Practical scheduling for stochastic event capture in energy harvesting sensor networks

Haipeng Dai, Xiaobing Wu* and Lijie Xu

State Key Laboratory for Novel Software Technology,
Nanjing University,
Nanjing, Jiangsu 210023, China
Email: dhpphd2003@gmail.com
Email: wuxb@nju.edu.cn
Email: ljxu83@gmail.com
*Corresponding author

Fan Wu

Department of Computer Science and Engineering,
Shanghai Jiao Tong University,
Shanghai 200240, China
Email: fwu@cs.sjtu.edu.cn

Shibo He

State Key Laboratory of Industrial Control Technology,
Zhejiang University,
Hangzhou, Zhejiang 310027, China
Email: shibohe@asu.edu

Guihai Chen

State Key Laboratory for Novel Software Technology,
Nanjing University,
Nanjing, Jiangsu 210023, China
and
Department of Computer Science and Engineering,
Shanghai Jiao Tong University,
Shanghai 200240, China
Email: gchen@nju.edu.cn

Abstract: Existing scheduling schemes for stochastic event capture with rechargeable sensors either adopt simplified assumptions on event's properties or provide no performance guarantee. Considering the stochasticity of event staying time and event capture utility, we investigate the sensor scheduling problem aiming to maximise the overall quality of monitoring (QoM) in event capture application of energy harvesting sensor networks. We first provide a paradigm to calculate the QoM of a point of interests (PoI) and formulate the scheduling problem as an optimisation problem. Although we find that this problem is NP-complete, we prove that the problem can be cast as maximisation of a submodular function subject to matroid constraints. Accordingly, we can design centralised and distributed algorithms, each of which achieves a factor of $1/2$ of the optimum. We evaluate the performance of our solution through simulations, and simulation results show that our scheme outperforms former works.

Keywords: energy harvesting; sensor networks; events capture; scheduling; submodularity.

Reference to this paper should be made as follows: Dai, H., Wu, X., Xu, L., Wu, F., He, S. and Chen, G. (xxxx) 'Practical scheduling for stochastic event capture in energy harvesting sensor networks', *Int. J. Sensor Networks*, Vol. x, No. x, pp.xxx–xxx.

Biographical notes: Haipeng Dai is currently pursuing the PhD at the Department of Computer Science and Technology in Nanjing University, Nanjing, China. His research interests are mainly in the areas of wireless sensor networks.

Xiaobing Wu received PhD in Computer Science from Nanjing University (NJU) in 2009 where he is an Assistant Professor. His research interests are mainly in the areas of wireless networking. He is a Member of the IEEE and the ACM.

Lijie Xu is currently a PhD student at Department of Computer Science and Technology, Nanjing University, China. He received his BS and MS from Yangzhou University in 2006 and 2009, respectively. His research mainly includes wireless sensor networks and ad-hoc networks.

Fan Wu is an Associate Professor in the Department of Computer Science and Engineering at Shanghai Jiao Tong University, China. He received his BS in Computer Science from Nanjing University in 2004, and PhD in Computer Science and Engineering from the State University of New York at Buffalo in 2009. He has visited the University of Illinois at Urbana-Champaign (UIUC) as a Post Doc Research Associate. His current research interests include wireless networking, algorithmic mechanism design, and privacy preservation. He received Excellent Young Scholar award of Shanghai Jiao Tong University in 2011, and Pujiang Scholar award in 2012. He is a member of ACM, CCF, and IEEE. Shibo He received PhD in Control Science and Engineering at Zhejiang

University, Hangzhou, China. He is a Member of the Group of Networked Sensing and Control (IIPC-nesC) in the State Key Laboratory of Industrial Control Technology at Zhejiang University. His research interests include coverage, cross-layer optimisation, and distributed algorithm design problems in wireless sensor networks.

Guihai Chen received BS in computer software from Nanjing University in 1984, ME in Computer Applications from Southeast University in 1987, and PhD in Computer Science from the University of Hong Kong in 1997. He is a professor and deputy chair of the Department of Computer Science, Nanjing University, China. He had been invited as a Visiting Professor by many foreign universities including Kyushu Institute of Technology, Japan in 1998, University of Queensland, Australia in 2000, and Wayne State University, USA during September 2001–August 2003. He has a wide range of research interests with focus on sensor networks, peer-to-peer computing, high-performance computer architecture and combinatorics.

This paper is a revised and expanded version of a paper entitled 'Practical scheduling for stochastic event capture in wireless rechargeable sensor networks' presented at *Proc. of the 2013 IEEE Wireless Communications and Networking Conference (WCNC)*, Shanghai, China, 7–10 April, 2013.

1 Introduction

Wireless sensor networks (WSNs) are mainly powered by small batteries, and the limited energy supply constrains the lifetime of WSNs. Fortunately, recent studies have shown that energy harvesting sensor networks have the potential to provide perpetual network operations by capturing renewable energy from the ambient environment. The benefits of using rechargeable batteries to prolong sensor network life by harvesting environmental energy such as vibration (Meninger et al., 2001), solar (Raghunathan et al., 2005), wind (Park and Chou, 2006), thermal (Stark, 2006), and RF signals (He et al., 2011; Dai et al., 2013a, 2013b, 2013c, 2014), have been well recognised.

However, as indicated by many real experiment results, usually the energy harvesting rate for rechargeable sensors is relatively slow compared with the discharging rate. Due to this reason, rechargeable sensors are forced to schedule their duty cycles for energy efficiency. Since continuous operations of sensors are no longer guaranteed, the activation policies need to be carefully designed for sensors to optimise system performance. In this paper, we consider the optimal activation policies for stochastic event capture with rechargeable sensors.

Event capture in WSNs has been studied for a long period of time, and it can be applied to various scenarios such as populated area protection against physical hazards, and forest monitoring for environmental changes like temperature and humidity. As for energy harvesting sensor networks, event capture is also by no means a new topic. However, as far as we know, almost all existing literatures either overlooked the stochasticities of event staying time or event capture utility, or made simple assumptions about sensor coverage. For example, Jaggi et al. (2007), Ren et al. (2012) and Jaggi et al. (2011) focuses on how to exploit on-line information of event occurrence by applying dynamic activation policies to optimise the QoM. Here QoM is defined as the ratio of interesting events captured to all occurred events. The events in these works are simply assumed to either leave some evidence after its disappearance such that they can ultimately be identified, or like 'blips' that can be observed only when the sensor is active in the time slot concerned. This assumption is not practical since Yau et al. (2010) showed that event staying time can be expressed as a stochastic process. Besides, existing literature in energy harvesting sensor networks typically focused on simple coverage cases to make the problem amenable. For example, Jaggi et al. (2007), Ren et al. (2012) and Jaggi et al. (2011) have studied the coverage problem where single or multiple sensors jointly cover one single PoI, which is definitely impractical.

In this paper, we study the stochastic event capture problem in energy harvesting sensor networks under practical concerns, namely, taking into account both stochasticities of event staying time and event capture utility and general sensor coverage settings (each sensor covers multiple PoIs, and adjacent sensors might cover nonempty PoIs in common). Particularly, we associate each PoI with a predefined weight to quantify its importance (Liu and Cao, 2012). The QoM of a single PoI is defined as the product of its weight and the average obtained utility of events, and the overall QoM of the network is given by the sum of QoMs of all PoIs. After all, our problem is how to schedule the sensor activations to achieve the optimal overall QoM. We show that this problem is NP-complete, and propose effective centralised and distributed scheduling schemes to address it, both of which achieve a constant factor of the theoretical optimum. More importantly, our theoretical can accommodate cases where events have arbitrary event staying time function and arbitrary concave event capture utility function.

Our theoretical analysis mainly exploits the submodular property of the overall QoM function. Informally, submodularity is an intuitive notion of diminishing returns, i.e., adding an element to a smaller subset of S helps more than adding it to a larger subset of S. Submodular functions play an important role in discrete optimisation. While submodular property has previously been used for sensor coverage (Abrams et al., 2004), placement (Krause and Guestrin, 2007; Krause, 2008; Bian et al., 2006; Krause et al., 2008), we are the first to disclose and exploit submodularity of scheduling for stochastic event capture in energy harvesting sensor networks. Besides, our problem is different from those partially observable stochastic optimisation problems, which are suitable for adaptive submodular optimisation (Golovin and Krause, 2011a, 2011b). This is because we assume that the stochastic property of events has already been determined by observation or other means.

Specifically, the contributions of this work are four-folds:

- We consider the scheduling problem in dense energy harvesting sensor networks for stochastic event capture in a practical way, i.e., taking into account the stochasticity of events in terms of event staying time and event capture utility. To facilitate the following analysis, we provide a paradigm to calculate the QoM of a PoI in the presence of single or multiple monitoring sensors.
- We formalise the scheduling problem, i.e., the question of how to best schedule the sensor activations to achieve the highest QoM. We show that this scheduling problem is NP-complete. Subsequently, we express the problem as a maximisation of a submodular function subject to matroid constraints.
- We use approximation results for submodular functions to design a centralised greedy algorithm which achieves a factor of $1/2$ from optimality. We proceed to devise a distributed algorithm and prove its approximation ratio is also $1/2$. Most importantly, these schemes accommodate general event utility function and probability distribution of the event staying time.
- We conduct extensive simulation to verify our findings. We compare the performance of our algorithm with that of optimal solution in small-scale networks. For large-scale networks, it is shown that our algorithm outperforms the former work, CSP.

The remainder of the paper is organised as follows. In Section 2, we give preliminaries and a formal definition of our problem. In Section 3, we prove the complexity of the problem and formulate it as a maximisation of a monotone submodular function subject to a partition matoid constraint. In addition, we also present a greedy algorithm to the problem. Section 4 presents extensive simulations to verify our theoretical findings. Before concluding this work in Section 6, we discuss the related work in Section 5.

2 Related work

We review related work in this section. In energy harvesting sensor networks, Jaggi et al. (2007) exploited the temporal correlations in the event occurrences to develop efficient activation policies. From a more general view, Ren et al. (2012) focused on general renewal processes, where the event arrival time can be drawn from an arbitrary probability distribution. There is also some work studied the cases where the derived utility, namely the probability of event detection, is determined only by the number of currently activated sensors (Kar et al., 2005; Jaggi, 2006; Tang et al., 2011). In particular, Tang et al. (2011) assumed that the utility function is monotone submodular, and provided a polynomial time algorithm which guarantees a constant approximation. However, his solution only suits homogeneous sensors whose discharging/recharging pattern are identical. Clearly, these utility based methods do not apply to stochastic event capture, as the QoM in this case depends on sensor schedules rather than their number.

