical walking speeds and relatively low density of MSs, simultaneous uploading and tasking results in none of the operations being fully completed using state of the art sensor network transports. More recently, Miluzzo, *et al.* have characterized [13] the severe radio attenuation caused by the body that will be prevalent in human-centric networks, that will tend to limit the average contact time between SAP and MS even if higher data rate radios are used. Because of this limited contact time, an interleaved approach to operation scheduling (*i.e.*, either preemptive or simultaneous uploads and downloads) may not be appropriate. Firstly, a preemptive approach leads to a longer average turnaround time than non-preemptive scheduling [18] for all operations. Also, with single channel radios, simultaneous operations implies more MAC layer overhead in terms of either backoffs or collisions in the case of contention-based MACs, or schedule maintenance and dissemination in the case of non-contention-based MACs. Instead, we take a non-interleaved or atomic approach with at most one active uploading or tasking session ongoing at each SAP.

Scheduling Discipline To determine the order in-sphere MSs will be served, a simple approach is to not actively manage the order of operations at all and just serve MSs in the order they arrive at the SAP until they move out of range. However, service disciplines like FIFO or even random selection ignore important features such as the size (*i.e.*, number of bytes) of tasking and uploading operations, and the MS dwell time in the SAP sphere of interaction. Thus, these naive approaches can lead to a lower operation throughput due to non-uniform MS inter-SAP-visitation times, hereafter *orbits*.

We propose a hybrid mobility-based/shortest-job-first (MB-SJF) scheduling algorithm to decide the order in which MSs are atomically served. Let A denote

the event that a tasking or uploading operation supported by the current set of MSs can be completed before the chosen MS exits the maximum SAP sphere of interaction obtainable without multi-hop, since multi-hop extension is not always possible. Uploading and tasking operations are ordered by $Prob(A)$. This probability reflects the size of the operation to be completed (*i.e.*, number of bytes b) and the estimated dwell time, t_{SAP}, of the associated MS in the sphere of interaction. We have

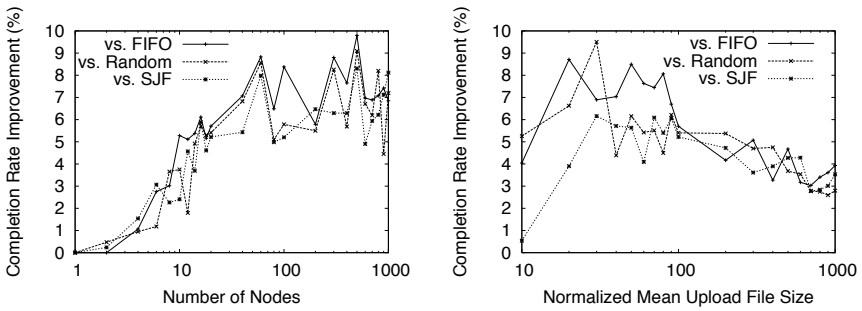
$$Prob(A) = Prob(t_{SAP} \geq \frac{b}{C} + \beta), \tag{2}$$

where C is the wireless channel rate in bytes per second, and $\beta = \overline{BO} \cdot \frac{b}{frame\_size}$, for a CSMA channel. Here, $\overline{BO}$ is the average MAC backoff interval across the packets needed to complete the operation, and $frame\_size$ is the maximum MAC frame size in bytes. It is worth nothing that with our atomic scheduling approach, the second term on the right side of the inequality can be driven to zero if the MAC parameters are tweaked such that a backoff window of zero is used during an upload/download session between a MS and the SAP (the standard backoff window would still be used for communication between MSs and for the MS↔SAP session setup [8, Appx II]).

