Intelligent Transmission of Patient Sensor Data in Wireless Hospital Networks

Danielle Bragg[1], Mira Yun, PhD[2], Haya Bragg[3], and Hyeong-Ah Choi, PhD[3]
[1] Princeton University, Princeton, NJ; [2]Wentworth Institute of Technology, Boston, MA;
[3]George Washington University, Washington, DC

Abstract

Medical data sensors on patients in hospitals produce an increasingly large volume of increasingly diverse real-time data. Because scheduling the transmission of this data through wireless hospital networks becomes a crucial problem, we propose a Reinforcement Learning-based queue management and scheduling scheme. In this scheme, we use a game-theoretical approach where patients compete for transmission resources by assigning different utility values to data packets. These utility functions are largely based on data criticality and deadline, which together determine the data's scheduling priority. Simulation results demonstrate the high performance of this scheme in comparison to a datatype-based scheme, with the drop rate of critical data as a performance measure. We also show how patients can optimize their policies based on the utility functions of competing patients.

Introduction

Many hospitals rely on wireless communication to monitor patients and alert doctors about their status. As hospitals grow in size and become more densely populated, the density of data transmitted through the hospital's wireless network grows proportionally. Furthermore, different types of data originating from patients with different medical conditions present a range of criticality, and also permit a range of acceptable latency for medical attention. Consequently, scheduling data transmission throughout the hospital network becomes a problem of great importance.

Hospitals increasingly rely on body area networks (BANs) of wireless sensors to monitor the status of patients[1,2]. These cyber-physical sensor networks must schedule the transmission of sensory data to a BAN controller node that aggregates and makes sense of the data gathered by the sensors. The BAN controller can then detect whether a significant event, like a heart attack or stroke, has occurred in the patient. Once an event has been detected, the hospital relies on a second layer of networking that allows the BAN controllers to alert a headquarters about the potentially life-threatening status of the patient. In this paper, we focus on this problem and provide a prioritized queue management and greedy scheduling scheme based on the criticality and deadline of patients' sensory data.

We can also analyze our system from a game theoretical perspective. In terms of game theory, each BAN controller acts as a player, acting on behalf of the corresponding patient. Each player is competing for networking resources, and attempts to maximize the throughput of raw data transmitted to the headquarters on the patient's behalf. Each player also attempts to minimize the amount of data dropped that is critical to the patient. Each player has a set of prioritized queues, and a policy consists of a strategy to place incoming data onto these queues. In this work, we propose a Reinforcement Learning (RL) protocol involving utility functions considering both data criticality and deadline to be employed by each player. Our proposed global scheduling policy, which defines the "rules" of the game, is also based in RL.

Related Work

Medical applications with wireless technologies are increasingly popular areas for development. Wireless networks provide many potential benefits in terms of net cost, user mobility, information accessibility, and network scalability. Consequently, many research and development teams are pursuing projects using wireless networking for medical applications including mobile healthcare, vital sign monitoring systems, and ambulatory data transmission[3,4].

Many powerful wireless medical technologies have been developed that would benefit from the type of throughput optimization protocols we present. For example, our protocols could help wireless sensor networks like Harvard's CodeBlue[5]. CodeBlue provides wireless sensors and accompanying software to track vital signs, and attempts to provide a wireless infrastructure for emergency and disaster medical care. Other systems involving patient monitoring and tracking could similarly benefit. For example, Johns Hopkins University's Advanced Health Disaster Aid Network (AID-N)[6] is an infrastructure-independent network that transmits patients' sensory data through an ad-hoc sensor network and EVDO wireless network. The European Commission has also developed a mobile healthcare project called MobiHealth[7] that uses BAN-based sensor networks and GPRS/UMTS wireless network technologies to measure and transmit bio signals, and thus facilitates full mobility of health monitoring systems. For e-health, Chen et al.[8] developed a vital sign monitoring system based on mobile telephony and internet infrastructure. Takizawa et al.[9] presented a different wireless monitoring system using IEEE 802.15.4a standard based on ultra wideband (UWB) communication. These and other ongoing research projects aim to improve the effectiveness and convenience of monitoring and treating patients by using wireless mobile and BAN sensor technologies.