In similar scenarios of duty cycle sensor networks, He et al. (2009, 2012a) considered the energy efficiency and the coordination issues between sensors in synchronous and asynchronous dense networks. In these literatures, He applied a simple (q, p) duty-cycle schedule, i.e., each sensor randomly starts their identical schedule independently. Basically, they analysed the event capture probability mainly from the view of probability theory. Furthermore, they also proposed a coordinated sleep protocol (CSP) to achieve an energy balance for maximum network lifetime when there is significant spatial overlap in the sensing regions of sensors. Though CSP performs well according to their simulation results, it is essentially an intuitive approach, and provides no performance guarantee with respect to QoM. He also covered a complementary problem with respect to connectivity (He et al., 2012b).

For mobile coverage, Yau et al. (2010) considered the impact of event arrival time and staying time on QoM, which were regarded as stochastic processes. Especially, mobile sensors were used to expand the area of coverage, and the associated periodic coverage problem was studied to optimise the overall QoM. He et al. (2010) extended the results of the periodic coverage problem by incorporating the energy constraints of mobile sensors, as well as the energy consumption of senor's motion. Jiang et al. (2011) used reader capable of mobility and functioning as energy distributors and data collectors to charge sensors. Reader mobile strategy for energy distribution and sensor schedules were jointly considered for efficient event capture. These results provide no performance guarantee and cannot be applied to optimal scheduling in energy harvesting sensor networks in terms of QoM maximisation.

3 Problem formulation

3.1 Network model

We assume that m sensors $V = \{v_1, v_2, \ldots, v_m\}$ distributed over a 2D region covering n PoIs $O = \{o_1, o_2, \ldots, o_n\}$. In particular, suppose that sensor v_i covers a subset of PoIs O_i. Adjacent sensors can cover nonempty PoIs in common. Accordingly, PoI o_i might be covered by a subset of sensors V_i. A base station serves as a sink, and requires each sensor to report its current energy level and other useful information to it hop by hop on a regular basis. The collection tree protocol (CTP) (Gnawali et al., 2009) is used as the routing protocol for sensors.

Assume that time is divided into time slots and the duration of a time slot is fixed and given a priori. Every T time slots, the base station determines the periodic schedules for the next T interval of all sensors, and disseminates them to sensors. The periodic schedules followed by sensors are of identical length $\mathscr{L}$. We name such a period starting from the scheduling process as *scheduling period*, and let T be a multiple of $\mathscr{L}$. A sensor can schedule itself to be active or inactive in any time slot. Therefore, the schedule of sensor v_i can be expressed by a vector $S_i = (a_{i1}, a_{i2}, \ldots, a_{i\mathscr{L}})$, where component $a_{ij} = 1$ indicates the sensor is active in time slot j while $a_{ij} = 0$ means the opposite. After all, we assume that the reporting processes of sensors take place with so low frequency (e.g., one time per hour) that the energy overhead can be ignored.

3.2 Recharging model and energy consumption model

Much existing work reports that the energy harvesting rates in many cases are of high variability, and the environmental energy model can be viewed as a stochastic process (Ren et al., 2012; Jaggi et al., 2011). However, for a wide range of application scenarios, such as indoor environment, the energy availability has been proved to be time-dependent and predictable (Tang et al., 2011; Yang et al., 2010, 2009; Hsu et al., 2006; Piorno et al., 2009). For instance, by effectively taking into account both the current and past-days weather conditions, Piorno et al. (2009) obtains a relative mean error of only 10%. Hence, with the knowledge of the accurate harvesting energy prediction of the next scheduling period for a sensor, along with the current residual energy, one can make a rational decision on energy budget for the sensor in the next scheduling period. Note that a valid decision should guarantee that sensors will never run out of energy.

Besides, the introduction of ultra-capacitors (up to 3000 F) can effectively offset the variability of the harvesting energy, and ensures stable power of harvesting (Zhu et al., 2009). To a great extent, this approach eliminates the dependence of activities of sensors on the environmental energy model. As a result, one can simply take all the residual energy in the capacitor at the end of the latest scheduling period as the energy budget in the next scheduling period.

Nevertheless, the energy budget determination is out of the scope of this paper, we simply assume that the energy budget of the next scheduling period T for sensor v_i has already been determined, and is denoted as e_T^i. In addition, assume that sensor v_i consumes δ_i energy for sensing and capturing an event in one time slot, but negligible energy being inactive, or switching between states provided that the duration of a time slot is set to be long enough. For simplicity, assume $l_i = \frac{e_T^i \mathscr{L}}{\delta_i T}$ for each sensor v_i is integral, which means that the charged power for sensor v_i can be equivalently converted to no more than l_i active time slots in periodic schedule S_i. Thus schedule S_i is subject to $||S_i||_1 \le l_i$ where $||S_i||_1$ is the L_1 norm of S_i. We call l_i the active time slot budget of sensor v_i.

A summary of the notations in this paper is given in Table 1.

Table 1 Definitions of notations

Notation	*Definition*
o_i	PoI i
v_i	Sensor i
O_i	Subset of PoIs covered by sensor v_i
V_i	Subset of sensors cover PoI o_i
$\mathscr{L}$	Sensor schedule length
l_i	Active time slot budget of sensor v_i
S_i	Schedule of sensor v_i
$\widehat{S}_i$	Equivalent monitoring schedule for PoI o_i
w_i	Weight of PoI o_i
$N(v_i)$	Neighbour set[1] of sensor v_i

[1]We adpot a new concept of neighbours in this paper: two sensors are neighbours to each other if and only if they cover PoIs in common.

3.3 Event model, QoM concept and properties

In this section, we first present a set of assumptions regarding the event dynamics and the properties of the sensors. Then we propose a general paradigm to compute the QoM for a PoI when it is monitored by one or more sensors.

For the event dynamics, we assume that the events at a PoI occur one after another, and the events at the same PoI or different PoIs are spatially and temporally independent (Yau et al., 2010; He et al., 2009; Jiang et al., 2011; Dai et al., 2013d, 2013e). After its occurrence, an event stays for some random time before it disappears. We denote by X the event staying time. Similarly, the time duration before the next event occurs, which we call the event inter-arrival time, is random and denoted by Y. Hence the sequence of event arrivals and departures forms a stochastic process. By renewable theory, the expected number of event arrivals in a time interval dt equals $\mu_i \mathrm{d}t$, where $\mu_i = 1/E(Y)$. As for the event staying time X, we assume that the probability distribution function of X is $f(x)$.

We use a binary sensing model for the sensors (Kar et al., 2003). Assume that the jth occurring event at PoI i is denoted as e_j^i, which is within range of a sensor for a total (but not necessarily contiguous) amount of time $t_j^i (t_j^i \geq 0)$. We assume that the sensor will, as a result, gain an amount of information $U_j^i(t_j^i)$ about e_j^i, where $U_j^i(x)$ is the utility function of e_j^i. For simplicity, we assume that $U_j^i(x) = U(x)$ for all the events at all the PoIs. We assume that the utility function has the following property.

Observation 3.1: The utility function $U(x)$ increases monotonically from zero to one as a function of the total observation time, i.e., $U(x) \geq 0$ and $U(y) - U(x) \geq 0$ for any $y \geq x \geq 0$.

Another important assumption is that the events are *identifiable* (Yau et al., 2010) (please refer to Yau et al. (2010) for a justification to the assumption). That is, when more than one sensor detects the same simultaneously, they will know that it is the same event. Furthermore, if more than one sensor observes the same event simultaneously, they learn exactly the same information.

To make our analysis more general, we associate with each PoI o_i a normalised weight w_i as in Liu and Cao (2012). Therefore, the QoM of sa single PoI is defined as the product of its weight and the average obtained utility of events, and the overall QoM is expressible as a sum of the individual QoMs. Formally, the overall QoM can be defined as:

$$\text{QoM} = \sum_{i=1}^{n} \text{QoM}(i) = \sum_{i=1}^{n} w_i \lim_{t \to \infty} \frac{\sum_{j=1}^{m_i} \widehat{U}_j(t)}{m_i} \tag{1}$$

where m_i denotes the number of events occur at PoI o_i until time t, and $\widehat{U}_j(t)$ denotes the aggregate achieved utility for event j at PoI o_i till time t. As will be seen below, it is extremely complicated to calculate $\widehat{U}_j(t)$ because the same event may be captured by more than one sensor, each of which may capture the event several times at different moments.

Prior QoM analysis either considers that a PoI is covered by only one sensor (Yau et al., 2010), or considers only special cases of the event utility function and event dynamics (He et al., 2009; Ren et al., 2012) (for example, only the step utility function is considered in He et al. (2009)). We generalise the prior analysis to cover other types of the events as well.

Definition 3.1 (Periodic extension function)**:** Given a schedule S_i of sensor v_i, the periodic extension function $\mathbf{S}_i(x)\,(\mathbf{S}_i : [0, +\infty] \mapsto \{0, 1\})$ of S_i is defined as:

$$\mathbf{S}_i(x) = \begin{cases} 1, & (x \in [k\mathscr{L} + j - 1, k\mathscr{L} + j], k \in \mathscr{N}, \\ & S_i(j) = 1) \\ 0, & \text{otherwise} \end{cases} . \tag{2}$$

We first present the following lemma, which is similar to Theorem 7 in Yau et al. (2010).

Lemma 1: *The QoM of a PoI, say o_i, covered by a single sensor $v_j\,(v_j \in V_i, |V_i| = 1)$ with schedule S_j, whose periodic extension function is $\mathbf{S}_j(x)$, is given by:*

$$\begin{aligned} &QoM(i|S_j) \\ &= \frac{w_i}{\mathscr{L}} \int_0^{\mathscr{L}} \int_t^{+\infty} U\left(\int_t^y \mathbf{S}_j(x)\mathrm{d}x\right) f(y-t)\mathrm{d}y\mathrm{d}t. \end{aligned} \tag{3}$$

Proof: The above formula follows from the fact that the overall utility available for any particular event, which starts at time t ($t \in [0, \mathscr{L}]$) and ends at time y ($y \in [t, +\infty)$), depends on the total length of the intersecting region $\int_t^y \mathbf{S}_j(x)\mathrm{d}x$. □

Notice that due to the stochastic property of event staying time and event capture utility, the above deterministic form of QoM actually embodies the expected performance of QoM in the long run. Besides, our problem is different from those partially observable stochastic optimisation problems (Ren et al., 2012) which can be resolved by theoretical techniques such as partially observable Markov decision processes and adaptive submodular optimisation (Golovin and Krause, 2011a,b), since we assume that the probabilistic distributions of event staying time and event capture utility have already been determined by observation or other means.

Suppose $S_i = (a_{i1}, \ldots, a_{i\mathscr{L}})$ and $S_j = (a_{j1}, \ldots, a_{j\mathscr{L}})$ are two different vectors, we define 'OR' operation of vectors as $S_i \vee S_j = (a_{i1} \vee a_{j1}, \ldots, a_{i\mathscr{L}} \vee a_{j\mathscr{L}})$.

