行政院國家科學委員會專題研究計畫 成果報告

iCare:社群化智慧型居家照護--子計畫二：感測網路於居家照護之應用(3/3)

研究成果報告(完整版)

計畫類別：整合型
計畫編號：NSC 95-2218-E-002-024-
執行期間：95年10月01日至96年09月30日
執行單位：國立臺灣大學電機工程學系暨研究所

計畫主持人：黃寶儀

計畫參與人員：碩士級-專任助理：吳瑞傑
博士班研究生-兼任助理：劉承榮、蕭俊杰
碩士班研究生-兼任助理：吳意曦、林宗翰,、陳星豪

報告附件：出席國際會議研究心得報告及發表論文

處理方式：本計畫可公開查詢

中華民國 97年02月22日

行政院國家科學委員會專題研究計畫 結案報告
感測網路於居家照護之應用 (3/3)
Sensor Network for Home Care (3/3)
計畫編號：NSC 95-2218-E-002 -024
執行期間：95 年 10 月 1 日 至 96 年 9 月 31 日 (第三年)
執行單位：國立台灣大學電機工程學系暨研究所
計畫主持人：黃寶儀
計畫參與人：吳瑞傑、劉承榮、蕭俊杰、陳星豪、吳意曦、林宗翰

一、中英文摘要

感測器(sensor)與無線感測網路(sensor network) 的研究及技術逐漸成熟，使其在智慧型系統的應用上受到廣泛的注目。智慧型系統可以利用感測網路收集到的環境參數，達成使用者與系統間輕鬆且聰明的互動。未來幾年，無線感測網路的發展將大幅推進至諸如軍事、工業、生態環境研究等方面的應用。因應現今社會人口老化，慢性疾病人口增加的趨勢，本計畫特別著眼於無線感測網路在智慧型居家照護(intelligent home care) 上的應用。在這個應用領域，無線感測網路扮演了收集老人居家生活情形中各種參數的重要角色。

有別於目前的技術，無線感測網路用於居家照護的設計上有三個特點：固定與行動節點並存、資料種類的多樣、及感測資料傳達的即時性。在設計創新及技術轉移可行性並重的前提下，我們提出了（一）adaptive diffusion 、（二）fast content-based forwarding 、及（三）two-class service differentiation 來因應前列三項特性的需要。Adaptive diffusion 是一個以資料為中心的調整型路由機制。Fast content-based forwarding 可望解決比對資料內容的速度問題。Service differentiation 則增加即時資料取得的可靠與容錯性。我們的目標是在三年內進行整體計劃的評估、實作、與整合，並期領先全球完成無線感測網路於智慧型居家照護的應用及實作。研究成果將奠定台灣在無線感測網路於一般消費市場研究與應用的全球策略性地位，並開創System on Chip (SOC) 在，以感測器與感測網路為基礎的智慧型應用上，一個全新的發展空間。

Recent technology and research advances have helped paving the way to the pervasive deployment of sensors and sensor networks everywhere. These give rise to a new generation of intelligent applications. Sensor data about the target of observation or the environment are essentially the insights to enable effortless interactions between the users and applications. The emergence of sensor network technology would impact a broad variety of applications from national security to infrastructure monitoring. In this project, we seek the applications of sensor networks in the domain of consumer electronics, and in particular intelligent home care. In that, the sensor networks play the critical role of collecting sensor data that indicate the well being of the elders living in place.

The design of sensor network for home care is unique in three aspects – co-existence of static and mobile sensors, variety of data types, and mission-criticalness of data for elder care. Emphasizing both the engineering innovation and practical business solution, we propose 1) adaptive diffusion, 2) fast content-based forwarding, and 3)

two-class service differentiation that address the above three challenges. The adaptive diffusion is a data-centric path finding (routing) mechanism, which prefers the selection of stable and energy-abundant routes. The fast forwarding algorithm handles, in low space and time complexity, the content-based table lookup problem. The service differentiation enables mission-critical data to be sent in high priority and redundancy. Our objectives in this project are to systematically evaluate, validate, implement, and integrate the sensor network to the overall home care system – iCare.

We expect, as a result, to lead the world in the research and development of sensor network based intelligent home care system. The expertise built up by the project will put Taiwan at a vintage point in the R&D of sensor networks in consumer electronics. This will also pave a brand new avenue for System on Chip (SOC) design and inspire a new generation of consumer demands for ubiquitous intelligent applications.

二、計畫目標與規劃

Our goal in this sub-project is to build a sensor network framework that meets the above challenges. That is to say the communication protocol should be:

1. configuration free
2. discriminative of urgent and non-urgent data
3. adaptive to the system residual energy on the mobile wireless nodes
4. fast in heterogeneous sensor data forwarding
5. efficient in computation, memory, and bandwidth use.

The communication system architecture is depicted in Figure 1. It contains a data-centric core and an extendible and composable application-dependent service level. With the building blocks supplied by the core and service level composability, we achieve in discriminating data by providing different services for application needs.

The **data-centric core** is responsible for the fundamental tasks of routing and forwarding. The routing component provides a broadcast service and a many-many routing service that is adaptive to the residual system power level of mobile wireless sensor nodes. The forwarding component provides a 2-level priority forwarding service.

The **application-dependent service level** could contain services composed by the primitives provided by the communication core. In the context of home care, the two data dissemination services defined are 1) broadcast and priority forwarding for the urgent data and 2) adaptive many-cast and regular forwarding for the non-urgent data. The urgent data in the home care system are disseminated aggressively --broadcast in high priority. This would avoid congestion, reduce delay, and prevent from data loss to the best quality the network can sustain.

The adaptive data-centric routing and fast data forwarding are the **key research components**. The adaptive data-centric routing protocol promises to avoid transiting data through more energy limited mobile wireless nodes and therefore prevent from the inconvenience caused by the repeated changes of batteries of the sensor nodes. The fast data forwarding mechanism is essentially a string matching problem. The efficiency will impact greatly the delay and loss rate the sensor network will experience. Our premise is to adopt a fast string matching algorithm for sensor data forwarding to ensure the quality of the data dissemination.

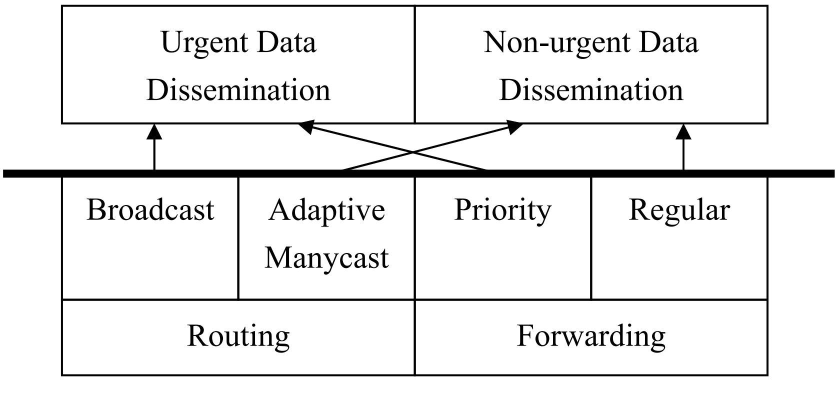


Figure 1. The communication system architecture for home care sensor networks.
Application-dependent services: urgent and non-urgent Data Dissemination
Data-centric core: broadcast/adaptive manycast; regular and priority forwarding

The task breakdown and project timeline is illustrated in Table 1. What highlighted are **tasks completed so far**. We have completed what have been planned for the first two years. In particular, for the service differentiation parts, we are ahead of the schedule.

Timeline	Tasks Work Flow	Adaptive Diffusion	Fast Forwarding	Service Differentiation
1st Year	Lit. Review	Data-centric delivery	String search algorithm	Sensor network QoS
	Design	Bandwidth usage and delay analysis Protocol Specification	Complexity analysis Pseudo code	Data prioritization for desired QoS Mechanism
2nd Year	Evaluation	ns-2-based simulation	Implementation on Linux-based embedded systems	ns-2-based simulation
	Validation	Implementation and APIs on sensor notes	Implementation and APIs on sensor nodes	Implementation and APIs on networked embedded Linux devices
3rd Year	Deployment	Integration and porting of the adaptive diffusion, fast forwarding, and service differentiation mechanisms to the actual devices		

Table 1. Task Breakdown

三、成果自評與展望

On results of magnetic diffusion [1] for data-centric routing is published in a selective international conference with the acceptance rate is 21.8%, the 8th ACM/IEEE International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM 2005). Subsequently, we submit the enhanced version [2] with more detailed results to Computer Communication (SCI) and the journal version is published August 2006. Our work on fast forwarding algorithm comparison is completed. The design, experimental results, and discussion are detailed in a master thesis [3]. A shorter version of the thesis is also submitted for publication [4]. Testbed establishment and evaluation on service differentiation for timely reliable data delivery is presented partly in [5] and detailed in a master thesis [6]. A concise version is under preparation for journal publication. The mechanisms and evaluations are detailed in the subsequent 分析與討論 sections of this report.

Other sensor network research on indoor localization has also received promising results. The idea of accelerometer-assisted energy-efficient location sampling technique is well received. An application of the idea on boundary detection [7] is published in IEEE PerCom 2006 and the extended version [8] is under revision to IEEE Transasction on Mobile Computing. Another application of the idea on our Zigbee-based indoor localization system [9] is published in IEEE SECON 2006. The extended version [10] of the work is invited and subsequently to appear in Ad Hoc Networks in 2008.

Our research activities have been noted by the international community. The project PI, Prof. Polly Huang, is invited to serve on the technical program committee of ACM SenSys 2007and HotEmNets 2008. ACM SenSys with a 10ish% is the flagship conference in the sensor network area and HotEmNets is the venue where the community shares the latest ideas.

On top of the research activities, our sensor network testbed, deployed in the Barry-Lam Hall of EECS NTU, is one of the largest and running testbed in Taiwan. The sensor networking mechanisms developed by this project and the indoor localization work are integrated and working reliably on the testbed. The project PI has hosted more than 20 visits and 100 visitors from the industry, academia, NSC, as well as NSF representatives from the US. The demo videos are available at: http://nslab.ee.ntu.edu.tw/demo/{BL-elevator.avi, BL-uhealth.wmv}

Looking forward to continue our work, we propose to investigate in depth the properties of data transmission over the physical sensor network testbed in office buildings. We observe in prior work that, even with routing and forwarding mechanisms designed for high quality of service (QoS), the delivery ratio of the sensor data varies depending on the (1) structure of the building, (2) building material, (3) human traffic, (4) the use of conflicting radio devices in the building, (5) weather, and (6) subtle differences in the sensor node hardware. These factors may impact the spatial and temporal stability of the sensor data delivery drastically. Given the advances in architectural and interior design, as well as very mobile and ubiquitous wireless communication lifestyle, we think that one of the key issues towards wide-spread use of sensor networks is the **scalability and robustness of sensor network communication in the presence of heterogeneous indoor environment**.

Knowing that the quality of sensor data delivery is not an issue that can be solved solely by a well-designed communication protocol suite, our aim in the follow-up project is to investigate in depth the **environmental**, **protocol design**, and **deployment** factors to the quality of data dissemination in sensor networks. Our premise is to develop, in a three year timeframe, a practical indoor sensor network deployment scheme, which may include a set of assistive software/hardware tools. The deployment scheme will enable easy deployment of quasi-reliable sensor networks in heterogeneous indoor environment. Our ultimate goal is to complete progressively a series of study towards effective sensor network deployment scheme for heterogeneous indoor environment: (1) to come to a fundamental understanding of the link quality distribution, (2) to perform a network- and MAC-layer protocol co-analysis, and (3) to design an effective sensor network deployment mechanism.

[1] Hsing-Jung Huang; Ting-Hao Chang; Shu-Yu Hu; Polly Huang, **Magnetic Diffusion: Disseminating Mission-Critical Data for Dynamic Sensor Networks,** *In the proceedings of the 8th ACM/IEEE International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM 2005)*, Montreal, Qc. Canada, October 10-13, 2005

[2] Hsing-Jung Huang; Ting-Hao Chang; Shu-Yu Hu; Polly Huang, **Magnetic Diffusion: Scalability, Reliability, and QoS of Data Dissemination Mechanisms for Wireless Sensor Networks**, *Computer Communications*, Vol. 29, No. 13,, pp. 2482-2493, Aug. 2006

[3] Jui-Chieh Wu, **Suffix Tree for Fast Sensor Data Forwarding**, Master Thesis, National Taiwan University, June 2006.

[4] Jui-Chieh Wu, Hsueh-I Lu, Polly Huang, **Suffix Tree for Fast Sensor Data Forwarding**, Under submission.

[5] Seng-Yong Lau, Ting-Hao Chang, Shu-Yu Hu, Hsing-Jung Huang, Lung-de Shyu, Chui-Ming Chiu, Polly Huang, "**Sensor Networks for Everyday Use: The BL-Live Experience**," *IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTC 2006)*, Industrial Program, Taichung Taiwan, Jun. 2006.

[6] Ting-Hao Chang, **Reliable Data Dissemination in Practical Sensor Networks**, Master Thesis, National Taiwan University, June 2006.

[7] Tsung-Han Lin, Polly Huang, Hao-Hua Chu, Hsing-Hau Chen, Ju-Peng Chen, **Enabling Energy-Efficient and Quality Localization Services,** *In the proceedings of the 4th IEEE International Conference on Pervasive Computing and Communications (PerCom 2006), Work in Progress Session,* Pisa Italy, March 2006

[8] Tsung-Han Lin, Chuang-Wen Yu, Polly Huang, Hao-hua Chu, E**nergy-efficient Boundary Detection for RF-Based Localization Systems,** In revision to *IEEE Transaction on Mobile Computing* (November 2006 submitted, September 2007 revision requested, December 2007 revision submitted)

[9] Chuang-wen You, Yi-Chao Chen, Hao-hua Chu, Polly Huang, Ji-Rung Chiang, Seng-Yong Lau, **Sensor-Enhanced Mobility Prediction for Energy-Efficient Localization,** *In the proceedings of the 3rd Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON 2006)*, Reston VA, USA, September 2006

[10] Chuang-wen You, Polly Huang, Hao-hua Chu, Yi-Chao Chen, Ji-Rung Chiang, Seng-Yong Lau, **Impact of Sensor-Enhanced Mobility Prediction for Energy-Efficient Localization,** Elsevier Ad Hoc Networks, Invited Paper, In Press, August 2008

四、分析與討論之一 — Magnetic Diffusion for Disseminating Mission-Critical Data for Sensor Networks

4.1. Introduction

The technological advances have enlightened a future of intelligent and pervasive computing and communication. In that, miniature and robust sensor nodes would be able to generate, pre-process, and communicate metrics about the environment. For instance, auto-sensing of the room temperature allows tuning of the air conditioning to the level just as necessary. Auto-detection of pulse anomalies allows prevention of irreversible damages caused by diseases that are preceded by arrhythmia.

Envisioning a new generation of sensor network applications in healthcare , we seek mechanisms that provide reliable and timely transmissions of mission-critical data. the sensor nodes are likely to run on limited battery power for the ease of deployment. Thus, energy efficiency remains an important design challenge. Much of the related work in sensor network data dissemination either emphasizes on the energy efficient design or the reliable data transfer. We have not yet been able to identify any mechanism comparable to the proposed mechanism, magnetic diffusion (MD) that aims at achieving timely delivery, reliability, and energy efficiency.

MD is a simple data dissemination mechanism that promises all the above properties. The inspiration comes from the magnetic interactions in the nature. Consider the data sink as a magnet and the data as nails. The data will be attracted towards the sink according to the magnetic field just as the metallic nails being attracted towards the magnet. The magnetic field is established by setting up the proper magnetic charges on the sensor nodes within the range of data sink. The strength of the charge is determined by the hop distance to the sink and the level of resource available at the sink. The data will be propagated based on the magnetic field from low to high magnetically charged nodes. This way of disseminating data results in optimal delay multi-path forwarding.

We are able to observe from the simulation results that MD does 1) perform the best in timely delivery of data, 2) achieve high data reliability in the presence of network dynamics, and yet 3) work as energy efficiently as the state of the art mechanisms. We thus conclude that MD is a promising data dissemination solution to the mission-critical applications such as home care and telemedicine.

4.2. Related Works

There is a significant volume of related work in sensor network data dissemination. MD is particularly related to research in energy efficient routing and reliable data delivery. To put our work in context, we compare and contrast the existing work to our mechanism. In general, we are not able to identify any mechanisms comparable to MD that aims at achieving timely delivery, reliability, and energy efficiency. This is primarily due to the limited attention in sensor network design for mission-critical applications.