In practice an empirically-derived mean value estimate of an MS's SAP dwell time $\overline{t}_{SAP}$ is easily tracked by the MS and shared with the SAP each rendezvous. Cold start effects on $\overline{t}_{SAP}$ are mitigated by seeding with values averaged across the MS population. Since we are considering the people-centric sensing domain, human activity inferred from on-board sensors can aid the MS in further refining its $\overline{t}_{SAP}$ estimate. In particular, samples from an accelerometer (embedded in many new mobile phone devices) can be processed to determine if a person is standing, walking, or running. In [12], the authors classify between these three states with an average accuracy of about 90%. Since average dwell times are likely to be highly correlated with human activity, we propose to keep a separate $\overline{t}_{SAP}$ for each classified activity and use this value in calculating Equation 2.

Once $Prob(A)_i$ is calculated for each {operation, MS} pair i in the sphere of interaction, the operation schedule is set in descending order of the value $Prob(A)_i * \nu_i$. The optional priority factor ν_i can be used to prioritize certain event types (*e.g.*, toxic spill) or users (*e.g.*, those with long average orbits), but the exact meaning is left up to the system administrator and it may be dropped altogether if user/operation priority need not be supported.

To evaluate the performance of MB-SJF, we simulate a one-SAP/multi-MS scenario where all MSs are assumed to have data to upload. We compare MB-SJF with common scheduling disciplines such as first-in-first-out (FIFO), random selection (RAND), and shortest-remaining-job-first (SJF) in terms of the time it takes to upload the data from all of the MSs. Each of the MSs is assigned a file to upload whose size is randomly chosen from an exponential distribution. MSs move between two states, at-SAP and not-at-SAP, where the dwell times (t_{SAP}) in each state are randomly drawn from different exponential distributions with means λ_{SAP} and $\lambda_{\overline{SAP}}$, respectively. To simulate a population of MSs with different mobility characteristics, each node is assigned a unique $\{\lambda_{SAP}, \lambda_{\overline{SAP}}\}$ pair, whose values are uniformly spread between 0 and $N\gamma$, where N is the



(a) (b)

Fig. 2: Advantage of MB-SJF versus other common scheduling disciplines in simulation. MB-SJF consistently completes the upload tasks faster than FIFO, RAND and SJF, across all tested parameter values for node population and mean file size.

number of MSs and γ is the spreading factor. The simulator updates MS states synchronously and both the file sizes and the location dwell times are normalized to its update period. For MB-SJF, it is assumed the MSs report their λ_{SAP} and $\lambda_{\overline{SAP}}$ values and remaining file size to upload (b) to the SAP upon entering the SAP's sphere of interaction (c.f., the *beacon reply* message in [8, Appx II]. Neglecting MAC effects, from Equation 2 the SAP computes $Prob(A) = e^{-\lambda_{SAP} b}$ for each node in its sphere. $\nu = e^{-\lambda_{\overline{SAP}}}$ prioritizes nodes with long orbits.

In Figures 2(a) and 2(b), we plot the completion rate improvement MB-SJF gives versus FIFO, RAND, and SJF. Each point represents the average of 1000 trials, each with a different seed for the pseudo-random number generator driving upload file sizes and location dwell times.

Figure 2(a) shows the completion rate improvement versus the number of MSs in the simulation, with a fixed mean upload file size of 100. As the MS population grows, the advantage of MB-SJF steadily increases until settling around a 6-10% improvement after 60 nodes. MB-SJF, like plain SJF, is able to finish off small upload tasks quickly, but also takes advantage of mobility information to opportunely upload from nodes that visit the SAP relatively rarely. Figure 2(b) shows the completion rate improvement versus the mean upload file size when there are 20 MSs. As expected, for medium size files (implying medium aggregate upload times), MB-SFJ provides for a relatively constant improvement in completion rate. As file sizes get larger, the improvement begins to diminish. As file sizes tend to infinity, so does the completion time regardless of scheduling discipline, and therefore the possible improvement goes to 0. However, we believe the typical case for opportunistic wireless uploads from mobile consumer devices will be small to medium size files (*e.g.*, a 1kB text file, a 1MB image file, a 10MB audio file). Based on these simulations, MB-SJF seems to be a good candidate for MS scheduling in the SAP sphere of interaction and we use MB-SJF for further evaluation in Section 4.