As these healthcare-related services emerge, wireless medical networks must provide the capacity to support them. To do this, they must accommodate vast amounts of diverse real-time medical data. The major issues in transmitting medical data through wireless networks are managing large volumes of real-time sensory data and satisfying the diversity of quality of service (QoS) requirements for medical data[10]. To solve these problems, Zubairi[11] presented a data aggregation and reduction scheme to transmit large amounts of medical data efficiently through bandwidth-limited cellular networks such as 2G GSM or 3G UMTS. In addition, Moghaddam et al.[12] provided a QoS-aware congestion control scheme by considering the different QoS weights of different bio signals.

While these existing systems attempt to manage medical data, their scheduling policies largely ignore important properties of medical data like criticality and deadline. Maximizing throughput is important, but it is often more important in medical systems to maximize throughput of crucial data. The protocol we present in this work addresses this goal by considering both data criticality and deadline in its transmission scheme.

System Model and Problem Formulation

We model the network within the hospital as a two-tiered hierarchy. Figure 1 displays an overview of the hospital network.

The first level of networking consists of BANs, each of which is associated with a particular patient. We consider a BAN architecture where each BAN consists of a number of BAN devices and a patient data controller (PDC). There are two types of BAN devices: implant devices and wearable devices, which will communicate with the PDC directly or via multi-hop. The PDC processes the data it receives and identifies the occurrence of meaningful medical events. It also relays raw data back to a headquarters.

Implant devices mainly refer to Implantable Medical Devices (IMDs) such as pacemakers, implantable cardiac defibrillators (ICDs), drug delivery systems, and neurostimulators. These IMDs are "smart" biosensors/actuators placed inside the body that communicate wirelessly with external control devices to automatically monitor and treat physiological conditions to manage a broad range of ailments. For example, the glucose level in the blood can be sensed by implanted glucose sensors. Based on these readings, an insulin controller can then dispense the appropriate amount of insulin into the blood for diabetes patients.

Wearable devices far exceed implanted ones in both quantity and heterogeneity. Wearable devices include sensors monitoring the cardiovascular system, for example electrodes on the chest for capturing ECG, Peizo sensors on the wrist to measure blood pressure, optical sensors on the toe and earlobe to measure pulse rate, and microphones on the chest to measure heart sounds. Motion sensors placed about the body, small cameras or video cameras attached to sunglasses, and radars attached to clothes to assist visually-disabled persons provide examples of other types of external data-gathering devices.

The second level of networking is the network that runs throughout the hospital, and it consists of the PDCs associated with the patients in communication with a headquarters. This centralized headquarters might represent a doctor's pager or some centralized command station in the hospital. Organizing and scheduling data transmission between the PDCs and the centralized headquarters so that the patients receive the necessary care is a focus of this work.

Our problem is to minimize critical data dropped during the transmission of sensory data from PDCs to the headquarters.

