Optimal Real-Time Sampling Rate Assignment for Wireless Sensor Networks

XUE LIU, QIXIN WANG, WENBO HE, MARCO CACCAMO, and LUI SHA
University of Illinois at Urbana-Champaign

How to allocate computing and communication resources in a way that maximizes the effectiveness of control and signal processing, has been an important area of research. The characteristic of a multi-hop Real-Time Wireless Sensor Network raises new challenges. First, the constraints are more complicated and a new solution method is needed. Second, a distributed solution is needed to achieve scalability. This article presents solutions to both of the new challenges. The first solution to the optimal rate allocation is a centralized solution that can handle the more general form of constraints as compared with prior research. The second solution is a distributed version for large sensor networks using a pricing scheme. It is capable of incremental adjustment when utility functions change. This article also presents a new sensor device/network backbone architecture—Real-time Independent CHannels (RICH), which can easily realize multi-hop real-time wireless sensor networking.

Categories and Subject Descriptors: C.2.2 [**Computer-Communication Networks**]: Network Protocols—*Applications*; C.3 [**Special-Purpose and Application-Based Systems**]—*Real-time and embedded systems*

General Terms: Algorithms, Design, Performance, Experimentation

Additional Key Words and Phrases: Sensor network, real-time wireless sensor network, optimization, pricing, distributed algorithms

1. INTRODUCTION AND RELATED WORK

Real-Time Wireless Sensor Network (RTWSN) is expected to carry out various applications such as remote control or video/audio monitoring in *ad hoc* environments. Instead of using conservative (lowest allowed) sampling/actuating rates (since sampling and actuating rate allocation are similar, unless explicitly denoted, "sampling rate" is used instead of "sampling/actuating rate" in

This research is supported by (listed in alphabetical order) MURI N00014-01-0576, NSF ANI 02-21357, NSF CCR 02-09202, and ONR N00014-02-1-0102. The first two authors are also supported by Saburo Muroga Fellowship and Vodaphone Fellowship, respectively.
Part of this research has been published at the 24th IEEE Real-Time Systems Symposium [Liu et al. 2003].
Authors' address: Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL 61801; email: {xueliu,qwang4,wenbohe,mcaccamo,lrs}@cs.uiuc.edu.

the following), a sampling rate allocation that maximizes global utility while maintaining real-time schedulability is wanted.

Resource allocation has been studied in general for computing systems [Kurose and Simha 1989; Waldspurger and Weihl 1994; Bolot et al. 1994]. Recently, the problem of resource allocation and congestion control in network has been studied together by Kelly and others [Kelly et al. 1998; Kelly 1997; Low and Lapsley 1999]. However, these works do not consider real-time constraints, and therefore cannot be directly applied to real-time systems.

Stoica et al. [1996] have studied a proportional share resource allocation algorithm for real-time, time-shared systems. The scheme is for single processor scheduling and is more focused on fairness of the scheduling algorithms rather than system optimality. Finding the optimal control rates subject to schedulability constraints was first studied by Seto et al. [1996] and by Sha et al. [2000] for analog and digital controllers respectively. An offline solution method is given based on the Kuhn-Tucker condition. However, the schedulability analysis is still for a single processor. Rajkumar et al. [1997] investigated the QoS-based Resource Allocation Model (Q-RAM), which is capable of handling complex multiple quality dimensions. But the solution can only be used with single constraint scenario. In the following works, Lee et al. [1999] and Ghosh et al. [2003] studied the scenario under multiple constraints. However, the problem they studied is an integer programming problem, which is different from the model we will discuss in this article. In Lee et al. [1999], the integer programming problem is proven to be NP-Hard. Several suboptimal algorithms are proposed. According to Ghosh et al. [2003], the one that scales well is Hierarchical Q-RAM. However, that algorithm requires the division of multiple constraints into independent groups, which is impractical for multi-hop RTWSN.

RTWSN presents new challenges for real-time resource allocation. Routes in a RTWSN may interleave with each other. The sampling rate optimization must take into consideration the traffic contention at each router. This makes the optimization problem harder than those studied before. We present an *Interior Point Method* (IPM)-based solution to show how the optimal rates can be found efficiently. On the other hand, though a centralized method is usually efficient for small and moderately large RTWSNs, it may not scale well for very large RTWSNs, because control messages for optimization converge at the central node and create a bottleneck. To dynamically find the global optimum in a very large network, a distributed solution is needed to generate balanced optimization control traffic that avoids bottlenecks. In addition, the solution should be incremental, so that when the utility functions at a few nodes change, an updated optimum can be found with small cost.

To the best of our knowledge, the work by Caccamo et al. [2002] is the first to provide real-time support for multi-hop RTWSN. In Caccamo et al. [2002], a cellular base station layout is deployed as the backbone for the whole RTWSN, as shown in Figure 1. The base station network uses seven nonoverlapping *Radio Frequency* (RF) bands. At the center of each cell, there is one base station, which also functions as a router. A router in a cell labeled i has a single transmitter that always transmits at RF band i, and a single receiver that

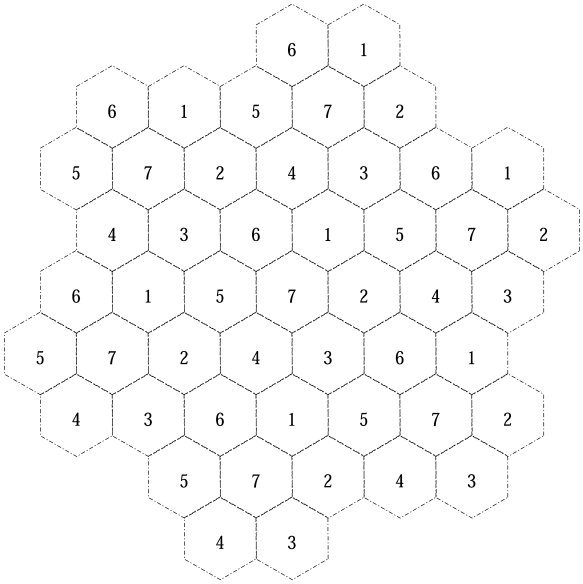


Fig. 1. A mixed FDMA-TDMA base station backbone layout.

receives from one neighbor at a time (i.e. listens to one of the six neighbors' RF bands at a time). All RF broadcasts are one-hop. The specific geographical layout makes each base station and its six neighbors transmit with distinct RF bands, and any two base stations sending with the same RF band are at least two cells apart. The intercell communication uses a globally synchronized TDMA scheme. Specifically, all base stations' receivers listen to their northeast neighbors at time slot 1, listen to east neighbors at time slot 2, so on and so forth. Therefore, the interbase station communication is a mixed FDMA-TDMA scheme. More recently, based on the mixed FDMA-TDMA scheme, Giannecchini et al. [2004] provide an online *suboptimal* approximation algorithm (CoRAl) to dynamically reconfigure sensing rates of RTWSN. CoRAl runs fast but only applies to exponential performance loss function. It is worth mentioning that inside of each cell, there can be randomly distributed wireless slave sensors with more constrained capabilities, which do the actual sensing and communicating with the cell's base station at other RF bands (which do not interfere with interbase station communications). *The intracell communication is not the focus of this article*, and therefore intracell sensors are not plotted in this article's figures.

We adopt Caccamo et al. [2002]'s cellular base station backbone layout. In this article, *we focus on the scenario that data flows among backbone base stations are unicast flows*. More sophisticated routing topologies such as multicast and convergecast are beyond the scope of this article.[1] Also, we assume routes are already decided by given algorithms, for example SPIN [Heinzelman et al.

[1]Intracell communications between slave sensors and the base station are more often convergecast, but are not the focus of this article.

1999], GPSR [Karp and Kung 2000], GEAR [Xu et al. 2001], SPEED [He et al. 2003] or Rumor Routing [Braginsky and Estrin 2002] and soforth[2]; and the total data bandwidth of each one-hop wireless link is adjusted so that the wireless medium is reliable enough for real-time communication. (According to information theory and CDMA theory [Viterbi 1995] (also see Appendix A), if the worst case wireless medium condition is given, there is an upper bound on data bit throughput so that the data bit error rate can be maintained below an upper bound.) How to incorporate routing and tolerate device failures and message drops into our sampling rate optimization, are future research issues beyond the scope of this article. According to information theory, the interbase station bandwidth available is determined by wireless medium quality and reception quality requirements, and is fundamentally irrelevant with specific multiple access schemes, such as FDMA, TDMA or CDMA. The mixed FDMA-TDMA multiple access scheme proposed in Caccamo et al. [2002] provides poorer flexibility on bandwidth allocation adjustment, and the scheduling mechanism is more complicated. Therefore, in this article, we propose a mixed FDMA and *Direct Sequence Spread Spectrum CDMA* (DSSS-CDMA)[3] scheme called *Real-time Independent CHannels* (RICH), which provides better flexibility and whose real-time scheduling is easier to implement and analyze. A brief tutorial on DSSS-CDMA is provided in Appendix A.

The major contributions of this article are:

—Study and model the optimal rate assignment problem in RICH multi-hop RTWSN using real-time schedulability analysis and nonlinear optimization.
—Using the state-of-the-art methods in optimization, two solutions to the optimal rate assignment problem are given. One is in a centralized fashion using the Interior Point Method, the other is in a distributed fashion based on a pricing scheme.
—Compare the trade-offs between the centralized and distributed algorithms under different situations.

The rest of the article is organized as follows. Section 2 describes our proposed RICH architecture, which can easily support multi-hop real-time networking. We also give the real-time schedulability constraints analysis in this section. An example of RICH RTWSN is presented to show the practicability of RICH. In Section 3, we formulate the optimal QoS sampling rate assignment problem into a nonlinear programming problem. Sections 4 and 5 give centralized and distributed solution for optimal QoS sampling rate assignment respectively. In Section 6, we first compare the numerical solutions based on the example discussed in Section 2, discuss in details the trade-offs between the centralized and distributed solution. Then we analyze the possible problems and solutions associated with both methods. Finally, conclusions and future work are discussed in Section 7.

[2]A good survey of routing protocols in sensor networks is given in Akkaya and Younis [2005].
[3]Nowadays, the term CDMA usually refers to DSSS-CDMA.

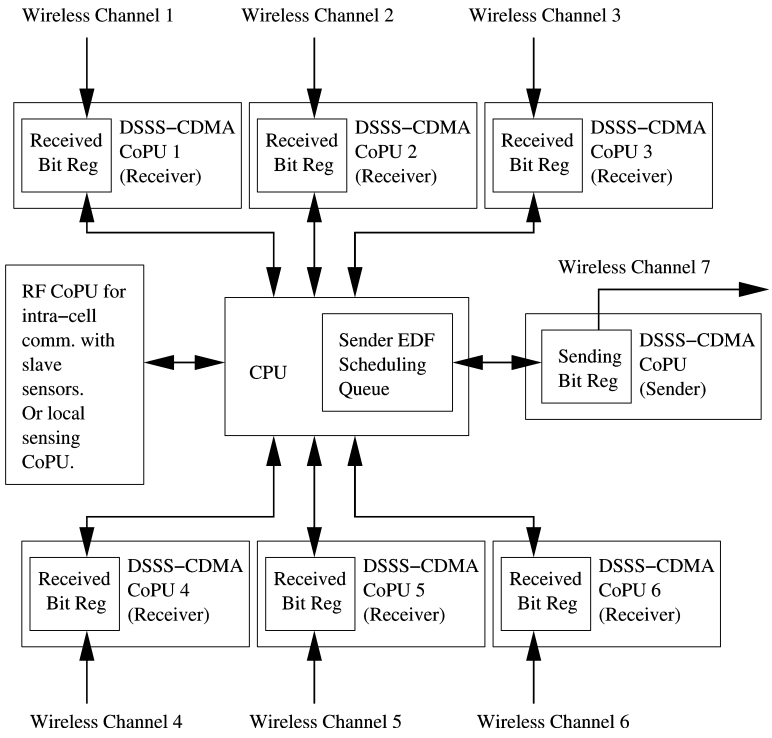


Fig. 2. Internal architecture of a RICH base station.

2. SUPPORTING MULTI-HOP RTWSN

2.1 RICH Architecture

We assume our wireless base stations (the so-called RICH base stations) have the internal architecture illustrated by Figure 2. Each RICH base station has seven DSSS-CDMA modulation/demodulation co-processors (CoPUs), each operates with a distinct DSSS-CDMA *Pseudo Noise* (PN) sequence at a distinct FDMA RF band. Among which, six of the DSSS-CDMA CoPUs are receivers, and the other one is the only transmitter at the base station. We allow the data bit bandwidths of each DSSS-CDMA CoPU to be distinct. For the time being, suppose the singular transmitter sends packets according to *Earliest Deadline First* (EDF) scheduling algorithm. A dedicated EDF scheduling queue is attached to it to buffer/schedule the outgoing packets. In addition, a RICH base station also includes a sensing CoPU, or an intracell communication CoPU that gathers data from intracell slave sensors. The CPU interacts with each CoPU by periodical polling.