Scheduling Epoch MSs may enter a SAP's sphere of interaction during an ongoing schedule. In this case, the SAP could ignore all newcomers until its current schedule is complete and then come up with a new schedule that incorporates the newcomers, or it might create a new schedule upon the completion of every

operation. In the former case, starvation is prevented, but new sensors that may rank higher are ignored. In the latter case, more energy is spent and time wasted by re-running the neighbor discovery after every operation. In Halo, we define a scheduling epoch of time length E to strike a middle ground. E is adaptive to the estimated mobility of the MSs involved in the current schedule. Let K_i represent this set of MSs for a given schedule; then $E = \max_{K_i}(t_{SAP})$. The next scheduling time is then defined as

$$t_{sched_{i+1}} = t_{sched_i} + min\left(\sum_{j=1}^{n-1} \ell_j^{sched_i}, \sum_{j=1}^{|K|} \ell_j^{sched_i}\right), \quad (3)$$

where $\ell_j^{sched_i}$ is the length of the jth operation in schedule i, and n is the ordinal of the first operation that makes the sum greater than E. MSs that depart the SAP's sphere of interaction before their schedule slot are skipped. This method of addressing starvation is more appropriate than standard aging techniques since MSs with a lower $P(A)$ (later in the schedule) are not likely to be around very long and would not be able to take advantage of the aging. Thus, with the schedule epoch, we focus on getting the higher $P(A)$ operations done rather than on fairness with respect to starvation.

4 Halo Evaluation

We base our evaluation of deadline-driven sphere of interaction management on the comparison of three schemes: MIN, ADAPT, and MAX. We use ten SAP sphere of interaction settings $S = \{s_1, ...s_{10}\}$, with $M = 1$ and $K = 10$, where setting s_j corresponds to a sphere radius of $j * 3$ distance units. This relationship implies a field without radio obstructions, which is unlikely in a people-centric network. We make this simplifying assumption to avoid making arbitrary assumptions about the deployment environment. Note, however, that transmission distance will always increase monotonically with transmission power. In the MIN scheme SAPs always use s_1; in the MAX scheme SAPs always use s_{10}; in the ADAPT scheme each SAP independently varies its radius according to the sensing deadlines of tasks it is managing. All schemes use MB-SJF scheduling.

Simulation Environment We implement Halo algorithms and the tasking and uploading communications protocols (see [8, Appx II] for details) in nesC, and simulate several multi-SAP multi-MS scenarios using TOSSIM/Tython. TOSSIM simulates Halo on the TinyOS platform, including packet exchange, timer events, etc. Tython is a Python/Java front end used to manage node mobility and connectivity.

Each simulation trial is conducted on a 500×500 field. MSs are initially placed uniformly at random across the field and move according to a modified random walk. MSs choose an activity uniformly at random from {standing, walking, running} and continue with that activity for a period of time chosen uniformly at random between 1 and 1200 seconds. According to the chosen activity MSs move at a rate of, respectively, $\{0, 3, 15\}$ distance units per second in a direction chosen uniformly at random, between 1 and 360 degrees inclusive, at the same time the activity is chosen. MSs bounce off the field boundaries. 50 SAPs are placed uniformly across the field and remain stationary throughout the simulation.