Lemma 2: *The QoM of PoI o_i covered by multiple sensors $V_i = \{v_{1'}, v_{2'}, \ldots, v_{m'}\}$, each of which has schedule $S_j\,(j = 1', 2', \ldots, m')$, is given by:*

$$\begin{aligned} QoM(i) &= QoM(i|S_{1'}, S_{2'}, \ldots, S_{m'}) \\ &= QoM\left(i|\bigvee_{v_j \in V_i} S_j\right). \end{aligned} \tag{4}$$

Hence, the QoM achieved by the multiple sensors can be equivalently viewed as that by one single sensor with schedule $\bigvee_{v_j \in V_i} S_j$.

Proof: This follows directly from the identifiable assumption. We omit the details to save space. □

For simplicity of exposition, we call $\widehat{S}_i = \bigvee_{v_j \in V_i} S_j$ the *equivalent monitoring schedule* for PoI o_i. We stress that our analysis can compute the QoM of a PoI in the presence of both single and multiple monitoring sensors. It can also accommodate general activation schedules, event utility functions, and probability distributions of the event staying times $f(x)$.

3.4 Problem statement

With the above assumptions, our problem can be described as follows.

Consider an energy harvesting sensor network consisting of m sensors and n PoIs with known weights. Given the active time slot budget for each sensor in the next scheduling period l_i, $i = 1, \ldots, m$, how to determine the periodic schedule for each sensor to maximise the overall QoM.

Mathematically, we can formally state our problem as follows:

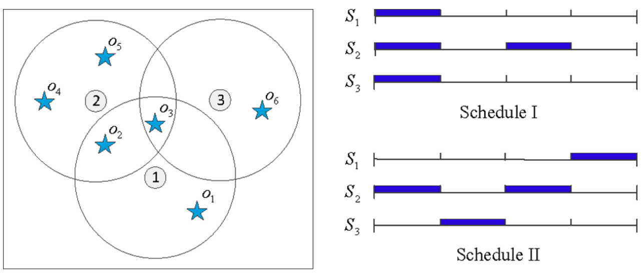
$$\begin{aligned} (P_1): \quad & \text{Max} \sum_{i=1}^{N} QoM(i) \\ & \text{s.t.} ||S_i||_1 \leq l_i \quad \forall i = 1, 2, \ldots, m, \end{aligned}$$

Notice that $QoM(i)$ can be calculated according to equations (3) and (4). During this procedure, the equivalent monitoring schedule $\widehat{S}_i$ for each PoI o_i needs to be calculated. $\widehat{S}_i$ is ultimately determined by those schedules of sensors covering o_i. We can see that such a relationship is quite complicated, which make our problem highly challenging.

3.5 *A numerical example*

In this section, we present a simple example for illustration. As shown in Figure 1(a), suppose there are three sensors covering six PoIs in the region. The sensor schedule length $\mathscr{L} = 4$, and the active time slot budgets for v_1, v_2 and v_3 are 1, 2 and 1, respectively. Besides, the weights are uniform among PoIs, namely, each PoI has weight $1/6$. In addition, we assume events have step utility function (Yau et al., 2010) (i.e., the utility reaches one instantaneously once an event is detected), and the event staying time follows the exponential distribution with mean $\lambda = 1$.

Figure 1 An example of scheduling, where three sensors monitor six PoIs: (a) topology and (b) two schedules of sensors (see online version for colours)



(a) (b)

We compare two schedules of sensors as depicted in Figure 1(b). It can be seen that the schedules of three sensors in Schedule I are set to $S_1 = (1,0,0,0)$, $S_2 = (1,0,1,0)$ and $S_3 = (1,0,0,0)$, respectively. We can then easily derive the equivalent monitoring schedule $\widehat{S}_i$ for each PoI, e.g., $\widehat{S}_1 = S_1 = (1,0,0,0)$ and $\widehat{S}_2 = S_1 \vee S_2 = (1,0,1,0)$.

Using equation (3), we can derive the exact QoM of a PoI for events with step utility function and exponential distribution of event staying time. Here we give the result directly to save space. Note that we obtained the same result by straightforward analysis in previous work (Dai et al., 2013a).

We first define the regularisation expression of the equivalent monitoring schedule $\widehat{S}$ as $R(\widehat{S}) = (p_0, q_1, p_1, q_2, p_2, \dots q_k, p_k)$, where q_i denotes the length of successive inactive time slots, and p_i the length of successive active time slots. For instance, $R(\widehat{S}_1) = (p_0, q_1, p_1) = (0,1,3)$ and $R(\widehat{S}_2) = (p_0, q_1, p_1, q_2, p_2) = (0,1,1,1,1)$. Then the QoM of a PoI can be written as

$$\text{QoM}(i) = w_i \left(\frac{\sum_{i=1}^{k} q_i}{\mathscr{L}} + \frac{\sum_{i=1}^{k-1}(1 - e^{-\lambda p_i}) + 1 - \mathrm{e}^{-\lambda(p_0 + p_k)}}{\lambda \mathscr{L}} \right).$$

Following this equation, the QoM of both PoI o_1 and o_6 is 0.0813, and the QoM of $o_2 - o_5$ is 0.1360. The overall QoM of the network is thus $QoM = 2 * 0.0813 + 4 * 0.1360 = 0.7066$.

Schedule I is not efficient since coverage waste arises as the first time slot of PoI o_2 is covered by both sensor v_1 and sensor v_2, while that of PoI o_3 is covered by sensor v_1, v_2, and v_3 simultaneously. Schedule II is an alternative to improve the performance of Schedule I. By rescheduling the active time slots of sensor v_1 and v_3, the QoM of o_2 and o_3 can be improved to 0.1513 and 0.1667 and the QoM of other PoIs remains unchanged, which results in an overall QoM 0.7526. In fact, Scheduling II yields the optimal overall QoM, since each QoM of PoI is maximised given the active time slot budgets of its monitoring sensors.

In the following section, we will first show that the problem stated above is NP-complete. After that, we formulate the problem into a maximisation of a monotone submodular function subject to a matroid constraint. Moreover, we present centralised and distributed algorithms both achieving provable results within a constant factor from optimality.

4 Theoretical analysis

In this section, we first show that the scheduling problem is NP-complete. Then we come up with centralised and distributed approximation algorithms with performance guarantee.

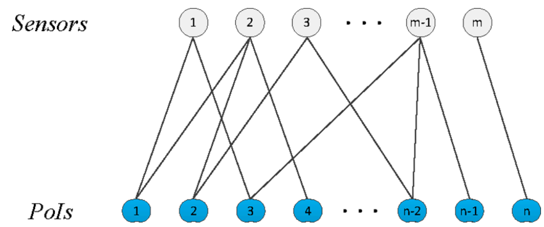
4.1 *Complexity analysis*

In the following theorem, we prove that the scheduling problem is NP-complete.

Theorem 1: *The scheduling problem is NP-complete.*

Proof: First of all, we represent the coverage relationship between sensors and PoIs in a bipartite graph, as is shown in Figure 2. Denote by bipartite graph $G = (V, O, E)$ the coverage graph in Figure 2 where V and O denote the set of sensors and PoIs respectively, and E denotes the set of edges between sensors and PoIs. If there is an edge between sensor v_i (depicted in a circle with text i centred) and PoI o_j (depicted in a rounded rectangle with text j centred), it means that v_i covers o_j. In practice, the coverage graph is determined by the locations of sensors and PoIs and coverage area of the sensors (not necessarily a disk).

Figure 2 PoI coverage illustration (see online version for colours)



To show that problem P_1 is NP-complete, we consider its decision version. Given the sensor schedule length $\mathscr{L}$, and a real number $Q \geq 0$, we need to answer whether there exists any scheduling policy of the sensors, such that $||S_i||_1 \leq l_i$ for any sensor v_i, and the objective function in the problem P_1 satisfies: $\sum_{i=1}^{N} QoM(i) \leq Q$.

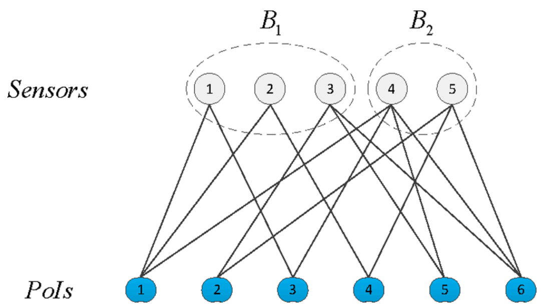
Denote by Ω the set of weights w_i for all PoIs, and L the set of l_i. Then the above problem instance can be denoted as $SDP(O,V,E,L,\mathscr{L},\Omega,Q)$. As in Golrezaei et al. (2012), we reduce an NP-complete problem called 2-Disjoint Set Cover Problem to the decision version of the scheduling problem (we call it scheduling decision problem hereafter).

Consider a bipartite graph $G = (A,B,E)$ with edges E between two disjoint vertex sets A and B. For each element $b_i \in B$, it has neighbourhoods in A which is denoted by $N(b_i)$. Assume that $A = \bigcup_{b_i \in B} N(b_i)$. Then it is proved to be NP-complete in Cardei and Du (2005) to determine whether there exist two disjoint sets $B_1, B_2 \subset B$ such that $|B_1| + |B_2| = |B|$ and $A = \bigcup_{b_i \in B_1} N(b_i) = \bigcup_{b_i \in B_2} N(b_i)$. For simplicity, we denote the above problem instance as $2DSC(A,B,E)$.

First of all, it is easy to see that $SDP(O,V,E,L,\mathscr{L},\Omega,Q)$ is in the class NP. Next we show that given a unit time oracle for scheduling decision problem, we can solve 2-Disjoint Set Cover Problem in polynomial time.

Consider an oracle which can solve any problem instance $SDP(O,V,E,L,\mathscr{L},\Omega,Q)$ in unit time. Then solving a problem instance $2DSC(A,B,E)$ is equivalent to solving $SDP(A,B,E,\{1,1,\ldots,1\},2, \{1/|A|,1/|A|,\ldots,1/|A|\},1)$. Consider A to be the set of PoIs, B to be the set of sensors, and E to be the edges representing the coverage relationship between sensors and PoIs. The sensor schedule length $\mathscr{L}$ is set to be 2 and the active time slot budget of all sensors is equal to 1. The PoI weights are assumed to be $1/|A|$ for all PoIs. We check if the overall QoM can be greater or equal to 1. If it is the case, the overall QoM has to be equal to 1 because QoM for each PoI is at most 1. This can only happen if the equivalent monitoring schedule $\widehat{S}_i$ for any PoI o_i is equal to $(1,1)$. It means there exist 2 disjoint set covers B_1 and B_2, while the entire sensors in B_1 are active in its first time slot and that in B_2 are active in its second time slot. Illustration is provided in Figure 3.