We adopt the cellular base station layout of Caccamo et al. [2002] (see Figure 3), and maintain the seven RF band coloring of the cells. Meanwhile, we deploy forty-nine DSSS-CDMA PN sequences, denoted as $A1, \ldots, A7$, $B1, \ldots, B7$, $C1, \ldots, C7$, $D1 \ldots, D7$, $E1, \ldots, E7$, $F1, \ldots, F7$, $G1, \ldots,$ and $G7$ respectively. In a cell labeled XY, the RICH base station transmitter deploys

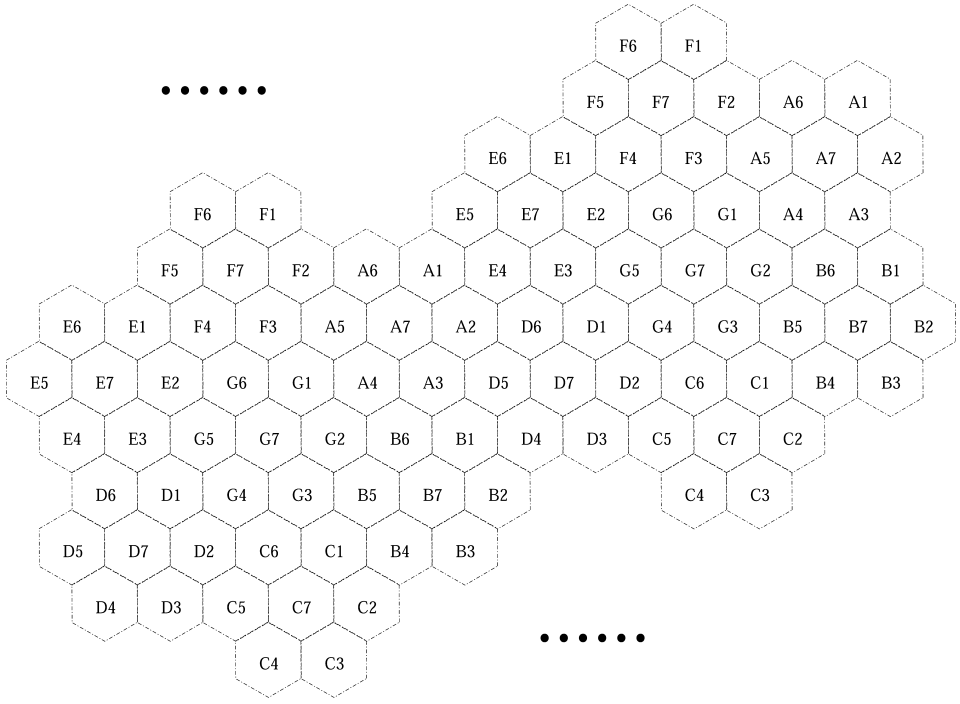


Fig. 3. The mixed FDMA-CDMA base station backbone layout.

the XY th PN sequence for DSSS-CDMA modulation, and transmits at the Y th RF band. For example, the base station in a cell labeled $G7$ transmits with the $G7$th DSSS-CDMA PN sequence at the 7th RF band. The transmission range of every transmitter in our RICH RTWSN is one-hop. Each of the six receivers on a RICH base station listens to one of its one-hop neighbor's transmissions. Take the RICH base station at a cell labeled $A5$ for example, its six receivers listen to the 6, 7, 4, 1, 3, 2th RF band respectively, and demodulate with DSSS-CDMA PN sequence $A6$, $A7$, $A4$, $G1$, $F3$, $F2$ respectively.

Under such a design, the broadcast of a base station is simultaneously received by its six one-hop neighbors. The effective receiver is designated by the broadcast packet's "destination" data segment. More importantly, because of the deployment of DSSS-CDMA, any transmission can be carried out independently. For example, in Figure 3, an $F2$ base station and an $A6$ base station can both send packets to an $A5$ base station (their common one-hop neighbor) at any time. Furthermore, the schedule can be independently adjusted to be specific per base station. For example, the $F2$ base station can dedicate 100% of its time sending to the $A5$ base station; at the same time, the $A6$ base station can dedicate $\frac{1}{3}$ of its time to the $A5$ base station. There are no synchronization requirements between any pair of transmissions. Under TDMA, however, this is impossible. For example, $F2$'s sending schedule must not overlap with $A6$'s. Such mutually exclusive relationships propagate throughout the network; finally all base stations' broadcast schedules are interlocked, which complicates analysis and reconfiguration.

Also, the layout guarantees two base stations transmiting with same DSSS-CDMA PN sequence (and therefore at the same RF band) are at least eight hops away. This implies that *a base station does not have to be at the center of its cell*.

2.2 Schedulability Analysis of RICH RTWSN

The broadcast of a RICH base station is overheard by all of its six neighbor base stations. Usually, the wireless medium conditions to the six neighbors are irregular [Zhou et al. 2004; Zhao and Govindan 2003]. According to DSSS-CDMA theory (see Appendix A), given RF band, worst case wireless medium conditions, and maximum acceptable bit error rate, the upper bound of data bit bandwidth is determined—which we call *affordable bandwidth*. Suppose for a RICH base station X, because of the irregularity of the wireless medium, the affordable bandwidths to its six neighboring RICH base stations are $B_1, B_2, \ldots, B_6$. We set the transmission data bit bandwidth of X to be $B = \min\{B_1, B_2, \ldots, B_6\}$. Therefore the broadcast of X is reliably received by all of its six neighbors—the bit error rate of one-hop transmission is always below the maximum acceptable bound. In another words, B models factors such as the impact of radio irregularity on the wireless medium.

For a RICH base station, the real-time scheduling is carried out in the EDF scheduling queue attached to its singular transmitter. Let $\mathcal{T}$ be the set of all routes that go through it. For a route $\tau \in \mathcal{T}$, suppose it has a sampling/reporting rate of f_τ, and each report is a packet of length l_τ. The corresponding transmission time of the packet is therefore $c_\tau = l_\tau/B$. When there are multiple contending routes through one RICH base station, the transmitter should be regarded as a *nonpreemptive* resource, because once a packet starts transmitting, it cannot be preempted until it is completely transmitted. Therefore, the real-time scheduler of a RICH base station's EDF queue should be a *nonpreemptive EDF scheduler* to ensure both the EDF behavior and nonpreemptive usage of the broadcast link. Under such scheduling, a packet (job) can be blocked by at the most, one other packet. Therefore we can apply the well-known schedulability bound for the nonpreemptive EDF scheduler as follows [Buttazzo 1997]:

$$\sum_{\tau \in \mathcal{T}} c_\tau f_\tau + C_j f_j \leq 1, \quad \text{for each} \quad j \in \mathcal{T}, \tag{1}$$

where C_j is the maximum blocking time for sending a packet of route j.

$$\begin{aligned} C_j &= \max_{\{\tau \in \mathcal{T} \text{ and } \tau \neq j\}} \{c_\tau\} \\ &= \max_{\{\tau \in \mathcal{T} \text{ and } \tau \neq j\}} \left\{ \frac{l_\tau}{B} \right\} \\ &= \max_{\{\tau \in \mathcal{T} \text{ and } \tau \neq j\}} \frac{\{l_\tau\}}{B}. \end{aligned}$$

Therefore (1) is transformed into:

$$\sum_{\tau \in \mathcal{T}} \frac{l_\tau}{B} f_\tau + \frac{L_j}{B} f_j \leq 1, \quad \text{for each} \quad j \in \mathcal{T}, \tag{2}$$

where $L_j = \max_{\{\tau \in \mathcal{T} \text{ and } \tau \neq j\}}\{l_\tau\}$.

Multiply both sides of Equation (2) with broadcast link bandwidth B; we get:

$$\sum_{\tau \in \mathcal{T}} l_\tau f_\tau + L_j f_j \leq B, \quad \text{for each} \quad j \in \mathcal{T}. \tag{3}$$

Besides being a router, a RICH base station can also simultaneously function as the source end of a route. The data are either from the base station's local sensing CoPU or by gathering intracell slave sensors' readings. Either way, the base station can be regarded as the virtual singular source end sensor for the route, and its sampling rate is upper bounded, creating the following constraint (for the time being, we assume each base station can be the source end of at the most *one* route):

$$f_j \leq f_j^{\max}, \tag{4}$$

where f_j is the sampling rate of route j, and $f_j^{\max}$ is the maximum affordable sampling rate at the route's source end base station.[4] By analyzing each base station of the RICH RTWSN according to inequality set (3) and each route according to inequality (4), we can derive a set of linear inequalities, which are the sufficient real-time schedulability constraints for RICH RTWSN sampling rate assignment. We can summarize them in the following form:

$$\begin{cases} \mathbf{A}f \leq W \\ f \leq f^{\max}, \end{cases} \tag{5}$$

where $f = (f_1, f_2, \ldots, f_N)^\mathsf{T}$ is the vector of sampling rates assigned to each of the N routes. $f^{\max} = (f_1^{\max}, f_2^{\max}, \ldots, f_N^{\max})^\mathsf{T}$ is the maximum sampling rate for the N end point sensors (see (4)). Matrix $\mathbf{A}$ and vector W are obtained by base station-wise analysis according to (3), which reflects the specific routing topology of the RICH RTWSN. Suppose this schedulability analysis generates M inequalities in total, then $\mathbf{A} \in \mathbb{R}^{M \times N}$, $W \in \mathbb{R}^{M \times 1}$.

Besides the constraints from real-time schedulability, there are often application specific minimum sampling/reporting rate requirements. These extra requirements can be written as:

$$f \geq f^{\min} = \left(f_1^{\min}, \ldots, f_N^{\min}\right)^\mathsf{T}. \tag{6}$$

Inequality sets (5) and (6) constitute a complete set of real-time schedulability constraints for RICH RTWSN sampling rate allocation. An example is given as follows:

[4]In this article, we assume the capacity of CPU, internal bus, sensing CoPU and RF CoPUs are big enough, so that as long as (4) is satisfied, we only need to be concerned about the wireless network bandwidth schedulability.

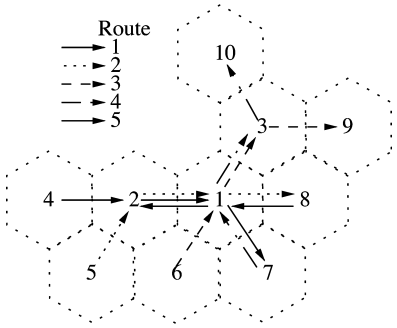


Fig. 4. A RICH RTWSN schedulability analysis example.

Example 1. As depicted by Figure 4, a RICH RTWSN consists of 10 base stations (as labeled $1, 2, \ldots, 10$) and 5 routes (identified with different arrow styles): Route 1: $4 \rightarrow 2 \rightarrow 1 \rightarrow 7$; Route 2: $5 \rightarrow 2 \rightarrow 1 \rightarrow 8$; Route 3: $6 \rightarrow 1 \rightarrow 3 \rightarrow 9$; Route 4: $7 \rightarrow 1 \rightarrow 3 \rightarrow 10$; Route 5: $8 \rightarrow 1 \rightarrow 2$.

Let $B_i (i = 1, \ldots, 10)$ be the transmitting bandwidth of base station i. Suppose the sampling rate assigned to Route j to be $f_j (j = 1, \ldots, 5)$. Let l_j, $f_j^{\min}$ and $f_j^{\max} (j = 1, \ldots, 5)$ be the data packet size, minimum and maximum sampling rate constraints for Route j.

According to the topology in Figure 4, we have the following real-time schedulability constraints:

Node 1: Route 1, 2, 3, 4, 5 are passing through it, hence:

$$\begin{aligned}
(l_1 f_1 + l_2 f_2 + l_3 f_3 + l_4 f_4 + l_5 f_5) + \max\{l_2, l_3, l_4, l_5\} f_1 &\leq B_1 \\
(l_1 f_1 + l_2 f_2 + l_3 f_3 + l_4 f_4 + l_5 f_5) + \max\{l_1, l_3, l_4, l_5\} f_2 &\leq B_1 \\
(l_1 f_1 + l_2 f_2 + l_3 f_3 + l_4 f_4 + l_5 f_5) + \max\{l_1, l_2, l_4, l_5\} f_3 &\leq B_1 \\
(l_1 f_1 + l_2 f_2 + l_3 f_3 + l_4 f_4 + l_5 f_5) + \max\{l_1, l_2, l_3, l_5\} f_4 &\leq B_1 \\
(l_1 f_1 + l_2 f_2 + l_3 f_3 + l_4 f_4 + l_5 f_5) + \max\{l_1, l_2, l_3, l_4\} f_5 &\leq B_1
\end{aligned}$$

Node 2: Route 1, 2 are passing through it, hence:

$$\begin{aligned}
(l_1 f_1 + l_2 f_2) + l_2 f_1 &\leq B_2 \\
(l_1 f_1 + l_2 f_2) + l_1 f_2 &\leq B_2
\end{aligned}$$

As the destination end for Route 5, there are no constraints (the corresponding constraints are analyzed at base station 1, Route 5's last sending hop).