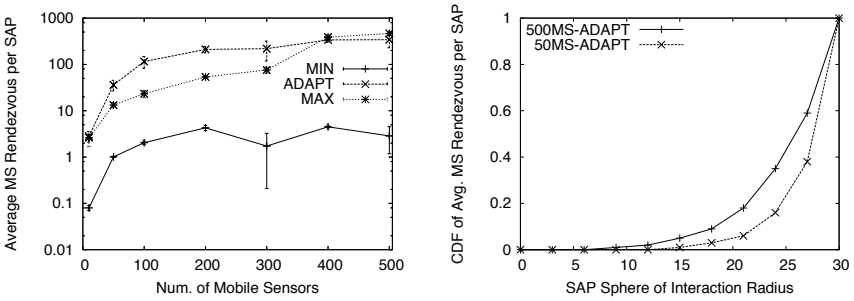
MSs estimate their t_{SAP} and communicate this estimate to the SAP during the tasking exchange [8, Appx II] to facilitate the MB-SJF schedule calculation (see Equation 2). We assume all MSs have an accelerometer that can be used for activity classification, and use the activity classifier confusion matrix published in CenceMe [12, Fig 6(a)] to drive the actual and reported mobility characteristics in the simulation. For example, a MS may be "running" as dictated by the mobility model, but the MS believes it is running with only 90.9% probability. With 8.37% probability it thinks it is walking and tells the SAP the wrong information. Real world effects such as classification inaccuracy (or GPS error if a GPS system is used as a basis for a dwell time estimate) degrade the performance of the MB-SJF scheduler. Yet, even under worst case classification accuracy, the scheduler just behaves as a random scheduler that does not consider mobility.

As no large-scale mobile sensor networks allowing external queries have yet been built, we make some best guesses at parameters characterizing the task arrival process. Emulating application requests from thousands of backend system users, tasks arrive independently at each SAP with inter-arrival times drawn randomly from an exponential distribution with a mean of 10s. Task sizes are drawn randomly from an exponential distribution with a mean of 10 packets (packets are 128 bytes long). The sensing deadline ($t_{s(max)}$) is randomly chosen for each task from an exponential distribution with a mean of 1000s. The deadline threshold T should reflect the time it takes on average to travel the distance from the tasking SAP to the sensing target (see Section 2). In our simulation, the T value for a given task is chosen uniformly at random in the interval from 1 to $t_{s(max)}$. This is equivalent to choosing a sensing target uniformly at random in the field and pre-filtering those tasks whose sensing deadlines do not allow enough travel time from SAP to target. A task whose sensing deadline passes before it is assigned to a MS is dropped from the SAP's task queue. The SAP's *beacon* interval [8, Appx II] is set to 5s.

Each MS has a task queue of size 1, meaning it can only serve one application query at a time. If a task is partially transferred to a MS before a particular tasking session completes, the MS caches the state of the suspended session and resumes the session at the next met SAP. If the sensing deadline of a task that is partially transferred to a MS expires, the partial state is expunged from the MS. MSs that are fully tasked and then successfully sense the target generate a number of data packets to upload chosen randomly from an exponential distribution with a mean of 100 packets. About 20% of the fully tasked MSs end up reaching their respective target sensing regions prior to their sensing deadlines. We use an uploading deadline of infinity for all tasks; a MS with data to upload maintains the state of its upload session across how ever many SAP rendezvous it takes to complete the upload.

In the following graphs, each data point represents the average of five one-hour trials, and error bars indicate the 95% confidence interval.

Impact on Tasking/Uploading Opportunity To characterize the impact of sphere of interaction radius on the opportunity for MS/SAP rendezvous, we run simulations across a range of MS densities. Results are summarized in Figure 3.



(a) Impact of MS density on the number of MS/SAP rendezvous.

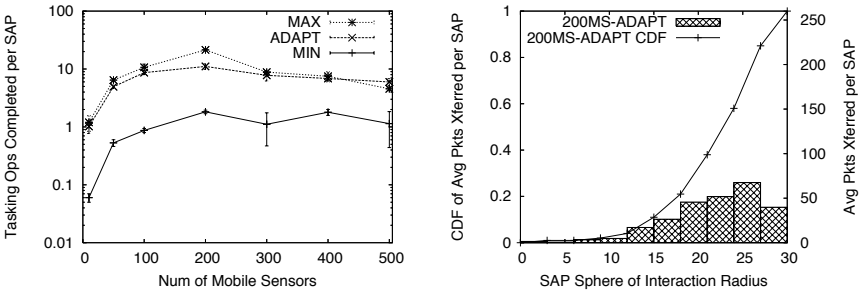
(b) Cumulative distribution of rendezvous vs. sphere radius for diffent MS densities (with ADAPT scheme).