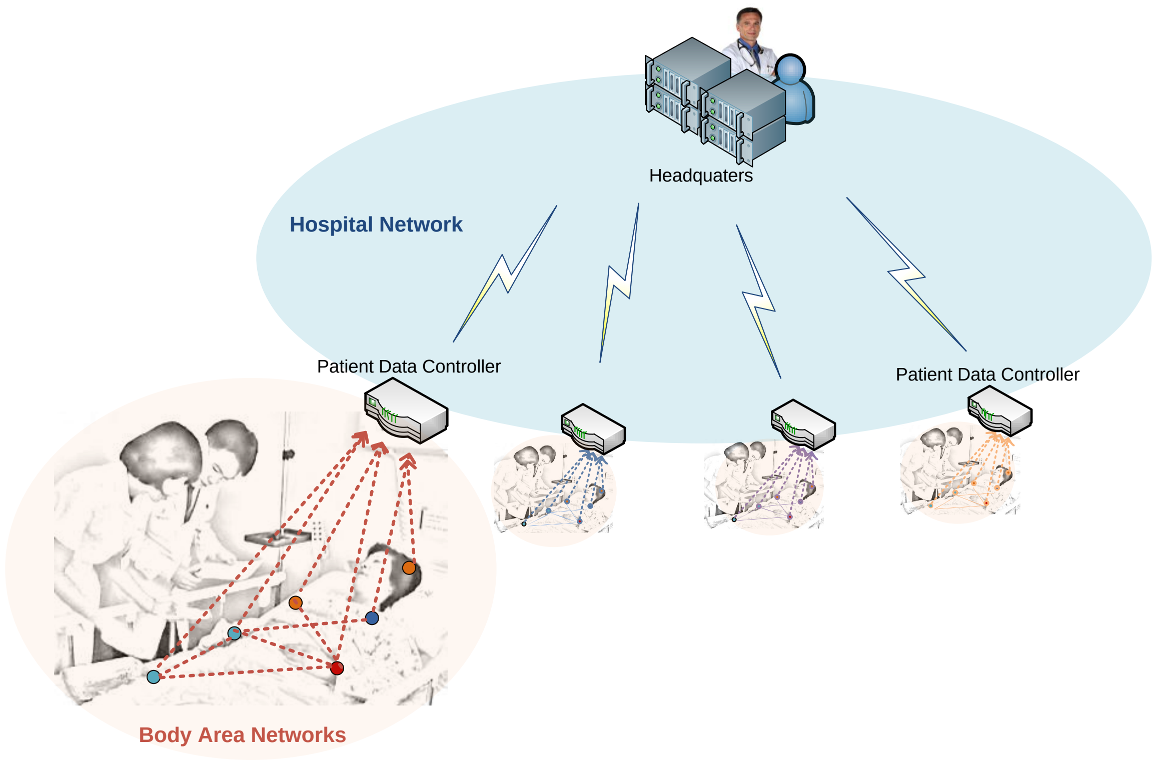

Figure 1: Hospital Network Architecture

Given:

- a graph $G(V, E)$ representing the connections between PDCs and the headquarters
- incoming data packets, each packet p with a criticality index c_p and maximal latency d_p
- a set of prioritized data queues Q for each PDC, such that higher priority queues must be emptied before data from lower priority queues are transmitted

Define: a strategy for each PDC to sort data into its queues and a global scheduling scheme to minimize the amount of critical data dropped.

Proposed Protocol

The main focus of this work is the transmission of raw data from PDCs to the headquarters. Our approach consists of two main steps: 1) the placement of BAN sensory data packets into queues at the PDCs, and 2) the scheduling of data transmission from the PDCs to the headquarters. Intuitively, transmission of critical data should be prioritized over less critical data. Similarly, highly time-sensitive data should be prioritized over less urgent data. To allow for appropriate prioritization, we sort data packages into prioritized queues according to an RL protocol. An integral component of this protocol is a utility function used to quantify the urgency of data based on the data's criticality and transmission deadline. Our centralized scheduler then selects the data package with the largest weight across all PDCs to transmit next.

We note that criticality and time-sensitivity are not the same. As an example of critical data that is not time-sensitive, consider a patient with a lethal reaction to a medication that would take weeks to actually kill him. As an example of time-sensitive data that is not critical, consider a mild reaction to a medication that causes immediate discomfort but not danger. Of course, doctors also frequently face data that is both critical and time-sensitive, or not critical and not time-sensitive.