Node 3: Route 3, 4 are passing through it, hence:

$$\begin{aligned}
(l_3 f_3 + l_4 f_4) + l_4 f_3 &\leq B_3 \\
(l_3 f_3 + l_4 f_4) + l_3 f_4 &\leq B_3
\end{aligned}$$

Node 4: As a base station along Route 1, we have $l_1 f_1 \leq B_4$.

Node 5: As a base station along Route 2, we have $l_2 f_2 \leq B_5$.

Node 6: As a base station along Route 3, we have $l_3 f_3 \leq B_6$.

Node 7: As a base station along Route 4, we have $l_4 f_4 \leq B_7$. As the destination end for Route 1, there are no constraints (the corresponding constraints are analyzed at base station 1, Route 1's last sending hop).

Node 8: As a base station along Route 5, we have $l_5 f_5 \le B_8$. As the destination end for Route 2, there are no constraints (the corresponding constraints are analyzed at base station 1, Route 2's last sending hop).

Node 9 and Node 10: As purely destination end for routes, there are no constraints (corresponding constraints are analyzed at the corresponding routes' last sending hops).

$$\begin{cases} \boldsymbol{A} = \begin{bmatrix} l_1+\max\{l_2,l_3,l_4,l_5\} & l_2 & l_3 & l_4 & l_5 \\ l_1 & l_2+\max\{l_1,l_3,l_4,l_5\} & l_3 & l_4 & l_5 \\ l_1 & l_2 & l_3+\max\{l_1,l_2,l_4,l_5\} & l_4 & l_5 \\ l_1 & l_2 & l_3 & l_4+\max\{l_1,l_2,l_3,l_5\} & l_5 \\ l_1 & l_2 & l_3 & l_4 & l_5+\max\{l_1,l_2,l_3,l_4\} \\ l_1+l_2 & l_2 & 0 & 0 & 0 \\ l_1 & l_1+l_2 & 0 & 0 & 0 \\ 0 & 0 & l_3+l_4 & l_4 & 0 \\ 0 & 0 & l_3 & l_3+l_4 & 0 \\ l_1 & 0 & 0 & 0 & 0 \\ 0 & l_2 & 0 & 0 & 0 \\ 0 & 0 & l_3 & 0 & 0 \\ 0 & 0 & 0 & l_4 & 0 \\ 0 & 0 & 0 & 0 & l_5 \end{bmatrix} \\ f = (f_1, f_2, f_3, f_4, f_5)^{\mathsf{T}} \\ W = (B_1, B_1, B_1, B_1, B_1, B_2, B_2, B_3, B_3, B_4, B_5, B_6, B_7, B_8)^{\mathsf{T}} \end{cases} \tag{7}$$

In addition, because of the minimum sampling rate constraints, we have: $f_j \ge f_j^{\min}$, where $(j = 1, \ldots, 5)$. The complete rate allocation constraints are: $\mathbf{A}f \le W$, $f \le (f_1^{\max}, \ldots, f_5^{\max})^{\mathsf{T}}$ and $f \ge (f_1^{\min}, \ldots, f_5^{\min})^{\mathsf{T}}$, which are detailed by (7).

Suppose the numerical values of parameters are as shown in Table I, then the complete rate allocation constraints are transformed into (8).

3. OPTIMIZING QOS IN WSN WITH REAL-TIME CONSTRAINTS—MATH MODELING

In this section, we model the optimal sampling rate allocation problem as a nonlinear convex optimization problem, using constraints set (5) (6) from the previous section.

3.1 Utility Loss Index

The base station at the source end of a route periodically samples and reports sensor readings. Let the sampling/reporting rate (or "frequency") for the jth route be f_j. For most applications, the higher the sampling/reporting rate f_j, the higher is the QoS. For example, for control applications, the faster the sampling rate, the better the control performance [Seto et al. 1996]. Ideally, the best performance is achieved if the sampling rate is approaching ∞, that is continuous sampling. In practice, this is not achievable, so we use the *Utility Loss Index* (ULI) function to capture the performance *loss* at a discrete sampling

Table I. Parameter Values for Example 1

Node	Bandwidth(B_i Mbps)	Max Sampling Capability ($f^{\max}$ Hz)
1	1.0	—
2	0.6	—
3	0.4	—
4	0.25	($f_1^{\max} =$)30
5	0.25	($f_2^{\max} =$)25
6	0.25	($f_3^{\max} =$)30
7	0.2	($f_4^{\max} =$)40
8	0.15	($f_5^{\max} =$)30

Route	Required Min Freq. ($f_j^{\min}$ Hz)	Affordable Max Freq. ($f_j^{\max}$ Hz)	Report Packet Size (l_j Mbit)
1	11	30	0.01
2	2.5	25	0.015
3	5	30	0.02
4	1	40	0.025
5	2	30	0.03

$$\begin{cases} \mathbf{A}f = \begin{bmatrix} .04 & .015 & .02 & .025 & .03 \\ .01 & .045 & .02 & .025 & .03 \\ .01 & .015 & .05 & .025 & .03 \\ .01 & .015 & .02 & .055 & .03 \\ .01 & .015 & .02 & .025 & .055 \\ .025 & .015 & 0 & 0 & 0 \\ .01 & .025 & 0 & 0 & 0 \\ 0 & 0 & .045 & .025 & 0 \\ 0 & 0 & .02 & .045 & 0 \\ .01 & 0 & 0 & 0 & 0 \\ 0 & .015 & 0 & 0 & 0 \\ 0 & 0 & .02 & 0 & 0 \\ 0 & 0 & 0 & .025 & 0 \\ 0 & 0 & 0 & 0 & .03 \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \end{bmatrix} \le W = \begin{bmatrix} 1.0 \\ 1.0 \\ 1.0 \\ 1.0 \\ 1.0 \\ .6 \\ .6 \\ .4 \\ .4 \\ .25 \\ .25 \\ .25 \\ .2 \\ .15 \end{bmatrix}, \\ f \le f^{\max} = (30, 25, 30, 40, 30)^{\mathsf{T}}, \text{ and } f \ge f^{\min} = (11, 2.5, 5, 1, 2)^{\mathsf{T}}. \end{cases} \tag{8}$$

rate f compared to continuous sampling. For control applications, Seto et al. [1996] show the ULI is in the following form:

$$U_j(f_j) = \omega_j \alpha_j e^{-\beta_j f_j}, \tag{9}$$

where f_j is the sampling rate for task j (i.e. route j), and ω_j, α_j, and β_j are application-specific constraints. The values of ω_j, α_j, and β_j can be determined through data fitting using real-world measurements. In this article, the form of the ULI function is generalized to be any strictly decreasing differentiable convex function with regard to rate f.

3.2 Mathematical Formulation

Assume each individual ULI function $U_j(f_j)$ is strictly decreasing differentiable and convex. Suppose ULIs are additive, the system's overall ULI is thereby the sum of the ULIs of all individual routes: $\sum_{j=1}^{N} U_j(f_j)$. The performance optimization problem becomes:

$$\min_{(f_1,\ldots,f_N)} \sum_{j=1}^{N} U_j(f_j) \tag{10}$$

$$\text{such that:} \quad \mathbf{A}f \leq W \tag{11}$$

$$f \leq f^{\max} \tag{12}$$

$$f \geq f^{\min} \tag{13}$$

where $\mathbf{A}$ is a constraint matrix with dimension $M \times N$. M is dependent upon the routing topology of the RTWSN, and N is the number of total routes. We call this problem the *Multiple Constraints Optimization Problem* (MCOP) in contrast to Seto et al. [1996]'s *Single Constraint Optimization Problem* (SCOP), and we denote the former as $MCOP(U, \mathbf{A}, W)$.[5]

The feasible set of MCOP is compact and convex and $U_j(f_j)$ is differentiable and convex, therefore MCOP has optimal solutions [Bertsekas 1995]. Furthermore, if $U_j(f_j)$ is strictly convex, the optimal solution is unique [Bertsekas 1995].

When $M = 1$ and there is no constraint set (12), and the ULIs are in the negative exponential form, then $MCOP(U, \mathbf{A}, W)$ becomes SCOP as follows [Seto et al. 1996]:

$$\min_{(f_1,\ldots,f_N)} \sum_{j=1}^{N} U_j(f_j) = \sum_{j=1}^{N} \omega_j \alpha_j e^{-\beta_j f_j} \tag{14}$$

$$\text{such that:} \quad Af \leq W \tag{15}$$

$$f \geq f^{\min} = \left(f_1^{\min}, \ldots, f_N^{\min}\right)^{\mathsf{T}}, \tag{16}$$

where W is the bandwidth (utilization) constraints, and $A \in \mathbb{R}^{1\times N}$, $f \in \mathbb{R}^{N\times 1}$, $W \in \mathbb{R}$.

Based on the Kuhn-Tucker condition, Seto et al. [1996] provide an algorithm to solve the SCOP problem analytically. MCOP is a generalization of SCOP. We will show that the approach for deriving an analytical solution to SCOP is not viable for solving MCOP. To this end, we first prove that the optimal solution f^* of MCOP will make at least one of the constraints in constraint sets (11) and (12) become the equality constraint, and show why it is not viable to tackle the MCOP in an analytical fashion similar to Seto et al. [1996]. In $MCOP(U, \mathbf{A}, W)$, constraint sets (11) and (12) can be combined as:

$$\mathbf{A}'f \leq W',$$
$$\text{where } \mathbf{A}' = \begin{bmatrix} \mathbf{A} \\ \mathbf{I} \end{bmatrix}_{(M+N)\times N}, W' = \begin{bmatrix} W \\ f^{\max} \end{bmatrix}_{(M+N)\times 1}. \tag{17}$$

[5]Notation used in Kelly et al. [1998].

THEOREM 3.1. *MCOP's optimal solution f^*, must ensure that at least one of the $(M + N)$ constraints $A_i' f^* \leq W_i', i = 1, \ldots, (M + N)$ reaches equality: $\exists i \in \{1, \ldots, M + N\}$ such that $A_i' f^* = W_i'$. Here A_i' is the ith row of $\mathbf{A}'$.*

PROOF. Please refer to Appendix B for the proof. □

Though we know for the optimal rate assignment f^*, there is at least one i such that $A_i' f^* = W_i'$, we do not know explicitly which constraints reach equality. This makes the Kuhn-Tucker based solution method not applicable. In contrast, the problem discussed in Seto et al. [1996], which is a SCOP ($M = 1$), is much easier because there is only one nontrivial constraint $A_1 f^* \leq W_1$, and it is exactly this constraint that should reach equality. In addition to Seto et al. [1996], Rajkumar et al. [1997] proposed a numerical solution. But that solution is also for a single constraint scenario. In their more recent works, Lee et al. [1999] and Ghosh et al. [2003] studied the scenario under multiple constraints. However, the problem they studied is an integer programming problem, which is different from the model we will discuss in this article. In Lee et al. [1999], the integer programming problem is proven to be NP-Hard. Several suboptimal algorithms are proposed. According to Ghosh et al. [2003], the one that scales well is Hierarchical Q-RAM. However, that algorithm requires the division of multiple constraints into independent groups, which is impractical for multi-hop RTWSN (see Section 6.5). Fortunately, as will become clear later, MCOP can be solved with the state-of-art Interior Point Methods [Nesterov and Nemirovsky 1994; Ye 1997a] and Internet pricing schemes [Low and Lapsley 1999; Kelly et al. 1998; Kelly 1997].

4. CENTRALIZED SOLUTION METHOD FOR MCOP

In this section, we apply the *Interior Point Method* (IPM) to solve the MCOP problem for RTWSN.

Definition 4.1. A *Constrained Optimization Problem* is expressed as $\min\{f(x) : x \in Q \subseteq \mathbb{R}^n\}$, where the constraint set Q is defined by multiple equalities and inequalities: $Q = \{x : h(x) = 0, g(x) \leq 0\}$, where $h : \mathbb{R}^n \to \mathbb{R}^p; g : \mathbb{R}^n \to \mathbb{R}^q$.

Definition 4.2. A *Convex Optimization* (CO) problem is a constrained optimization problem whose objective function $f(x)$ is continuous and convex, and whose constraint set Q is compact (closed and bounded) and convex.