Figure 3 Reduction from 2-disjoint set cover problem (see online version for colours)



Conversely, if there exist two disjoint set covers, we can set the sensors in the first set cover to be active in the first time slot, and that in the second set cover to be active in the second time slot. By doing so, the *SDP* instance will be satisfied since every PoI o_i is continuously covered as $\widehat{S}_i = (1,1)$, and the overall QoM is equal to 1.

Hence, we conclude that 2-Disjoint Set Cover Problem $\leq_L$ Scheduling Decision Problem, where $\leq_L$ means a polynomial time reduction.

In summary, our scheduling problem is NP-complete. □

4.2 Problem reformulation

In this section, we begin with some necessary definitions. Then we cast the problem P_1 as maximising a monotone submodular function subject to a partition matroid constraint.

Definition 4.1 (Schrijver, 2003)**:** Let S be a finite ground set. A real-valued set function $f : 2^S \mapsto \mathbb{R}$ is *normalised*, *nondecreasing* (or *monotonic*) and *submodular* if and only if it satisfies the following conditions, respectively:

- $f(\emptyset) = 0$
- $f(A) \leq f(B)$ for any $A \subseteq B \subseteq S$, or equivalently: $f(A \cup \{e\}) - f(A) \geq 0$ for any $A \subseteq S$ and $e \in S \backslash A$
- $f(A) + f(B) \geq f(A \cup B) + f(A \cap B)$ for any $A, B \subseteq S$, or equivalently: $f(A \cup \{e\}) - f(A) \geq f(B \cup \{e\}) - f(B)$ for any $A \subseteq B \subseteq S$ and $e \in S \backslash B$.

For simplicity, we use $f_A(e) = f(A+e) - f(A)$ to denote the marginal value of element e with respect to A. Note that here we use $A + e$ instead of $A \cup \{e\}$ to simplify notation. Then f is monotone if $f_A(e) \geq 0$ and submodular if $f_A(e) \geq f_B(e)$ whenever $A \subseteq B$. This can be interpreted as the property of *diminishing returns*.

Definition 4.2 (Schrijver, 2003)**:** A *matroid* $\mathscr{M}$ is a tuple $\mathscr{M} = (S, \mathscr{I})$, where S is a finite ground set and $\mathscr{I} \subseteq 2^S$ is a collection of independent sets, such that:

- $\emptyset \in \mathscr{I}$
- if $X \subseteq Y \in \mathscr{I}$, then $X \in \mathscr{I}$
- if $X, Y \in \mathscr{I}$, and $|X| < |Y|$, then $\exists y \in Y \backslash X$ such that $X \cup \{y\} \in \mathscr{I}$.

In this work, we will pay our attention on the following specific class of matroids.

Definition 4.3 (Schrijver, 2003)**:** Given $S = \bigcup_{i=1}^{k} S'_i$ is the disjoint union of k sets, $l_1, l_2, \ldots, l_k$ are positive integers, a *partition matroid* $\mathscr{M} = (S, \mathscr{I})$ is a matroid where $\mathscr{I} = \{X \subset S : |X \cap S'_i| \leq l_i$ for $i = 1, 2, \ldots, k\}$.

We will demonstrate that the problem P_1 fits perfectly well in the realm of maximising a monotone submodular function subject to a partition matroid. We start with a definition of ground set S. Denote by $\mathbf{a}_{ij}$ the activating time slot a_{ij} of sensor v_i, then S is given by:

$$S = \{\mathbf{a}_{11}, \mathbf{a}_{12}, \ldots, \mathbf{a}_{1\mathscr{L}}, \ldots, \mathbf{a}_{m1}, \mathbf{a}_{m2}, \ldots, \mathbf{a}_{m\mathscr{L}}\}. \tag{5}$$

Whenever there is no confusion, we use S_i to denote the set version of v_i's sensor schedule, which is a subset of S, namely $S_i = \{\mathbf{a}_{i1'}, \mathbf{a}_{i2'}, \ldots, \mathbf{a}_{i\mathscr{L}'}\}$ if and only if $a_{ij'} = 1\,(j' = 1', 2', \ldots, \mathscr{L}')$. Further, S can be partitioned into m disjoint sets, $S'_1, S'_2, \ldots, S'_m$, which is given by $S'_i = \{\mathbf{a}_{i1}, \mathbf{a}_{i2}, \ldots, \mathbf{a}_{i\mathscr{L}}\}$.

We say S'_i is the *candidate schedule* of sensor v_i, as any feasible schedule S_i is the subset of S'_i. It is obvious that any scheduling policy X, consisting of all sensor schedule S_i, namely $X = \{S_1, S_2, \ldots, S_m\}$, is subject to $|X \cap S_i'| = |S_i| \le l_i$. Then we can write the independent sets as:

$$\mathscr{I} = \{X \subseteq S : |X \cap S_i'| \le l_i \quad \text{for } i = 1, 2, \ldots, m\} \tag{6}$$

On the other hand, it can be easily proved that $\mathscr{M} = \{S, \mathscr{I}\}$ is a matroid by verifying the three properties proposed in Definition 4.2. Hence we have the following lemma.

Lemma 4: *The constraint in the scheduling problem P_1 can be written as a partition matroid on the ground set S.*

As a consequence, we can rewrite the optimisation problem in P_1 as following:

$$\begin{array}{lll}(P_2): & \text{Max} & f(X) = \sum_{i=1}^{N} QoM(i) \\ & \text{s.t.} & X \in \mathscr{I} \\ & & S_i = X \cap S_i' \quad \forall i = 1, 2, \ldots, m,\end{array}$$

The new optimisation function $f(X)$ bears a desirable property as is stated in the following lemma.

Lemma 5: *If the utility function $U(x)$ is* concave*, then the objective function $f(X)$ in the optimisation problem P_2 is a monotone submodular function.*

Proof: According to Definition 4.1, we have to check if the three conditions hold for $f(X)$. First of all, it is obvious that $f(\emptyset) = 0$ holds. Secondly, we consider the monotonicity property of $f(X)$. Given set $A \subseteq S$ and $e_1 \in S \backslash A$. Assume that $e_1 = \mathbf{a}_{ij}$, then $f(A + e_1)$ can be regarded as the resulting overall QoM obtained by activating the time slot a_{ij} of sensor v_i based on the original scheduling policy. Consequently, the equivalent monitoring schedule of PoI o_k, which is covered by v_i ($o_k \in O_i$), may be changed. For simplicity of exposition, we denote the original and changed equivalent monitoring schedule of o_k as $\widehat{S}_k^{<A>}$ and $\widehat{S}_k^{<A+e_1>}$, respectively. In particular, the time slot a_{kj} is activated for $\widehat{S}_k^{<A+e_1>}$. We say $\mathbf{a}_{ij} \in \widehat{S}_k^{<A>}$ if the j_{th} time slot of $\widehat{S}_k^{<A>}$ is active, namely $a_{kj} = 1$, to simplify the notation. To save space, we only consider the case $\mathbf{a}_{ij} \notin \widehat{S}_k^{<A>}$. Note that:

$$\begin{aligned}&\widehat{\mathbf{S}}_k^{<A+e_1>}(x) - \widehat{\mathbf{S}}_k^{<A>}(x) \\ &= \begin{cases} 1, & (x \in [k\mathscr{L} + j - 1, k\mathscr{L} + j], k \in \mathscr{N}) \\ 0, & \text{otherwise} \end{cases}.\end{aligned} \tag{7}$$

Thus we have $\int_t^y \widehat{\mathbf{S}}_k^{<A+e_1>}(x)\mathrm{d}x - \int_t^y \widehat{\mathbf{S}}_k^{<A>}(x)\mathrm{d}x \ge 0$ for any $y \ge t \ge 0$. According to equation (3):

$$\begin{aligned}&Q(k|\widehat{S}_k^{<A+e_1>}) - Q(k|\widehat{S}_k^{<A>}) \\ &= \frac{1}{\mathscr{L}} \int_0^{\mathscr{L}} \int_t^{\infty} [U\left(\int_t^y \widehat{\mathbf{S}}_k^{<A+e_1>}(x)\mathrm{d}x\right) \\ &\quad - U\left(\int_t^y \widehat{\mathbf{S}}_k^{<A>}(x)\mathrm{d}x\right)] \cdot f(y-t)\mathrm{d}y\mathrm{d}t \ge 0.\end{aligned}$$

Note that the last inequality holds due to the monotonicity of $U(x)$, which follows from Observation 3.1. Hence:

$$\begin{aligned}f_A(e_1) &= \textstyle\sum_{o_k \in O_i} w_k [Q(k|\widehat{S}_k^{<A+e_1>}) - Q(k|\widehat{S}_k^{<A>})] \\ &\ge 0.\end{aligned}$$

The monotonicity property of $f(X)$ holds.

Thirdly, we verify the diminishing returns property of $f(X)$. Given set $A \subseteq B \subseteq S$ and $e_1 \in S \backslash B$. Similarly, we assume that $e_1 = \mathbf{a}_{ij}$, and the original equivalent monitoring schedules for o_k in A and B are $\widehat{S}_k^{<A>}$ and $\widehat{S}_k^{<B>}$, while the changed ones are $\widehat{S}_k^{<A+e_1>}$ and $\widehat{S}_k^{<B+e_1>}$, respectively. Due to space limit, we only consider the case when $\mathbf{a}_{ij} \notin \widehat{S}_k^{<A>}$ and $\mathbf{a}_{ij} \notin \widehat{S}_k^{<B>}$.

We first show that $U(x+\delta) - U(x) \ge U(y+\delta) - U(y)$ for $y \ge x \ge 0$ and $\delta \ge 0$. Note that $x \le x + \delta \le y + \delta$ and $x \le y \le y + \delta$. Due to the concavity of $U(x)$, we have:

$$(x+\delta) \ge \frac{y-x}{y+\delta-x} U(x) + \left(1 - \frac{y-x}{y+\delta-x}\right) U(y+\delta), \tag{8}$$

$$U(y) \ge \frac{\delta}{y+\delta-x} U(x) + \left(1 - \frac{\delta}{y+\delta-x}\right) U(y+\delta). \tag{9}$$

Adding up the left sides and the right sides of equations (8) and (9), the result follows.