We choose to compare MD to the mechanisms that we are able to finnd actual implementations for the frequently-used Motes/TinyOS platform [28][29]. Directed diffusion [30][31] and flooding are the mechanisms considered in our evaluation. To facilitate discussions in the later sections, we give an overview of the directed diffusion mechanism, following the summary of existing work.

4.2.1 Energy efficient data dissemination

There are three major classes of energy efficient data dissemination mechanisms, the cluster-based, random-walk, and location-aware mechanisms. The cluster-based approaches [3] [32] [33] evenly distribute energy load among the sensors in the network. The idea is to form clusters for localized data dissemination and send only the aggregated data across the clusters to conserve energy. Random-walk approaches [26] [34] randomly select the next hop targets to disseminate data. These mechanisms are energy efficient in that they avoid the sending of control messages and the maintenance of routing states. Location-aware schemes [35][36] exploit the nodes' geographical location information to minimize the cost of disseminating data. The location information might not be easy to obtain for certain sensor network applications. More closely related to our work are the mechanisms dealing with efficient data dissemination to mobile sinks [37][38]. MD is similar in that our mechanism also aims at energy efficient data dissemination for dynamic networks. MD is yet different for that the mechanism design takes into account mobile non-sink nodes and considers the on-off type of network dynamics.

4.2.2 Reliable data dissemination

There are two major approaches towards better data delivery reliability. The passive approach retransmits data when losses are detected, whereas the active approach aims, instead, at avoiding losses. More specifically in the passive approach category, some proposed mechanisms [39][40] recover data end-to-end, and the others [41][42] retransmit hop-by-hop. In the end-to-end mechanisms, the sink nodes track the status of data delivery and send request messages to other nodes for recovery.

The hop-by-hop mechanisms implement the data recovery between two neighboring nodes. When a data packet is lost, the intermediate sending node retransmits. In the active approach category, several schemes [43][44][45] attempt to improve reliability by avoiding congestions. By avoiding congestions, these schemes indirectly lower the loss rate and improve the overall reliability. In addition, some schemes [46][47][48] avoid losses by selecting less lossy data paths.

Our work is closely related to [49], a multi-path mechanism. This work has suggested that disseminating data over multiple paths improves the reliability. The number of paths selected in the mechanism depends on the priority of the data and the network situation. Our MD is similar to this mechanism in that we also disseminate data over multiple paths for reliability. The primary difference is in our choices of the multiple paths. Opting for energy efficiency and timeliness of data delivery, we scope our choice to all the available shortest paths.

4.2.3 Directed Diffusion

Directed diffusion [1][2] is a data-centric data dissemination protocol for wireless sensor networks. The mechanism achieves energy efficiency by means of selecting empirically good paths and in-network aggregation. There are two dissemination modes in directed diffusion: one-phase pull and two-phase pull.

One-Phase Pull (OPP). In OPP, the sink periodically broadcasts an interest message to each of its neighbors. The interest message specifies the data the sink is interested. The neighboring nodes continue to broadcast the interest message to their own neighbors if the interest message is not a duplicate. This action will be repeated until all nodes have received the interest message. Then, the source with the matching data selects the path of shortest latency according to the interest arrival time, and disseminates data to the sink. The data path is periodically refreshed by interest messages.

Two-Phase Pull (TPP). TPP operates in a similar way with a slight complication. The sink also periodically broadcasts an interest message to its neighbors, and the neighbors operate similarly to those in OPP. The main difference is at the broadcasting of exploratory data from the data source to all nodes in the sensor network. After the sink receives this exploratory data packet, it reinforces a good path back upstream by sending a reinforcement message. This reinforced path is determined according to metrics such as delay, loss rate, and available bandwidth. The node that receives the reinforcement message acts in the same way and forwards the reinforcement message back upstream until the source receives this reinforcement message. After that, the source sends data along the reinforced path.

To adapt to network dynamics, the reinforced path is periodically refreshed by periodic exploratory data. To conserve energy due to the flooding of data packets, the interval of periodic exploratory data is set to be significantly longer than that of the periodic interest messages. Please note following important differences between OPP and TPP that will be referred later while comparing the performance of OPP, TPP and MD:

A. To avoid collisions, TPP implements its own random wait mechanism to prevent synchronized sending of exploratory data. Each node selects a random waiting time when receiving an exploratory data. There is only one data transmission path in OPP. It does not need to schedule a random wait before propagating data.
B. To adapt to failures, OPP refreshes the data path by periodic interest message, whereas TPP refreshes the reinforced path by periodic exploratory data.
C. The frequencies of sending exploratory data and interest messages are different. The frequency of periodic exploratory data is twice as low as that of periodic interest messages.

4. 3. MAGNETIC DIFFUSION

Magnetic diffusion is simple and yet powerful. Consider the data sink as a magnet and the data as metallic nails. The data will be attracted towards the sink according to the magnetic field just as the nails are attracted towards the magnet. The magnetic field is established by setting up the proper magnetic charges on the sensor nodes within the magnetic influence of the data sink. The strength of the charge is determined by both the hop distance to the sink and the level of resource available on the sink. The data will be propagated based on the magnetic field from low-charge to high-charge nodes (Figure 2-a). This way of disseminating data promises the following properties.

Multiple Paths. By the simple principle of data traveling towards the center of attraction, the sensor nodes forward data coming strictly from the nodes with lower charges (Figure 2-b). Forwarding data this way, the paths selected by MD are optimal. Furthermore, there can be multiple next hops to forward the data. Thus, MD sends data in an optimal multi-path fashion.

Load Balancing. When there are multiple magnets in the environment, the multiple magnetic fields will find their balance. The nails will thus be attracted by the magnets nearby. The same principle applies to MD. When there exist multiple sinks, their magnetic influences, decrementing MD charges from the sinks, will strike a balance (Figure 2-c) and attracts the data nearby. One advantage of such property is that whenever a new sink joins or an old sink leaves this system, a new balance will be re-established automatically.

Resource Awareness. In the real world, some data sinks might be more resource-limited than the others. For example, there could be sink nodes supplied by unlimited wall-power, while others run on batteries. By assigning sink charges based on the amount of resource at hand, we can achieve the goal of balancing the use of heterogeneous re- source and potentially prolonging the network lifetime without human intervention.

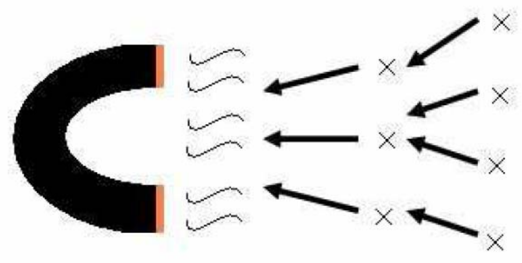


Figure 2-a. Data move toward magnet (i.e. sink).

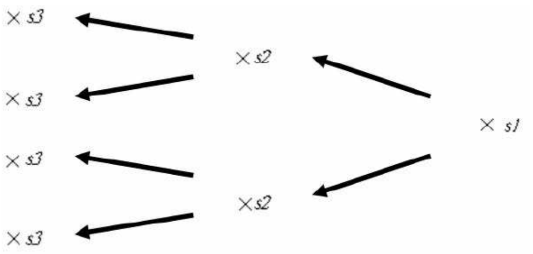


Figure 2-b. Multiple paths: multiple nodes with equal charge, s1 < s2 < s3.

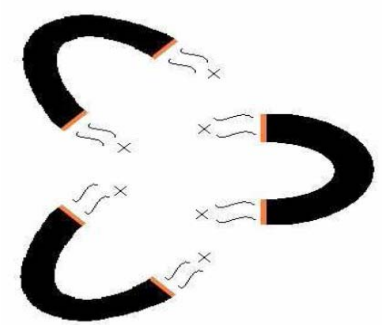


Figure 2-c. Load Balancing: multiple magnets attracting nearby nails.
Figure 2. Illustration of Magnetic Diffusion

In a healthcare example, a patient with chronic diseases may wear biometric sensors for long-term monitoring of the patient's body temperature, blood pressure, and heart rate. These sensor data need to be collected to a remote patient record server for further analysis. When the patient goes outdoors where there are no infrastructural sinks, the sensor data need to be transmitted to a resource-limited personal device such as the patient's PDA. Through a GPRS interface, the sensor data can be transmitted to the patient record server. When the patient comes back home, or wherever there are more resourceful sinks such as a full edged PC with low-cost ADSL connection, one can assign a stronger magnetic charge to the PC. Then the data, aware of the stronger attraction, will go through the PC rather than the PDA(Figure 3). Such resource-aware feature brings not only convenience but also optimized utilization of the resources in the environment.

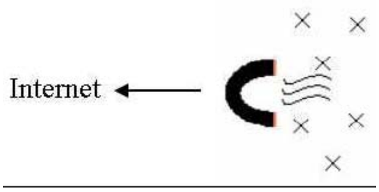


Figure 3-a. Small magnet (resource limited sink, ex:PDA) attract data and transmit via resource limited network (ex:GPRS) to Internet.

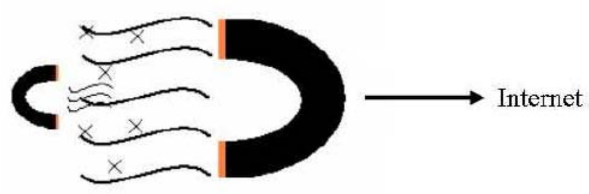


Figure 3-b. Large magnet (more resourceful sink, ex:PC) which overwhelm small magnet, attract data and transmit via wider bandwidth network (ex:ADSL) to Internet
Figure 3. Resource-awareness: larger magnet out-attracting the small magnet

MD operates in two phases: the interest broadcast and data propagation. The magnetic field is established in the interest broadcast phase such that the data can be disseminated towards the sink in the data propagation phase. In the subsections below, we describe the details of the MD operation and the implementation options.

4.3.1 Interest Broadcast

We first introduce two data entities involved in the interest broadcast phase: the interest and the entry. The interest is a message used to establish the magnetic field to facilitate the proper flow of data toward the sink. An interest message consists of two elements, the data type and magnetic charge. The former records what data type the sink wants to collect. The latter is an integer value recording the strength of the magnetic influence from the sink. The entry is where a node records the data type and the magnetic charge in each node.

When a sink wants to collect data, it sends an interest message to its neighbors. When a node receives an interest message for the first time, it creates an entry for this interest. Then the node decrement the magnetic charge in the interest by one. The node records the data type and the magnetic charge into its entry and then forwards the interest message to its neighbors. The decrementing magnetic charges from the sink to source will guide the flow of data in the reverse direction, mimicing the traveral of metallic nails from the low-charge to high-charge points in the magnetic field.

If a node receives an interest message and the corresponding entry exists, it will compare the decremented magnetic charge of the interest with the one in the entry. If the magnetic charge of the interest is greater than the one in the entry, the node will update the value in the entry to that of the interest and forward the interests to its neighbors. Otherwise, the node will discard the interest knowing the interest is not from a node closer to the current sink. The sink node will broadcast interest messages periodically to re-establish proper magnetic charges in case of network dynamics. It provides a robust environment especially for the dynamic network.

In Figure 4(a), the sink node broadcasts the interest to its neighbor nodes. In Figure 4(b), both nodes, A and B receive the interest, create corresponding entries, set the data type and proper magnetic charges in the entries, and then propagate the interests to their neighbors. In Figure 4(c), node C, D, and E receive the interests from node A and B and repeat the actions taken by A and B in Figure 4(b). Each node may receive and discard duplicate interests or interests of weaker magnetic charges.

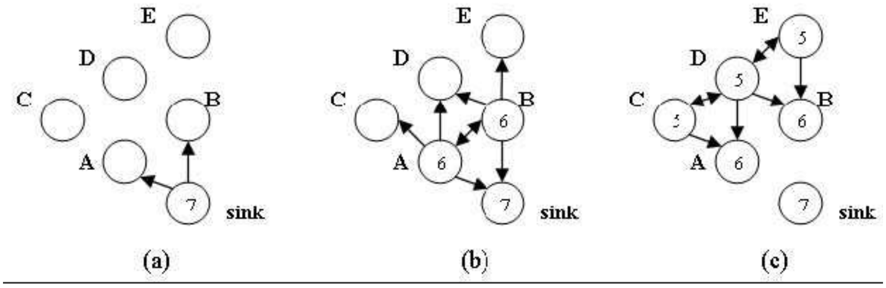


Figure 4. An example of interest propagation.

4.3.2 Data Propagation

We next describe how the data is disseminated through a network. We have two implementation strategies for data propagation. One is gradient-based and the other is broadcast based.

4.3.2.1 Gradient-based

In the interest broadcast phase, when a node receives an interest from the neighboring nodes with a stronger magnetic charge, the node will establish a gradient toward the interest-sending node. This gradient will lead the data through the network from the source to the sink.

When a node senses data, it checks if it has an entry matching the data type. If it does, the node sends the data to the nodes pointed by the existing gradients. The receiving nodes check for the matching entry and then send the data according to the gradients as well. The forwarding process goes on and the data will reach the sink eventually. Figure 5(a) shows the gradients established by the interest broadcast in Figure 4. In Figure 5(b), the source sends data to the two neighbor nodes pointed by the gradients from the source. In Figure 5(c), the receiving nodes continue to send data to their neighbor nodes based on their gradients. The sink finally receives the data.

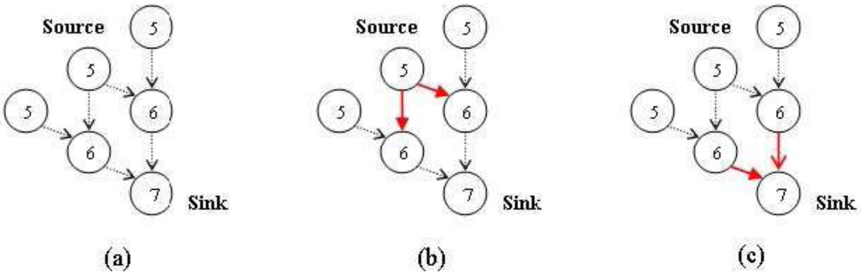


Figure 5. An example of data propagation with gradient-based mechanism.

4.3.2.2 Broadcast-based

In the broadcast-based mode, the nodes do not establish gradients in the interest broadcast phase. Instead, the magnetic charge is included in the data being disseminated. The receiving node can tell from the charge carried in the data where the data is coming from and whether to forward the data further down the stream.

More specifically, when a node receives the data, the node checks if it has any matching entry. If so, it compares the magnetic charge in the entry with the magnetic charge of the data. If the magnetic charge in the entry is greater than that of the data, it sets the magnetic charge of the data to the charge in the entry and then broadcasts the data. This means the data is sent from the node whose magnetic charge is lower than the intermediate node and the intermediate node should continue to broadcast them toward the sink. If a node receives a duplicate data or the data whose magnetic charge is greater than or equal to that in the entry, the data will be discarded.

This dissemination mode is easy to implement and we do not need to maintain the gradients. The node simply compares the magnetic charges to decide whether to broadcast the data. The drawback is every data message will have to carry the magnetic charge of the sending node.

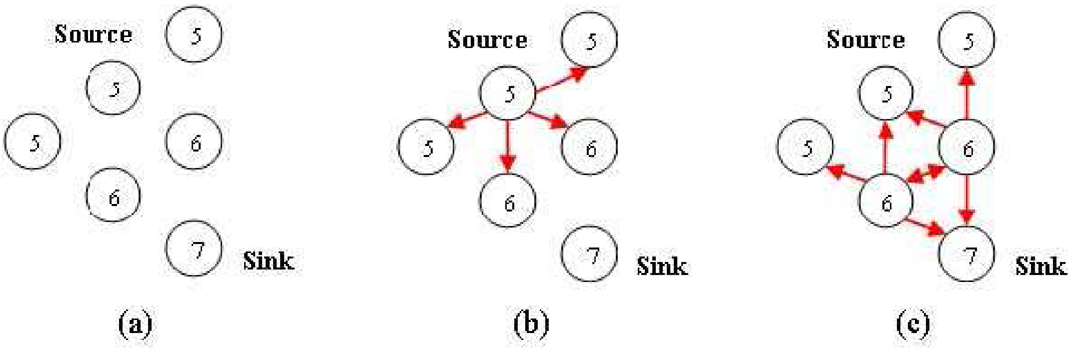


Figure 6. An example of data propagation with broadcast-based mechanism.

In Figure 6(a), the magnetic charge of every node is established with the sink having change strength 7. In Figure 6(b), when the source wants to send data to the sink, it will broadcast the data to its neighbor nodes. In Figure 6(c), the nodes with charge strength 6 broadcast the data because the magnetic charge of the nodes is greater than that of the data. Thus, the sink receives the data.