It's easy to see that an MCOP is a constrained convex optimization problem with linear constraints. For solving a convex optimization problem, the major difficulties come from the multiple inequality constraints. Closed form solutions are generally unavailable. However, IPMs [Nesterov and Nemirovsky 1994; Ye 1997a] can solve linear constraint convex optimization problems numerically. IPM is a numerical method that iterates in the interior of the solution space defined by constraint set Q to find the optimal solution. IPMs can be further divided into two subcategories: *primal methods* and *primal-dual methods*. Primal-dual methods try to solve the primal and dual optimization problems [Luenberger 1984] together. In practice, the primal-dual methods are more

efficient. The advantages of the interior-point method based numerical solution includes: (1) Efficient. IPMs give the correct solution very fast; (2) Multi-hop application scenarios: The objective function need not be confined as an exponential form as in Seto et al. [1996], but can be a general strictly decreasing differentiable convex function.

To solve the MCOP problem, we use optimization library COPL_LC [Ye 1997b]. It is easy to transform our MCOP problem to the form used by COPL_LC. The transformation method can be found in Appendix C. To implement the IPM based centralized solution, the whole RICH RTWSN elects a central computing node **C**, which gathers ULI and constraints information from all the network, carries out the optimization algorithm, and returns the final results.

5. DISTRIBUTED ALGORITHMS FOR OPTIMAL RATE ASSIGNMENT

However, a direct application of IPM results in a centralized solution that requires collecting data from each node. This will create a traffic bottleneck around the central computing node (detailed discussion is in Section 6.5). To overcome the bottleneck problem, we give a distributed algorithm for solving the MCOP. The distributed algorithm lets routers and routes' end point nodes collaborate to find the optimal rates. The algorithm is based on the recent researches of Internet pricing schemes [Low and Lapsley 1999; Kelly et al. 1998; Kelly 1997], especially Low and Lapsley [1999].

The main idea is to impose a price on each constraint in (11)∼(13). Each route will accumulate its relevant constraints' prices and solve a local optimization problem based on its own ULI function. The result is the next proposed sampling rate for the route (to simplify, we call it *rate proposal* in the following). The rate proposal is then delivered to each of the route's routers, where each of the route's relevant constraint updates (imagine each constraint as an active agent) its price (called *constraint price*) accordingly. This procedure works in an iterative manner until it converges.

The distributed algorithm has two main attributes:

(1) It converges to the optimal rates of MCOP (Theorem 5.1).
(2) Each route's computation is only based on local information.

Notations used in the Distributed Algorithm:

s. is the algorithm's iteration step, $s = 0, 1, \ldots$.

$p(s)$. is the updated constraint price vector for each constraint $i, i \in \{1, \ldots, M\}$ at iteration step s. $p(s) = (p_1(s), \ldots, p_M(s))$.

$f(s)$. is the updated rate proposal vector for each route $j, j \in \{1, \ldots, N\}$ at iteration step s. $f(s) = (f_1(s), \ldots, f_N(s))^{\mathsf{T}}$.

The Distributed Algorithm:

The distributed algorithm is made up of iterations. Each iteration consists of two consecutive steps: the *Constraint Algorithm*, and the *Route Algorithm*.

At the very beginning of the distributed algorithm, set $f(0) = f^{\min}$, and $p(0) \geq 0$.

(1) *Constraint Algorithm*

During iteration $s = 1, 2, \ldots$, for each constraint $i (i = 1, \ldots, M)$:

C1. Receives rate proposal $f_j(s)$ from each relevant route j. Route j and constraint i are relevant if $\mathbf{A}_{ij} \neq 0$.

C2. Computes a new constraint price for itself using the following price update equation:

$$p_i(s+1) = [p_i(s) + \gamma(f^i(s) - W_i)]^+ \tag{18}$$

Here $f^i(s) = A_i f(s)$, and A_i is the ith row of $\mathbf{A}$. Function $[\bullet]^+$ is defined as $[x]^+ = \max\{x, 0\}$, where x is a real number.

C3. Delivers new price $p_i(s+1)$ to all routes that are relevant to constraint i.

(2) *Route Algorithm*

During iteration $s = 1, 2, \ldots$, for each route $j (j = 1, \ldots, N)$:

R1. Receives from the network the sum of all the constraints' prices $p^j(s)$ imposed on this route:

$$p^j(t) = \sum_{i=1}^{M} p_i(s)\mathbf{A}_{ij} \tag{19}$$

R2. Update the route's rate proposal $f_j(s+1)$ for the next iteration according to the local optimization of:

$$\begin{aligned} &\min \quad U_j(f_j) + f_j p^j(s) \\ &\text{such that:} \quad f_j^{\min} \leq f_j \leq f_j^{\max} \end{aligned}$$

$$i.e.\ f_j(s+1) = \arg \min_{f_j^{\min} \leq f_j \leq f_j^{\max}} (U_j(f_j) + f_j p^j(s)) \tag{20}$$

The iteration of Constraint and Route Algorithms stops until the predefined convergence criterion is reached. For example, when both of the following criteria are met.

$$\| f(s) - f(s-1) \|_n \leq \varepsilon_f, \tag{21}$$

$$\| q(s) - q(s-1) \|_n \leq \varepsilon_q, \tag{22}$$

where $f(s) = (f_1(s), f_2(s), \ldots, f_N(s))^{\mathsf{T}}$, $q(s) = (p^1(s), p^2(s), \ldots, p^N(s))^{\mathsf{T}}$. $\varepsilon_f > 0$ and $\varepsilon_p > 0$ are sufficiently small real numbers. $\|v\|_n$ denotes the nth-norm of vector $v = (v_1, \ldots, v_k)$. If $n = 1$, $\|v\|_1 = \max(v_i), i \in \{1, \ldots, k\}$. If $n \in \mathbb{Z}^+$, then $\|v\|_n = (\sum_{i=1}^{k} v_i^n)^{\frac{1}{n}}$.

Now we prove the convergence and correctness of the above iterative algorithm. This is summarized in *Theorem* 5.1. First, we give the assumptions and notations to be used.

Assumptions:

A1. The feasibility condition holds for each constraint $i (i = 1, \ldots, M)$, such that $\sum_{j=1}^{N} \mathbf{A}_{ij} f_j^{\min} \leq W_i$.

A2. For each route j, on the interval $I_j = [f_j^{\min}, f_j^{\max}]$, the utility function U_j is *strictly decreasing, strictly convex, and twice continuously differentiable*.

A3. The curvature of U_j for each route satisfies the following condition on $I_j = [f_j^{\min}, f_j^{\max}]$, $\exists \bar{\alpha}_j$, such that $U_j''(f_j) \geq \frac{1}{\bar{\alpha}_j} > 0$, for all $f_j \in I_j$.

Notations Used in Theorem 5.1:

- $L(j) = \sum_{i=1}^{M} \mathbf{A}_{ij}$. It is the column sum of $\mathbf{A}$;
- $\bar{L} = \max_{j=1,\ldots,N}\{|L(j)|\}$, which is the maximum absolute value of the column sum of $\mathbf{A}$;
- $S(i) = \sum_{j=1}^{N} \mathbf{A}_{ij}$. It is the row sum of $\mathbf{A}$;
- $\bar{S} = \max_{i=1,\ldots,M}\{|S(i)|\}$, which is the maximum absolute value of the row sum of $\mathbf{A}$;
- $\bar{\alpha} = \max_{j=1,\ldots,N}\{\bar{\alpha}_j\}$. $\bar{\alpha}$ is the upper bound on $\frac{1}{U_j''(f_j)}$, $j = 1, \ldots, N$.

THEOREM 5.1. *Suppose assumptions A1 ∼ A3 hold and the step size γ satisfies $0 < \gamma < 2/(\bar{\alpha}\bar{L}\bar{S})$. Then starting from any initial rates $f^{\min} \leq f(0) \leq f^{\max}$ and prices $p(0) \geq 0$, the sequence $\{(f(s), p(s))\}$ generated by the above distributed algorithm will converge to a accumulation point (f^*, p^*), and f^* is the solution of MCOP$(U, \mathbf{A}, W)$.*

PROOF. See Appendix D. □

6. EVALUATION

In this section, we shall present the simulation results and discuss the tradeoffs between the centralized algorithm and the distributed algorithm, showing which is more appropriate in what situations. We also show that the distributed algorithm has the desirable incremental adjustment property.

6.1 Implementation

First let us look at the real-world feasibility of RICH architecture. Real-world DSSS Chip-Sets consist of multiple parallel independent transmitters and receivers are already available. For example, the QualComm CSM2000 chipset [Qualcomm 2004a], originally designed for low-cost lightweight cellular base stations, supports eight parallel users, which is enough for the seven-transceiver RICH architecture. Higher performance chipsets can be QualComm CSM5000 [Qualcomm 2004b], CSM5500 [Qualcomm 2004c] and so on, which can be easily reconfigured to build RICH base stations, providing no less than 1.8Mbps data bandwidth for each of the seven transceivers. The sizes and power consumption of these chip sets are also satisfactorilly small. For example, a CSM5500 chip complies with BGA560 packaging, which is $35 \times 35 \times 2.5$ mm in dimension; and is of $3 \sim 3.6$ volt I/O voltage and 1.8 volt core voltage.

Based on the above real-world parameters, we carry out simulation using J-Sim [DRCL 2004]. The centralized algorithm is straightforward. To simulate the distributed algorithm, we need to devise a network protocol that matches the algorithm described in Section 5, which is as following:

Network Protocol for Distributed Algorithm:

The protocol is carried out in iterations, each iteration s consists of two steps:

Step 1. Each constraint's price is updated by the router that creates that constraint based on (18). Next, each of these updated prices must be propagated to all the relevant routes. To do that, each route's source end sends an empty packet toward the destination end. The packet's payload is just one floating-point number (4 bytes), dedicated to carry the total price $p^j(s)$, where j refers to the jth route. As this packet travels toward the destination along the route, on each hop, it will accumulate onto $p^j(s)$ all relevant constraints' prices maintained by the local router. When the packet reaches the destination end, the total price $p^j(s)$ is obtained.

Step 2. After Step 1, each route's destination end carries out the route algorithm (20) to update the sampling rate proposal. Then the destination end sends another packet towards the source end, to notify every router along this route about the updated sampling rate proposal. This packet's payload is also just one floating-point number (4 bytes), which is enough to carry the updated sampling rate proposal.

If the payload of the control traffic is piggybacked to the data traffic, it will add a 4 bytes overload. If the control traffic is sent separately from the data traffic, it can be encoded into a 16 byte packet. Within this 16 byte packet, 4 bytes are the control payload, 4 bytes are for the source address, 4 bytes are for the destination address, and the remaining 4 bytes are for other purposes, such as checksum and so on. In the following, we discuss the separate control message scheme, that is, the distributed algorithm incurs a 16 byte packet in Step 1 and Step 2 respectively for each route.

For the distributed algorithm, we also assume all the involved routes of the RTWSN are coarse-grain synchronized in the sense that in each iteration, all the routes finish Step 1 and then move on to Step 2; and when all the routes finish Step 2, they move on to the next iteration. This can be achieved, for examples, by synchronizing all the nodes and starting each step at time kT_{step}, where $k \in \mathbb{Z}$, and T_{step} is the empirical upper bound of end-to-end packet travel time along the network diameter, assuming there is a specified upper bound on network diameter. The GPS System [Getting 1993] can already provide global time synchronization with an accuracy of within 0.25 msec [Exit Consulting 2004], which is enough for our application. For example, in our simulation setup, a synchronization granularity of 2msec is enough for the testbed.

6.2 Numerical Example of the Centralized and the Distributed Algorithm

First both the centralized and distributed algorithms are applied to the scenario discussed in Example 1 of Section 2.2. The setup involves 5 routes. The ULI function for each route j is in the form of $\omega_j \alpha_j e^{-\beta_j f_j}$, so the $MCOP(U, \mathbf{A}, W)$'s total ULI (the objective function) is: $\sum_{j=1}^{5} U_j(f_j) = \sum_{j=1}^{5} \omega_j \alpha_j e^{-\beta_j f_j}$, with parameters shown in Table II. These parameters are taken from those reported in Sha et al. [2000]. Constraints are listed in equation (8).

Table II. Parameters for ULI in Example 1

Route	α_j	β_j	ω_j
1	0.66	0.3	1
2	0.66	1.0	2
3	0.66	0.5	3
4	0.66	0.7	4
5	0.66	0.3	5

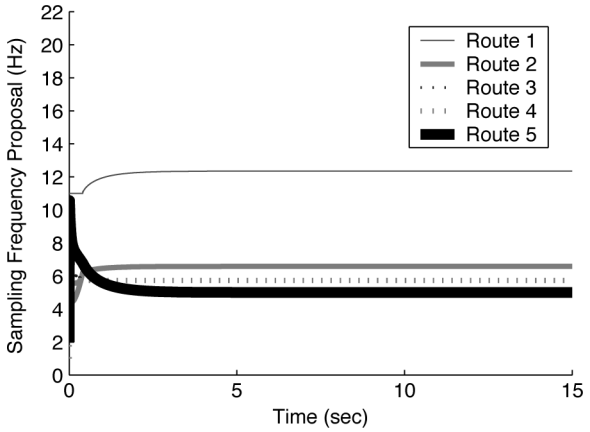


Fig. 5. Rate proposal update trace.