1) Queue Placement: To place incoming data onto queues at the PDCs, we rely on RL, a branch of Machine Learning. An RL-based approach is appropriate here because it allows for long-term optimization in a dynamic environment. An RL system is typically characterized by a set of states, a set of actions, an expected immediate cost (or reward) for taking each action, and a value function for each state reflecting the expected long-term cost (or reward). At each time step, the system observes the state that it is in, and selects an action to take based on the immediate cost and expected long-term cost. It finds itself in a new state, observes the impact that the action has made, and updates its expected long-term costs. In our system, we define each of these components as follows:

- Set of states S: the set of possible preferences for an incoming packet. We say that a PDC node is in a particular state if it is in the state of preferring a particular queue for incoming data. This model is practical because incoming data is often similar to the other data generated around the same time.
- Set of actions A: the set of possible incoming packet placements. Each PDC has K possible actions, where K is the number of queues it supports. An action consists of placing an incoming packet in one of these queues.
- Cost function $r(s, a)$: the immediate cost of placing an incoming packet according to action a from state s. Each action is associated with a particular cost. For example, placing a highly critical piece of data at the end of a long queue can be seen as quite costly, since transmitting all preceding data will cost quite a bit of time and system resources.
- Value function $V(s)$: the expected long-term reward for state s. This value reflects the payoff that one can expect to incur throughout the running of the algorithm, given that we have entered state s. This value function is continuously updated as the algorithm runs and gathers more information about the system.

At each iteration of the algorithm, a packet is placed in a queue, a new state is entered, and the value function V for the previous state is updated based on the reward incurred. We refer to the state at iteration n as s_n, so that taking an action transitions the node to state s_{n+1}, and the next iteration begins.

Utility Functions: Each time a packet arrives at a PDC, it must be placed onto a queue in such a way as to optimize the system performance. Intuitively, "good" performance allows for packets with higher urgency to be transmitted before those with lower urgency. Our use of multiple prioritized queues allows for greater flexibility, and potentially increased performance. To represent the urgency of the transmission of a data package from a PDC to the headquarters, we use utility functions, similar to Time-Utility Functions (TUFs)[13]. Each data package is assigned a utility value determined by the appropriate utility function.

Let p be the arriving packet, with criticality c_p and maximal latency d_p. Let t_0 be its time of generation, and t be the current time. Then p is assigned a utility value according to the following function:

$$U(p) = c_p^{\beta} (\frac{t - t_0}{d_p})^{\zeta} \tag{1}$$

In Equation 1, β and ζ are constants. A larger value of β will place a greater emphasis on the criticality of incoming packets, and a larger value of ζ will place an increased value on the time that has passed in relation to the time until the packet expires and is dropped from the queue. Each PDC sets its values of β and ζ, and in this way chooses the policy it employs in competition for system recourses. The utility values of packets in the queues are periodically recomputed as time passes.

Immediate Cost: Based largely on the utility value assigned to the packet, we compute the immediate cost associated with the action of assigning an incoming packet to a particular queue. More specifically, we define the immediate cost of choosing action a when in state s to be

$$r(s,a) = \frac{\sum_{p} U(p)R(p)}{Q(n)} \tag{2}$$

$R(p)$ is the absolute ranking of packet p at the node, considering that all packets in higher prioritized queues have higher rankings, in addition to higher ranking packets in the same queue. Multiplying $U(p)$ by $R(p)$ weights each packet by both its utility, or urgency, and its placement in the queues, so that packets with large utility values contribute more, as do packets located towards the bottom of the series of queues. $Q(n)$ is the number of packets in queues at node n, and dividing by this value allows the cost function to represent a weighted average over all packets. The design of this reward function is based on the work of Zhou et al[14].