4.3.3 Random Wait

When the interests or data are being broadcast hop by hop through the network, the messages might collide to each other. To avoid collisions, the node waits a random period of time before sending a message. This technique might decrease the probability of collision and, in the meantime, increase the transmission delay.

4.4. EVALUATION

We have conducted an extensive set of simulations that examines the scalability of MD to the size of the network, the change of the data rate, and the level of the network dynamics. In this evaluation, we are particularly interested in the performance of MD in dynamic networks. Thus, we focus our discussion on the quality of data delivery and the message overhead of MD in the presence of network dynamics.

Nodes	50
Size	160 x 160
Radio range	40 m
Data rate	5 sec
Periodic Interest	30 sec
MAC	802.11

Antenna	OmniAntenna
Simulation time	600 sec

Table 2. Simulation setup

4.4.1 Simulation Setup and Metrics

In this section, we describe the simulation setup and the metrics examined for the performance evaluation. We implemented MD in the ns-2 simulator [50]. In order to get the average behavior, there are 10 distinct runs for each setup. In each run, there are 50 nodes randomly placed in a 160m by 160m square. Each node has a radio range of 40m. We use one source and one sink randomly selected from 50 nodes in our simulation. The source sends data every 5 seconds, and the interests are periodically sent every 30 seconds. The simulation time is 600 seconds, but source will stop sending data after 300 seconds. Table 2 summarizes the parameters of our choice.

We choose three metrics to evaluate the performance of MD and to compare it to other schemes: *Overhead* measures the amount of interest and data packets transmitted. The metric is closely related to the energy consumption of the system in wireless sensor networks when an energy efficient MAC protocol is applied. *Reachability* measures the probability the sink receives the data packet successfully. This metric is important for medical applications in that the sensor data are mission critical and data losses can be life-or-death matters. *Latency* measures the data transmission time from the source to the sink. For medical applications, the metric represents the timeliness and temporal reliability of the data.

The MD implementation used in our experiments is the broadcast mode. There are several advantages of the MD broadcast (MDB) mode over the gradient (MDG) mode. MDB is more energy efficient. The interest packet overhead of the two modes are the same because interests are transmitted in an identical way in both modes. The difference lies in the data packet overhead. In MDG, if a node has five gradients to its neighbors, there are five packets sent, one for each gradient. But in MDB, it broadcasts only one data packet. Therefore, the overall overhead of MDB is much smaller than MDG. Additionally, the data latency is small in MDB. It does not need to send the handshake packets, e.g., the RTS/CTS in IEEE 802.11, before sending a data packet, but MDG does. As a result, MDB behaves better in latency. However, without the handshaking, there are more collisions observed in MDB simulations, and this results in a slight lower reachability in some cases. MDB performs also better in the presence of network dynamics. Given the various advantages of the MDB over MDG, we use MDB for the rest of the comparison.

4.4.2 Impact of Dynamics

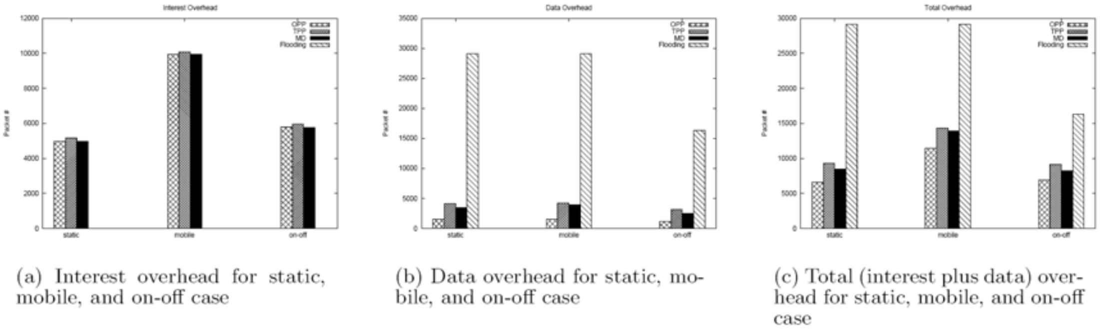
To be realistic, we simulated two kinds of dynamics in our experiments. First of all, we simulated mobile environment in which nodes moved around every 120 seconds. It means that the first movement of each node occurs at 120 seconds. After a node moves to its new position, it waits another 120 seconds to move again. The amount of time that a node takes to move to its new position varies from node to node because of the random new position selection. As a result, the movement of the nodes are not simultaneous.

Second of all, we simulated random node on and off according to the following probabilistic distribution. At the beginning, all nodes are in the on state for a random period between 5 to 65 seconds. Then each node goes into the off state for 25 to 35 seconds, and wake up again for 55 to 65 seconds. All the on or off time durations are uniform randomly selected. and this process will continue until the end of the simulations. These sets of simulations are necessary because the two dynamics are common in reality. The choices of the parameters represent the extreme cases to highlight the distinct properties in MD to the other two mechanisms, DD (See Section 4.2.3.) and flooding, compared.

4.4.2.1 Overhead

Figure 7 shows the amount of overhead for the static, mobile, and on-off cases. Note that in the static case, all nodes remain static and on for the entire duration of the simulations. We show the static case result here to compare to the mobile and on-off cases. In Figure 7(a), the interest packet overhead of TPP is slightly higher than that of OPP and MD in all cases. This is because TPP has to disseminate additional control packets such as the positive and negative reinforcement messages, whereas MD and OPP do not. Note that we set the interval of periodic interest to 30 seconds in all mechanisms. That is the reason that the interest packet overhead of MD and OPP are almost identical.

Figure 7(b) shows that MD has a lower data packet overhead than that of the TPP and a higher data overhead over that of the OPP in all cases. OPP selects only one way to disseminate data, and thus has the least amount of data packet overhead. This result suggests that, independent of the static, mobile, or on-off case, the resulting data packet overhead of TPP's exploratory data is higher than that of the multi-path delivery in MD, and that MD is more energy efficient than TPP. It seems that MD is worse than OPP in terms of data packet overhead. However, the higher data overhead, in return, provides better data reachability and latency which will be discussed next. Figure 7(c) shows that flooding has the largest total packet overhead even though the mechanism does not require any control packets.



(a) Interest overhead for static, mobile, and on-off case

(b) Data overhead for static, mobile, and on-off case

(c) Total (interest plus data) overhead for static, mobile, and on-off case

Figure 7. Overhead.

4.4.2.2 Reachability

Figure 8 shows the reachability results of the static, mobile, and on-off cases. In the static case, OPP and TPP experience an 100% reachability, MD and flooding's reachability is at about 98%. The reason of the difference is at the way the data are propagated. OPP and TPP are gradient-based, whereas MD and flooding are broadcast-based. If data packets collide, the gradient-based mechanisms will retransmit assuming IEEE 802.11 as the underlying MAC protocol. The broadcast-based mechanisms, however, do not detect, nor recover for errors, and thus the slightly lower reachability observed in MD and flooding.

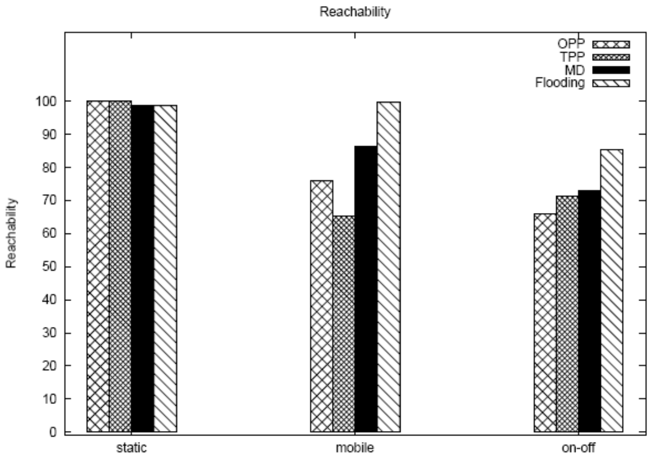


Figure 8. Reachability for static, mobile, and on-off case

In the mobile case, the reachabilities of OPP and TPP drop significantly. MD's reachability is reduced to 85%, which is about 20% higher than TPP (65%) and 10% higher than OPP (75%). This is because MD sends data in multiple paths. When the nodes are mobile, the probability is higher to get the data through the network to the sink with multiple paths.

Note that TPP performs worse than OPP in the mobile case. The difference lies in the path update frequency. OPP reconfigures data path at the time of periodic interest, but TPP does so at the time of periodic exploratory data. The interval of periodic exploratory data is twice as much as the periodic interest. Therefore, TPP results in a lowest reachability in this case. If we shortened the time of periodic exploratory data, there will be a higher reachability, but the data packet overhead will also increase. From this set of results, we can see the advantages of MD over OPP and TPP in terms of reachability in mobile sensor networks. Note that the reachability of flooding is close to 100%. The result suggests that if the data is absolutely critical and energy is not a constraint, flooding is the best choice for reliable data dissemination. Otherwise, MD provides as a reliable solution for environment that energy resource is limited.

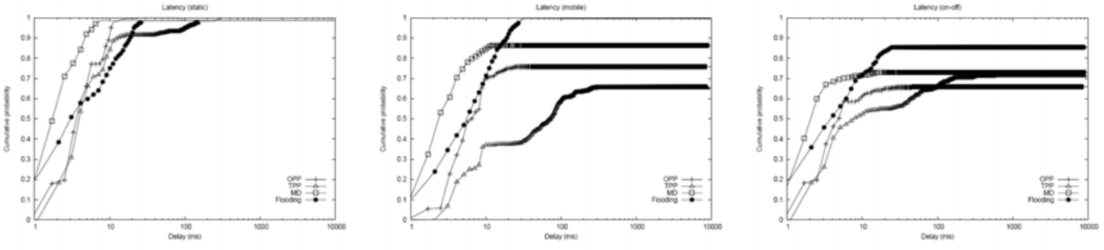
In the on-off case, we observe similar results. But the performance is unsatisfactory., Even the flooding mechanism manages only an 85% reachability. This is because of the potential of broken paths, or even a disconnected network, when certain bottleneck nodes are turned off. Such conditions result in a lower overall reachability. However, we think that a good deployment strategy may compensate for such situations.

We find that OPP performs the worst in the on-off case. In some sense, the on-off type dynamic presents greater challenge to reliable delivery of data and OPP, or single-path mechanisms, is less suitable for networks with extreme dynamics.

4.4.2.3 Latency

Figure 9 shows the cumulative probability of data delivery latency for the static, mobile, and on-off cases. In Figure 9(a), MD has the least latency because it broadcasts data and bypasses the RTS/CST handshake in IEEE 802.11. This is also why OPP performs much worse than MD, even if it always finds the fastest path. Flooding is better than OPP and TPP only in the preceding 60%. We observe an unusual amount of traffic and collisions in the flooding set of simulations, As a result, some data packets get retransmitted repeatedly and routed for a long way to finally reach the sink. This explains why flooding has a lot of data packets with long latency. TPP is very close to OPP in the preceding 90%, but there is a long tail caused by the random wait mechanism in disseminating exploratory data in the later 10%. This plot shows that the broadcast-based mechanisms work better for applications with latency requirements.

In Figure 9(b), MD has the least latency yet. Note that Flooding performs better than OPP completely. This is because OPP has to retransmit a packet many times as a result of high frequency of packet loss and collision in mobile networks. The same reason in OPP and lost data packets result in the poor performance of TPP. Figure 9(c) shows similar result as well. These results indicate that MD is a better solution for applications with requirements of restricted latency in dynamic network.



(a) Cumulative distribution of delay time for static case

(b) Cumulative distribution of delay time for mobile case

(c) Cumulative distribution of delay time for on-off case

Figure9. Latency.

4.4.3 Impact of Random Wait

Figure 10 shows the reachability after adding random wait mechanism to MD and flooding. The figure clearly shows that MD and flooding are improved significantly in all static, mobile, and on-o_ cases. In the static case, MD and flooding are better than the previous result (Figure 8) and the reachabilities are close to 100%. In the mobile case, MD and flooding maintain the same level of reachability. In the on-off scenario, however, advances on MD and flooding can be clearly identified. These results show that random wait mechanism is effective in reducing collisions and thus improve reachability.

Figure 11(a) shows cumulative distribution of delay for static environment. Contrary to the previous result, MD performs slightly worse than OPP and TPP here. This is because the random wait mechanism incurs a longer data delivery latency. It does not have the long tail as in TPP though. As expected, the curve of flooding shifts to the right as what happened to MD. Figure 11(b) shows also the trend that the curves of MD and flooding shift to the right. But MD performs as well as OPP. TPP is the worst in latency. Figure 11(c) shows similar results to Figure 11(a) except the overall amount of data received in MD and flooding simulations is higher.

These results show a trade-off between reachability and latency. The cost of a higher reachability is often the longer latency. One may trade-off a little reachability for much shorter latency. Whether to implement random wait depends on the requirements from the applications. If reachability is more important, turning on random wait will be a good option. If the latency is more critical, it is more appropriate to turn off the random wait mechanism.

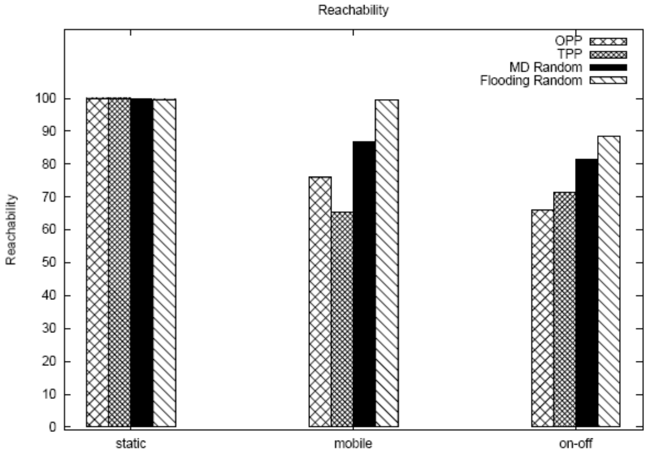
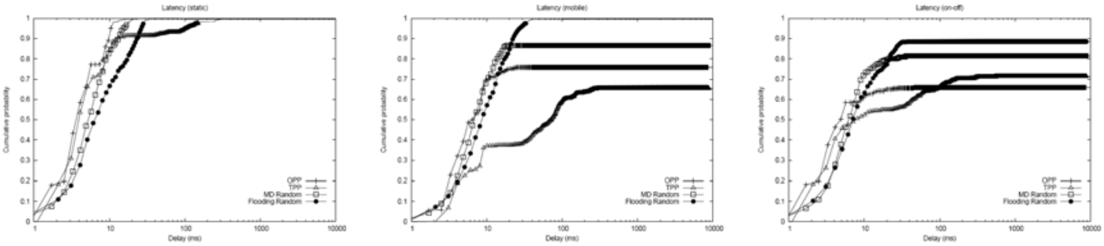


Figure 10. Reachability for static, mobile, and on-off case after adding random wait to MD and Flooding



(a) Cumulative distribution of delay time for static case after adding random wait to MD and Flooding

(b) Cumulative distribution of delay time for mobile case after adding random wait to MD and Flooding

(c) Cumulative distribution of delay time for on-off case after adding random wait to MD and Flooding

Figure 11. Latency.

4.5. CONCLUSION

Inspired by the physics of magnets and nails, we propose magnetic diffusion, a simple and yet efficient dissemination mechanism for mission-critical data. MD is able to identify all the shortest paths available from the data source to sink. By transmitting data over the multiple shortest paths, MD performs well in the timeliness and reliability of data delivery while the overhead and energy consumption of MD is kept low. These properties are confirmed by the simulations. Therefore, we conclude that MD is particularly suitable for sensor network applications in healthcare and workplace safety. For these applications, the timeliness and reliability of data, and the energy efficiency of the system are all required properties.

五、分析與討論之二 — Fast Forwarding: Suffix Trees for Fast Sensor Data

5.1. INTRODUCTION

On the address-centric Internet, communication nodes are numerically addressed, for example, 140.112.42.220. A source node sends data by the destination node's address. Forwarding, also known as the routing table lookup problem, involves how, given the destination address of an incoming data packet, each intermediate router locates a matching entry in the routing table. From the matching entry, the router identifies the network interfaces (or ports) towards the next hops that the data packet should be forwarded further. Efficient algorithms and data structures such as [50] are proposed to speed up the number matching, which lead to the design of very high-speed IP switches today. Motivated to achieve high-speed forwarding for sensor networks, we seek efficient algorithms and data structures to speed up string matching for data-centric sensor networks.