Fig. 3

In Figure 3(a), we quantify the impact of MS density on the number of MS/SAP rendezvous, plotting the average number of MS/SAP rendezvous per SAP versus the number of MSs. The y-axis is in log scale to better show the detail despite the wide spread between MIN and MAX. Unsurprisingly, the number of rendezvous generally increases with increasing MS density for MIN, ADAPT and MAX. Of interest, the ADAPT scheme actually results in more rendezvous for the intermediate MS densities tested. This is likely due to the dynamism of the sphere of interaction, resulting in "re-rendezvousing" for MSs that have moved little (*e.g.*, standing) during the sphere adaptation time scale. The effect becomes negligible at the lowest densities (10 mobile sensors) since the overall probability of rendezvous shrinks dramatically. At high densities (*i.e.*, above 400 MSs), this effect is overwhelmed by the sheer number of mobility-based rendezvous, and at the highest density (500 MSs) MAX yields more rendezvous than ADAPT since it always uses the largest sphere radius. Another way to see the effect of MSs density is to consider the sphere of interaction radius at which most rendezvous occur. In Figure 3(b), we show the cumulative distribution function (CDF) of the average number of MS rendezvous per SAP versus SAP sphere of interaction radius. Curves for 500 MSs and 50 MSs show how at lower densities the majority of rendezvous occur at higher values of sphere radius. For example, at a sphere radius of 27 (second largest), in the 50 MS scenario only 40% of the rendezvous have occurred, while in the 500 MS scenario already 60% have occurred.

Impact on Bytes Transferred and Operations Completed Figure 4 summarizes the performance of the ADAPT scheme in terms of number of task and upload packets transferred between MSs and SAPs, and the number of tasking operations completed. Simulations are run across a range of MS densities.

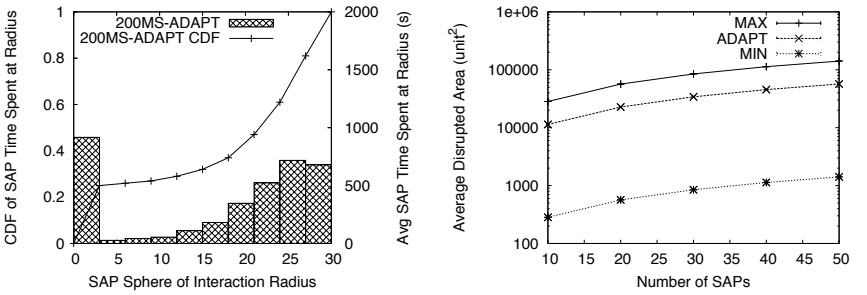
Figure 4(a) shows the average number of tasking operations completed per SAP plotted in log scale across a range of MS densities. We see that the behavior of adapting the sphere of interaction radius based on proximity to the sensing-deadline for a particular task leads to excellent comparative performance for ADAPT. On average ADAPT completes 85.5% of the tasks MAX does and nearly 10 times as many as MIN does.



(a) MIN completes an order of magnitude fewer tasking ops than ADAPT or MAX, giving poor service to apps.

(b) Packets xferred generally increases with radius, but the largest radius shows a diminishing return.

Fig. 4



(a) For ADAPT, SAPs spend a plurality of their time at the lowest setting, but the majority at the highest settings.

(b) Impact of SAP density on disrupted area. On avg., ADAPT disrupts $\approx \frac{2}{3}$ the area that MAX does.

Fig. 5

In Figure 4(b), we provide insight into why the operation completion performance of ADAPT is able to remain close to that of MAX. Figure 4(b) shows the distribution (on the right axis) and cumulative distribution function (on the left axis) of packets transferred across sphere of interaction radius for the median density scenario (200 MSs). We see that, in contrast to the rendezvous distribution shown in Figure 3(b), the packet transfer distribution does not monotonically increase with increasing sphere radius. Rather, the amount of additional packets transferred at the maximum sphere radius is less than the penultimate radius, indicating a diminishing return for increasing the sphere radius.