Policy Selection: Once the immediate cost of all possible actions has been established, the system must choose which action to take, or equivalently it must select the queue into which the incoming packet will be placed. In RL, the rules by which a system determines which action to take, given the system state, is called a policy. In our system, the policy π that the system follows is dynamically updated as the system learns from the actions it takes. The queue q^{π} that the system selects at iteration n of the algorithm according to policy π is:

$$q^{\pi} = \underset{a \in A}{\text{argmin}}\, E\{r(s_n, a) + \gamma V(s_{n+1})\} \tag{3}$$

In this formula, s_n represents the current state at iteration n. All possible actions a are examined, and s_{n+1} represents the state into which that action would bring the system. In RL, $0 \leq \gamma \leq 1$ is called the "discount rate," and is a parameter that reflects the importance of the value of the long-term expected reward in comparison to the immediate reward[15]. According to this formula, the system selects the action, or queue, that minimizes the combined immediate and long-term expected costs.

Updating the Value Function: Once the system takes an action determined by policy π described in Equation 3, the system transitions from state s_n to state s_{n+1} and incurs some cost. The value function must be updated accordingly. To do this, we employ a Temporal Difference (TD) technique. Specifically, we use TD(0), which is a one-lookahead protocol[15]. It only considers the expected reward of the next immediate state, in addition to the cost incurred from the transition to this state, according to the following formula:

$$V(s_n) = V(s_n) + \alpha_n[r_n + \gamma V(s_{n+1}) - V(s_n)] \tag{4}$$

Intuitively, this equation updates the cost expected to be incurred from a state s_n by considering the return expected from the subsequent state s_{n+1}. We cannot simply update $V(s_n) = V(s_{n+1})$ since at other times s_n might transition to other states through other actions, so TD provides a weighted update according to weight α_n. Thus, $0 \leq \alpha_n \leq 1$ is the "step-size parameter," and allows for the value function to be a weighted average of the expected returns of subsequent states.

2) Scheduling: Finally, the headquarters invokes a centralized scheduler to decide which data package is transmitted at every time unit. The rules of the scheduler define the game theoretical space in which the PDCs compete for transmission resources. We define this centralized algorithm to act by examining the following weight function for each PDC node:

$$w(n,t) = V(s)Q(n)C(i,t) \sum_{p \in T} c_p \tag{5}$$

In Equation 5, t is the current time, n is the node under examination, s is the current state of the node, and T is the set of packets that can be transmitted in the next scheduling period by node n. $C(i,t)$ is the channel rate of channel i at time t. The centralized protocol selects the data package at the node with the highest weight, indicating that it is the highest priority, and transmits that piece of data. Our system consists of a tree of depth one, with the leaf nodes (the PDCs) transmitting to the root node (the hospital headquarters), and can only transmit data from a single node at a time. In a network with $c > 1$ communication channels, it would theoretically select the c data packages at the head of queues with the highest utility values to transmit at any given time.

Analysis

We run simulations with 10 patients, each with a corresponding PDC. The topology of our network is still a tree of depth one, as shown in Figure 1, where each leaf node is a PDC, and the root is the hospital headquarters. Each leaf node streams data to the root node. To highlight the performance of our scheduling protocol, our system uses a single channel for data transmission.

An average of 8900 bps can be expected to be generated by sensors on a patient monitoring vital signs[16]. These vital data types include EKG, blood pressure, pulse, respiration (tidal volume), and SpO2 (blood oxygenation). In our simulations, an average of $8900 + \delta$ bits of data are generated per second by each patient. We accommodate δ additional bits per second to account for other data that the doctors might want, including large data types like pictures and videos.

Our simulation settings were chosen in accordance with this model of data generation. In our simulations, packets are of uniform size. Queue assignment takes place immediately for every incoming packet, but data transmission is scheduled every 5 time slots. Our simulations run for 10,000 time slots, and each simulation result is the average of 10 runs. Packets are of uniform size; the distribution of data criticality is Normal $N(3, 1)$; and the distribution of the data deadlines is uniform $[25, 75]$. If data channel rate is 1-2 Mbps, as in 802.11b WiFi, then our simulation settings correspond to a data generation rate of up to 100 Kbps for each patient. The discount factor γ is 0.8, and $\alpha_n = \frac{1}{n}$.