Since $A \subseteq B$, we have $\widehat{\mathbf{S}}_k^{<B>}(x) - \widehat{\mathbf{S}}_k^{<A>}(x) \ge 0$ for $x \in [0, +\infty]$. Another important observation is that $\widehat{\mathbf{S}}_k^{<A+e_1>}(x) - \widehat{\mathbf{S}}_k^{<A>}(x) = \widehat{\mathbf{S}}_k^{<B+e_1>}(x) - \widehat{\mathbf{S}}_k^{<B>}(x)$ as $\mathbf{a}_{ij} \notin \widehat{S}_k^{<A>}$ and $\mathbf{a}_{ij} \notin \widehat{S}_k^{<B>}$ for $x \in [0, +\infty]$. Therefore, it is easy to see:

$$\begin{aligned}&[Q(k|\widehat{S}_k^{<A+e_1>}) - Q(k|\widehat{S}_k^{<A>})] - [Q(k|\widehat{S}_k^{<B+e_1>}) - Q(k|\widehat{S}_k^{<B>})] \\ &= \frac{1}{\mathscr{L}} \int_0^{\mathscr{L}} \int_t^{\infty} \{\left[U\left(\int_t^y \widehat{\mathbf{S}}_k^{<A+e_1>}(x)\mathrm{d}x\right) - U\left(\int_t^y \widehat{\mathbf{S}}_k^{<A>}(x)\mathrm{d}x\right)\right] \\ &\quad - \left[U\left(\int_t^y \widehat{\mathbf{S}}_k^{<B+e_1>}(x)\mathrm{d}x\right) - U\left(\int_t^y \widehat{\mathbf{S}}_k^{<B>}(x)\mathrm{d}x\right)\right]\} f(y-t)\mathrm{d}y\mathrm{d}t \\ &\ge 0.\end{aligned}$$

Hence:

$$\begin{aligned}f_A(e_1) - f_B(e_1) = &\sum_{o_k \in O_i} w_k \{[Q(k|\widehat{S}_k^{<A+e_1>}) - Q(k|\widehat{S}_k^{<A>})] \\ &- [Q(k|\widehat{S}_k^{<B+e_1>}) - Q(k|\widehat{S}_k^{<B>})]\} \\ \ge &\ 0.\end{aligned}$$

Then we conclude that $f(X)$ is indeed submodular. To sum up all the analysis, we complete the proof. ☐

It is easy to see that the step utility function, exponential utility function and linear utility function are concave, while the S-shaped utility function and delayed step utility function are not (Yau et al., 2010). We assume that the utility function $U(x)$ is concave hereafter to make the problem amenable. Nevertheless, since in most of the cases utility functions are really concave (Yau et al., 2010; He et al., 2009; Tang and Yang, 2012; Ren et al., 2012; Tang et al., 2011), such treatment will not reduce our contribution significantly.

4.3 Centralised algorithm

Having proved that the objective function of our problem is monotone submodular, now we can resort to a simple greedy algorithm to find an optimised QoM. The details of the algorithm can be found in Algorithm 1.

Algorithm 1: *Greedy Algorithm*

Input: *The sensors set* $V = \{v_1, v_2, \ldots, v_m\}$*, the PoIs set* $O = \{o_1, o_2, \ldots, o_n\}$*, the objective function* f*, the ground set* S*, the candidate schedule* S_i'*, active time slot budget* $l_1, l_2, \ldots, l_m$*.*

Output: *The sensor schedules* $S_1, S_2, \ldots, S_m$*.*

$D = S$;
$S_i = \emptyset$ *for* $i = 1, 2, \ldots, m$;
$X = \emptyset$;
$k = 1$;
while $k \leq m \times \mathscr{L}$ ***do***
  $\boldsymbol{a}_{ij} = \arg\max_{\boldsymbol{d} \in D} f_X(\boldsymbol{d})$;
  if $f_X(\boldsymbol{a}_{ij}) = 0$ ***then***
    break;
  end if
  $X \leftarrow X + \boldsymbol{a}_{ij}$;
  $S_i \leftarrow S_i + \boldsymbol{a}_{ij}$; *(namely set* $S_{ij} = 1$*)*
  $D \leftarrow D \backslash \boldsymbol{a}_{ij}$;
  $S_i' \leftarrow S_i' \backslash \boldsymbol{a}_{ij}$;
  if $||S_i||_1 = l_i$ ***then***
    $D \leftarrow D \backslash S_i'$;
  end if
  $k = k + 1$;
end while

It can be seen that at each step, the algorithm adds one element with the highest marginal value to set. However, if the marginal value is zero, it means that all the PoIs covered by the sensors with nonempty residual active time slots budgets are already covered at all time slots. As a result, the algorithm should stop. Moreover, at every iteration, we remove the selected element $\mathbf{a}_{ij}$ from set D. If the active time slot budget for sensor v_i is used up, we need to remove the candidate schedule S_i' from D.

We have the following theorem with respect to Algorithm 1.

Theorem 6: *The greedy scheduling algorithm can achieve 1/2-approximation.*

Proof: By Lemma 4, the constraint in the optimisation problem P_2 is indeed a partition matroid. Further, by Lemma 5, the objective function in P_2 is a monotone submodular function. We thus claim that the reformulated problem P_2 can be expressed as maximisation of a monotone submodular function subject to a partition matroid constraint. According to the classical results obtained by Nemhauser et al. (1978), a simple greedy heuristic to maximisation of a monotone submodular function subject to a partition matroid constraint can achieve 1/2-approximation. Hence, the result follows. □

Now we analyse the time complexity of Algorithm 1. It can be seen that the time complexity lies mainly in the execution of the while loop in the greedy algorithm. The greedy algorithm for searching of solution has at most $O(m\mathscr{L})$ iterations. Each iteration checks at most $O(m\mathscr{L})$ time slots, each of which relates to $O(n)$ QoM calculations of PoIs. Therefore, the overall time complexity is $O((m\mathscr{L})^2 nT)$.

It is worthwhile to mention that (Calinescu et al., 2009) provided a randomised algorithm which is optimal in terms of approximation, and achieves $(1 - 1/e)$-approximation. However, this algorithm is too computationally demanding to implement, especially when the number of sensors or sensor schedule length becomes large.

As we mentioned above, the overall QoM is computed in such a way that it indeed reflects the expected performance of event monitoring in the long run. So does the performance of our greedy algorithm.

4.4 Distributed algorithm

Since the proposed centralised algorithm is not scalable well, it is desirable for us to design a distributed algorithm for this scheduling problem. In this section, we begin with the description to our distributed algorithm, and then analyse its performance.

4.4.1 Algorithm description

For simplicity of presentation, our algorithm is divided into rounds. We stress that the concept of rounds can be removed to cater to real cases by asynchronously executing the algorithm. During each round, the schedules of a subset of sensors can be determined. After a finite number of rounds, we can obtain all the schedules of sensors, which guarantee overall performance within a constant factor of the global optimal as to be elaborated.

First of all, we define a new concept of neighbours. Two sensors are neighbours to each other if and only if they cover PoIs in common. We assume that the communication range of each sensor is at least twice of the sensing range (Wang et al., 2003), which is widely adopted in previous literature. Therefore, the neighbouring sensors can directly communicate with each other, and the neighbour set, say $N(v_i)$, of each sensor can be determined beforehand. In addition, the control message exchanged between sensors are defined as $msg(ID, type, sch, \Delta QoM_i^{\max})$, where the fields ID and sch denote the sensor ID and its schedule respectively, $type$ is either COL or UPD, which stands for the colouring notice or the update report, and $\Delta QoM_i^{\max}$ is the 'maximum' additional QoM for sensor v_i.

We list the pseudo code in Algorithm 2. Next we introduce the details of this algorithm. In the very beginning, each sensor is uncoloured and has a null schedule. At each round, each uncoloured sensor computes $\Delta QoM_i^{\max}$ and the corresponding schedule, and broadcasts them to its neighbours. Note that $\Delta QoM_i^{\max}$ for sensor v_i is obtained by greedily activating the time slot that can provide the maximum additional QoM until its active time slot budget l_i is exhausted. In this sense, $\Delta QoM_i^{\max}$ is not really maximum in terms of additional QoM. Nevertheless, it requires little computation cost and does not impair the performance much as demonstrated in Section 5.

After collecting all the replies from its uncoloured neighbours, each sensor will try to determine whether or not it has the largest $\Delta QoM_i^{\max}$ among its uncoloured neighbours. If it is the case, it will colour itself and send the notification of colour decision along with the corresponding schedule to the local neighbourhood. Note that if there are two sensors have the same 'maximum' additional QoM and thereby lead to a tie, it can be broken based on the *ID* of sensors. Subsequently, each sensor receiving the colour notice will update the schedule of the related neighbour and recalculate $\Delta QoM_i^{\max}$, clear the stored information for all its uncoloured neighbours and send $\Delta QoM_i^{\max}$ to its neighbours.

Algorithm 2: *Distributed Algorithm (at each sensor v_i)*

```
Input: The neighbor set N(v_i), the set of PoIs that v_i covers, the neighboring schedules, the QoM computation function QoM(i).
Output: The state (color or uncolor), the sensor schedule S_i if it is colored.
1:  Calculate the "maximum" additional QoM ΔQoM_i^max based on the schedules of its colored neighbors, and determine the corresponding schedule;
2:  Broadcast msg(i, UPD, NULL, ΔQoM_i^max);
3:  while ΔQoM_i^max > 0 do
4:      if The candidate "maximum" additional QoMs of all uncolored neighbors are collected, and ΔQoM_i^max is larger than any of them then
5:          Color itself;
6:          Broadcasts msg(i, COL, sch, ΔQoM_i^max);
7:          Exit;
8:      end if
9:      if msg(k, COL, sch, ΔQoM_k^max) is received then
10:         Update the stored schedule of its neighbor v_k to sch, recalculate ΔQoM_i^max and determined its schedule based on the schedules of its colored neighbors (including the updated schedule of v_k);
11:         Clear the candidate "maximum" additional QoMs of all its uncolored neighbors;
12:         Broadcast msg(i, UPD, NULL, ΔQoM_i^max);
13:         Go to Line 3;
14:     end if
15:     if msg(k, UPD, NULL, ΔQoM_k^max) is received then
16:         Update the stored candidate "maximum" additional QoM of its neighbor v_k, i.e., ΔQoM_k^max;
17:         Go to Line 3;
18:     end if
19: end while
```

On receiving the update report, each sensor will update the recorded 'maximum' additional QoM of the corresponding neighbour. This information will then be used to help the sensor identify whether it has the largest $\Delta QoM_i^{\max}$ among its neighbours.

Essentially, our algorithm is equivalent to finding a maximal independent set at each round (Basagni, 2001) (in graph theory, an independent set is a set of vertices in a graph, no two of which are adjacent; a maximal independent set is an independent set that is not a subset of any other independent set). At the end of each round, the coloured sensors are removed from the considerations of its neighbours, and the algorithm is then conducted by the remaining uncoloured sensors. It is easy to see that this algorithm is bounded to terminate, since at each round there must be at least one sensor colouring itself. Eventually, the algorithm stops when all the sensors are coloured.