In data-centric wireless sensor networks [51], nodes are no longer addressed. Data do not carry the destination address. Instead, each sink node sends an explicit interest through the network to draw in a particular type of data. The intermediate nodes in turn disseminate the data based on the data content rather than the destination node's address. This data-centric style of communication is particularly promising for that it alleviates the effort of node addressing and address reconfiguration in large-scale mobile sensor networks.

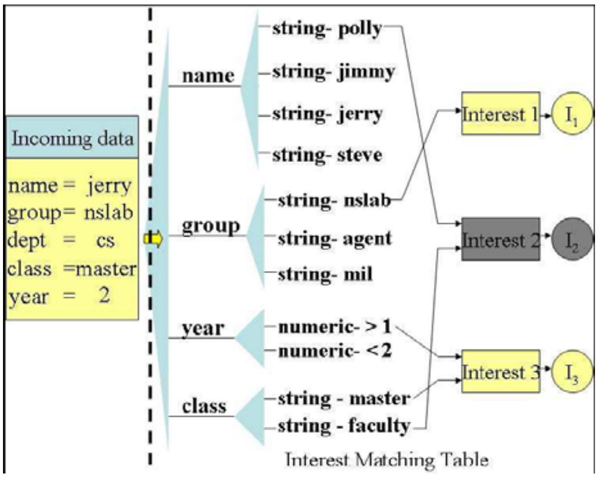


Figure. 12. An example of interest matching.

Forwarding in data-centric sensor networks is particularly challenging. It involves matching of the data content, i.e., string-based attributes and values, instead of numeric addresses. This content-based forwarding problem is well studied in the domain of publish-subscribe systems. It is estimated in [52] that the time it takes to match an incoming data to a particular interest ranges from 10s to 100s milliseconds. This processing delay is several orders of magnitudes higher than the propagation and transmission delay.

Consider the MicaZ [53] and TinyOS [54] sensor network development platform. The packet size limit is 36 bytes. The wireless radio transmits at 100s kbps. The transmission delay is thus at the scale of 1s milliseconds. Assume 10 meter radio range and $2x10^8$ m/s propagation speed The propagation delay can be found at the scale of 0.01s microseconds. The processing delay is evidently the bottleneck of the per hop forwarding delay. The interest lookup delay is contributed by 2 levels of matching - interest and predicate matching. Each interest may consist of multiple predicates. At the higher level, the system needs to identify, among various interests, a particular interest that matches the incoming data. At the lower level, the system verifies whether a predicate in an interest matches the incoming data. Illustrated in Figure 12 are 3 example interests composed by a number of predicates. Interest 1 is looking for anything related to the nslab group. Interest 2 looks for data about a faculty member whose name is polly, and interest 3 looks for data about all 2nd year and above master students. The incoming data in Figure 12 matches both interests 1 and 2.

Much of the recent work [55][56][57][58] focus on the strategies of structuring interests or content types to enable fast interest matching. Their objective is to reduce the number of predicate matching required. Our work complements these earlier studies in that we focus on improving the efficiency of individual predicate matching.

Predicate matching involves attribute matching and value matching. Attribute matching is essentially an exact string matching problem. One common practice to speed up attribute matching is to fix the bit position of all possible attributes in the data packet as well as the routing table. This method, although simplifies the attribute matching process, will be memory and bandwidth consuming when the number of different data types is high. When there are different sensors to be added to the network, the system will not be easily extendable without changing the packet format and interest table data structure. Value matching is also a string matching problem, when the data type is string. Depending on the operator of the predicate, value matching may require exact or sub-string matching. In essence, the efficiency of predicate is determined by the efficiency of the string matching algorithm used.
For efficient string matching, prior work [7] suggests the use of ternary search tree (TST). It is a string matching algorithm with O(|P|+log(N)) time complexity and O(|S|)

space complexity. P denotes the input string, typically an attribute or value string in the incoming data. S denotes the training word set which concatenates all the strings appear in the entire interest table. N denotes the total number of strings in the training set. To speed up the string matching process further, we propose and evaluate the use of suffix tree (ST), a linear time string matching algorithm that can be easily extended to perform efficient prefix, suffix and substring matching. ST has an O(|P|) time and O(|S|) space complexity [59][60].

Although the large-scale performance of TST and ST's memory requirement is the same, we find, in real implementations, the amount of memory required by ST is significantly higher than that of TST. This is a serious problem for sensor nodes in which the memory space is very limited. To tackle the problem, we further propose a scheme to optimize the memory consumption for ST.

To observe how the algorithms will perform in practice, we implement TST and ST, as well as a simple hash-based method for comparison. The experimental results show that the computation time is reduced by 29% to confirm a match and 48% to confirm a non-match at best. The memory usage of ST is improved by 24% with the memory optimization scheme.

Our contribution is three-fold: (1) we identify ST for scalable sensor data forwarding, (2) we propose a novel optimization for ST to reduce the space requirement, and (3) we implement four string matching algorithms and evaluate how well they will perform in practice. The remainder of this thesis is organized as follows. The related work is presented next in Section 5.2. We describe then in Section 5.3 the string matching algorithms. Next in Section 5.4, we provide an analytical comparison of the algorithms. In Sections 5.5, 5.6, and 5.7 we detail the experimental setup, results, and our findings on how the algorithms perform in practice.

5.2. RELATED WORK

5.2.1. Data-Centric Communication

In the traditional IP network, nodes communicate to each other by the fixed IP addresses. This communication model is proven, by the daily operation of Internet, effective in supporting applications running on static and full-fledged computers. For mobile and resource-limited sensor networks, how to configure and reconfigure node addresses in the presence of node dynamics poses a great challenge. This problem is first raised and addressed in one of the pioneer work on sensor networks [51]. In that, the authors propose the data-centric communication paradigm.

In data-centric communication, digital information are disseminated based on the feature/attribute/content of the information itself, not the addresses of issuers or receivers. In the first data-centric routing mechanism for wireless sensor networks [1], sinks send explicit interest packets to set up routing states at the intermediate nodes. These interest specific routing states in turn draw in the data of interest for the sinks. Such dissemination scheme relies on well-defined naming system to describe data attributes and sink interests. The corresponding naming system and the filter-based forwarding mechanism are detailed in [55]. The string matching problem, although recognized as the performance bottleneck, is not addressed.

5.2.2. Publish/Subscribe Systems

In [7][61], the authors design a set of efficient forwarding and routing mechanisms for content-based data dissemination. The notion of content-based communication is essentially the same as data-centric communication. The mechanisms proposed, although descends from the literature of publish/subscribe systems, are applicable to sensor networks. There are two different kinds of publish/subscribe systems: channel-based and content-based. In both systems, multiple users may subscribe to the data of interest. In channel-based systems, the users subscribe to a particular channel and the corresponding data broker pushes particular data to the channel from which the subscribing users receive the data. In content-based systems, the concept of channel is refined as rules and interests, traveling through the intermediate nodes between brokers and subscribers. If the rules belonging to some subscribers are matched by certain data, the data will be forwarded further to the indicated output ports, otherwise the data will be dropped by the node. Much of the improvement in rule matching concentrates on the management of the subscriptions, i.e., organization of predicates and rules. For instance, index algorithms, used also extensively in database management, are adopted by [56] to manage the subscriptions and speed up the matching process.

Other data structures such as button-up selection trees and binary decision diagrams are proposed to improve the matching performance by [57] and [58] respectively. Little work has addressed formally the problem of string matching.

5.2.3. Ad Hoc Publish/Subscribe Systems

The close relationship between publish/subscribe systems and data-centric sensor networks is first formally noted by [62]. The limited energy, computing power, and memory space on a typical wireless sensor node give rise to unique challenges in the design of data forwarding algorithms. Authors of [63] found that the interest diffusion mechanisms used by general publish/subscribe systems will not be suitable for resource-limited wireless sensor networks. Energy-efficient routing schemes such as [64][65] are proposed to minimize the interest dissemination overhead. Despite the level of activities in energy-efficient publish-subscribe routing for sensor networks, the problem of efficient forwarding receives little attention.

5.2.4. String Matching Algorithms

String matching problems are well studied in the domain of bio-informatics, AI, and data mining. From the classic KMP [66] matching algorithm to the more recent ones, there has been a range of schemes proposed. TST [67] is a state-of- the-art algorithm that enables efficient binary search of strings. It is widely used in dictionary lookup and information retrieval.

ST [68] is a data structure for very high speed string matching. To handle dynamics in wireless sensor networks, an efficient algorithm must be able to insert attribute and value strings onto the suffix tree in an efficient ways. The online tree construction algorithm for suffix tree proposed by Ukkonen [59] satisfies the requirement. In [60], a memory improvement technique is presented to remove the redundancy inside internal nodes of suffix trees . Through the new data management of suffix tree, the space requirement of a static suffix tree can be reduced and be less than four integers per node in average. In this work, we adopt ST for string matching involved in the data forwarding process for sensor networks.

5.3. ALGORITHMS

We introduce in this section the four string matching algorithms in comparison. We begin with the straightforward algorithms, hash and TST. Details of the ST algorithm and its improvement will follow.

5.3.1. Hash

We use a rotation-based hash function [69] to construct the hash table of attribute and value strings and to search for strings in the hash table. The result of the hash function is calculated from the whole input string. Suppose every character in the string is coded as a number and s[i] represent the ith character in the string. For an n-character string, the hash value, h(n), is defined as follows:

$h(n) = h(n - 1) << d + s[n - 1]$

$h(0) = C$

C is a pre-defined constant. We can obtain the hash value of an n-character string by iterating the shift and add operations n times. h(n) further takes a modulo m for a hash table of m slots. The final hash value is used to associate the string with the string’s location in the hash table. A string with hash value j will be stored in the jth slot.

Multiple strings might be hashed to the same slot. A simple linked list is used to handle the collisions. When a new interest packet is received, the attribute or value string will be inserted into the linked list at the corresponding slot. Similarly, when a data packet is received, the hash function is applied to obtain the hash values of the attribute and value strings. The system then may look up the hash table to see if there's a matching attribute and value, i.e., a matching predicate.

5.3.2. Ternary Search Tree (TST)

String search using TST is similar to binary search. In TST, the data structure of a node contains a character and 4 pointers. The character is also known as the key for string comparison. Three of the pointers are used to track the decendant nodes cl, cm and cr, cl leads to substrings that begin with a character alphabatically smaller than the key. cm leads to substrings that begin with a character that equals the key. Likewise, cr leads to substrings that begin with a character alphabatically greater than the key. When a match is identified, one follows the 4th pointer to the matched entry. Figure 13 shows the TST after inserting five words: egg, gas, get, aids, and bad. Following the principle of binary search, for an incoming string ”bad”, the search path on the tree will
be e → a → b → a → d.
The TST search algorithm is detailed below in Algorithm 1. To insert a string, one search on the existing TST first. From the branch the search stops, the remaining substring is attached to extend the TST. Let gdp be a string to be inserted into the TST in Figure 13, for example. The search stops at the g node on the right. To this end, part of the incoming string, g, is already be matched. To complete the insertion, new nodes representing the remaining substring, dp, will be created and attached one by one onto the branch.

Algorithm 1 TST Search (TST T, String s)

```
$parent \leftarrow child \leftarrow ROOT(T)$
$index \leftarrow 0$
while $child \neq NIL$ and $index < s.length$ do
    $parent \leftarrow child$
    if $s[index] > parent.key$ then
        $child \leftarrow patrnt.c_r$
    else if $s[index] < parent.key$ then
        $child \leftarrow parent.c_l$
    else if $s[index] = parent.key$ then
        $child \leftarrow parent.c_m$
        $index = index + 1$
    end if
end while
if $index < s.length$ then
    $return\ NOTFOUND$
else
    $return\ parent.e$
end if
```

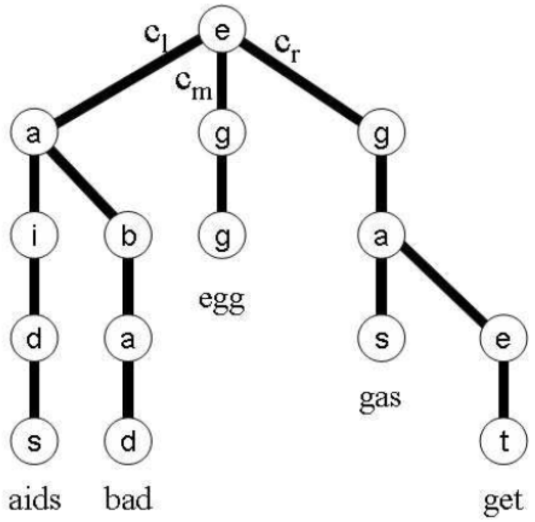


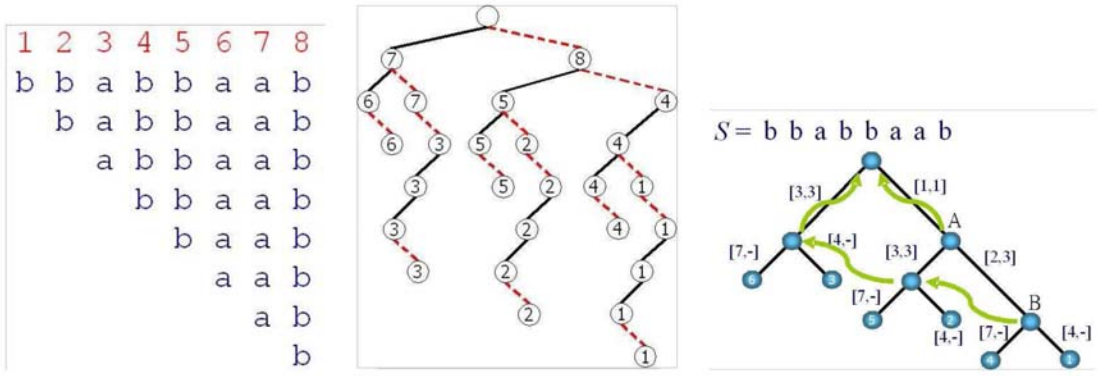
Figure 13. An example of a ternary search tree.

5.3.3. Suffix Tree (ST)

The concept of suffix tree is better explained by a more intuitive data structure called *suffix trie*. A suffix trie for a string S is a joint tree of its suffixes S1, S2, ..., to S|S|. Let Si denotes the suffix of S starting from the ith character. For instance, in Figure 14(a), there are 8 suffixes for string bbabbaab, and bbaab is the forth suffix. Once we have the suffixes, we can build a suffix trie for the original string. Figure 14(b) shows the suffix trie of bbabbaab. The right (red dash) edge accounts for the character b and the left (black) one for a. The number on the leaves shows which suffix of the string on the interest table the incoming string matches to. Given the original string and the suffixes, it is trivial to extend the algorithm for substring matching. Any substring of bbabbaab must be the prefix of one of its suffixes. A naive method to create a suffix trie is iteratively inserting the suffixes into the trie from the longest to the shortest. The construction cost is O(|S|2) steps, but it is clear that any substring search can finish in O(|P|) steps for any input string P.

ST retains this strong suit because it is essentially a compact representation of suffix trie. In a suffix trie, each edge carries one character. ST is more compact in that an edge on ST may carry a sequence of characters. A suffix trie can be transformed to an ST, and vice versa. Figure 14(c) is the ST of Figure 14(b). The pair-wise number on each edge denotes the beginning and end indices of the original string. For example [2,3] is to indicate the substring from the 2nd character to the 3rd, i.e., ba in the example. The numbers on the leaves represent the matched suffix. The curly (green) edges are traveled when none of the child matches. In this case, we can declare the exact match fails.

As for the insertions, we adopt Ukkonen's online construction [59]. With that, one can insert a new string T into ST by gradually adding characters T1, T2, ... , T|T| into the data structure without breaking any properties of the original tree.
Taking advantage of the suffix edges, each insertion can be completed in O(|T|) time. This fast construction of ST allows us to build a suffix tree in linear time. We apply ST on the attribute and value matching for sensor data forwarding. To enable exact string matching, each suffix tree is initialized by adding a special character $, which is considered a character that will never appear in any input strings. At the initialization stage, the ST, T , begins with an empty string followed by the special character, $. In each insertion, an input string Ti will be extended by adding one $ at its end and concatenates to the ST set, T . The string Ti$ is then added to the ST structure. After k insertion operations, the ST set, T , equals string $T1$T2$...$Tk−1Tk. An exact search for string P can be achieved by searching the string P in T . Other string operations such as prefix and suffix can also be achieved by using strings like $P or P$ as input. We apply also a memory requirement reduction technique proposed in [60] to improve the scalability of ST in space.