Impact on Disrupted Area Figure 5 summarizes the extent to which an increased sphere radius impacts the disrupted area. Since MS energy depletion is proportional to the sphere of interaction radius, we omit explicit energy-related results here. Figure 5(a) shows the distribution (and CDF) of the time that SAPs on average spend at each of the 10 sphere of interaction settings when using the ADAPT scheme. The data are from the median density 200 MS scenario, but are similar for the other tested MS densities. SAPs spend a plurality of their time (about 25%) at the lowest sphere setting, implying the minimum possible disruption. Yet, in aggregate most of the time is spent at or above the eighth

setting. In fact, the average sphere radius is about two thirds of that used by the MAX scheme. Figure 5(b) reflects this relationship and also indicates that the disturbed area in the field can quickly get very large as the SAP density increases (number of MSs is fixed at 200). The MIN scheme disrupts a much smaller area, but as we see in Figure 4 MIN also transfers and completes tasking operations at a rate an order of magnitude lower rate than ADAPT.

We define a unified efficiency metric as the average number of operations completed per unit area disrupted $\eta = Ops/Area$. On average across our tested desities $\eta_{\text{ADAPT}}/\eta_{\text{MIN}} = 0.27$ and $\eta_{\text{ADAPT}}/\eta_{\text{MAX}} = 2.18$; ADAPT gives a 200% improvement over MAX., while facilitating a nearly 10× improvement over MIN in terms of completed operations. While η provides a notion of efficiency, it also reflects the tradeoff between resource conservation, *i.e.*, disrupted area and MS energy, and a coarse-grained quality of service in terms of system responsiveness to application queries.

In Figure 6, we illustrate how adjusting the deadline threshold T can be used to move the operating point of the ADAPT scheme from more resource conserving to offering a lower average completion delay to application queries. For the 200-MS scenario, Figure 6 shows the CDF of the time that SAPs on average spend at each of the 10 sphere of interaction settings when using the ADAPT scheme for different values of the average deadline threshold, $\overline{T}$.

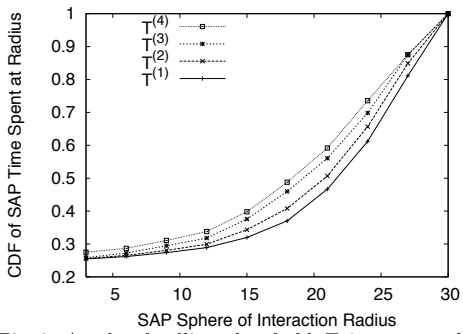


Fig. 6: As the deadline threshold T increases, the SAPs spend proportionally more time at higher sphere of interaction settings.

In the previous simulations, T is chosen uniformly between 1 and $t_{s(max)}$, so $\overline{T} = t_{s(max)}/2$. In Figure 6, we adopt the following shorthand: $T^{(i)} = t_{s(max)}/2 \cdot i$. As $\overline{T}$ increases, SAPs spend proportionally more time at higher sphere settings, resulting in more packets transferred and better delay service to the applications but consuming more resources of the MS cloud (*i.e.*, energy and peer-to-peer communications opportunities).

Discussion The MIN scheme provides the lowest energy consumption and disrupted area; the MAX scheme provides the greatest opportunity for packet transfer between MSs and SAPs, and thus the highest operation completion rate. In the previous sections we show that by dynamically adjusting the sphere of interaction, *i.e.*, transmit power and hop extension, according to the sensing deadlines of submitted application tasks ADAPT hits the sweet spot and provides increased opportunities for packet transfer and operation completion compared with MIN, and at a lower cost in terms of disruption to MS P2P communication compared to MAX. While we do not present explicit MS energy results, the packet reception energy is directly related to the disrupted area (see Figure 5(b)) and the packet transmission energy is directly related to the SAPs' transmit power setting (see Figure 5(a)). MS energy consumption and disrupted

area can be considered as system design parameters based, for example, on MS battery characteristics and expected P2P application traffic in the mobile cloud, to drive the selection of the deadline threshold T.