Let $N_{\max}$ and $n_{\max}$ be the maximum number of neighbours and covered PoIs for a sensor, respectively. Suppose that our distributed algorithm will terminate in K rounds. On one hand, apparently the message complexity for each sensor is $O(K)$. On the other hand, the computation complexity of each round consists of two parts. One comes from the computation of $\Delta QoM_i^{\max}$, which checks $\mathscr{L}$ time slots, each of which relates to at most $n_{\max}$ QoM calculations of PoIs. The other stems from the comparisons of $\Delta QoM_i^{\max}$, which is conducted at most $N_{\max}$ times. Thus, the overall computation complexity is given by $O(KN_{\max} + K\mathscr{L}n_{\max})$. Typically, K is a small number, and therefore, the overhead of our distributed algorithm is low.

As to be elaborated, our distributed algorithm has comparable performance to the greedy algorithm in terms of QoM. In addition to low computational cost, it brings about significant advantages than a centralised algorithm. For example, it does not incur the communication overhead for collecting node information by the sink.

4.4.2 Algorithm analysis

Unlike the centralised algorithm which is performed sequentially and thereby easy to be analysed theoretically, the distributed algorithm is not executed in a well-organised sequence for sensors, and even not executed round by round as we assumed before. That is, in practice the algorithm can be conducted in a totally asynchronous way among sensors, and thereby the concept of rounds is removed. This causes difficulty in the theoretical analysis. Nevertheless, we claim that the performance of the distributed algorithm can be provably guaranteed, as the following theorem states.

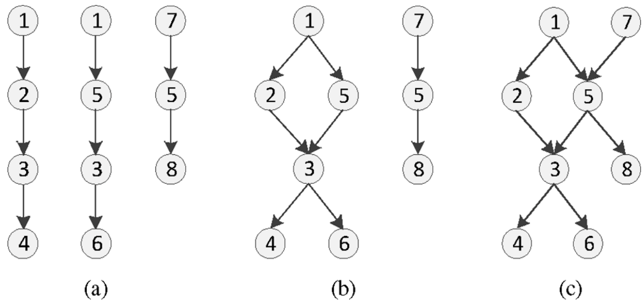
Theorem 7: *The distributed algorithm achieves an approximation factor of 1/2 for the scheduling problem.*

Proof: First of all, we show that we can reorder all the colouring sensors obtained by the distributed algorithm in an order, such that these sensors can be equivalently viewed as sequentially determining its schedule based on global knowledge of QoM.

To begin with, we remove the concept of rounds here to make our solution practical, as in reality the algorithm can be executed asynchronously. After this, the only information learned by each sensor is the local algorithm execution order among neighbourhood. Assume that the algorithm execution order for sensor v_i is R_i, then we can plot it in a directed chain where a directed edge v_iv_j from vertex v_i to vertex v_j means the execution time of v_i is left behind v_j. For example, suppose that the execution order for sensor v_1, v_3 and v_5 are $R_1 : v_1 < v_2 < v_3 < v_4$, $R_3 : v_1 < v_5 < v_3 < v_6$ and ,$R_5 : v_7 < v_5 < v_8$, respectively (we use sensor label rather than execution time to simplify the expression), we can depict their directed chains as shown in Figure 4(a).

Next, we can combine these chains by merging same vertex. For instance, Figure 4(b) demonstrates the resulting directed graph obtained by combining execution chains of v_1 and v_3 wherein vertex v_1 and v_3 are merged, and the result of further combining with chain of v_5 is shown in Figure 4(c). By sequentially combining all chains of sensors until no further connection is possible, we can build a directed graph $G(V,E)$ indicating the relative execution time order between sensors. More importantly, it is easy to verify that $G(V,E)$ is also a directed acyclic graph (in computer science, a directed acyclic graph is a directed graph with no directed cycles), as otherwise the algorithm execution time of a sensor v_i will come ahead of that of itself. As a result, we can employ some topological sorting algorithm to order all sensors. A topological sort of a directed graph is a linear ordering of its vertices such that for every directed edge v_iv_j from vertex v_i to vertex v_j, v_i comes before v_j in the ordering. There exist a number of effective algorithms for computing the topological order of a directed acyclic graph, for example, the well-known linear time algorithm proposed by Cormen et al. (2001). Suppose some good topological sorting algorithm is used and the exact solution is obtained by which all sensors are sorted in an order. Besides, it can be easily verified that the additional 'maximum' QoM, i.e., $\Delta QoM_i^{\max}$, computed by each sensor is exactly equal to the additional QoM of the overall QoM at that moment. It thus can be assumed that all sensors have access to an incremental oracle for global QoM (Goundan and Schulz, 2007).

Figure 4 An example of directed acyclic graph construction. The three directed chains belong to sensor v_1, v_3 and v_5 respectively



Now we prove the approximation ratio for our distributed algorithm partly based on the results obtained by Goundan and Schulz (2007). Let $S_1, S_2, \ldots, S_m$ represent the schedule for each sensor obtained by the distributed algorithm, and $S_1^*, S_2^*, \ldots, S_m^*$ the optimal schedule to the scheduling problem. Hence, the solution to the problem as well as the optimal one can be written as $X = \bigcup_{i=1}^m S_i$ and $X^* = \bigcup_{i=1}^m S_i^*$, respectively.

Following the submodularity of the objective function f, we have that:

$$f(X^*) \leq f(X) + \sum_{j \in X^* \setminus X} f_X(j). \tag{10}$$

Suppose $X^* \setminus X = \bigcup_{i=1}^m Y_i$ where $Y_i \subseteq S_i'$, or equivalently, $Y_i = S_i^* \setminus S_i$. Let e_i be the element in S_i that was selected with the lowest marginal value during the local procedure of the algorithm in sensor v_i. Assume that the current solution just before the addition of e_i is $X_i^{e_i}$ (this is obtained based on the new ordering), then we have:

$$f_{X_i^{e_i}}(e_i) = \min_{e \in S_i} f_{X_i^e}(e). \tag{11}$$

Further, since $\sum_{j \in X^* \setminus X} f_X(j) = \sum_{i=1}^m \sum_{j \in Y_i} f_X(j)$, it follows that:

$$\begin{aligned} f(X^*) &\leq f(X) + \sum_{j \in X^* \setminus X} f_X(j) \\ &\leq f(X) + \sum_{i=1}^m \sum_{j \in Y_i} f_X(j) \\ &\leq f(X) + \sum_{i=1}^m \sum_{j \in Y_i} f_{X_i^{e_i}}(j). \end{aligned} \tag{12}$$

The last inequality in equation (12) follows from the submodularity of f as $X_i^{e_i} \subseteq X$ for $i = 1, 2, \ldots, m$. Due to the fact that $Y_i = S_i^* \setminus S_i \subseteq S_i^* \setminus X_i^{e_i}$, we have:

$$\begin{aligned} f(X^*) &\leq f(X) + \sum_{i=1}^m \sum_{j \in Y_i} f_{X_i^{e_i}}(j) \\ &\leq f(X) + \sum_{i=1}^m |Y_i| f_{X_i^{e_i}}(e_i) \\ &\leq f(X) + \sum_{i=1}^m |S_i| f_{X_i^{e_i}}(e_i) \\ &\leq f(X) + f(X) \\ &\leq 2f(X). \end{aligned} \tag{13}$$

Note that $f_{X_i^{e_i}}(j) \leq f_{X_i^{e_i}}(e_i), \forall j \in Y_i$ since e_i can provide the maximum marginal value of QoM. This completes the proof. □

5 Performance evaluation

We present simulation results to verify our findings.

5.1 Simulation setup

Unless otherwise stated, we use the following simulation settings:

- the event staying time $X \in \text{exponential}(\lambda)$, $\lambda = 1$
- each sensor has a sensing range $r = 1\,m$
- the weight $w_i = 1/n$ for each PoI o_i.

As most literature does, we set the communication range is at least twice of the sensing range: sensing range is set to be 1 and communication range to be 2.

5.2 Baseline setup

In Section 5.3, we compare our algorithms to the optimal solution. The optimal solution is obtained by enumerating all possible scheduling policies under the same active time slot budgets constraints. This exhaustive search method is extremely computationally-demanding (up to $(P(\mathscr{L},l))^m$ given that the active time slot budget for all sensor sensors is uniform and is equal to l). We are thus only able to make comparisons in small-scale networks.

In Section 5.4, we compare our algorithm to CSP proposed in He et al. (2009), an energy-efficient protocol for stochastic events capture, which accommodates both synchronous and asynchronous networks, in large-scale sensor networks.

Generally, there are two versions of CSP: S-CSP for synchronous networks, where all the sensors employ the same (q,p) schedule and start their on periods at the same time; and A-CSP for asynchronous networks, where each sensor employs the same (q,p)-periodic schedule, but starts their on periods independently at a uniformly random point in time within the period p. Since A-CSP is only suitable for asynchronous networks, we extend it to synchronous networks in the discrete time model by letting each sensor start their on periods independently at a random time slot in the schedule of length $\mathscr{L}$, which we call A-CSP-S. Note that, in essence, A-CSP-S is a randomised algorithm.

We use the example stated in Section 3.5 to illustrate these two algorithms. For S-CSP, the schedules of three sensors should be $S_1 = (1,0,0,0)$, $S_2 = (1,1,0,0)$ and $S_3 = (1,0,0,0)$. All of these schedules starts at the first time slot, and all their active time slots cluster together in consecutive time slots (see S_2). In contrast, if the sensors employ A-CSP-S algorithm, their schedules may be $S_1 = (1,0,0,0)$, $S_2 = (0,1,1,0)$ and $S_3 = (0,0,1,0)$. Comparing with S-CSP, A-CSP-S allows sensors to start their active time slots at a random time slot.

In addition, to make the comparison between CSP and our scheme feasible, we assume that there is no sensor whose sensing region is completely covered by those of its active neighbours, which means each sensor should not sleep.

5.3 Performance compared to the optimum

We compare our proposed approximation algorithm with the optimal solution for small-scale networks in this section.

We randomly distribute sensors in a 3×3 m region in this scenario. As for PoIs, the deployment region is discretised into square cells of dimensions 0.5×0.5 m, and each vertex of each cell is a PoI. We vary the number of sensors between 4 and 8, and only record the data where the total number of covered PoIs is 36, in order to make the comparisons reasonable among different cases with a different number of sensors. We compute the overall QoMs of the schedules output by the greedy algorithm (GA) and distributed algorithm (DA) for three scenarios:

- sensor schedule length $\mathscr{L} = 8$, and the active time slot budget $l_i = 1$ for every sensor v_i
- $\mathscr{L} = 5$, and $l_i = 1$
- $\mathscr{L} = 5$, and l_i is randomly selected from $\{1,2\}$.

The simulation results are shown in Figure 5. Specifically, Figure 5(a) demonstrates the overall QoMs obtained by the optimal solution, GA and DA corresponding to the first scenario. It can be seen that for networks with small size, the performances of GA and DA are exactly the same, and quite close to that of the optimal solution. For example, the largest performance gap between the optimal solution and that of GA is only 1.8%, attained when the number of sensors is 8. The overall QoMs for these three solutions rise when the number of sensors increases.