We evaluate our system in terms of the sum of the criticality indexes of all packets dropped, divided by the sum of the criticality indexes of all packets. We call this the Criticality-Weighted Drop Rate (CWDR). Consequently, a system that drops a given number of packets with higher criticality indexes is evaluated to perform worse than another system that drops the same number of packets with lower criticality indexes, given identical patient data. Similarly, a system that drops few packets with high criticality might perform worse than another system that drops more packets with lower criticality indexes.

More specifically, let D be the set of dropped packets, and P be the set of all packets dropped or transmitted. Then the CWDR of a network N is expressed as follows:

$$\text{CWDR}(N) = \frac{\sum\limits_{d \in D} c_d}{\sum\limits_{p \in P} c_p} \tag{6}$$

Comparison to Naive Protocol: To exemplify the strength of our solution, we compare our RL-based protocol to a naive solution. In the naive solution, there are several classes of data, such as pictures, video, or numerical data, each with different QoS requirements from the network but with identical distributions of criticality and deadline. Data is sorted into queues according to its class. Data is scheduled for transmission according to these classes as well. Data from the first class of queues is always transmitted before data from the second class of queues, and so on. The improvement that our RL-based protocol has over this naive protocol is demonstrated in Figure 2, which was generated by simulations run with inter-arrival time (IAT) of 1.0, corresponding to a relatively high data generation rate for the network. The improvement in performance is particularly notable for a high number of queues, which is to be expected because a larger number of queues allows the RL-based assignment of packets to queues to have an increased effect. The RL-based weight function used for packet scheduling also contributes to the performance by considering criticality in scheduling, rather than relying solely upon QoS requirements for the packet's data type. Because our RL-based assignment considers both packet criticality and deadline, it shows up to 25% improvement over the naive solution, as shown in Figure 2.

Varied IAT and Queue Number: We also ran a set of simulations to demonstrate the effect of the number of queues and IAT on the drop rate. We define the IAT to be the average expected time between data packet arrivals at a single

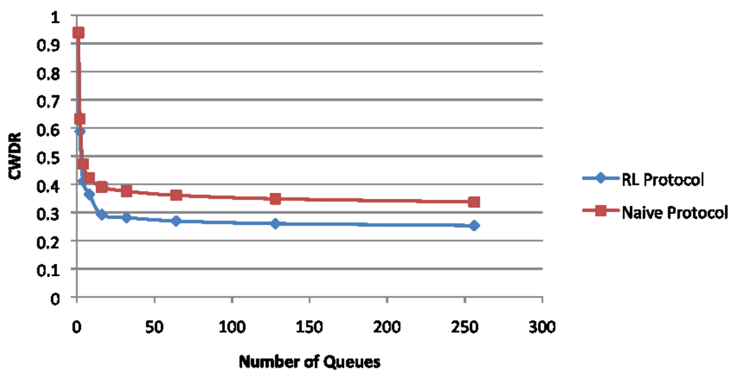

Figure 2: Impact of IAT and Queue Number on CWDR

PDC. The inter-arrival time for a patient can easily be computed based on the patient's data generation rate. A smaller IAT corresponds to a larger data generation rate, and a larger IAT indicates a smaller data generation rate. The results are displayed in Figure 3. As IAT increases, our protocol performs better. This performance improvement stems directly from the fact that an increased IAT corresponds to a lower average data generation rate. Our protocol must handle fewer data packets, and so a lower CWDR can be expected.

Similarly, when the queue number is increased, performance improves. Again, this increased performance stems from the RL-based queue placement protocol that considers both data criticality and deadline. We also note that as the number of queues approaches infinity, the improvement in system performance decreases. This makes sense since each additional queue introduces a smaller fraction of increased resources for our protocol to use. In other words, the increase in queue resources between 1 queue and 2 queues is 100%, while the increase between 100 queues and 101 is 1%, leading to diminishing returns in performance improvement.