5 Related Work

Managing the SAP sphere of interaction is related to adaptive clustering. Work from the MANET community proposes various clustering techniques, but invariably to increase routing efficiency and/or reduce routing protocol overhead (*e.g.*, [9] [7] [6] [17]). Zone Routing Protocol [9] sets a zone boundary between proactive and reactive routing to reduce the number of route request packets while providing good route acquisition delay. However, a method of determining an appropriate zone radius is not specified.

A study of adaptive clustering with the end of maximizing operation completion rate does not exist. The relationship between node density, transmission power, and neighbor set cardinality has been studied in the context of wireless graph connectivity [1] [23]. The effects of the relationships between transmission power, node density, and node mobility patterns on the operation completion rate have not been reported. A number of existing scheduling policies may be appropriate for SAP operation scheduling, but none have been evaluated with the unique combination of constraints present in a large scale mobile sensor network, to the best of our knowledge.

Halo's adaptation of a SAP's sphere of interaction is analogous to the "cell breathing" approach used in cellular telephony and proposed [3] for 802.11 access nodes for system load balancing. With a similar aim, the authors of the SoftRepeater [2] system use a combination of network coding and channel utilization to decide when 802.11 AP clients should become repeaters for others' traffic in order to address the "rate anomaly problem". Rather than dealing with SAP overloading, we address what is in some ways the opposite problem of application starvation due to under-utilization of the SAP tier.

Power control in cellular systems aims to save handset energy and secondarily to reduce adjacent cell interference. While we share the first concern, cellular mechanisms are too complex, and use a separate control channel which is not generally available on embedded sensing platforms.

6 Conclusion

We have shown how by adapting SAPs' spheres of interaction Halo can manage the opportunity for interaction between mobile sensors and sensor access points, while striking a balance between resource consumption and operation completion. Halo uses a new scheduling discipline (MB-SJF), based on mobility statistics and sensor-based inputs, that is tailored for the typical characteristics of the people-centric sensing domain. MB-SJF is shown to provide up to a 10% increase in operation completion rate compared to FIFO, random selection, and shortest-job-first scheduling, independent of the gains achievable from SAP sphere of interaction management. Halo provides improved support for delay-aware applications in the people-centric sensing context.

Acknowledgment

This work is supported in part by Intel Corp., Nokia, NSF NCS-0631289, and the Institute for Security Technology Studies (ISTS) at Dartmouth College. ISTS support is provided by the U.S. Dept. of Homeland Security under Grant Award Number 2006-CS-001-000001 and by award 60NANB6D6130 from the U.S. Dept. of Commerce. The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the U.S. Dept. of Homeland Security.