(a) The suffixes of string "bbabbaab" (b) The suffix trie of string "bbabbaab" (c) The suffix tree of string "bbabbaab"

Figure 14. Illustration of the relationship between suffix trie and tree

5.3.4. Suffix Tree with Space Improvement (ST+)

The drawback of ST is the high memory consumption. We conducted preliminary experiments to examine different possible sensor network related word sets on ST to check the space efficiency. The results shown in Figure 15 indicate that there are only 10% of nodes that have more than 5 children regardless of the size or the characteristics of the word set.

We exploit this property and present a memory optimization scheme for ST , namely, ST+. The main strategy concentrates on optimizing the memory usage of internal tree node. This is because of the observation that the larger the training/table word set is, the more internal nodes there will be. Therefore, the memory consumption can be effectively improved. will be higher.

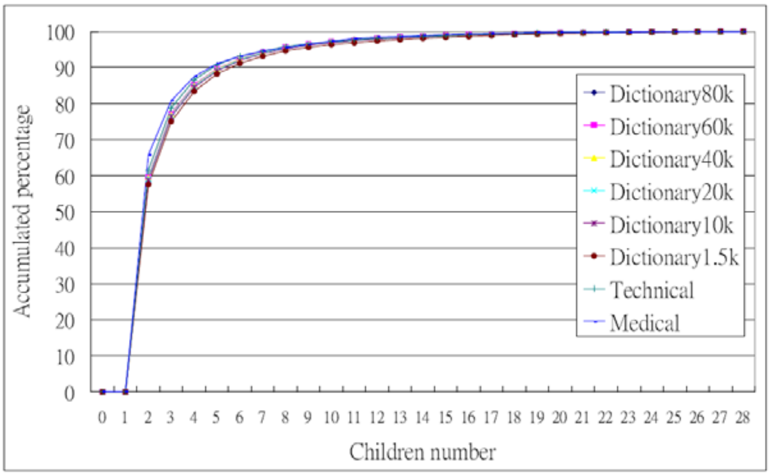


Figure. 15. ST children number distribution over different wordsets

As described in 5.3.3, each node maintains a direct mapping of (NextCharacter,NextNode) pairs. This is memory consuming. To mitigate this effect, we use two types of internal nodes, M-type and H-type internal nodes. M-type node is the same as internal nodes described in 6.3.3. H-type node uses another hash function to map its child nodes to a smaller hash table. The difference between ST and ST+ is that ST+ uses M-type nodes when a node is frequently traversed, such as the attribute strings, or when the number of its child nodes exceeds a certain threshold. Using these two criteria, we classify the internal nodes in ST , and then keep the mapping in the critical nodes direct for speed and the mapping in other nodes indirect, i.e., another hash table, for space efficiency.

5.4. ANALYTICAL COMPARISON

5.4.1. Definitions and Operations Description

	Hash	TST	ST	ST +
search	O(1)	O(|P | + log N)	O(|P |)	O(|P |)
insert	O(1)	O(|P | + log N)	O(|P |)	O(|P |)
memory	O(|S|)	O(|S|)	O(|S| + N)	O(|S| + N)

Table 3. A comparison of theoretical attributes of matching algorithms.

Table 3 provides as a concise overview of the theoretical average search time, insertion time and memory space requirement for the algorithms in comparison. We will elaborate in this subsection the results presented in the table in detail. The terminology in this section and the rest of the paper related to strings are listed below:

- S: The training word set consist of N strings from S1 to SN, the total word length of S is represented by |S|
- P: The test string, and the length of the string is |P|

Meanwhile, the operations we would use for fast forwarding are defined as follows:

- Search exact match through the data structure (exact match): For a input string P, search through all the strings in the training set S to see if there’s a string P* that is exactly the same as P; If such string exists, then return the index of the string, otherwise, the operation returns a search fail notice.
- Search for prefix/suffix/substring match through the data structure: For a input string P, search all the strings in the training set S to see if there’s a set of string {P’} such that every strings in {P’} has more than one prefix/suffix/substring that exactly matches P.
- Insertion: Insert a string P into the data structure.
- Deletion: Delete a given string P from the data structure. In our implementation, the prefix/suffix/substring matching are both applying the algorithms listed in [52]. In consequences, the performance of these operations thus can analogize to exact match.

As for deletion, to keep the designation simple, we add a valid bit on each leaf or entry for all the data structures and 2 counters to represent the number of valid/invalid strings in the training word set. If the ratio of valid/invalid strings is too low, then we automatically rebuild the whole structure with respect to the current word set. This deletion strategy is also highly related to an exact match operation.

5.4.2. Search Time

TST is an efficient algorithm, it can complete an exact string search in $O(\log(N) + |P|)$ steps. In substring search cases, TST can finish a search in $O(|P| \log(N)+ |P|*2)$ steps according to [3]. Each step can be viewed as a jump from one tree node to another. ST can complete a single attribute-value matching, a search or an insertion in $O(|P|)$ steps. For substring search, ST can accomplish a single task in $O(|P|+\beta)$ steps if the substring relationships among all training strings are figured out in the insertion operations, whereβdenotes the number of matched strings in S. Each step can either be a pointer jumping. (The pointer moves one step on the edge). The performance of hash table really depends on the design and adjustment of the hash function. It is generally fast and low memory consumption but still has some weak points. For example, it is hard to implement a substring or super string matching by using merely a hash table while there are already efficient ways for such requirements. The time complexity of a hash table based attribute-value matching with rotation algorithm is $O(|P|)$ steps for hash value calculation and $O(|P|)$ string comparisons. Notice that the asymptotic notation of the three algorithms are based on the ”number of steps”, the real time performance still needs to be tested.

5.4.3. Insert Time

As paper [66] suggests, ST can have insertion time linear to the length of the input string input. For TST, an insertion can have a cost of $O(|P| + \log(N))$. For hash based method, it takes an amortized $O(1)$ steps for any insertion when the hash table is sparse enough.

5.4.4. Memory Consumption

For memory usage, both the four data structures/algorithms consume memory space linear to the length of the training string set S. But the coefficients attached with asymptotic compounds for these algorithms are not the same. Generally speaking, the memory complexity coefficient comes with suffix tree-based data structure is much greater than TST and Hash. The effect will show in the experiment results.

5.5. EXPERIMENTAL DESIGN

5.5.1. Experimental Environment

The test environment we use consists both x86 and sensor-based environments.

1) For x86 platform, we use laptop with Intel Pentium-M 1.4G CPU, 768MB RAM. The operating system is Linux with kernel version 2.6.12. The test programs are written in C and compiled by gcc v3.4 with optimization flag ”O3”. We randomly pick up strings in each word set to form the training sets and test sets. To avoid the influence of HDD I/O, the program will load all strings into main memory before the insertion and searching procedures start. For time measurement, we use standard GNU library functions in < time.h > for general tests and < sys/time.h > for tests that need higher resolution data (Execution time is less than 10ms).

2) For sensor based environment, we use Telos with MSP430 series MCU designed for sensor applications [70] , Its correspondent development tool, and benchmark IAR Workbench [71] as our test environment. The hardware simulation tool contained in the workbench is used to run each test and calculate the number of time cycles consumed by each string insertion and search for all three algorithms. The clock cycle of the chip is set to 2MHz, and the timer frequency is set to 3.27KHz.

5.5.2. Word Sets

In consideration of potential functions of real world sensor applications, we choose five different word sets in the experiments:

1) Dictionary word set: Dictionary words that are collecting from free dictionaries and word lists. Totally 80000 words.

2) Technical word set: Frequently used words from the book ”Longman Glossary of Scientific & Technical Terms”, totally 1500 words.

3) Medical word set: Excerpt from the index of a medical sensor textbook, totally 340 words.

4) BL-Life word set: Word set which accounts for rooms (for example: office, lab621) and elevators’ status such as ”up”, ”down”, ”close”, or ”floor”, and people who are using these infrastructure (user names like ”Polly”, ”Tim”, ”Jerry”, ... etc). There are totally 50 attributes and values in BL-Life set.

5) USNames word set: Excerpt the top 20 men’s and womem’s names from statistics of the demographic distribution of the United States from 1980 to 1990. The USNames word set is divided into men and women names.

On x86 based platform, word set Dictionary, Technical, Medical and BL-Life are used; On sensor-based platform, we only use word set BL-Life and USNames in consideration of the limited resource of the system.

5.5.3. Metrices

We use 3 different metrices to evaluate the performance of fast forwarding algorithms:

1) Average insertion time: The average time to insert one string into the data structure.

2) Search time for exist strings: The average time to search for one test string which exactly exist in the training set.

3) Search time for not-exist strings: The average time to search for one test strings which exactly NOT exist in the training set.

In a real sensor data forwarding environment, most matching events are case (2) event. Cases (1) and (3) may happen consecutively with a much lower probability. Generally speaking, case (2) is the dominating factor of the performance of forwarding algorithms.

5.6. EXPERIMENTA RESULTS

In this section we show the experimental result of insertion time, search time and memory consumption of Hash, TST and ST. Each test operates on both x86 and sensor-based platforms. The results are also represented and explained with respect to each specific platforms.

5.6.1. Search Time

In this part, the test string sets are divided into two parts: We and Wne. All strings inside every We sets do exist in the training word set while all strings inside Wne sets does not. Words of Wne Words in Wne are choosing from all word sets except strings in We. Wne for x86 platform are chosen from Dictionary, Technical, Medical, and BL-Life word set while those for sensor-based platform are chosen from only BL-Life and USNames.

1) Search Time on x86 Platform: The result shows in Figures 16 and 17. Every method has a better search time in Wne than in We and the rank of performance for We word set group is Hash $\geq$ ST $\geq$ ST+ $\geq$ TST . On the other hand, the search performance rank for word set group Wne is generally ST > ST+ > Hash > TST except the word set BL-Life.

From the graph we observe three interesting trend:

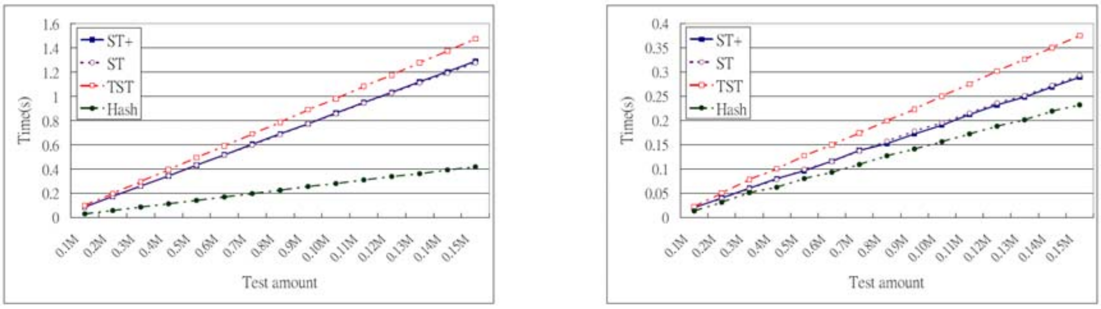
1) Hash method’s performance seems to be influenced by the length of the strings. We think the reason may come from the implementation of hash method. We use C language standard library call to do direct string match and such kind of call can use more time when the given string is long.

2) In We tests, Hash method has the best performance and the time gap between

Hash and other data structures can reach 1 second level when the number of search input reaches 1.5 million. However, in the Wne test, the performance of ST becomes better. The rank of performance from this perspective is

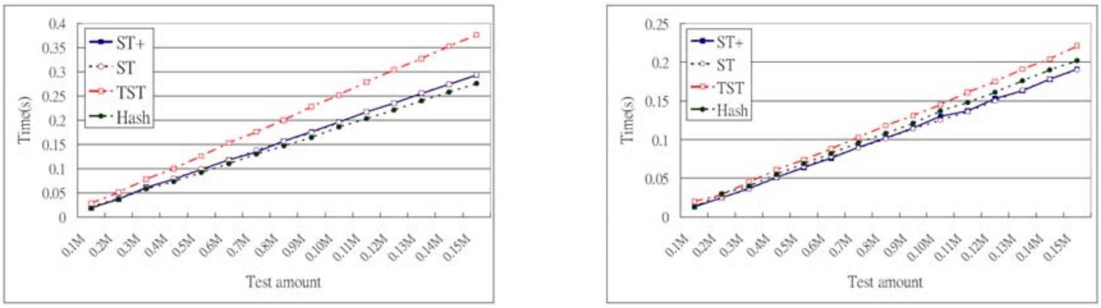
$$ST \geq STmemory-improve \geq Hash \geq TST .$$

3) The search time difference between ST+ and the original ST is less than 5%. Experiments show that on general purpose computers, ST+ is viable.



(a) Search time on Dictionary word set

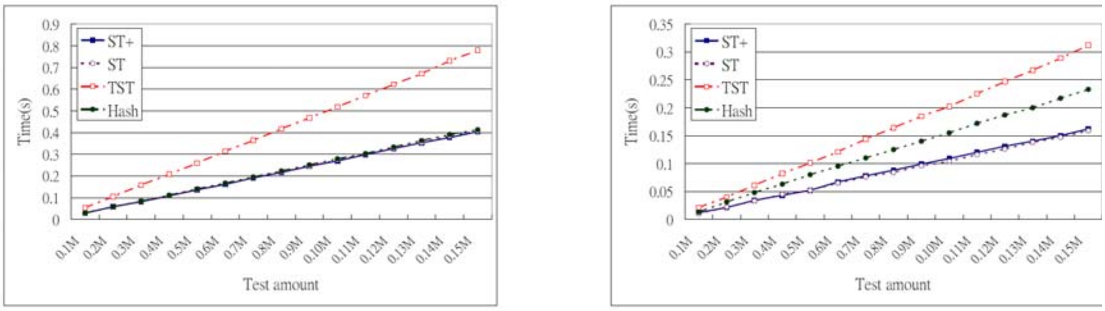
(b) Search time on Technical word set



(c) Search time on Medical word set

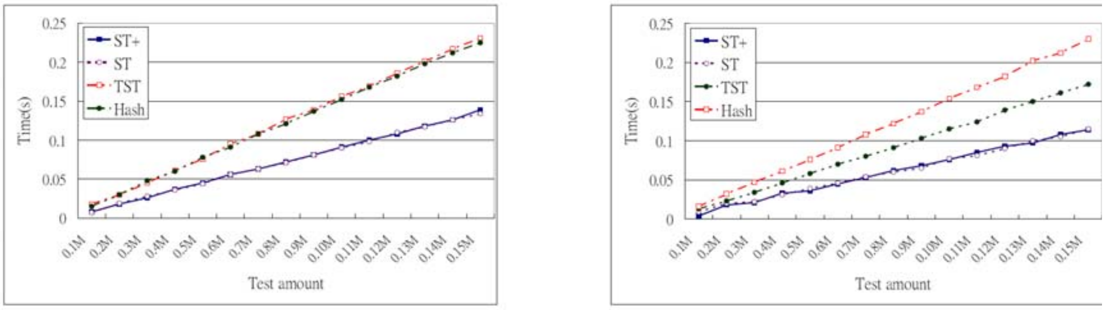
(d) Search time on BL-Life word set

Figure 16. Result of exact string search over different word sets and algorithms



(a) Search time on Dictionary word set

(b) Search time on Technical word set



(c) Search time on Medical word set

(d) Search time on BL-Life word set

Figure 17. Result of non-exist string search over different word sets and algorithms

2) Search Time on Sensor-Based Platform: Table 4 shows the average time cycles of 10000 independent search operations over word set BL-Life and USNames using Hash, TST and ST Both We and Wne results are listed. In Wne tests, all three algorithms runs faster than that in We tests. It is obvious that Hash method consumes the longest time to complete all searches. ST and TST both perform well, ST is faster than TST, but the gap is not large. We analyze the execution of the Hash method and observe that almost 40% of clock cycles are used in calculating hash values. The possible reason can be the arithmetic instruction set used by MSP430 microcontrollers. When modular-based hash functions are used, hash function uses more time to output hash values.

	Hash	TST	ST
BL-Life	2.094	1.349	1.143
USNames-mem	2.494	1.428	1.264
USNames-womem	2.493	1.472	1.214
BL-Life(non exist)	1.762	1.099	0.891
USNames-mem(non exist)	1.462	0.949	0.746
USNames-womem(non exist)	1.461	1.037	0.849

Table 4. The search time of Hash, TST and ST on sensor-based platform.

5.6.2. Insert Time

1) Insert Time on x86 Platform: The insertion time is the average time to insert one string in the training word set into the given data structures (including ST, TST and hash table). Our results in Figure 18 show that TST and hash method outperform ST in every word set. The memory efficient version of ST uses even more time in insertion due to the effort maintaining data structure and tag changes. The difference between each method is no more than 10−6 second hence and still tolerable when the processing speed of the node is fast enough and attributes do not update too often. In truth, each step of suffix tree insertion can be an edge traversing, a single node creation or even several suffix link constructions. To insert a character into the suffix tree consists one to several operations listed above depending on the input strings, and some operations can even use more time than inserting a string into Hash or TST. In consequences, ST may consume more time in each loop round then TST and Hash.