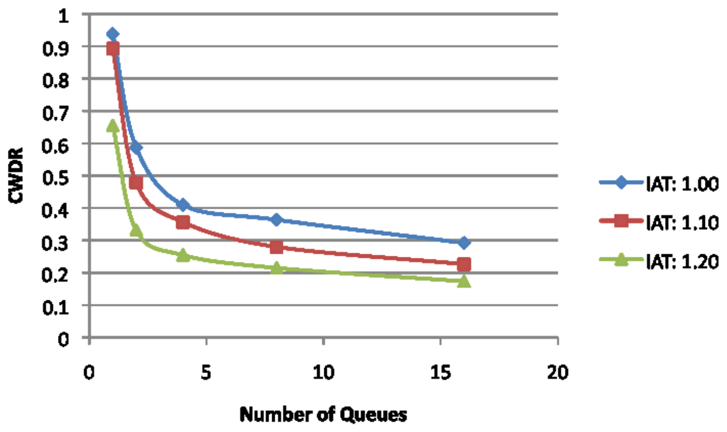

Figure 3: Impact of IAT and Queue Number on CWDR

Competing Strategies: Our next set of simulations explores different combinations of strategies, defined by the BAN controller choices of ζ and β of its utility function $U(p)$. To explore the effect of different combinations of competing strategies on the service to individual patients, we divide the group of patients into two sets of equal size, and assign a strategy to each set separately. This type of approach might be appropriate in a medical scenario where there are two distinct types of patients whose data allows for different drop rates, or CWDRs.

In our simulations, we divide the group of 10 patients into two sets of 5, and assign a strategy to each set separately. This yields a total of $2^4 = 16$ combinations of strategies. The average CWDR for each group and scenario are presented in a payoff matrix in Table 1. We also present the average CWDR over all patients in Table 2.

From Table 1, we see that there is a Nash equilibrium when both sets of patients set $\zeta = 2$ and $\beta = 2$, meaning that neither set can achieve improved performance by choosing another strategy, given that the other set has set $\zeta = 2$ and $\beta = 2$. This table also shows that the best CWDR for an individual set is achieved when one uses $\zeta = 2$ and $\beta = 2$ while the other uses $\zeta = 2$ and $\beta = 1$. Such a setting might be useful if one set of patients is producing data that cannot afford to be dropped, while another produces data that can be dropped with less risk to the patients.

Table 2 presents the average CWDR over both groups. It shows that the system as a whole achieves the best performance when all patient data collectors assign $\zeta = 1$ and $\beta = 2$, $\zeta = 2$ and $\beta = 1$, or $\zeta = 2$ and $\beta = 2$. Similarly, it achieves the worst performance when one set of patients assigns $\zeta = 1$ and $\beta = 1$ while the other assigns $\zeta = 2$ and $\beta = 1$.

Strategy: (ζ, β)	(1,1)	(1,2)	(2,1)	(2,2)
(1,1)	0.42, 0.42	0.49, 0.37	0.37, 0.44	0.40, 0.35
(1,2)	0.37, 0.49	0.36, 0.36	0.47, 0.36	0.39, 0.45
(2,1)	0.44, 0.37	0.36, 0.47	0.36, 0.36	0.43, 0.31
(2,2)	0.35, 0.40	0.45, 0.39	0.31, 0.43	0.36, 0.36

Table 1: Payoff matrix showing CWDR for competing patient groups

Strategy: (ζ, β)	(1,1)	(1,2)	(2,1)	(2,2)
(1,1)	0.42	0.43	0.40	0.38
(1,2)	0.43	0.36	0.41	0.42
(2,1)	0.40	0.41	0.36	0.37
(2,2)	0.38	0.42	0.37	0.36

Table 2: Average CWDR over all competing patients

Conclusion

In this paper, we consider the problem of medical sensor data transmission in wireless hospital networks, and present a novel consideration of criticality and deadline in utility functions representing scheduling preference. By learning from experience, our system has the ability to adapt to changes in dynamic environments. We also introduce a novel perspective for this problem of medical data transmission management by analyzing our system in terms of game theory, where patients compete for resources.