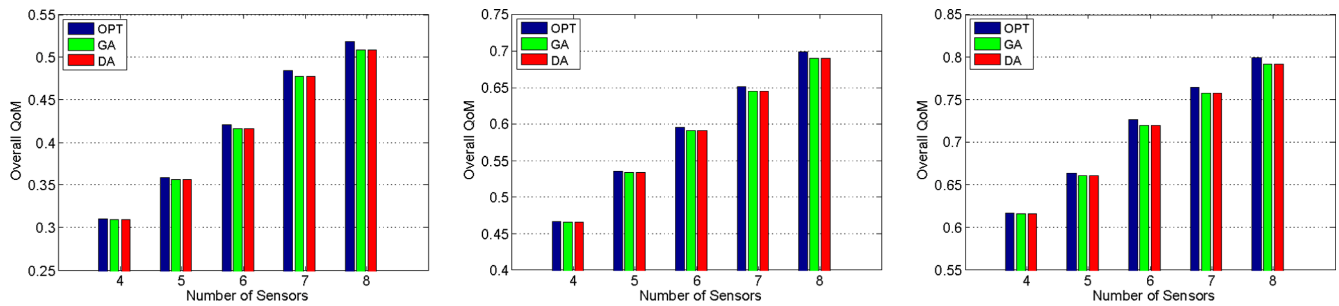
The overall QoMs under the second and third scenarios, which are depicted in Figure 5(b) and (c) respectively, show the same trend when the number of sensors increases. The performance differences between the optimal solution and the that of GA are no more than 1.4% for both cases. As the energy budget (different from the active time slot budget l_i, see Section 3.2) for the third scenario is better than the second scenario, which in turn exceeds that of the first one, we conclude that the overall QoM grows with a larger energy budget. Again, the outcomes of GA and DA are exactly the same, and come pretty close to that of the optimal solution.

5.4 Performance compared to CSP

Throughout the simulation, sensors are distributed randomly in a 20×20 m region. The sensor schedule length $\mathscr{L}$ is set to 4, and the active time slot budget $l_i = 1$ for any sensor v_i. Further, the distance between adjacent PoIs is increased with the average number of sensors increasing from 50 to 500, such that the number of covered PoIs maintains 500 all the time. As for A-CSP-S, we simulate the algorithm for 100 times and record the mean value of the outputs.

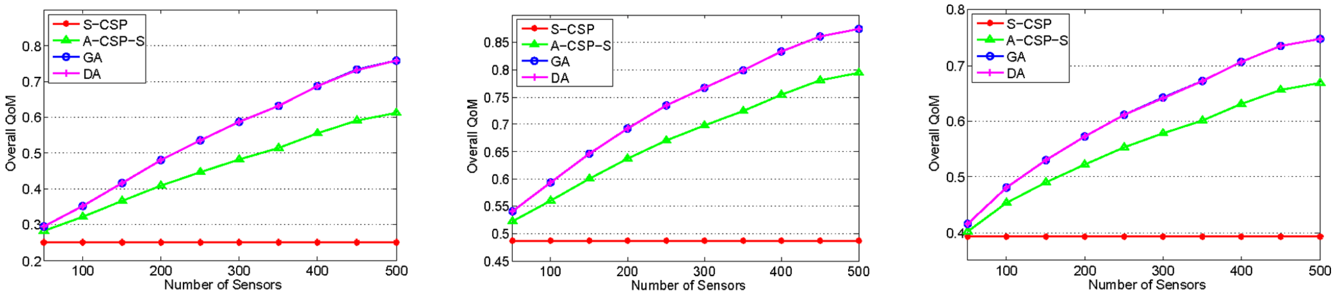
In particular, we first compare our algorithms to S-CSP and A-CSP-S for the case where events have step event staying time function and step utility function, and plot their achieved overall QoM in Figure 6(a). It can be seen that S-CSP has the constant yet worst performance, since all sensors have

Figure 5 Optimum vs. greedy algorithm vs. distributed algorithm (see online version for colours)



(a) (b) (c)

Figure 6 S-CSP vs. A-CSP-S vs. greedy algorithm vs. distributed algorithm: (a) achieved QoM for events with step utility function and step staying time; (b) achieved QoM for events with exponential utility function and exponential staying time (see online version for colours)



(a) (b) (c)

the same schedule and, therefore, the increase of number of sensors does not contribute to the overall QoM. A-CSP-S performs much better than S-CSP, but is still inferior to GA and DA (almost coincide with each other in Figure 6(a)). To be more specific, on average, both the overall QoMs achieved by GA and DA are roughly 119% and 18% higher than that of S-CSP and A-CSP-S, respectively. On the other hand, though the difference between GA and DA is not obvious, our simulation data indicates that DA has a slight advantage on the GA, with 0.69 performance gain on average.

For the cases where events have exponential event staying time function and step utility function, or exponential event staying time function and exponential utility function, the situations are very similar to the former one. As Figure 6(b) and (c) suggest, both GA and DA achieve substantially higher QoMs compared with S-CPS and A-CSP-S, with improvements 9% and 50%, respectively. Most importantly, the results of these three cases imply that DA has completely comparable performance with GA in terms of overall QoM.

Next, we study the trend of QoMs with varying number of PoIs. Figure 7 shows GA and DA have the same performance while S-CSP has the worst, when we set the number of sensors be 50. Furthermore, the QoMs of each algorithm remain nearly constant when the number of PoIs increases. The reason behind this phenomenon is as follows. Suppose the whole region is divided into multiple subregions, each of which is covered by a different set of sensors. Apparently, PoIs in the same subregion have the same QoM. Since PoIs are randomly distributed in the interested region, the number of PoIs in each subregion should increase roughly at the same rate. Such increasing rate exactly smoothes the decrease of PoI weights, and consequently, the accumulated QoM contributed by each subregion remains unchanged. So does the overall QoM.

5.5 *Varying length of time slot*

Note that the default value of duration of time slot is 1 s in above sections. Now we vary the duration of time slot, and plot the overall QoMs of both GA and DA in Figure 8. Note that we set the sensor schedule length $\mathscr{L} = 4$, and the active time slot budget $l_i = 1$ for every sensor v_i. Similarly, the performance of DA is exactly the same with that of GA. Moreover, the overall QoMs rises with decreasing duration of time slot for both schemes, and achieves up to 69% performance gain for $1/10$ s compared with 1 s. This is because events with the same staying time become more likely to be detected (as the time interval between non-consecutive active time slots shrinks) and captured in its early phase with high utility. It is noteworthy that this result is consistent with the theoretical analysis of Yau et al. (2010) and Jiang et al. (2011).

Figure 7 Impact of PoI number (see online version for colours)

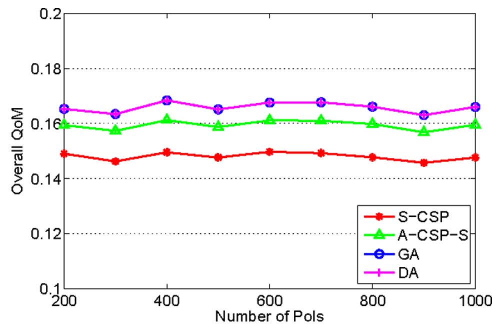
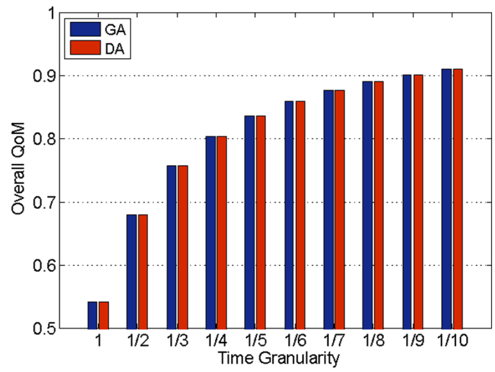


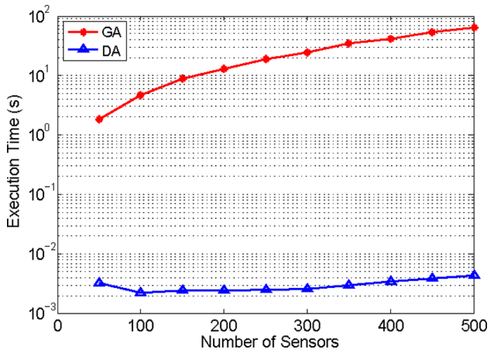
Figure 8 Impact of time granularity (see online version for colours)



5.6 *Scalability evaluation of distributed algorithm*

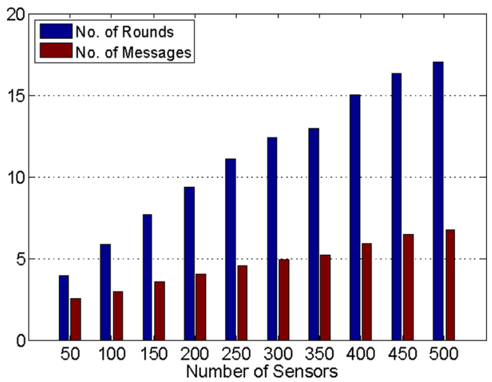
We study the scalability of DA by comparing with that of GA. Figure 9 depicts the execution time needed for GA and DA respectively. It shows that the latter is highly competitive against the former with up to three order improvement. Note that we show the average execution time per node for DA and the overall execution time for GA due to their distributed and centralised natures. Moreover, it can be seen that the required execution time of DA remains low as the number of sensors increases, with the run time being 0.0043 s for 500 sensors. On the contrary, the run time of GA rises rapidly with an increasing number of sensors, and balloons to 64.6 s in the presence of 500 sensors. In other words, DA scales effectively with network size, which greatly outperforms GA. Though the simulation is performed on our Lenovo notebook which is much more powerful than typical sensors, the results still demonstrate the huge advantage of DA over GA.

Figure 9 Execution time (see online version for colours)



We proceed to investigate the average number of rounds and the communication overhead of DA. In Figure 10, we can observe that the average number of rounds and that of messages per node increases at a moderate rate with an increasing number of sensors. In particular, the number of messages grows slowly from 2.55 to 6.72, when the number of sensors jumps from 50 to 500. Therefore, we claim that DA scale well in terms of communication overhead. Besides, the growth rate of the number of rounds is relatively higher, yet still acceptable, than that of the number of message.

Figure 10 Cost analysis (see online version for colours)



6 Conclusion

In this paper, we consider the scheduling problem in order to maximise the QoM of stochastic event capture in WRSNs. Specifically, we propose centralised and distributed algorithms both with constant approximation ratios. Simulation results show that our algorithm has performance close to the optimal, and outperforms the former work.

There are some directions to refine our solution in future work. For instance, if the event occurrence process exhibits significant correlation in time, the question is how such dynamic on-line information can be exploited to further improve system performance.