References

1. A. Agarwal and P. R. Kumar. Capacity bounds for ad-hoc and hybrid wireless networks. In *ACM SIGCOMM CCR, Special Issue on Science of Networking Design*, pp. 71-81, Vol. 34, No. 3, July 2004.
2. P. Bahl, R. Chandra, P.-C. Lee, V. Misra, J. Padhye, D. Rubenstein and Y. Yu. Opportunistic Use of Client Repeaters to Improve Performance of WLANs. In *Proc. of ACM CoNEXT'08*, Madrid, Dec 2008.
3. P. Bahl, M. Hajiaghayi, K. Jain, V. Mirrokni, L. Qiu, and A. Saberi. Cell Breathing in Wireless LANs: Algorithms and Evaluation. In *IEEE Trans. on Mobile Computing*, Vol. 6, No. 2, Feb 2007.
4. A. Campbell, S. Eisenman, N. Lane, E. Miluzzo, and R. Peterson. People-Centric Urban Sensing. In *Proc. WICON'06*, Boston, Aug 2006.
5. A. Chaintreau, A. Mtibaa, L. Massoulie and C. Diot. The Diameter of Opportunistic Mobile Networks. In *Proc. of ACM CoNEXT'07*, New York, Dec 2007.
6. M. Chatterjee, S. K. Das, and D. Targut. WCA: A Weighted Clustering Algorithm for Mobile Ad hoc Networks. In *Journal of Cluster Computing (Special Issue on Mobile Ad hoc Networks)*, Vol. 5, Apr 2002.
7. S. Du, et al. Self-Organizing Hierarchical Routing for Scalable Ad Hoc Networking. In *ACM Ad Hoc Networks*, Vol 6, Iss 4, Jun 2008.
8. S. B. Eisenman and A. T. Campbell. Managing Node Rendezvous is Opportunistic Sensor Networks. Tech. Report, `http://www.ee.columbia.edu/~shane/halo.pdf`
9. Z. J. Haas. A New Routing Protocol for the Reconfigurable Wireless Networks. In *Proc. ICUPC'97*, San Diego, Oct 1997.
10. B. Hull, *et al.* CarTel: A Distributed Mobile Sensor Computing System. In *Proc. of 4th ACM Int'l Conf. on Embedded Networked Sensor Systems*, pp 125–138. Boulder, Nov 2006.
11. M. Laibowitz and J. A. Paradiso. Parasitic Mobility for Pervasive Sensor Networks. In *Proc. of 3rd Int'l Conf on Pervasive Computing*, pp. 255-278, 2005.
12. E. Miluzzo, N. D. Lane, S. B. Eisenman, and A. T. Campbell. CenceMe - Injecting Sensing Presence into Social Networking Applications. In *Proc. EUROSSC'07*, pp. 1–28, 2007.
13. E. Miluzzo, X. Zheng, K. Fodor and A. T Campbell. Radio Characterization of 802.15.4 and its Impact on the Design of Mobile Sensor Networks. In *Proc. EWSN'08*, Bologna, Jan 30/31 - Feb 1, 2008.
14. A. Parker, S. Reddy, T. Schmid, K. Chang, G. Saurabh, M. Srivastava, M. Hansen, J. Burke, D. Estrin, M. Allman and V. Paxson. Network System Challenges in Selective Sharing and Verification for Personal, Social, and Urban-scale Sensing Applications. In *Proc. of HotNets-V*, Irvine, Nov 2006.
15. N. Ravi, P. Stern, N. Desai, L. Iftode. Accessing ubiquitous services using smart phones. In *Proc. PERVASIVE'05*, Mar 8-12, 2005.
16. Sensorplanet. `http://www.sensorplanet.org/`.
17. J. Sharony. A Mobile Radio Network Architecture with Dynamically Changing Topology Using Virtual Subnets. In *Proc. ICC'96*, pp. 807–812, Vol. 2, Jun 1996.
18. A. Silberschatz, P. B. Galvin, and G. Gagne. *Operating Systems Concepts, Seventh Edition*, John Wiley & Sons, Inc., 2004.
19. K. Srinivasan and P. Levis. RSSI is Under Appreciated. In *Proc. EMNETS'06*, Cambridge, MA, 2006.
20. S. Srinivasan, A. Moghadam, S. G. Hong, H. G. Schulzrinne 7DS - Node Cooperation and Information Exchange in Mostly Disconnected Networks. In *Proc. ICC'07*, Jun 1, 2007.
21. M. Srivastava, et al. Wireless Urban Sensing. In *CENS Tech. Report #65*, Apr 2006.
22. V. Tuulos, J. Scheible and H. Nyholm. Combining Web, Mobile Phones and Public Displays in Large-Scale: Manhattan Story Mashup. In *Proc. of the 5th Int'l Conf. on Pervasive Computing*, Toronto, May 2007.
23. X. Zhang and N. F. Maxemchuk. The Effects of the Number of Neighbors in Multihop Wireless Networks. To appear in *Int'l Journal of Wireless and Mobile Computing, Special Issue on Group Communications in Ad hoc Networks*.