Through our work, we provide a solid theoretical infrastructure and simulated results. Our results demonstrate significant improvement over a traditional naive protocol, and allow for individualized patient service. In the future, we would like to analyze system performance on raw medical data, and tune the system parameters accordingly. Our work is also easily extendable to incorporate event detection through the use of sensory signatures. Each medical event can be identified by the presence of a particular set of sensory data. Once the event has been identified, the criticality of the data can be adjusted accordingly. The development and use of this type of intelligent medical data transmission protocol would enhance existing medical networks, and greatly improve patient care.

References

1. Cao H, Leung V, Chow C and Chan H. Enabling technologies for wireless body area networks: A survey and outlook. *IEEE Communications Magazine*, 47:84–93, 2009.

2. Ren Y, Pazzi N and Boukerche A. Monitoring patients via a secure and mobile healthcare system. *IEEE Wireless Communications*, 17:59–65, 2010.

3. Shobha G, Chittal RR and Kumar K. In *Second International Conference on Systems and Networks Communications*.

4. Ullah S, Khan P, Ullah N, Saleem S, Higgins H and Kwak KS. A review of wireless body area networks for medical applications. *International Journal of Computer Science and Network Security*, 2:797–803, 2009.

5. Malan D, Fulford-Jones T, Welsh M and Moulton S. An ad hoc sensor network infrastructure for emergency medical care. In *Proceedings of MobiSys Workshop on Applications of Mobile Embedded Systems*, pages 12–14, 2004.

6. Gao T, Massey T, Selavo L, Crawford D, Chen B, Lorincz K et al. The advanced health and disaster aid network: a light-weight wireless medical system for triage. *IEEE Transactions on Biomedical Circuits and Systems*, 1: 203–216, 2007.

7. van Halteren AT, Bults RA, Wac KE, Konstantas D, Widya IA, Dokovski NT et al. Mobile patient monitoring: The mobihealth system. *Journal on Information Technology in Healthcare*, 2:365–373, 2004.

8. Chen W, Wei D, Zhu X, Uchida M, Ding S and Cohen M. A mobile phone-based wearable vital signs monitoring system. In *The Fifth International Conference on Computer and Information Technology*, pages 950–955, 2005.

9. Takizawa K, Huan-Bang , Kiyoshi L and Kohno HR. Wireless vital sign monitoring using ultra wideband-based personal area networks. In *IEEE Annual International Conference on Engineering in Medicine and Biology Society*, pages 1798–1801, 2007.

10. Skorin-Kapov L and Matijasevic M. *International Journal of Telemedicine and Applications*.

11. Zubairi JA. Aggregation scheme implementation for vital signs data over the network. In *Sixth International Symposium on High-Capacity Optical Networks and Enabling Technologies*, pages 129–133, 2009.

12. Moghaddam MHY and Adjeroh D. A novel congestion control protocol for vital signs monitoring in wireless biomedical sensor networks. In *IEEE Wireless Communications and Networking Conference*, pages 1–6, 2010.

13. Garroppo RG, Giordano S, Iacono D and Tavanti L. Game theory and time utility functions for a radio aware scheduling algorithm for WiMAX networks. *Wireless Networks*, 17:1441–1459, 2011.

14. Zhou Y, Yun M, Kim T, Arora A and Choi HA. RL-based queue management for QoS support in multi-channel multi-radio mesh networks. In *IEEE Eighth International Symposium on Network Computing and Applications*, pages 306–309, 2009.

15. Sutton RS and Barto AG. *Reinforcement Learning*, volume 9. MIT Press, Cambridge, MA, 1998.

16. Zubairi JA and Misbahuddin S. Ambulatory data aggregation and reduction for transmission over limited bandwidth network. In *International Symposium on Collaborative Technologies and Systems*, pages 356–360, 2009.