Using the centralized algorithm and the COPL_LC package [Ye 1997b], by 14 iterations, the optimal solution is derived: $f^*_{central} = (12.35, 6.58, 5.70, 5.73, 5.00)^\mathsf{T}$, with an optimal value of 0.916.[6]

For the distributed algorithm, we choose the initial values to be $f(0) = f^{\min}$, $p(0) = (0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)$ and $\gamma = 2 \times 10^{-7}$. The network parameter settings follow Table I. The convergence criteria are described by equations (21) and (22), where we pick $\varepsilon_f = 1 \times 10^{-9}$ and $\varepsilon_q = 1 \times 10^{-9}$. The rate proposal update trace is shown in Figure 5. The trace shows that the algorithm converges in a short time (no more than 9.48sec). The converged rate proposal value is: $f^*_{distributed} = (12.35, 6.58, 5.70, 5.73, 5.00)^\mathsf{T}$, which matches the results obtained from the centralized algorithm.

It is worth noting that, for many RTWSN applications, it is not necessary to derive the exact optimum sampling rate. Instead, getting a quasi-optimum in a relatively shorter time is often preferrable. In Table III, the convergence time for each route with certain error bound is listed. We see that if coarser error bound is allowed, the convergence time is even shorter.

6.3 Monte Carlo Simulation on Convergence Speed

In order to give a feeling of how fast the distributed algorithm converges, the following Monte Carlo simulation is carried out:

[6]The primal objective values reported in the 14 iterations are {1.649, 1.339, 1.237, 1.072, 0.834, 0.504, 0.649, 0.683, 0.792, 0.899, 0.915, 0.916, 0.916, 0.916}.

Table III. Convergence Time

Route	1	2	3	4	5
Convergence Time (CT) (sec)	6.58	5.80	5.59	5.66	9.48
CT when ± 1% error is allowed (sec)	1.79	1.39	0.45	0.27	2.73
CT when ± 5% error is allowed (sec)	0.74	0.46	0.23	0.13	1.6

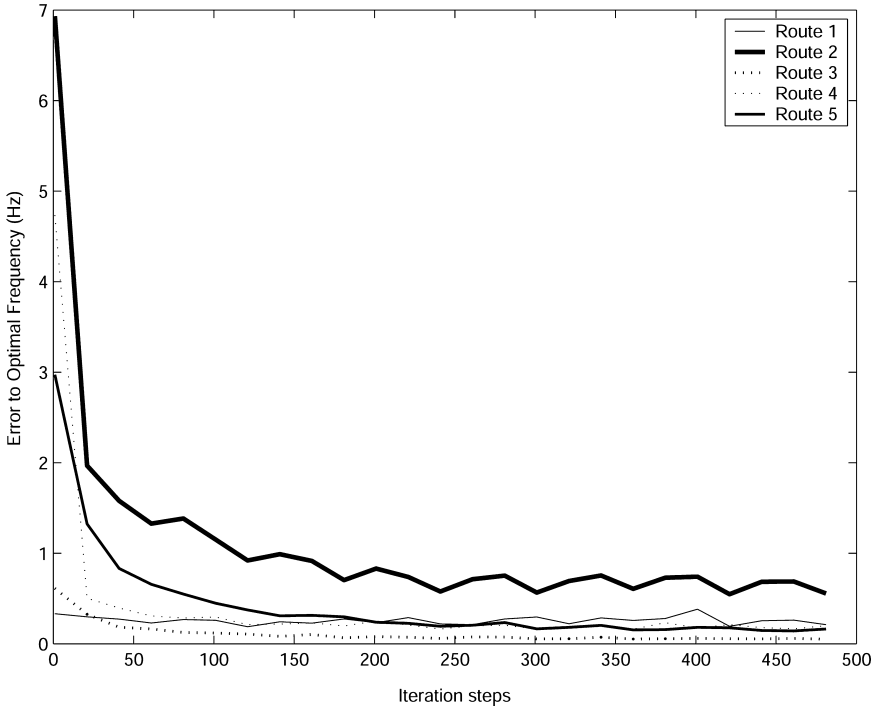


Fig. 6. Trace of error between proposed and optimal rates. Empirically, in 100 iterations, the proposed rates converge to a satisfactory range around the optimum.

Rate error of route j at the sth iteration of the distributed algorithm is defined as: $e\_f_j(s) = |f_j(s) - f_j^*|$, where $f_j(s)$ is the rate proposal for route j at the sth iteration; f_j^* is the optimal sampling rate for this route.

We still use the testbed depicted by Example 1. But in each trial of the Monte Carlo, a different ULI function for each route is picked by setting the coefficients of ω_j, α_j, and β_j randomly. Then the distributed algorithm is carried out. The rate error for iteration $s = 1, \ldots, 500$ is traced. Eight hundred trials are run. For each route, the eight hundred rate error traces are averaged and plotted in Figure 6.

According to Figure 6, the rate error converges in a negative exponential form. Empirically, in 100 iterations the distributed algorithm reaches a satisfactory quasi-optimum.

It is worth noting that by exploiting the "incremental adjustment" property discussed in Section 6.4, the distributed algorithm can converge even faster.

Table IV. New $MCOP(U', \mathbf{A}', W')$ Parameters

Route	α_j	β_j	ω_j
1	0.33	0.3	4
2	0.22	0.2	3
3	1.32	0.5	2
4	1.98	0.7	1
5	0.66	0.3	6

6.4 Incremental Adjustment Property of the Distributed Algorithm

In real-world conditions, there are often times that the ULI functions and the constraint set change dynamically. These changes transform the original optimization problem $MCOP(U, \mathbf{A}, W)$ into a new optimization problem $MCOP(U', \mathbf{A}', W')$, hence the optimal sampling rate f^* has to be recalculated. If the distributed algorithm is used, new iterations can be carried out from the existing optimum (f^*, p^*), so as to reach the new optimum (f'^*, p'^*) faster. We call this the "incremental adjustment property." An example is given as follows:

Continue with the $MCOP(U, \mathbf{A}, W)$ simulation example in Section 6.2. Suppose at time 15sec, the ULI coefficients switch from the old value set (see Table II) to the new value set depicted in Table IV. Figure 7 and Table V show the comparison between incremental and nonincremental adjustment schemes: the incremental adjustment scheme starts with $MCOP(U, \mathbf{A}, W)$'s optimum sampling rate and its corresponding price vector (f^*, p^*); the nonincremental adjustment scheme starts with a constant tuple (f^0, p^0), where $f^0 = f^{\min}$ and $p^0 = (0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)$. All other settings of the testbed are the same as those for Section 6.2, Under the incremental adjustment scheme, in about 10.30sec, the rate proposal converges to the new optimum f'^*. In contrast, starting from (f^0, p^0), the convergence to the new optimum f'^* is much slower, taking 12.67sec.

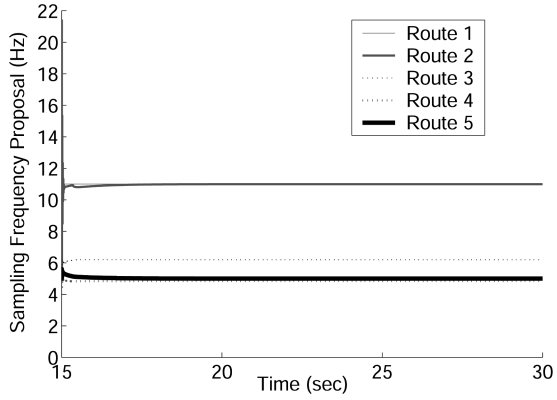
Furthermore, if quasi-optimum is allowed, the new solutions can be derived faster. The convergence time costs are listed in Table V. The incremental adjustment scheme still runs faster than the nonincremental version.

6.5 Control Traffic and Scalability Analysis for the Distributed and Centralized Algorithms

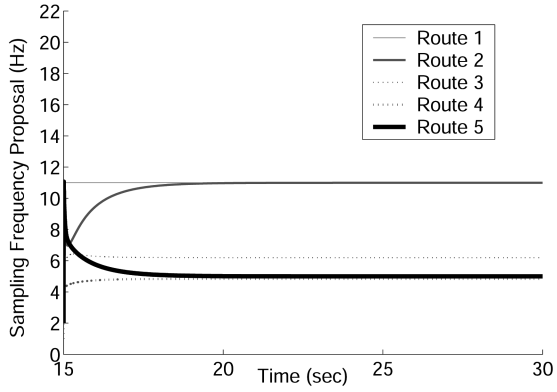
In this section, the control traffic for both distributed and centralized algorithms are analyzed. The centralized algorithm is efficient even when the network is moderately large. However, as the network continues to scale up, the centralized algorithm would finally reach its bottleneck. In contrast, under certain assumptions, the distributed algorithm provides better scalability, though it may be inefficient for smaller networks.

Control Traffic Analysis for the Distributed Algorithm:

Let $\mathcal{N}$ be the set of all base stations in a RICH RTWSN. Let ϕ_i^{dis} be the accumulated control traffic (in bytes) passing through base station $i (i \in \mathcal{N})$ under the distributed algorithm. *Let* Φ^{dis} *be the maximum accumulated control traffic (in bytes) passing through any of the base stations:* $\Phi^{dis} = \max_{i \in \mathcal{N}} \{\phi_i^{dis}\}$.



(a) Based on $MCOP(U, \mathbf{A}, W)$'s solution (f^*, p^*)



(b) Based on constant initial value (f^0, p^0)

Fig. 7. Illustration of incremental adjustment property.

Table V. Convergence Time of $MCOP(U', \mathbf{A}', W')$

Route		1	2	3	4	5
starts with (f^*, p^*)	Convergence Time (CT) (sec)	8.67	7.68	6.98	7.32	10.30
	CT when ±1% error is allowed (sec)	0.01	1.13	0.25	0.07	1.24
	CT when ±5% error is allowed (sec)	0.01	0.05	0.03	0.02	0.15
starts with (f^0, p^0)	Convergence Time (CT) (sec)	11.04	10.05	9.35	9.69	12.67
	CT when ±1% error is allowed (sec)	0.002	3.51	1.25	1.61	3.61
	CT when ±5% error is allowed (sec)	0.002	1.93	0.03	0.39	2.01

Let $\mathcal{R}$ be the set of all routers (RICH base stations that serve as routers) in the same RICH RTWSN. Let d_r be the number of routes passing through router $r (r \in \mathcal{R})$: the out-degree of router r. Let D be the maximum number of routes passing through any routers: $D = \max_{r \in \mathcal{R}}\{d_r\}$.

During each iteration, in Step 1, totally d_r control packets pass through router r. In Step 2, totally d_r packets pass through router r. Without loss of generality, we assume all control packets have 12 bytes of headers. As mentioned previously, each packet's payload length is 4 bytes. Therefore the total

control traffic passing through each router r during each iteration is $32d_r$ bytes, we therefore have the following proposition:

PROPOSITION 6.1. *For any base station $i \in \mathcal{N}$, during each distributed algorithm iteration, the number of control packets-passing through it is no more than $32D$ bytes.*

According to our simulation results in Section 6.3, the distributed algorithm usually converges or reaches a very good approximation in $K \leq 100$ steps (exploiting the incremental adjustment property discussed in Section 6.4, or allowing quasi-optimum, the number of iterations may be even less). Hence after all the iterations, the accumulated control traffic passing through any router is no more than $32KD$. That is, for any node $i \in \mathcal{N}$, $\phi_i^{dis} \leq 32KD \leq 32 \times 100D = 3200D$ (byte). Therefore we have:

$$\Phi^{dis} \leq 3200D \tag{23}$$

$$\Phi^{dis} = O(D). \tag{24}$$

Traffic Load Analysis for Centralized Algorithm:

Let $\mathcal{N}$ be the set of all base stations in a RICH RTWSN. Let ϕ_i^{cen} be the accumulated control traffic passing through base station $i (i \in \mathcal{N})$ under the centralized algorithm. *Let Φ^{cen} be the maximum accumulated control traffic passing through any of the base stations:* $\Phi^{cen} = \max_{i \in \mathcal{N}}\{\phi_i^{cen}\}$.