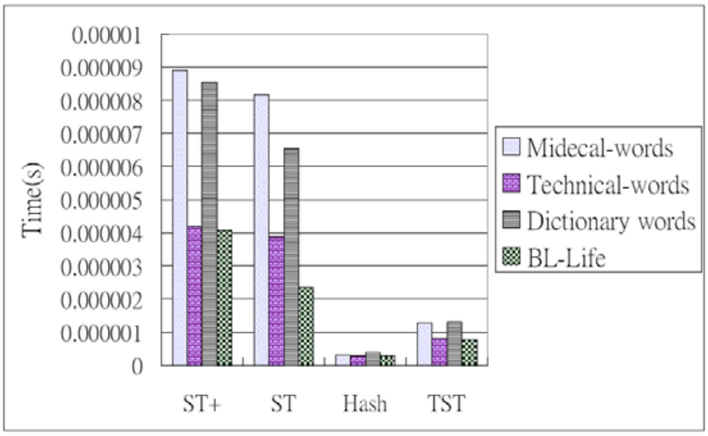


Figure 18. Insertion time of different word set and algorithms

2) Insert Time on Sensor-Based Platform: Table 5 shows the average insertion time cycles consumed in word set BLLife and USNames tests. The result shows that the performance rank of Hash, TST and ST remains the same as the rank on x86 platform. All BL-Life word set tests have better performance than USNames word set ones.

	Hash	TST	ST
BL-Life	3.25	6.95	11.76
USNames-mem	3.85	9.45	16.80
USNames-women	3.95	10.75	11.60

Table 5. The insertion time of Hash, TST and ST on sensor-based platform.

5.6.3. Memory Consumption

1) Memory Consumption for x86 Platform: The memory consumption of each data set

and algorithm is listed in Figure 19. From Figure 19 we can see for BL-Life, Medical Sensor word sets, ST uses a hundreds of Kilobytes than TST. In dictionary word set case, with the growth of the training word set, ST attempts to consume more memory than TST. ST memory improvement is not obvious in small word sets. However, when the word set size greater than 10000, the improvement becomes noticeable. In average, the memory optimization can save 20% to 24% memory usage. Note that any process in linux has a basic size of about 2MB.

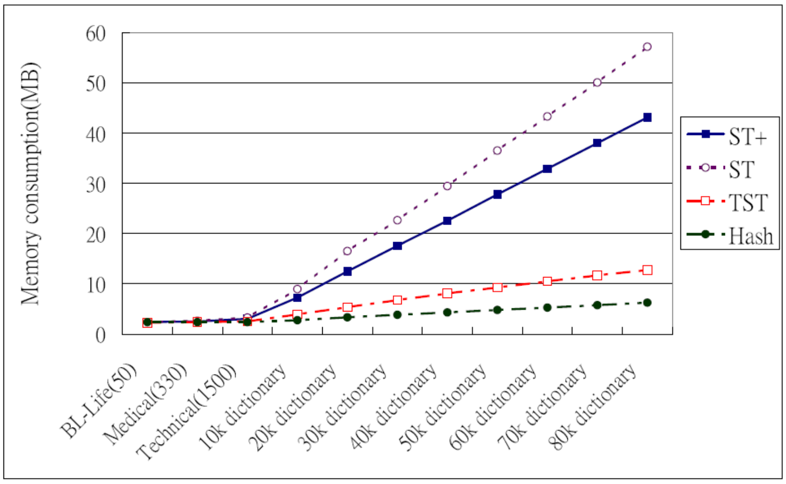


Figure 19. Memory consumption

2) Memory Consumption for Sensor-Based Platform: Table 6 list the total memory usage of Hash, TST and ST over word set BL-Life and USNames, the unit used here is byte. Current Telos spec only support totally 32KB flash memory. Assuming that 80% memory are accessible, it means ST can serve well in a system consists less than 100 human names or approximately 150 normal attributes. TST and Hash can contain more attributes but it may take longer time on search operations, which can be the bottleneck of applications with frequent message updates.

	Hash	TST	ST
BL-Life	865	2064	7840
USNames-mem	645	1464	4667
USNames-women	645	1212	4914

Table 6. The memory consumption of Hash, TST and ST on sensor-based platform.

5.7. DISCUSSION

5.7.1. Word Set Dependency

On x86 based platforms, observed from the experimental data (Figures 16,17), we can point out that the number of training strings can affect the performance of ST, TST, and hash. Unlike Hash method which can still complete search tasks fast under big word sets, ST tempt to consume more time on larger word sets. Algorithm of ST shows the linear relationship between input string length and search time. The real implementation confirms the difference for theory and experiments. The fact reflect that in real applications, hardware effect severely contributes to the performance of matching process. On sensor-based platforms, different word sets also have different search speed. For example, the search time of BL-Life word set in every mechanism outperforms the search time of USNames. But the size effect is not obvious. Based on our results, we consider that size effect in smaller word set tests are less obvious.

5.7.2. Trade-off Between Memory and Time

Based on the results, we observed some trade-offs between space and speed. Among three data structures, Hash method has the best performance in exact string search. But a simple hash table does not contain sufficient information to perform substring or super string match. If the given sensor data forwarding system has to process prefix suffix or substring matching, other mechanisms should be imported. TST performs averagely in the middle of the three algorithms in each test but have long search time. ST has long insertion time and occupies large memory space. On the other hand, it also has a short exact string search time and an even better search time for non-match cases. The weakness of ST is definitely blamed to the large memory space. For small word sets, ST, TST and Hash are similar in memory size, but with the word set grows, the space requirement of ST dramatically goes up. The memory efficient version of ST can reduce 20%-24% memory space and still keeps the search time in our experiments when the training word set contains more than 10000 strings, but the reduced memory size still large compared to TST and Hash.

However, when we move our test environment to sensor-based platforms, the search time rank inverses. Due to the instruction set dependency, modular-based instructions takes much longer time on sensor-based platforms. To have better performance, hash-based matching method should carefully evaluate the properties of the given platform before choosing hash functions. In our experiences, rotation-based hash function has the least conflict in average cases, but it takes almost two times longer to finish a search than TST.

5.8. CONCLUSION

We study and evaluate the use of ST for efficient predicate matching for data-centric sensor networks. With the proposed memory optimization scheme, we are able to implement ST on a sensor network development platform. The experimental results show that ST performs significantly better than TST and hash which suggest ST being the most promising solution for fast sensor data forwarding.

六、分析與討論之三 — Sensor Network for Everyday Use: The BL-Live Testbed Experience

6.1. Introduction

There have been avid research activities on sensor networks worldwide. Research labs, such as Berkeley WEBS and UCLA CENS, have initiated research projects and related hardware/software platform development early in 1999-2000. NSF of the United States started to call for research proposals in the area of *Sensors and Sensor Networks* in 2003. Early vision on the military use of the sensor networks also prompted the establishment of sensor and sensor network related programs in DARPA.

It is now 2006. After over 7 years of R&D in sensors and sensor networks, we see now sporadic reports of sensor networks for short-term experimental purpose [72][73][74]. There lacks still any long-term, everyday-use deployment. We wonder why. Is the deployment of sensor networks too difficult or practically impossible? Motivated to address this question, we deploy a 30+ node wireless sensor network in a university campus building, the Barry Lam (BL) Hall of Electrical Engineering at the National Taiwan University main campus.

This project is referred to as *BL-Live* for that the seemingly cold concrete BL Hall is transformed into a lively smart office building. The main objective is to obtain practical experience and to discover problems that otherwise will be difficult to observe in small-scale test-beds or in simulations.

The sensor network in the BL Hall facilitates two everyday services: 1) Elevator Report and 2) Smart Office. The first service reports the status of the slow-paced elevators located on two opposite sides of the building. This service allows the building residents to select an elevator that will arrive earlier at the floor they desire. The second service detects the presence meeting participants in an office. This allows automated control of the camera to broadcast publicly the progress of a meeting in the office whose door is better off kept closed to conserve energy in the summer. The two services, although casual, address real needs of the BL Hall residents. More importantly, the two services function as the applications that drive the use of the sensor network infrastructure.

To support the Elevator Report and Smart Office services, there involve three major system and network components, hardware, sensor networking, and sensor signal processing. Hardware is the most fundamental element. We have successfully cloned and manufactured 40 pieces of an ultra low-power wireless sensor node named Telos [75] from scratch. Most of these Telos clones are deployed in the building to form the sensor network infrastructure for the Elevator Report service. The rest are deployed in a number of offices and carried by the volunteers for the Smart Office service.

Each node on the sensor network infrastructure runs the Magnetic Diffusion [76], a routing protocol that enables the collection of the sensor data from the elevators to the data sink in one of the student laboratories in BL Hall. The nodes deployed in the offices runs a simple protocol that periodically probes for the existence of any sensor nodes alive within the radio range. Each node carried by a volunteer answers the probes from the office nodes. This enables the detection of the volunteer's presence in the offices.

The sensor nodes placed in the elevators are integrated with accelerometers. The accelerometers sense the acceleration and breaking patterns of the elevators. From these motion data, we derive the status of the elevators on-node. These status reports are then relayed by the sensor network to the data sink. The data sink is connected to a Web server and the statuses of the elevators are made available as a public Web page. The office sensor nodes deliver the presence information via Internet to the Web server and the presences of the volunteers are made available as well as a public Web page.

After more than 9 months of experience building, deploying and operating such a wireless sensor network, we come to a number of observations and suggestions on the cost, deployment, placement, data delivery, energy efficiency, and usability issues towards practical, commercialize-able sensor networks in the future.

6.2. BL-Live Services

We elaborate more in this section the problems we intend to solve with and our implementation of the Elevator Report and Smart Office services.

6. 2.1. Elevator Report

The BL Hall has 7 levels above the ground and 1 level underground. The resident population is approximately 900+ with the 7th floor still vacant. The building is equipped with two slow-paced elevators that a round-trip from a high level floor in the peak hours takes more than 10 minutes of time. The two elevators are located on two opposite sides of the building which makes it difficult for the residents to decide which elevator will arrive earlier. The purpose of the Elevator Report service is to provide continuous status reports of the two elevators including the current level and the direction of moving. This will allow the residents to select an elevator that will arrive at the floor they desire earlier and to leave offices just in time to catch the arriving elevator.

Figure 20. Elevator Report (Left) and Smart Office (Right) Service Web Interface

One sensor node with accelerometer is deployed in each elevator. The accelerometer data are taken and analyzed on board. The status derived is transmitted wireless to the relay sensor nodes. There are 11 relay nodes deployed on the 4th to the 7th floor of the building. These relay nodes forms the sensor network infrastructure and runs MD routing to disseminate the status reports to the data sink. The data sink is connected to a Web server in a student laboratory on the 6th floor and the status reports are used to update the Elevator Report service Web page as shown in the left plot of Figure 20.

6.2.2. Elevator Report

The BL Hall is equipped with a central air conditioning unit, but it does not cover all offices in the building. The corridors are not air conditioned. The north side offices from the 6th floor and up are air conditioned by individual air conditioners per office. In the meantime, the university advises the meetings between the faculty member and students to be kept public. The dilemma is that when there are meetings, the office doors will need to be kept open with the air conditioning running. The purpose of the Smart Office service is to detect the presence of meeting participants in an office. This allows automated control of the in-office camera to take the progress of the meeting and broadcast publicly on a flat panel display outside the office. This way, the office door can remain closed to conserve the energy in the summer.

A PC-based control center is installed in two offices. One of them is a faculty office (6th floor north side) and the other is a student laboratory. Connected to the control center include a camera pointing towards the meeting area, a flat panel display embedded on the wall facing the corridor, and a sensor node emitting radio signals periodically to probe for the presence of meeting participants. Each volunteer wears a sensor node. The wearable sensor node serves as an active RFID tag and transmits the pre-configured ID in response to the probes from the sensor node connected to the control center. When the sensor node at the control center side receives a signal, it will extract the ID from the radio packet and pass the information to the control center via the USB port. When there are multiple wearable nodes detected, the camera will be turn on and the screen will broadcast the meeting live. The control center will further transmit the information to a Web server which in turn highlights the name of the volunteer on a BL resident information page as shown in the right plot of Figure 20.

6.3. System Components

We elaborate in detail in this section the technical components in hardware cloning, sensor networking, and elevator signal processing. Provided also are our preliminary assessments on the costs of sensor node manufacturing, the success ratio of sensor data delivery, and the accuracy of elevator status inference.

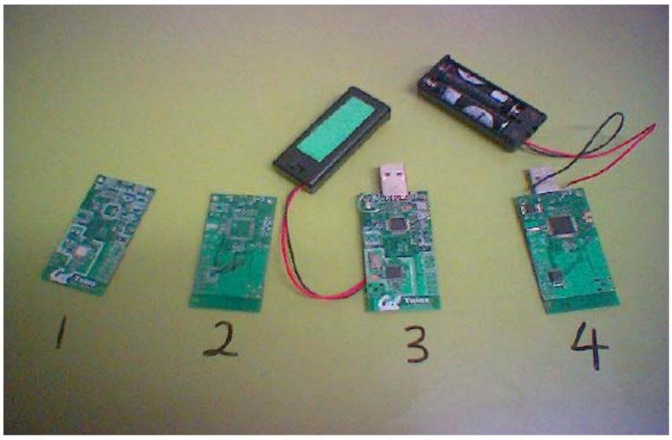


Figure 21. Telos Revision B. Before soldering (1, 2) & after soldering (3, 4)

6.3.1. Cloning Sensor Nodes

Cloning Processing. We choose to clone the ultra low-power sensor node, Telos (See Figure 21). The cloning process involves three steps: (1) PCB manufacturing (2) parts purchasing and (3) part soldering. First of all, the schematic, printed circuit board (PCB) layout, also known as the Gerber file, and bill of materials are open source and available from the TinyOS website. We send the Gerber file to the PCB manufacturer to produce the PCBs from which we receive the printed boards within two weeks.

The most time consuming part is the purchasing step. We need to acquire all the parts we need. Before we know what parts to buy, we spent a significant amount of time studying the parts and the corresponding functionalities in the bill of materials.

The datasheets are carefully examined to make sure that we order the parts with correct footprints. Our attempt purchasing from local suppliers is not successful for the reason that most of the suppliers do not take low quantity orders. We have no choice but to turn to Digikey , a major electronic component distributor that offers a breadth of product lines, provides with online catalogs and accepts low-quantity orders.

After getting all the parts ready, the third step is to solder all the parts on the PCBs. The components used by Telos are very small and some have special footprints that are almost impossible to solder by hand. We take a stencil and toaster oven approach5. The idea is to use the stencil to paste the solder paste on the PCB, place the components on the solder paste, and melt the solder paste using a baking oven. A microscope is necessary to check whether the components are well aligned before sending the solderpaste-and-component ready PCB to the baking oven. Temperature control is also important. The components will malfunction if they are overheated for too long.

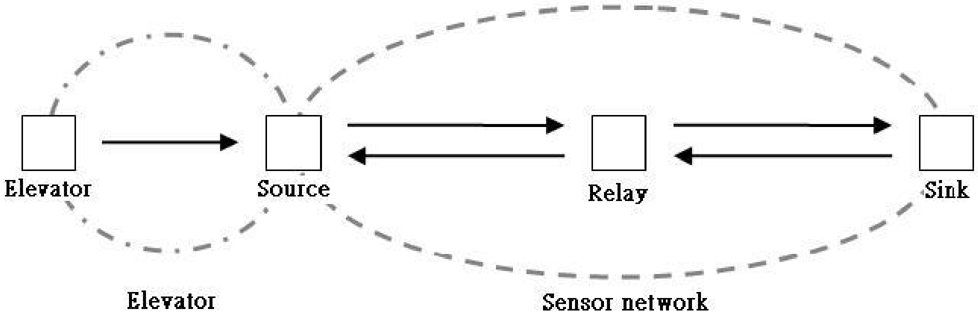


Figure 22. The illustration of wireless sensor network

Turn-Out Rate. We clone 50 pieces of Telos. After we produce the hardware, install the program, and test, only 5 of them function correctly. Thisnumber is far lower than what we anticipate. Hardware debugging is essentially to identify abnormal voltage level, resistance level and waveform using the electric multi-meter and oscilloscope. Once the abnormal component is identified, we use the microscope to check the quality of soldering. Two kinds of problems are common: (1) component placement and (2) soldering precision. For example, a 100 ohms resistor is soldered at places for a 100K ohms resistor. Either too much or too little soldering paste will not be good. The former creates short circuit over two consecutive pins. In case of the latter, the IC might not be soldered firmly on the PCB. After a round of hardware debugging, we fix most of the malfunctioning Telos and 40 of them function fine at the end. Problems of the remaining Telos' are unknown and need substantial rework.