Acknowledgements

This work is supported by National 973 project of China under Grant No. 2012CB316200, No. 2012CB316201, No. 2014CB340300 and No. 2014CB340303, National Natural Science Foundation of China under Grant No. 61133006, No. 61373130, No. 61321491 and No. 61272443, NSF of Jiangsu Province Grant No.BK20141319, EU FP7 CROWN project under grant number PIRSES-GA-2013-610524, National Key Project of China under Grant No. 2013ZX01033002-003, Research and Innovation Project for College Graduate Students of Jiangsu Province in 2012 under Grant No.CXZZ12_0056, and the Shanghai Science and Technology fund 12PJ1404900.

References

Abrams, Z., Goel, A. and Plotkin, S. (2004) 'Set k-cover algorithms for energy efficient monitoring in wireless sensor networks', *Proceedings of the 3rd International Symposium on Information Processing in Sensor Networks*, ACM, pp.424–432.

Basagni, S. (2001) 'Finding a maximal weighted independent set in wireless networks', *Telecommunication Systems*, Vol. 18, Nos. 1–3, pp.155–168.

Bian, F., Kempe, D. and Govindan, R. (2006) 'Utility-based sensor selection', *IPSN'06*, ACM, pp.11–18.

Calinescu, G., Chekuri, C., Pál, M. and Vondrák, J. (2009) 'Maximizing a monotone submodular function subject to a matroid constraint', *SIAM Journal on Computing*, Vol. 40, No. 6, pp.1740–1766.

Cardei, M. and Du, D. (2005) 'Improving wireless sensor network lifetime through power aware organization', *Wireless Networks*, Vol. 11, No. 3, pp.333–340.

Cormen, T.H., Leiserson, C.E., Rivest, R.L. and Stein, C. (2001) *Introduction to Algorithms[M]*, MIT Press, Cambridge, Vol. 2.

Dai, H., Wu, X., Xu, L. and Chen, G. (2013a) 'Practical scheduling for stochastic event capture in wireless rechargeable sensor networks', *WCNC*, IEEE, pp.986–991.

Dai, H., Wu, X., Xu, L., Chen, G. and Lin, S. (2013b) 'Using minimum mobile chargers to keep large-scale wireless rechargeable sensor networks running forever', *ICCCN*, IEEE, pp.1–7.

Dai, H., Xu, L., Wu, X., Dong, C. and Chen, G. (2013c) 'Impact of mobility on energy provisioning in wireless rechargeable sensor networks', *WCNC*, IEEE, pp.962–967.

Dai, H., Jiang, L., Wu, X., Yau, D.K.Y., Chen, G. and Tang, S. (2013d) 'Near optimal charging and scheduling scheme for stochastic event capture with rechargeable sensors', *MASS*, IEEE, pp.10–18.

Dai, H., Wu, X., Xu, L. and Chen, G. (2013e) 'Practical scheduling for stochastic event capture in wireless rechargeable sensor networks', *WCNC*, IEEE, pp.986–991.

Dai, H., Liu, Y., Chen, G., Wu, X. and He, T. (2014) 'Safe charging for wireless power transfer', *INFOCOM*, IEEE, pp.1105–1113.

Gnawali, O., Fonseca, R., Jamieson, K., Moss, D. and Levis, P. (2009) 'Collection tree protocol', *SenSys*, ACM, pp.1–14.

Golovin, D. and Krause, A. (2011a) *Adaptive Submodular Optimization Under Matroid Constraints*, arXiv preprint arXiv:1101.4450.

Golovin, D. and Krause, A. (2011b) 'Adaptive submodularity: theory and applications in active learning and stochastic optimization', *Journal of Artificial Intelligence Research (JAIR)*, Vol. 42, pp.427-486.

Golrezaei, N., Shanmugam, K., Dimakis, A., Molisch, A. and Caire, G. (2012) 'Femtocaching: wireless video content delivery through distributed caching helpers', *INFOCOM*, IEEE, pp.1107–1115.

Goundan, P.R. and Schulz, A.S. (2007) 'Revisiting the greedy approach to submodular set function maximization', *Optimization Online*, pp.1-25.

He, S., Chen, J., Yau, D.K., Shao, H. and Sun, Y. (2009) 'Energy-efficient capture of stochastic events by global- and local-periodic network coverage', *MobiHoc*, p.155.

He, S., Chen, J., Sun, Y., Yau, D. and Yip, N. (2010a) 'On optimal information capture by energy-constrained mobile sensors', *IEEE Transactions on Vehicular Technology*, Vol. 59, No. 5, pp.2472–2484.

He, S., Chen, J., Jiang, F., Yau, D., Xing, G. and Sun, Y. (2011) 'Energy provisioning in wireless rechargeable sensor networks', *INFOCOM*, pp.2006–2014.

He, S., Chen, J., Yau, D., Shao, H. and Sun, Y. (2012a) 'Energy-efficient capture of stochastic events under periodic network coverage and coordinated sleep', *IEEE Transactions on Parallel and Distributed Systems*, Vol. 23, No. 6, pp.1090–1102.

He, S., Chen, J. and Sun, Y. (2012b) 'Coverage and connectivity in duty-cycled wireless sensor networks for event monitoring', *IEEE Transactions on Parallel and Distributed Systems*, Vol. 23, No. 3, pp.475–482.

Hsu, J., Zahedi, S., Kansal, A., Srivastava, M. and Raghunathan, V. (2006) 'Adaptive duty cycling for energy harvesting systems', *Proceedings of the 2006 International Symposium on Low Power Electronics and Design*, ACM, pp.180–185.

Jaggi, N., Kar, K. and Krishnamurthy, A. (2007) 'Rechargeable sensor activation under temporally correlated events', *Wireless Networks*, Vol. 15, No. 5, December, pp.619–635.

Jaggi, N., Madakasira, S., Mereddy, S. and Pendse, R. (2010) 'Adaptive algorithms for sensor activation in renewable energy based sensor systems', *Ad Hoc Networks*, Vol. 11, No. 4, pp.1405–1420.

Jaggi, N. (2006) 'Robust threshold based sensor activation policies under spatial correlation', *Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks*, IEEE, pp.1–8.

Jiang, F., He, S., Cheng, P. and Chen, J. (2011) 'On optimal scheduling in wireless rechargeable sensor networks for stochastic event capture', *MASS*, IEEE, pp.69–74.

Kar, K. and Banerjee, S. (2003) 'Node placement for connected coverage in sensor networks', *WiOpt*, p.2.

Kar, K., Krishnamurthy, A. and Jaggi, N. (2005) 'Dynamic node activation in networks of rechargeable sensors', *INFOCOM*, Vol. 3, pp.1997–2007.

Krause, A. (2008) *Optimizing Sensing: Theory and Applications*, PhD Thesis, Carnegie Mellon University.

Krause, A. and Guestrin, C. (2007) 'Near-optimal observation selection using submodular functions', *Proc. of the 22nd Conference on Artificial Intelligence (AAAI)*, pp.1650–1654.

Krause, A., Leskovec, J., Guestrin, C., VanBriesen, J. and Faloutsos, C. (2008) 'Efficient sensor placement optimization for securing large water distribution networks', *Journal of Water Resources Planning and Management*, Vol. 134, No. 6, pp.516–526.

Liu, C. and Cao, G. (2012) 'Distributed critical location coverage in wireless sensor networks with lifetime constraint', *INFOCOM*, IEEE, pp.1314–1322.

Meninger, S., Mur-Miranda, J., Amirtsharajah, R., Chandrakasan, A. and Lang, J. (2001) 'Vibration-to-electric energy conversion', *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, Vol. 9, No. 1, pp.64–76.

Nemhauser, G., Wolsey, L. and Fisher, M. (1978) 'An analysis of approximations for maximizing submodular set functions', *Mathematical Programming*, Vol. 14, No. 1, pp.265–294.

Park, C. and Chou, P. (2006) 'Ambimax: autonomous energy harvesting platform for multi-supply wireless sensor nodes', *SECON'06*, Vol. 1, pp.168–177.

Piorno, J., Bergonzini, C., Atienza, D. and Rosing, T. (2009) 'Prediction and management in energy harvested wireless sensor nodes', *Wireless Communication, Vehicular Technology, Information Theory and Aerospace and Electronic Systems Technology, 2009. Wireless VITAE 2009. 1st International Conference on*, IEEE, pp.6–10.

Raghunathan, V., Kansal, A., Hsu, J., Friedman, J. and Srivastava, M. (2005) 'Design considerations for solar energy harvesting wireless embedded systems', *Proceedings of the 4th International Symposium on Information Processing in Sensor networks*, IEEE Press, p.64.

Ren, Z., Cheng, P., Chen, J., Yau, D.K.Y. and Sun, Y. (2012) 'Dynamic activation policies for event capture with rechargeable sensors', *ICDCS*, IEEE, pp.152–162.

Schrijver, A. (2003) *Combinatorial Optimization: Polyhedra and Efficiency*, Springer-Verlag, New York, Vol. 24.

Stark, I. (2006) 'Invited talk: thermal energy harvesting with thermo life', *Wearable and Implantable Body Sensor Networks*, pp.19–22.

Tang, S. and Yang, L. (2012) 'Morello: a quality-of-monitoring oriented sensing scheduling protocol in sensor networks', *INFOCOM, 2012 Proceedings IEEE*, IEEE, pp.2676–2680.

Tang, S., Li, X-Y., Shen, X., Zhang, J., Dai, G. and Das, S.K. (2011) 'Cool: on coverage with solar-powered sensors', *ICDCS*, pp.488–496.

Wang, X., Xing, G., Zhang, Y., Lu, C., Pless, R. and Gill, C. (2003) 'Integrated coverage and connectivity configuration in wireless sensor networks', *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems*, ACM, pp.28–39.

Yang, Y., Wang, L., Noh, D., Le, H. and Abdelzaher, T. (2009) 'Solarstore: enhancing data reliability in solar-powered storage-centric sensor networks', *Proceedings of the 7th International Conference on Mobile Systems, Applications, and Services*, ACM, pp.333–346.

Yang, Y., Su, L., Gao, Y. and Abdelzaher, T. (2010) 'Solarcode: utilizing erasure codes for reliable data delivery in solar-powered wireless sensor networks', *INFOCOM, 2010 Proceedings IEEE*, IEEE, pp.1–5.

Yau, D.K.Y., Yip, N.K., Ma, C.Y.T., Rao, N.S.V. and Shankar, M. (2010) 'Quality of monitoring of stochastic events by periodic and proportional-share scheduling of sensor coverage', *ACM Transactions on Sensor Networks*, Vol. 7, No. 2, August, pp.1–49.

Zhu, T., Zhong, Z., Gu, Y., He, T. and Zhang, Z. (2009) 'Leakage-aware energy synchronization for wireless sensor networks', in *Proceedings of the 7th International Conference on Mobile Systems, Applications, and Services*, ACM, pp.319–332.