Suppose the total number of *routes* in a RICH RTWSN is Γ^{total}. Under the centralized algorithm, each route at least needs to send the central computing node $\mathbf{C}$ its ULI information together with at least one constraint. Without loss of generality, we suppose each ULI function is expressed by 3 floating point numbers (12 bytes) and each constraint is at least represented by 2 floating point numbers (8 bytes). To be consistant, we still assume the control packet header has 12 bytes. Thus the accumulated control traffic payload at node $\mathbf{C}$ is $\phi_{\mathbf{C}}^{cen} \geq 32\Gamma^{total}$; $\phi_{\mathbf{C}}^{cen} = \Omega(\Gamma^{total})$. Because $\Phi^{cen} \geq \phi_{\mathbf{C}}^{cen}$, we have:

$$\Phi^{cen} = \Omega(\Gamma^{total}). \tag{25}$$

One may argue that the routes in a RICH RTWSN may not be all directly or indirectly connected, but rather partitioned into several disjoint *maximal subgraphs* (routes within each maximal subgraph are directly or indirectly connected). So that there need not be *ONE* central computing node. Each maximal subgraph can elect its own central computing node, which takes charge of the optimal sampling rate planning for, and only for, the routes within that maximal subgraph. In this case, let $\mathcal{G}$ be the set of all maximal subgraphs, for a specific maximal subgraph $g \in \mathcal{G}$, let Γ_g be the number of routes in g. Let $\mathbf{C}_g$ be the elected central computing node for g. Because of the same reasoning with which we derived (25), we have $\phi_{\mathbf{C}_g}^{cen} \geq 32\Gamma_g$. By the definition of Φ^{cen}, we still have $\Phi^{cen} \geq \phi_{\mathbf{C}_g}^{cen} \geq 32\Gamma_g$. That is: $\forall g \in \mathcal{G}, \Phi^{cen} \geq 32\Gamma_g$, which implies $\Phi^{cen} \geq \max_{g \in G}\{32\Gamma_g\}$. Let $\Gamma = \max_{g \in G}\{\Gamma_g\}$, the maximum number of directly

or indirectly connected routes of the whole RTWSN; then we have:

$$\Phi^{cen} \geq 32\Gamma$$
$$i.e.\ \Phi^{cen} = \Omega(\Gamma). \tag{26}$$

In the following simulation for wide area monitoring, we shall show $\Gamma \approx \Gamma^{total}$, (26) is empirically equivalent to (25). We shall also show Φ^{dis} is empirically insensitive to the scale of the RICH RTWSN while Φ^{cen} increases at least quadratically with the scale of the RICH RTWSN. *This means the centralized algorithm's central computing node is a control message exchanging bottleneck.* Hence the centralized algorithm does not scale up well while the distributed algorithm does.

Comparison of the Distributed and Centralized Algorithms:

In many cases, even for moderately large networks, the centralized algorithm is efficient enough. But there are certain cases where the distributed algorithm is more scalable than the centralized algorithm. An example scenario is as follows:

Suppose all routes are desseminated in a square area of $l \times l$ km^2, where l is the square edge length. The square area deploys a RICH RTWSN cellular division with a hexagon cell edge length of 0.1km. All routes between base stations are unicast, and we assume the network diameter is upper bounded by a fixed constant, which, without loss of generality, is set to 10. Specifically, the source end base stations of all routes are uniformly distributed across the square area with density $\rho = 10/\text{km}^2$; and the destination end is also uniformly distributed within 10 hops from the source end. This also implies the total number of routes is ρl^2. The route is determined by the source/destination end and a simple geographical routing protocol that always forwards the packet closer to the destination in each hop.

For each RICH RTWSN scale ($l = 5, \ldots, 100$(km)), thirty trials are carried out. In each trial, ρl^2 routes are generated according to the previous description; then the maximum number of routes passing through any router (D), and the maximum number of directly or indirectly connected routes (Γ) are counted. The results are shown in Table VI. For ease of comparision, we plot the same data in Figure 8. We can see from Figure 8, *D is bounded by a relatively small constant, insensitive to the network scale l, while Γ rises in a roughly l^2 speed*.[7]

According to (24) and (26), $\Phi^{dis} = O(D)$ and $\Phi^{cen} = \Omega(\Gamma)$. Therefore for the distributed algorithm, the maximum per-base station accumulated control traffic (Φ^{dis}) is upper bounded by D, and D is insensitive to the scale of the network. In contrast, for the centralized algorithm, the maximum per-base station accumulated control traffic (Φ^{cen}) is lower bounded by Γ, and Γ is growing quadratically with the network scale l. The underlying reason is that for the centralized algorithm, the control traffic is bottlenecked at the central computing node, while for the distributed algorithm, the control traffic is evenly the distributed among all the nodes. Hence, the distributed algorithm shows better scalability.

[7]In other scenarios, the sensitivity analysis of D to network scale is left for future research.

Table VI. Scalabillity Comparison

l	Γ			D		
(km)	mean	min	max	mean	min	max
5	242	236	247	7	6	10
10	977	65	988	8	7	10
15	2203	2185	2220	8	7	11
20	3920	3894	3935	8	7	10
25	6126	6100	6145	9	8	11
30	8835	8810	8860	9	8	11
35	12020	11975	12057	9	8	11
40	15713	15681	15760	9	8	11
45	19889	19843	19951	9	8	11
50	24564	24510	24611	10	9	11
55	29726	29675	29766	10	9	12
60	35380	35317	35420	10	9	11
65	41506	41446	41551	10	9	11
70	48161	48103	48208	10	9	11
75	55294	55224	55362	10	9	11
80	62908	62831	62983	10	9	12
85	71029	70943	71106	10	9	11
90	79647	79559	79720	10	9	11
95	88722	88619	88835	10	9	11
100	98315	98214	98432	10	9	13

l(km) is the edgelength of the square area. The route density $\rho = 10/\text{km}^2$.
Γ is the maximum number of directly or indirectly connected routes. The centralized algorithm incurs a time cost $\Phi^{cen} = \Omega(\Gamma)$.
D is the maximum number of routes passing through a router (base station). The distributed algorithm incurs a time cost $\Phi^{dis} = O(D)$.
Thirty trials are carried out for each l. The data shows the distributed algorithm has better scalability than the centralized algorithm.

According to Section 6.2, the centralized algorithm works efficiently when the network scale is small or even moderately large. For the distributed algorithm, from the above analysis, because of better scalability, when the network continues scaling up, there will be a point where the distributed algorithm starts to outperform the centralized algorithm. It is useful to set up a threshold to determine when to switch from the centralized to the distributed algorithm. According to Figure 8 and Table VI, D is always less than 15; whereas Γ is never less than 1500 when $l \geq 15$km. According to (23), $\Phi^{dis} \leq 3200D = 48000$ (byte). According to (26), $\Phi^{cen} \geq 32\Gamma \geq 32 \times 1500 = 48000$ (byte), when $l \geq 15$km. That is, when l is bigger than 15km, there is $\Phi^{dis} \leq \Phi^{cen}$, which means the distributed algorithm is more desirable.

In the end, it's worth mentioning that there is a possible "live lock" problem for the distributed algorithm. In the distributed algorithm, each route needs to communicate with all its routers to exchange the updated constraint price and rate proposal. Ideally, control messages are sent in the background using spare bandwidth. We call this the "best-effort" approach since the bandwidth assigned for background traffic is not guaranteed. According to Theorem 3.1, when the optimal rate $f^* = (f_1^*, \ldots, f_N^*)^\intercal$ is reached, at least one of the constraints will reach equality. This means at the router where that constraint is created, all the bandwidth is used up by the data traffic, and *no control messages can be*

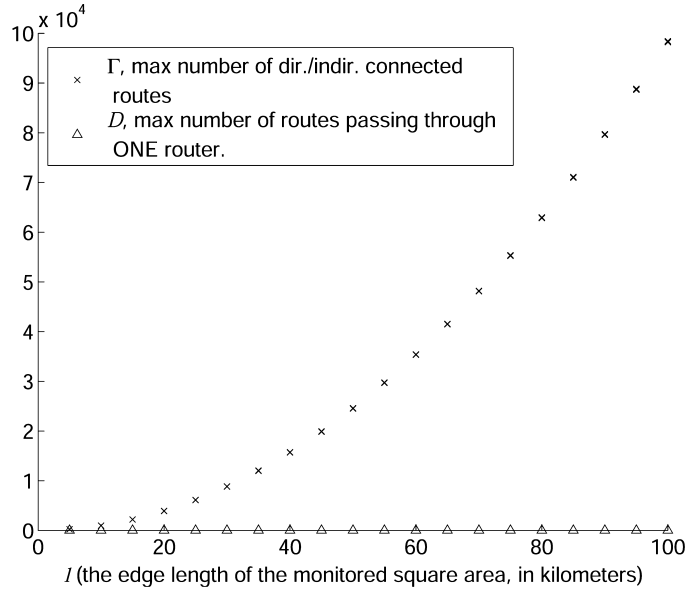


Fig. 8. Scalability Comparison. l(km) is the edgelength of the square area. The route density $\rho = 10/\text{km}^2$. Γ is the maximum number of directly or indirectly connected routes. The centralized algorithm incurs a time cost $\Phi^{cen} = \Omega(\Gamma)$. D is the maximum number of routes passing through any router (base station). The distributed algorithm incurs a time cost $\Phi^{dis} = O(D)$. Thirty trials are carried out for each l. The results shows the distributed algorithm has better scalability than the centralized algorithm.

sent through that router any more! This causes a "live lock" problem since if there is future need for exchanging control messages, the saturated router can no longer participate.

A solution to the "live lock" problem is to preserve a small amount of dedicated bandwidth for exchanging the control messages. According to Proposition 6.1, the maximum amount of control message payload bytes passing through each RICH RTWSN node during each iteration is no more than $32D$, where D is the maximum number of routes passing through any router in the whole RTWSN. Figure 8 and Table VI show when the maximum route length and density of routes in the RTWSN are fixed, and the end points of routes are uniformly the distributed; empirically, D is bounded by a constant, which can be estimated via simulation. Therefore, the bandwidth to be reserved for control message exchange can be planned accordingly. For example, according to Figure 8 and Table VI, D is empirically bounded by 15. Therefore each iteration causes a control traffic of $32 \times 15 = 480$ bytes. If 100 iterations are needed to get the result, and the distributed algorithm is supposed to finish in 4 seconds, then the control traffic bandwidth should be $480 \times 8 \times 100/4 = 96$ (kbps). Note according to Section 6.1, a RICH base station can achieve 1.8Mbps transmitting bandwidth with lightweight hardware. Also note the above bound is based on worst case analysis. In practical applications, if more detailed network information, such as the number of routes passing through each router is available, then different router nodes can reserve different bandwidth based on this information.

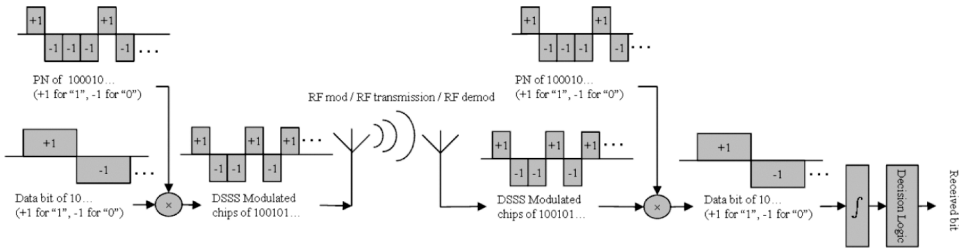


Fig. 9. DSSS modulation/demodulation process: a simplified view.

7. CONCLUSIONS AND FUTURE WORK

In this article, we study the optimal sampling rate assignment in RTWSN, and formalize it into a nonlinear optimization problem. By using the state-of-art methods in optimization, two solutions are given. One is in a the centralized fashion, the other is in a the distributed fashion. Our solutions can handle multi-hop routing scenarios, which are not covered by previous research. We compare the trade-offs between the centralized and the distributed algorithms under different situations. Specifically, we quantitatively analyze the node-wise control traffic under both algorithms. We show that though the centralized algorithm works efficiently with small and even moderately large RTWSNs, it has a bottleneck problem, which limits its scalability. On the other hand, the distributed algorithm is a better choice for large-scale RTWSN and has the desirable incremental adjustment property. Also, the convergence of the distributed algorithm is guaranteed and empirically shown to be fast. Note that using the GPS for synchronization in the distributed algorithm is not an obligation; an asynchronous algorithm can be designed, for example, similar to the asynchronous flow control algorithm in Low and Lapsley [1999]. However, an asynchronous algorithm usually converges, more slowly.

Our ongoing research topics include: (1) Integrating QoS optimization and error modeling for WSN; (2) Theoretical analysis of the distributed algorithm's convergence rate for specific WSN applications; (3) ULI function formulation based on stochastic models.

A. A BRIEF TUTORIAL ON DSSS-CDMA

DSSS is a physical layer baseband modulation/demodulation scheme for digital communication. Without loss of generality, we assume digital "1" and "0" are represented by $+1$ and -1 (volt) rectangular pulses. Unlike conventional baseband modulation schemes, where each bit is represented with a *single* $+1$ or -1 pulse, DSSS multiplies a *Pseudo Noise* (PN) sequence onto the stream of user data bits, as shown in Figure 9.