Manufacturing Cost. The manufacturing cost consists of three parts: component, equipment, and labor. The electronic components cost approximately 1600NTD per piece of Telos, and the cost of PCBs is 400NTD per piece. We bought some equipment for the making and testing. Those include the stencil, microscope, oscilloscope, electric multi-meter, and baking oven. The total cost of these equipments is approximately 150,000NTD. The final part is the labor cost which is difficult to estimate. We spent time on studying the datasheet, finding supplier, purchasing components, trying out the stencil and toaster oven method, placing components, testing, debugging and fixing. It takes 4 months and 2 graduate-level man power to complete the Telos cloning process. The manufacturing cost is summarized in Table 7 below. The overall cloning cost, 250,000NTD, might not be low. However, going through this cloning process helps us to understand the hardware. We also gain insights on the manufacturing cost if the sensor nodes will be mass produced later.

Electronic Components	1600/piece NTD
Printed Circuit Board	400/piece NTD
Material Cost per Piece	2000/piece NTD
Total Material Cost	100,000 NTD
Total Equipment Cost	150,000 NTD

Table 7. Cost of successfully cloning 40 pieces of Telos

6.3.2. Sensor Network Infrastructure

Elevator. When the elevator arrives at a particular floor, the mote inside the elevator will transmit a message which contains the information of the estimated floor number and moving direction. There are nodes placed on the ceiling nearby the elevator each floor. When the node receives the message from the elevator, it will check the correctness of the floor number first. If the number is right, the node will transmit the message toward the sink through a wireless sensor network. The node inside the elevator will be the 'Elevator' node and the node outside the elevators will be the 'Source' node as indicated in Figure 22.

Magnetic Diffusion. The 'Source', 'Relay', and 'Sink' nodes form a wireless sensor network in the building. To route data through the network, we adopt a routing protocol called Magnetic Diffusion. In that, the sink, functioning like the magnet, propagates the magnetic charge to set up the magnetic field. Under the influence of the magnetic field, the sensor data, functioning like the metallic nails, are attracted towards the sink. The magnetic field is established by setting up the proper magnetic charges on the sensor nodes within the range of sink. The strength of the charge is determined by the hop distance to the sink. In Figure 22, the sink node broadcasts interest periodically and then builds a magnetic field upon other source and relay nodes. Once the source node detects elevator door opening and has data (elevator status information) to send to sink, the data will travel from the low to strong charge relay nodes, and finally arrive at the sink which has the strongest charge.

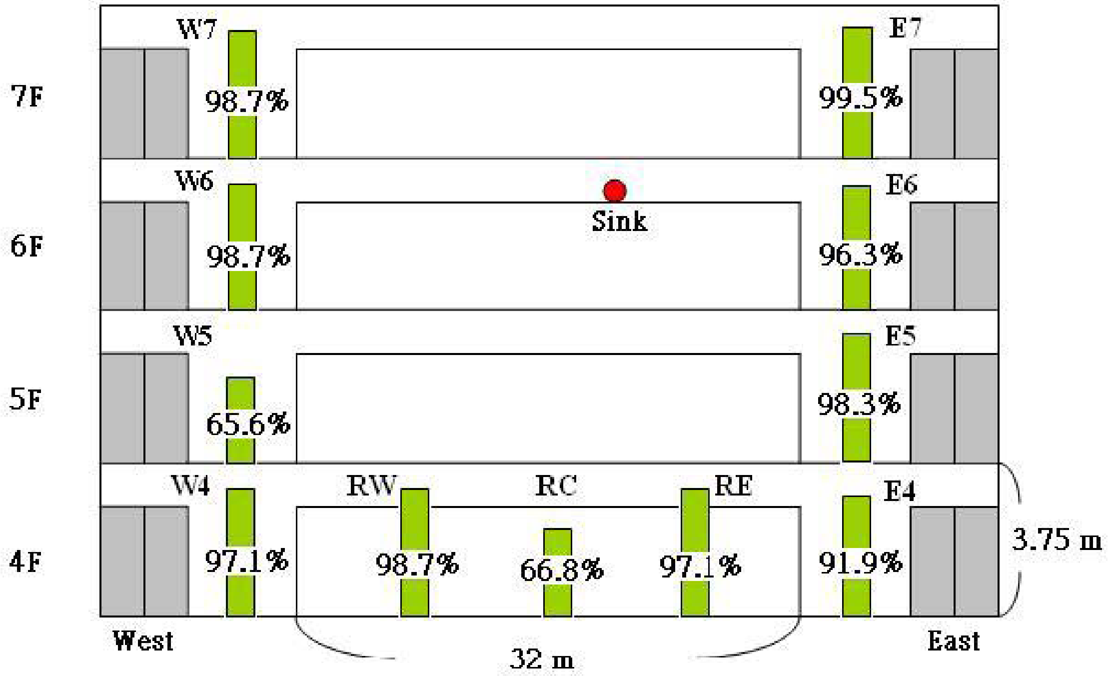


Figure 23. Reachability of individual nodes in wireless sensor network

Data Reachability. For the evaluation, we deployed two sensor nodes with accelerometers in the two elevators and 14 Telos to build the sensor network. The nodes nearby the elevator send the data twice for each elevator arrival for better reachability. The deployment extends for 4 floors, from the fourth to the seventh. Figure 23 illustrates the placement of the nodes in BL Hall. The duration of experiment is two hours. During the time, we logged the message received by the sink node.

The reachability of the elevators is shown in Table 8. The reachability of the east elevator is 95%, about 5% higher than the west elevator. This is because the two nodes shown in Figure 23, west 5th floor (W5) and relay center (RC), were out of power during the experiment, and thus cannot transmit messages to the sink. The reachability of the west elevator should be close to that of the east elevator without the failing nodes.

Node ID	# Packets Sent	# Packets Received	Reachability
E. Elevator	60	57	95%
W. Elevator	56	50	89.2857%

Table 8. Reachability of the elevators

6.3.3. Accelerometer Signal Processing

We measure the acceleration of the elevator using the accelerometers to decide the status of the elevator. There are other means to acquire the status of the elevator. For example, one may obtain the elevator status from the elevator itself. Without working with the elevator maintenance, the concern is the safety of the building residents.

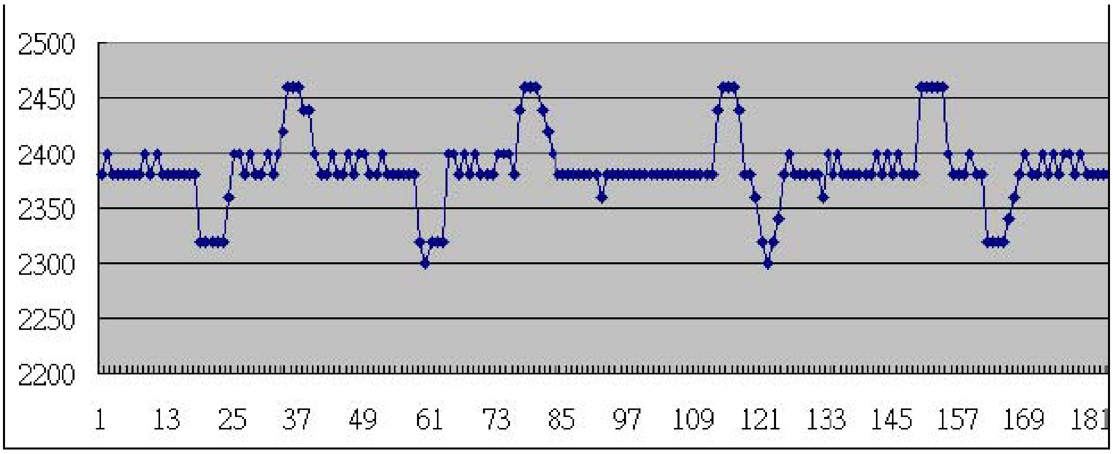


Figure 24. Typical accelerometer data pattern of the elevators

Using a camera and broadcasting the video openly on the Web is another option, but there is still privacy concern. The use of accelerometer is less intrusive and we can avoid needing the elevator engineers' attentions.

Figure 24 shows the typical up-and-down patterns in the acceleration data collected over time. We can identify four phases of operations of the elevator: STILL, ACCELERATING, MOVING, and BREAKING. A pair of up and down represents the elevator ACCELERATING and BREAKING. In between such an up and down pair, the elevator is in the MOVING phase. In between the up and down pairs, the elevator is in the STILL phase and it stops and opens its door to let people in and out.

The amount of time the elevator stays in each phase is unique for each floor movement. We collect the acceleration data of the elevator every 0.5s for hours, including periods with heavy and low elevator traffic. There are in total 500 floor movements in 3 hours when collecting the acceleration data. According to the time the elevator spends in two phases, the ACCELERATING and MOVING phases, we can identify well which floor the elevator goes to.

We construct four tables of the maximum values and the minimum values of the accelerating time and the moving time for all-pair movements. When the elevator finishes an operation, the sensor checks these tables to tell which floor the elevator goes to. The system begins on the seventh floor, the initial floor, when the elevator stops and opens the door, the sensor would tell which floor it is and send the status out of the elevator to a sensor network outside the elevator. Our algorithm can identify the floor movement 100%. The four tables also allow us to identify the noises. We create 90 noises by shaking and lightly jumping in the elevator. 90 noises are all detected.

6.4. Experience and Lessons Learned

The BL-Live project starts from June 2005. Cloning of the sensor nodes takes about 3 months. Implementation of the services and sensor network protocols takes slightly more than a month of time. Deployment, testing and debugging are the most time consuming part. This takes pretty much all the rest of the 9 months. We learn a few lessons in manufacturing, choice of hardware, node placement, data delivery, and usability. Some lessons rise from the quantitative evaluations of the system components; others come from the experience deploying BL-Live hands on.

Manufacturing Cost. The cost of building this 30+ node sensor network is approximately 250,000NTD. The material cost per unit is approximately 2000NTD that is lower than the catalog price from all manufactures worldwide. When there is demand, the sensor nodes can be mass-produced and assembled automatically. The price is expected to drop below 1000NT per unit. We, therefore, think cost is likely not the bottleneck commercializing sensornetwork- based products.

Hardware Platform Choice. The deployment of sensor nodes in a building depends highly on the physical structure and material used by the builder. The deployment is generally easier in factory-style building where the space is more open and connected, as opposed to in office-style and apartment-style where floors, ceiling, walls lined up in a significantly higher density. BL Hall is a hybrid building. It is constructed with open space from the 4th floor and up. Hence, a small 20-node sensor network is able to cover the whole 4th to the 7th floor area. From the 3rd floor and down, it is office-style. The floors are separated by solid concrete which blocks the wireless signals from the low-power radio on the sensor node. The implication is that more powerful gateway nodes that have higher-power wireless or wired communication capability will be necessary for sensor network deployment in modern office buildings.

Node Placement. In the course of deploying the sensor network infrastructure, we have changed the placement of the relay nodes for a number of times. These sensor nodes eventually are either hidden or placed high up on the wall such that it is not easy to see or remove. It appears that human curiosity can sometimes supersede integrity. We find in two occasions that the sensor nodes are taken from where they are. The sensor nodes taken the first time are returned after notes are posted at the places they are taken. We are not so lucky the second time. That is when we decide to place the sensor nodes not for the best coverage but for the best security.

Data Delivery. The delivery rate of the sensor data varies from 90%-95% depending on the weather, the use of conflicting radio devices in the building, and the battery lifetime. Among the three factors, battery lifetime of the bottleneck relay node affects the data delivery success ratio the most. For everyday applications, the cost of wiring the relay nodes one time may not be higher than the reoccurring cost of changing the batteries. For better data integrity and consistency, the infrastructure nodes in commercial sensor networks are more practical connected to wall power. That is saying, for the wired infrastructural sensor nodes, energy efficiency will not be the most critical issue. For the mobile sensor nodes, however, energy efficiency is still a serious problem.

Usability. From the Smart Office experience, we find the current form factor is too large and uncomfortable to be wearable. The lack of alert when the batteries are running low is also an issue. Most willing volunteers suggest that a thin card size node with recharge capability and low-power alert will be optimal. Most volunteers have neglected to bring along the wearable sensor nodes when they move between offices from time to time. This problem can be improved by implementing the office door access functionality on the wearable node. What is perhaps more serious is the privacy concern. Observed also isthat a small number of volunteers in few occasions turn their wearable node off on purpose to hide their presence. Running of the system must base on the mutual agreement between the building residents and managers.

6.5. Summary and Outlook

With 250,000 NTD material and equipment purchased, 9 month time elapsed, and 6 graduate student power allocated, we present BL-Live, a 30+ node sensor network deployed for everyday services, Elevator Report and Smart Office. Based on our experience, we revisit technical issues such as manufacturing cost, deployment, communication, energy efficiency, and usability. While we think it is important to tackle these technical issues, the current bottleneck is at our ability to identify needs and the experience using sensors.

Without actual deployment and experimentation, the proposed solutions might never be practical. Having observed the progress of sensor network R&D over the years, we think what lack to advance and commercialize sensor network technologies are *actions* and experience sharing. Thus, with this work, we would also like to advocate for more deployment work and an open forum for experience sharing.

七、參考文獻

[1] C. Intanagonwiwat, R. Govindan and D. Estrin "Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks", submitted to ACM MobiCom 2000. http://citeseer.nj.nec.com/intanagonwiwat00directed.html

[2] C. Perkins and P. Bhagwat. Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers. In Proc. of the ACM SIGCOMM, October 1994. http://citeseer.nj.nec.com/perkins94highly.html

[3] David B. Johnson, David A. Maltz, and Josh Broch. DSR The Dynamic Source Routing Protocol for Multihop Wireless Ad Hoc Networks. In Ad Hoc Networking, edited by Charles E. Perkins, chapter 5, pages 139--172. Addison-Wesley, 2001. http://citeseer.nj.nec.com/johnson01dsr.html

[4] IETF Mobile Ad-hoc Networks (manet) Working Group, http://www.ietf.org/html.charters/manet-charter.html

[5] Charles E. Perkins, Elizabeth M. Belding-Royer, and Samir Das. Ad Hoc On Demand Distance Vector (AODV) Routing. IETF RFC 3561

[6] William Adjie-Winoto, Elliot Schwartz, Hari Balakrishnan, and Jeremy Lilley. The Design and Implementation of an Intentional Naming System. In Proceedings of the ACM Symposium on Operating Systems Principles, pages 186{ 201, Charleston, SC, 1999. Page 14 http://citeseer.nj.nec.com/adjie-winoto99design.html

[7] A. Carzaniga and A. L. Wolf. Forwarding in a Content-Based Network. Proceedings of ACM SIGCOMM 2003. p. 163-174. Karlsruhe, Germany. August, 2003

[8] Sensors and Sensor Networks, NSF Program Solicitation 04-522, http://www.cise.nsf.gov/funding/pgm_display.cfm?pub_id=9478&div=cns

[9] Defense Advanced Research Project Agency, http://www.darpa.mil/

[10] Center of Embedded Networked Sensors, http://www.cens.ucla.edu/default.htm

[11] Berkeley Wireless Embedded Systems (WEBS), http://webs.cs.berkeley.edu/

[12] Advanced Technologies and Applications for Next Generation Information Communication Network, http://www.ccrc.nthu.edu.tw/excellence/

[13] S. Deering and R. Hinden. Internet Protocol, Version 6 (IPv6) Specification. RFC 2460. December 1998

[14] Google. http://www.google.com.tw/

[15] P. Mockapetris. Domain Names--Implementation and Specification. RFC 1035. November 1987

[16] C. Cheng, R. Riley, S. Kumar, and J. Garcia-Lunes-Aceves. A Loop-Free Extended Bellman-Ford Routing Protocol Without Bouncing Effect. In Proceedings of ACM Sigcomm, pages 224--236, August 1989

[17] D. Waitzman, C. Partridge, and S. Deering, "Distance vector multicast routing protocol (DVMRP)," Internet Engineering Task Force (IETF), RFC 1075, November 1988

[18] L. Breslau, D. Estrin, K. Fall, S. Floyd, J. Heidemann, A. Helmy, P. Huang, S. McCanne, K. Varadhan, Y. Xu, H. Yu, "Advances in Network Simulation", IEEE Computer, vol. 33, No. 5, p. 5967, May 2000. http://citeseer.nj.nec.com/breslau00advances.html

[19] David M. Thompson, “Electromyography (EMG)”, http://moon.ouhsc.edu/dthompso/pk/emg/emg.htm

[20] Philip Levis, Sam Madden, David Gay, Joe Polastre, Robert Szewczyk, Alec Woo, Eric Brewer and David Culler, “The Emergence of Networking Abstractions and Techniques in TinyOS”, Proceedings of the First USENIX/ACM Symposium on Networked Systems Design and Implementation (NSDI 2004).