The PN sequence is also a sequence of ± 1 rectangular pulses, with a $+1$ pulse representing digit "1" and a -1 pulse for digit "0". Each digit of a PN sequence is called a *chip*. The number of PN chips generated per second is called *chip rate*, represented by r_c; chip duration T_c is defined as $T_c \stackrel{def}{=} \frac{1}{r_c}$. Correspondingly, the number of data bits generated per second is called *bit rate*, represented by r_b, and bit duration $T_b \stackrel{def}{=} \frac{1}{r_b}$. Usually r_c is a positive integer multiple of r_b;

the ratio is called *processing gain*, denoted as $g \stackrel{def}{=} \frac{r_c}{r_b}$. According to the DSSS modulation scheme, each modulated bit consists of g chips (see Figure 9). The process of DSSS modulation: multiplying PN sequence chips onto user data bits, is often called *scrambling*.

At the demodulator, if the same PN sequence with 0 phase shift (synchronized, or say, *coherent*) is again multiplied to the scrambled data chip stream, the original user data bit stream (the ± 1 sequence at bit rate r_b) recovers. If the received data stream is scrambled with another PN sequence, or the phase shift is more than a chip, the data stream cannot be recovered; instead, it looks like a stream of random ± 1s generated by independently flipping a fair coin at chip rate r_c. The multiplication carried out at the demodulator is also called *descrambling*. Next, by integrating over T_b units of time, a decision logic can decide whether a wanted user data bit is received or not.

Therefore, a specific PN sequence decides a unique DSSS data transmission *channel*. Different data streams scrambled with different PN sequences are allowed to occupy the same RF spectrum. In the time domain, data streams of different DSSS channels may be sent out in parallel without TDMA (Time Division Multiple Access). The matching PN sequence at the receiver can filter out the wanted user data signal from the shared spectrum. Such a parallel multiple access scheme is called *Code Division Multiple Access* (CDMA). Note DSSS is a physical layer concept, while CDMA is a MAC layer concept. Other MAC layer schemes, such as TDMA can also be deployed on top of the DSSS physical layer.

Quantitatively, a number of important features of DSSS communication are captured by its *Bit Error Rate* (BER) upper bound (27), which assumes QPSK RF modulation, and per-connection pilot tone [Viterbi 1995]; [Muqattash and Krunz 2003] (different implementation alternatives may affect details of the formula, but will not cause fundamental differences):

$$\mathcal{P}_{ber} \leq \exp\left(-\frac{g\,P_u}{J + \sum_{i=1, i\neq u}^{\Xi} P_i + \sum_{h=1}^{H} A_h + P_u}\right), \tag{27}$$

.

where $\mathcal{P}_{ber}$ is the BER; g is processing gain; J is the received power of *External RF Interference* (EI), which specifically refers to EMI, thermal noise and the RF interference from RF devices that are turned on accidentally or maliciously. P_i $(i = 1 \ldots \Xi)$ is the received power of CDMA channel i, Ξ is the total number of received CDMA channels. u is the intended channel, whose corresponding received power is P_u. Each transmitting node may send out several CDMA channels in parallel. To ease the reception, the node may also transmit at additional pilot tone. In (27) the pilot tone of transmitting node h $(h = 1, \ldots, H)$ is of power A_h. $\sum_{i=1, i\neq u}^{\Xi} P_i + \sum_{h=1}^{H} A_h$ is therefore the upper bound of total *Multiple Access Interference* (MAI), the interference caused by other CDMA channels and pilot tones received in parallel with the intended channel. Note P_u also appears in the denominator, adding up to the total interference power. This is to provide a pessimistic estimation on *Inter Symbol Interference* (ISI), which is usually a result of multipath fading. To simplify, we can merge $\sum_{i=1, i\neq u}^{\Xi} P_i$ and P_u together to be denoted as $\sum_i P_i$. The $g\,P_u/(J + \sum_i P_i + \sum_h A_h)$ part

shows the effective SNR for the intended channel, where $J + \sum_i P_i + \sum_h A_h$ represents the upper bound of noise power and $g P_u$ represents effective signal power. The bigger the SNR, the smaller the probability of bit error $\mathcal{P}_{ber}$. When $\mathcal{P}_{ber}$ is below a certain threshold Θ_{ber}, the wireless communication is acceptable for real-time communication. Therefore, to maintain a real-time DSSS-CDMA channel in fact means to maintain the SNR of the channel from dropping below an acceptable threshold Θ_{snr}.

(27) implies that the SNR of the intended channel can be raised by increasing the processing gain g. Meanwhile, g is defined as the ratio of chip rate and bit rate: $g \stackrel{def}{=} r_c/r_b$. Usually, chip rate r_c is fixed by hardware because of the multipath effect and hardware cost constraints [Price and Green 1958]; [Viterbi 1995], therefore raising processing gain means slowing down user data bit rate r_b. DSSS hereby provides a mechanism to leverage between SNR and data bit rate.

B. PROOF OF THEOREM 3.1

Using Lagrangian multipliers $\lambda_j, j = 1, \ldots, N$, and $\mu_i, i = 1, \ldots, M+N$, we can write the Kuhn-Tucker condition of $MCOP(U, \mathbf{A}, W)$ as:

$$\frac{dU_j(f_j^*)}{df_j} + \mu_1 \mathbf{A}'_{1j} + \ldots + \mu_{(M+N)} \mathbf{A}'_{(M+N)j} - \lambda_j = 0, \tag{28}$$

where $(j = 1, \ldots, N)$

$$f_j^{\min} - f_j^* \leq 0 \tag{29}$$

$$\lambda_j \left(f_j^{\min} - f_j^*\right) = 0 \tag{30}$$

$$\sum_{j=1}^{N} \mathbf{A}'_{ij} f_j^* \leq W'_i, (i = 1, \ldots, M+N) \tag{31}$$

$$\mu_i \left(\sum_{j=1}^{N} \mathbf{A}'_{ij} f_j^* - W'_i \right) = 0 \tag{32}$$

$$\mu_i \geq 0, \lambda_j \geq 0, (i = 1, \ldots, M+N, j = 1, \ldots, N) \tag{33}$$

Suppose for all $i = 1, \ldots, M+N$, $\sum_{j=1}^{N} \mathbf{A}'_{ij} f_j^* < W'_i$, then from (32), we know $\mu_i = 0$. Then $\frac{dU_j(f_j^*)}{df_j} + \mu_1 \mathbf{A}'_{1j} + \ldots + \mu_{(M+N)} \mathbf{A}'_{(M+N)j} - \lambda_j = \frac{dU_j(f_j^*)}{df_j} - \lambda_j < 0$, since $\frac{dU_j(f_j^*)}{df_j} < 0$. This contradicts equation (28). So we know Theorem 3.1 holds.

C. CONVERSION OF MCOP TO COPL_LC

The original COPL_LC package is used to solve the problem of equality constraints as follows:

$$\begin{aligned} \min \ & g \ (x) \\ \text{such that: } & Tx = b, x \geq 0 \\ \text{where } & \mathbf{A} \in \mathbb{R}^{m \times n}, b \in \mathbb{R}^m \end{aligned}$$

This is NOT the form of our problem formulation of MCOP. We have to do some transformations to transform MCOP into the framework of COPL_LC. Here, we use the combined constraints set (17) of MCOP:

$$\min_{(f_1,\ldots,f_N)} \sum_{j=1}^{N} U_j(f_j) \qquad (10)$$

$$\text{such that: } \mathbf{A}'f \le W' \qquad (17)$$

$$\text{where } \mathbf{A}' = \begin{bmatrix} \mathbf{A} \\ \mathbf{I} \end{bmatrix}_{(M+N)\times N}, \quad W' = \begin{bmatrix} W \\ f^{\max} \end{bmatrix}_{(M+N)\times 1}.$$

$$f \ge f^{\min} \qquad (13)$$

First, we let $\tilde{f}_j = f_j - f_j^{\min}$, so that the constraints $f_j \ge f_j^{\min} \Leftrightarrow \tilde{f}_j \ge 0$. We also add a slack variable $y = (y_1, \ldots, y_{M+N})^{\mathsf{T}}$, so that

$$\begin{aligned} \mathbf{A}'f \le W' &\Leftrightarrow \mathbf{A}'f + y = W', y \ge 0 \\ &\Leftrightarrow \mathbf{A}'\tilde{f} + y = W' - \mathbf{A}'f^{\min}, y \ge 0. \end{aligned}$$

MCOP is thereby transformed to the following form:

$$\min_{(\tilde{f}_1,\ldots,\tilde{f}_N)} \sum_{j=1}^{N} U_j(\tilde{f}_j)$$

$$\begin{aligned} \text{such that: } \tilde{f}_j &\ge 0 (j = 1, \ldots, N) \\ y &\ge 0 \\ \sum_{j=1}^{N} \mathbf{A}'_{ij}\tilde{f}_j + y_i &= W'_i - \sum_{j=1}^{N} \mathbf{A}'_{ij} f_j^{\min}, \\ (\text{where } i &= 1, \ldots, M+N). \end{aligned}$$

This is in the form of COPL_LC. To see this, simply let $x = [\tilde{f}_1, \ldots, \tilde{f}_N, y_1, \ldots, y_{M+N}]^{\mathsf{T}}$, $T = [\mathbf{A}'_{(M+N)\times N} | \mathbf{I}_{(M+N)\times(M+N)}]$, where $\mathbf{I}_{(M+N)\times(M+N)}$ is the $(M+N)\times(M+N)$ identity matrix and $b = W' - \mathbf{A}'f^{\min}$.

D. PROOF OF THEOREM 5.1

First by defining $I_j = [f_j^{\min}, f_j^{\max}]$ and $V(\bullet) = -U(\bullet)$, we can rewrite the $MCOP(U, \mathbf{A}, W)$ as:

$$\textbf{Primary: } \max_{f_j \in I_j} \sum_{j=1}^{N} V_j(f_j) \qquad (34)$$

$$\text{such that: } \quad \sum_{j=1}^{N} \mathbf{A}_{ij} f_j \le W_i \quad (i = 1, \ldots, M). \qquad (35)$$

We call this constraint optimization problem the Primary problem. The following proof follows Low and Lapsley's result in Low and Lapsley [1999] with modifications for our problem.

Let's first convert the Primary problem to its dual form.
Define the Lagrangian with multipliers vector p for the Primary problem as:

$$\begin{aligned} L(f,p) &= \sum_{j=1}^{N} V_j(f_j) - \sum_{i=1}^{M} p_i \left(\sum_{j=1}^{N} \mathbf{A}_{ij} f_j - W_i \right) \\ &= \sum_{j=1}^{N} \left(V_j(f_j) - f_j \sum_{i=1}^{M} \mathbf{A}_{ij} p_i \right) + \sum_{i=1}^{M} p_i W_i. \end{aligned}$$

Notice the first term is separable in f_j, and hence:

$$\max_{f_j \in I_j} \sum_{j=1}^{N} \left(V_j(f_j) - f_j \sum_{i=1}^{M} \mathbf{A}_{ij} p_i \right) = \sum_{j=1}^{N} \max_{f_j \in I_j} \left(V_j(f_j) - f_j \sum_{i=1}^{M} \mathbf{A}_{ij} p_i \right). \quad (36)$$

Then, by defining the objective function:

$$\begin{aligned} D(p) &= \max_{f_j \in I_j} L(f,p) \\ &= \sum_{j=1}^{N} B_j(p^j) + \sum_{i=1}^{M} p_i W_i, \end{aligned}$$

where

$$B_j(p^j) = \max_{f_j \in I_j} (V_j(f_j) - f_j p^j), \quad (37)$$

$$p^j = \sum_{i=1}^{M} A_{ij} p_i. \quad (38)$$

The dual problem is:

$$\textbf{Dual: } \min_{p \geq 0} D(p). \quad (39)$$

We call the unique optimizer of (37), $f_j(p^j)$. From the Kuhn-Tucker theorem, it is easy to see:

$$f_j(p^j) = \left[V'^{-1}_j(p^j) \right]_{f_j^{\min}}^{f_j^{\max}}, \quad (40)$$

where $[z]_a^b = \min\{\max\{z,a\},b\}$, $V'^{-1}_j(\bullet)$ represents the inverse of derivative function $V'_j(\bullet)$ (with respect to f_j).
The three Lemmas given below are used in the proof of Theorem 5.1.

LEMMA D.1. *Under assumptions A1 and A2, the dual objective function $D(p)$ is convex, lower bounded, and continuously differentiable.*

PROOF. Directly follow assumption A1, A2. □

For any price vector p, define $\beta_j(p)$ by

$$\beta_j(p) = \begin{cases} \frac{1}{-V_j''(f_j(p))} & \text{if } V_j'(f_j^{\max}) \le p^j \le V_j'(f_j^{\min}) \\ 0 & \text{otherwise.} \end{cases} \tag{41}$$

Here, $f_j(p)$ is the unique maximizer of (37), which is defined in equation (40). Let $\mathbf{B}(p) = Diag(\beta_j(p)), (j = 1, \ldots, N)$ be the $N \times N$ diagonal matrix. Note from assumption **A3**, we know for $\forall p \ge 0$, $\frac{1}{-V_j''(f_j(p))} = \frac{1}{U_j''(f_j(p))} \le \overline{\alpha_j}$, so

$$0 \le \beta_j(p) \le \overline{\alpha_j} \le \infty. \tag{42}$$

LEMMA D.2. *Under assumption A1, A2, the Hessian matrix of D is given by* $\nabla^2 D(p) = \mathbf{A}\mathbf{B}(p)\mathbf{A}^{\mathsf{T}}$*, where it exists.*

PROOF. Let $\frac{\partial f}{\partial p}(p)$ denote the $N \times M$ Jacobian matrix whose (j, i) element is $\frac{\partial f_j}{\partial p_i}(p)$. When it exists,

$$\frac{\partial f_j}{\partial p_i}(p) = \begin{cases} \frac{\mathbf{A}_{ij}}{V_j''(f_j(p))} & \text{if } V_j'(f_j^{\max}) \le p^j \le V_j'(f_j^{\min}) \\ 0 & \text{otherwise} \end{cases}.$$

using (42), we have:

$$\left[\frac{\partial f_j}{\partial p_i}\right] = -\mathbf{B}(p)\mathbf{A}^{\mathsf{T}}. \tag{43}$$

We know $\frac{\partial D}{\partial p_i}(p) = W_i - \sum_{j=1}^{N} \mathbf{A}_{ij} f_j(p)$, *i.e.* $\nabla D(p) = W - \mathbf{A} f(p)$, hence

$$\nabla^2 D(p) = -\mathbf{A}\left[\frac{\partial f}{\partial p}(p)\right]; \tag{44}$$

substituting equation (43) into (44) yields the results. □

LEMMA D.3. *Under conditions A1 ∼ A3,* ∇D *is Lipschitz with* $\|\nabla D(q) - \nabla D(p)\|_2 \le \overline{\alpha}\overline{LS}$*, for* $\forall p, q \ge 0$.

PROOF. Using Lemma D.2, we will show that $\nabla^2 D(p) = \|\mathbf{A}\mathbf{B}(\omega)\mathbf{A}^{\mathsf{T}}\|_2 \le \overline{\alpha}\overline{LS}$. The lemma then follows from Rudin [1976]. We know:

$$\|\mathbf{A}\mathbf{B}(\omega)\mathbf{A}^{\mathsf{T}}\|_2^2 \le \|\mathbf{A}\mathbf{B}(\omega)\mathbf{A}^{\mathsf{T}}\|_\infty \|\mathbf{A}\mathbf{B}(\omega)\mathbf{A}^{\mathsf{T}}\|_1;$$

$\|\mathbf{A}\mathbf{B}(\omega)\mathbf{A}^{\mathsf{T}}\|_2$ is upper bounded by the product of the maximum row sum and the maximum column sum of the $M \times M$ matrix $\mathbf{A}\mathbf{B}(\omega)\mathbf{A}^{\mathsf{T}}$. Since $\mathbf{A}\mathbf{B}(\omega)\mathbf{A}^{\mathsf{T}}$ is symmetric, $\|\mathbf{A}\mathbf{B}(\omega)\mathbf{A}^{\mathsf{T}}\|_1 = \|\mathbf{A}\mathbf{B}(\omega)\mathbf{A}^{\mathsf{T}}\|_\infty$, and hence:

$$\begin{aligned} &\|\mathbf{A}\mathbf{B}(\omega)\mathbf{A}^{\mathsf{T}}\|_2 \le \|\mathbf{A}\mathbf{B}(\omega)\mathbf{A}^{\mathsf{T}}\|_\infty \\ &= \max_l \sum_{l'} [\mathbf{A}\mathbf{B}(\omega)\mathbf{A}^{\mathsf{T}}]_{ll'} \\ &= \max_l \sum_{l'} \sum_j \beta_j(\omega)\mathbf{A}_{lj}\mathbf{A}_{l'j} \\ &= \max_l \sum_j \beta_j(\omega)\mathbf{A}_{ij}|L(j)|. \end{aligned}$$

By definition $|L(j)| \leq \overline{L}$, $\beta_j(\omega) \leq \overline{\alpha}$, and hence $\|\mathbf{AB}(\omega)\mathbf{A}^{\mathsf{T}}\|_2 \leq \overline{\alpha}\overline{L}\max_l \sum_j |\mathbf{A}_{lj}| \leq \overline{\alpha}\overline{LS}$. □

Proof of Theorem 5.1: The dual objective function D is lower bounded and ∇D is Lipschitz from Lemma D.1 and D.3. Then any accumulation point p^* of the sequence $\{p(s)\}$ generated by the gradient projection algorithm (the distributed algorithm) for the dual problem is dual optima [Bertsekas and Tsitsiklis 1989].

Let $\{p(s)\}, s = 1, 2, \ldots$ be a subsequence converging to p^*. At least one exists since the level set $\{p \geq 0 | D(p) \leq D(p(0))\}$ of D is compact and that the sequence $\{D(p(s))\}$ is decreasing in s and hence in the level set, provided $0 < \gamma < 2/(\overline{\alpha}\overline{LS})$. To show that the subsequence $\{f(s) = f(p(s))\}, s = 1, 2, \ldots$ converges to the primal optimal node rate $f^* = f(p^*)$, note that $V_j'(f_j)$ is defined on a compact set I_j. Moreover it is continuous and one-to-one, and hence its inverse is continuous on $[f_j^{\min}, f_j^{\max}]$ [Rudin 1976]. From (40), $f(p)$ is continuous. Therefore $\lim_{s\to\infty} f(s) = f(p^*)$. Because our Primal problem is the same as $MCOP(U, \mathbf{A}, W)$, so Theorem 5.1 holds.

ACKNOWLEDGMENTS

The authors would like to thank Jiawei Zhang and Professor Yinyu Ye at Stanford University for providing and giving helpful advice on the COPL_LC package. We would also like to thank Professor Steven Low at CalTech for the discussion on the distributed algorithm.

REFERENCES

AKKAYA, K. AND YOUNIS, M. 2005. A survey of routing protocols in wireless sensor networks. *Elsevier Ad Hoc Network Journal 3*, 3.

BERTSEKAS, D. 1995. *Nonlinear Programming*. Athena Scientific, Belmont, MA.

BERTSEKAS, D. AND TSITSIKLIS, J. 1989. *Parallel and the Distributed Computation*. Prentice Hall.

BOLOT, J.-C., TURLETTI, T., AND WAKEMAN, I. 1994. Scalable feedback control for multicast video distribution in the internet. In *SIGCOMM*. 58–67.

BRAGINSKY, D. AND ESTRIN, D. 2002. Rumor routing algorithm for sensor networks. In *International Conference on the Distributed Computing Systems*.

BUTTAZZO, G. C. 1997. *Hard Real-Time Computing Systems: Predictable Scheduling Algorithms and Applications*. Kluwer Academic Publishers.

CACCAMO, M., ZHANG, L., SHA, L., AND BUTTAZZO, G. 2002. An implicit prioritized access protocol for wireless sensor networks. In *Proceedings of IEEE Real-Time Systems Symposium'02*.

DRCL. 2004. Drcl j-sim [online]. Available at: http://www.j-sim.org.

EXIT CONSULTING. 2004. Gps synchronization clock—model 200. Available at: http://www.gpsclock.com/specs.html.

GETTING, I. A. 1993. Perspective/navigation-the global positioning system. *Spectrum, IEEE 30*, 36–38, 43–47.

GHOSH, S., RAJKUMAR, R., HANSEN, J., AND LEHOCZKY, J. 2003. Scalable resource allocation for multiprocessor qos optimization. In *Proceedings of the 23rd IEEE International Conference on the Distributed Computing Systems (ICDCS 2003)*. Providence, RI.

GIANNECCHINI, S., CACCAMO, M., AND SHIH, C. 2004. Collaborative resource allocation in wireless sensor networks. In *IEEE Euromicro Conference on Real-Time Systems*. Catania, Italy.

HE, T., STANKOVIC, J. A., LU, C., AND ABDELZAHER, T. 2003. Speed: A stateless protocol for real-time communication in sensor networks. In *International Conference on the Distributed Computing Systems (ICDCS 2003)*. Providence, RI.

HEINZELMAN, W., KULIK, J., AND BALAKRISHNAN, H. 1999. Adaptive protocols for information dissemination in wireless sensor networks. In *5th ACM/IEEE Mobicom Conference*.

KARP, B. AND KUNG, H. 2000. Greedy perimeter stateless routing for wireless networks. In *Sixth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom 2000)*. Boston, MA.

KELLY, F. 1997. Charging and rate control for elastic traffic. *European Tran. on Telecommunications 8*.

KELLY, F., MAULLOO, A., AND TAN, D. 1998. Rate control in communication networks: shadow prices, proportional fairness and stability. *J. Oper. Res. Soc. 49*, 237–252.

KUROSE, J. F. AND SIMHA, R. 1989. A microeconomic approach to optimal resource allocation in the distributed computer systems. *IEEE Trans. Comput. 38*, 5, 705–717.

LEE, C., LEHOCZKY, J., SIEWIOREK, D., RAJKUMAR, R., AND HANSEN, J. 1999. A scalable solution to the multi-resource qos problem. In *Proceedings of the IEEE Real-Time Systems Symposium*.

LOW, S. H. AND LAPSLEY, D. E. 1999. Optimization flow control, i: Basic algorithm and convergence. *IEEE/ACM Tran. Netw. 7*, 6 (Dec.), 861–875.

LUENBERGER, D. 1984. *Linear and Nonlinear Programming*. Addison-Wesley, Reading, Massachusetts.

MUQATTASH, A. AND KRUNZ, M. 2003. Cdma-based mac protocol for wireless ad hoc networks. In *Proceedings of the 4th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc 2003)*. Annapolis, Maryland, USA, 153–164.

NESTEROV, Y. AND NEMIROVSKY, A. 1994. *Interior Point Polynomial Methods in Convex Programming*. SIAM.

PRICE, R. AND GREEN, JR., P. E. 1958. A communication technique for multipath channels. *Proceedings of the IRE 46*, 555–570.

QUALCOMM. 2004a. Csm2000 cell site modem [online]. Available at: http://www.cdmatech.com/solutions/products/csm2000.jsp.

QUALCOMM. 2004b. Csm5000 cell site modem [online]. Available at: http://www.cdmatech.com/solutions/products/csm5000.jsp.

QUALCOMM. 2004c. Csm5500 cell site modem [online]. Available at: http://www.cdmatech.com/solutions/products/csm5500.jsp.

RAJKUMAR, R., LEE, C., LEHOCZKY, J., AND SIEWIOREK, D. 1997. A resource allocation model for qos management. In *Proceedings of IEEE Real-Time Systems Symposium*.

RUDIN, W. 1976. *Principles of Mathematical Analysis*. McGraw-Hill Inc.

SETO, D., LEHOCZKY, J. P., SHA, L., AND SHIN, K. G. 1996. On task schedulability in real-time control systems. In *Proceedings of IEEE Real-Time Systems Symposium'96*.

SHA, L., LIU, X., CACCAMO, M., AND BUTTAZZO, G. 2000. Online control optimization using load driven scheduling. In *Proceedings of the 39th IEEE Conference on Decision and Control*.

STOICA, I., ABDEL-WAHAB, H., JEFFAY, K., BARUAH, S., GEHRKE, J., AND PLAXTON, C. G. 1996. A Proportional Share Resource Allocation Algorithm for Real-Time, Time-Shared Systems. In *IEEE Real-Time Systems Symposium*.

VITERBI, A. J. 1995. *CDMA: Principles of Spread Spectrum Communication*. Prentice Hall.

WALDSPURGER, C. A. AND WEIHL, W. E. 1994. Lottery scheduling: Flexible proportional-share resource management. In *Operating Systems Design and Implementation*. 1–11.

XU, Y., HEIDEMANN, J., AND ESTRIN, D. 2001. Geography-informed energy conservation for ad hoc routing. In *International Conference on Mobile Computing and Networking*. Rome, Italy.

YE, Y. 1997a. *Interior Point Algorithms: Theory and Analysis*. Wiley.

YE, Y. 1997b. User's guide of *copl∠c*, computational optimization program library: Linearly constrained convex programming. Available at: http://www.stanford.edu/~yyye/Col.html.

ZHAO, J. AND GOVINDAN, R. 2003. Understanding packet delivery performance in dense wireless sensor networks. In *First international conference on Embedded networked sensor systems*. LA, CA.

ZHOU, G., HE, T., KRISHNAMURTHY, S., AND STANKOVIC, J. A. 2004. Impact of radio irregularity on wireless sensor networks. In *Proceedings of the 2nd international conference on Mobile systems, applications, and services (MobiSys '04)*. ACM Press, 125–138. Boston, MA, USA.

Received February 2005; revised August 2005; accepted January 2006