[21] Jason Hill, Robert Szewczyk, Alec Woo, Seth Hollar, David Culler, Kristofer Pister, “System architecture directions for network sensors”, . ASPLOS 2000, Cambridge, November 2000.

[22] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, T. Berners-Lee, “Hypertext Transfer Protocol -- HTTP/1.1”, Internet Engineering Task Force Request For Comments 2616, http://www.ietf.org/rfc/rfc2616.txt

[23] W. Heinzelman and A. Chandrakasan and H. Balakrishnan, “Energy-Efficient Communication Protocol for Wireless Microsensor Networks”, In Proccedings of the Hawaii Conference on System Sciences, January 2000

[24] Yan Yu, Ramesh Govindan and Deborah Estrin, “Geographical and Energy Aware Routing: A Recursive Data Dissemination Protocol for Wireless Sensor Networks”, UCLA Computer Science Department Technical Report UCLA/CSD-TR-01-0023, May 2001

[25] Stephanie Lindsey, Cauligi Raghavendra e Krishna Sivalingam, “Data Gathering in Sensor Networks using the Energy Delay Metric”, In International Workshop on Parallel and Distributed Computing Issues in Wireless Networks and Mobile Computing, (San Francisco, CA), Apr. 2001

[26] S. D. Servetto and G. Barrenechea., “Constrained Random Walks on Random Graphs: Routing Algorithms for Large Scale Wireless Sensor Networks.”, In Proceedings of the 1st ACM International Workshop on Wireless Sensor Networks and Applications (WSNA 2002), pages 12-21, 2002

[27] Wendi Rabiner Heinzelman, Joanna Kulik and Hari Balakrishnan, “Adaptive protocols for information dissemination in wireless sensor networks”, In Proceedings of the fifth annual ACM/IEEE international conference on Mobile computing and networking. Pages 174-185, August 15 - 19, 1999, Seattle, WA USA

[28] Joanna Kulik, Wendi Rabiner Heinzelman, and Hari Balakrishnan, “Negotiation-based Protocols for Disseminating Information in Wireless Sensor Networks”, ACM/IEEE Int. Conf. on Mobile Computing and Networking, Seattle, WA, Aug. 1999

[29] J. M. Kahn, R. H. Katz and K. S. J. Pister, “Mobile Networking for Smart Dust”, in the Proceedings of the ACM/IEEE Intl. Conf. on Mobile Computing and Networking (MobiCom99), Seattle, WA, August 17-19, 1999

[30] Chalermek Intanagonwiwat, Ramesh Govindan and Deborah Estrin, "Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Network", in Proceedings of the Sixth Annual International Conference on Mobile Computing and Networking (MobiCOM '00), August 2000, Boston, Massachusetts

[31] John Heidemann, Fabio Silva, and Deborah Estrin, "Matching Data Dissemination Algorithms to Application Requirements", in Proceedings of the ACM SenSys Conference, pp. 218-229. Los Angeles, California, USA, ACM. November, 2003.

[32] Seema Bandyopadhyay, Edward Coyle, "An Energy-Efficient Hierarchical Clustering Algorithm for Wireless Ad Hoc Networks", in Proceedings of the 22nd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM), 2003.

[33] Ossama Younis, Sonia Fahmy, "Distributed Clustering in Ad-hoc Sensor Networks: A Hybrid, Energy-Efficient Approach", in Proceedings of the 23nd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM), 2004.

[34] David Braginsky, Deborah Estrin, "Rumor Routing Algorithm for Sensor Networks", in Proc. of the 1st ACM International Workshop on Wireless Sensor Networks and Applications(WSNA), Atlanta, GA, September 2002.

[35] Yan Yu, Ramesh Govindan, and Deborah Estrin, "Geographical and Energy Aware Routing: a recursive data dissemination protocol for wireless sensor networks", Technical Report UCLA/CSD-TR-01-0023, UCLA Computer Science Dept., May 2001.

[36] Ananth Rao, Christos Papadimitriou, Scott Shenker, and Ion Stoica, "Geographic Routing without Location Information", in Proceedings of the 9th annual international conference on Mobile computing and networking, San Diego, CA, USA, September 14 - 19, 2003.

[37] Hyung Seok Kim, Tarek F. Abdelzaher, and Wook Hyun Kwon, "Minimum-Energy Asynchronous Dissemination to Mobile Sinks in Wireless

Sensor Networks", in Proceedings of the 1st international conference on Embedded networked sensor systems, Los Angeles, California, USA, November 05 - 07, 2003.

[38] CFan Ye, Haiyun Luo, Jerry Cheng, Songwu Lu, and Lixia Zhang, A Two-Tier "Data Dissemination Model for Large-scale Wireless Sensor Networks", in Proceedings of the 8th annual international conference on Mobile computing and networking, Atlanta, Georgia, USA, September 23 - 28, 2002.

[39] Chieh-Yih Wan, Andrew T. Campbell, and Lakshman Krishnamurthy, "PSFQ:a reliable transport protocol for wireless sensor networks", in Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications, WSNA 2002, September 2002.

[40] Sankarasubramaniam, Y., Akan, O.B., and Akyildiz, I.F., "ESRT: Event-to-Sink Reliable Transport in Wireless Sensor Networks", in Proc. ACM MobiHoc'03, Annapolis, Maryland, USA, June 2003.

[41] Omprakash Gnawali, Mark Yarvis, John Heidemann, and Ramesh Govindan, "Interaction of Retransmission, Blacklisting, and Routing Metrics for Reliability in Sensor Network Routing", in Proc. of the First IEEE International Conference on Sensor and Ad hoc Communications and Networks, SECON 2004.

[42] Sukun Kim, Fonseca, R., Culler, D., "Reliable transfer onwireless sensor networks", in Proc. of the First IEEE International Conference on Sensor and Ad hoc Communications and Networks, SECON 2004.

[43] Bret Hull, Kyle Jamieson, and Hari Balakrishnan, "Techniques for Mitigating Congestion in Sensor Networks", in Proc. ACM SenSys 2004, Baltimore, MD, November 2004.

[44] Chieh-Yi Wan, Shane B. Eisenman, and Andrew T. Campbell, "CODA: COngestion Detection and Avoidance in Sensor Networks", in Proc. of First ACM Conference on Embedded Networked Sensor Systems, SenSys 2003, pp. 266-279, Los Angeles, November 5-7, 2003.

[45] Yaling Yang; Kravets, R., "Distributed QoS Guarantees for Realtime Traffic in Ad Hoc Networks", in Proc. of the First IEEE International Conference on

Sensor and Ad hoc Communications and Networks, SECON 2004.

[46] Douglas De Couto, Daniel Aguayo, John Bicket, and Robert Morris, “A High-Throughput Path Metric for Multi-Hop Wireless Routing”, in Proceedings of the 9th annual international conference on Mobile computing and networking, MobiCom 2003, pp 134 - 146, September 2003.

[47] Baruch Awerbuch, David Holmer, and Herbert Rubens, “High throughput route selection in multi-rate ad hoc wireless networks, in Proc. of the Wireless On-Demand Network Systems”, First IFIP TC6 Working Conference, WONS 2004, January 21-23, 2004.

[48] Medidi, S.R. Vik, K., “QoS Aware Source-Initiated Ad-hoc Routing”, in Proc. of the First IEEE International Conference on Sensor and Ad hoc Communications and Networks; Santa Clara, CA, USA; October 2004.

[49] Budhaditya Deb, Sudeept Bhatnagar and Badri Nath, ReInForM: “Reliable Information Forwarding using Multiple Paths in Sensor Networks”, in 28th Annual conference on Local Computer Networks, LCN-2003.n 28th Annual conference on Local Computer Networks, LCN-2003.

[50] M. Degermark, A. Brodnik, S. Carlsson, and S. Pink, “Small forwarding tables for fast routing lookups,” in Proceeings of ACM SIGCOMM Conference (SIGCOMM), 10 1997, pp. 3–14.

[51] D. Estrin, R. Govindan, J. Heidemann, and S. Kumar, “Next century challenges: Scalable coordination in sensor networks,” in Proceedings of the Fifth Annual International Conference on Mobile Computing and Networks (MOBICOM), Seattle, 8, 1999.

[52] R. Zhang and Y. C. Hu, “Hyper: A hybrid approach to efficient content-based publish/subscribe,” in ICDCS, 2005.

[53] (2004) Micaz datasheet. Crossbow Technology. [Online]. Available: http://www.xbow.com/

[54] W. Weber, J. Rabaey, and E. Aarts, “Tinyos: An operating system for wireless sensor networks,” in Ambient Intelligence, 2005. [Online]. Available:

http://www.tinyos.net/

[55] J. Heidemann, F. Silva, C. Intanagonwiwat, R. Govindan, D. Estrin, and D. Ganesan, “Building efficient wireless sensor networks with low-level naming,” in ACM Symposium on Operating Systems Principles (SOSP), 2001.

[56] M. K. Aguilera, R. E. Strom, D. C. Sturman, M. Astley, and T. D. Chandra, “Filtering algorithms and implementation for very fast publish/subscribe systems,” in ACM SIGMOD, Santa Barbra, California, USA, 5 2001.

[57] F. Fabret, H. Arno, J. Kenneth, and A. Ross, “Matching events in a content-based subscription system,” in Proceedings of the eighteenth annual ACM symposium on Principles of distributed computing, 1999.

[58] A. Campailla, S. Chaki, E. Clarke, S. Jha, and H. Veith, “Efficient filtering in publish-subscribe systems using binary decision diagrams,” in ICSE, 2001.

[59] E. Ukkonen, “On-line construction of suffix trees,” in Algorithmica, 1995.

[60] S. Kurtz, T. Fakultat, and U. Bielefeld, “Reducing the space requirement for suffix trees,” Journal of Software-Practice and Experience, 1999.

[61] A. Carzaniga, M.J. Rutherford and A.L. Wolf. A Routing Scheme for Content-Based Networking. In proceedings of IEEE INFOCOM, March, 2004.

[62] Paolo Costa, Gian Pietro Picco. Dep di Elettronicae Informazione. Politecnico di Milano, Italy. “Publish/Subscribe on Sensor Networks: A Semi-probablistic Approach”. ICDCS, 2005.

[63] U. Farooq, S. Majumdar, E.W. Parsons. “Engineering Mobile Wireless Publish/Subscribe Systems for High Performance”. MASCOTS. 2004.

[64] Yongqiang Huang and Hector Garcia-Molina. “Publish/Subscribe in a Mobile Environment”. Workshop on Data engineering for Wireless and Mobile Access. 2001.

[65] Yuan Chen and Karsten Schwan, College of Computing, Georgia Institute of Technology, USA. “Opportunistic Overlays: Efficient Content Delivery in

Mobile Ad Hoc Networks."

[66] Knuth D.E., Morris Jr J.H., and Pratt V.R. Fast Pattern Matching in Strings. SIAM Journal on Computing 6(1):323-350 1977.

[67] K. Mehlhorn. Dynamic Binary Search. SIAM Journal on Computing 8(2):175-198 May 1979.

[68] P. Weiner. *Linear Pattern Matching Algorithms*. Proc. 14th IEEE Annual Symp. on Switching and Automata Theory, pp1-11, 1973

[69] Minglu Li, Xian-He Sun, Qianni Deng, Jun Ni. Grid and Cooperative Computing. Second International Workshop, pp. 26 - 33. GCC 2003, Shanhai, China, December 7-10, 2003.

[70] Joseph Polastre, Robert Szewczyk, and David Culler. Telos: Enabling Ultra-Low Power Wireless Research. ISPN 2005.

[71] IAR Workbench, embedded Development Tools from IAR Systems http://www.iar.com/.

[72] Gilman Tolle, Joseph Polastre, Robert Szewczyk, Neil Turner, Kevin Tu, Stephen Burgess, David Gay, Phil Buonadonna, Wei Hong, Todd Dawson, David Culler, "A Macroscope in the Redwoods," In Proceedings of the 3rd international Conference on Embedded Networked Sensor Systems (SenSys 2005), San Diego, California, USA, November, 2005.

[73] Robert Adler, Phil Buonadonna, Jasmeet Chhabra, Mick Flanigan, Lakshman Krishnamurthy, Nandakishore Kushalnagar, Lama Nachman, Mark Yarvis, "Design and Deployment of Industrial Sensor Networks: Experiences from the North Sea and a Semiconductor Plant," In Proceedings of the 3rd international Conference on Embedded Networked Sensor Systems (SenSys 2005), San Diego, California, USA, November, 2005.

[74] Iuliu Vasilescu, Keith Kotay, Daniela Rus, Peter Corke, Matthew Dunbabin, "Data Collection, Storage and Retrieval with an Underwater Optical and Acoustical Sensor Network," In *Proceedings of the 3rd international Conference on Embedded Networked Sensor Systems (SenSys 2005)*, San Diego, California,

USA, November, 2005.

[75] J. Polastre, R. Szewczyk, and D. Culler, “Telos: Enabling Ultra-Low Power Wireless Research,” In Proceedings of the Fourth International Symposium on Information Processing in Sensor Networks (IPSN), Special track on Platform Tools and Design Methods for Network Embedded Sensors (SPOTS), UCLA, Los Angeles, California, USA, April 2005.

[76] Hsing-Jung Huang; Ting-Hao Chang; Shu-Yu Hu; Polly Huang, “Magnetic Diffusion: Scalability, Reliability, and QoS of Data Dissemination Mechanisms for Wireless Sensor Networks,” Computer Communications, In Press, Mar. 2006

Conference Trip Report

Presenter:

Prof. Polly Huang, National Taiwan University

Conference Venue:

Workshop of Internet Measurement Technology and its Applications to Building Next Generation Internet in conjunction with IEEE SAINT 2007

Presentation Topic：

Path Selection Criteria for Peer-to-Peer Voice

Co-Authors:

Cheng-Ying Ou, Chia-Li Huang, Cheng-Chun Lou, Ming-Tsang Tsai, Kuan-Ta Chen, Polly Huang

I. Conference Attendance

IEEE SAINT 2007 is held in Hiroshima, Japan. The conference is a collective sessions of special workshops ranging from QoS issues in wireless networks to Internet measurement studies. The total attendance is approximately 200. The attendance of the workshop we participate in is about 40-50. There are 4 sessions in the workshop and in each session there is an invited speech. Prof. Polly Huang is invited to talk on “Path Selection Criteria for Peer-to-Peer Voice”. Other invited speakers includes Prof. Sue Moon of KAIST on “Automatic Signature and Behavior Capture for Accurate Traffic Classification” and Dr. Patrice Abry of Physics Lab., CNRS UMR 5672, Ecole Normale Supérieure de Lyon, France on “Statistical Sketch based Anomaly Detection and Validation using an Anomaly Database.”

II. Visit to Local Institutes

Prof. Polly Huang is subsequently invited to visit the institute of network and communication research of NEC and the CSIE department of 北九州大學. The visit is intense as well. The hosting faculty member organized a special half day workshop. Many of the network engineers/scientists in East Asia, particularly Japan, are invited to exchange research work and to seek collaboration opportunities.

III. Audience Feedback and Observations

The talk is well received. We presented a detailed examination of our own Sigcomm 2006 work. The talk draws attention. Our frank acknowledgement, based on the recent results, on the flaws of the Sigcomm 2006 work also draws smiles. The audience seems particular appreciate the very open sharing of our insights to our work on quantifying the user satisfaction of Skype. That work is considered one of the earliest results on the study of application-specific QoS. In addition to application-specific QoS, analysis of online activities in the Web2.0 era is also on the rise. Other classic issues such as traffic analysis, scheduling and control are still active and show a sign of slowing down. The trend seems to be going for more understanding of user perception and usage model.

IV. Results and Documentations

- IEEE SAINT 2007 Conference Proceedings, hard copy and the CDROM.
- Invitation to submit the extended work to IEICE Transaction on Communication (SCI). The paper is published as: Te-Yuan Huang, Kuan-Ta Chen, Polly Huang, Chin-Laung Lei, "**A Generalizable Methodology for Quantifying User Satisfaction**," *IEICE Transaction on Communication*, Invited Paper, Vol. E91-B, No. 5, May 2008