Adaptive Medical Feature Extraction for Resource Constrained Distributed Embedded Systems

Roozbeh Jafari[†], Hyduke Noshadi[††], Soheil Ghiasi[‡] and Majid Sarrafzadeh[‡‡]
Computer Science Department [†] [††] [‡‡]
University of California, Los Angeles
Los Angeles, CA 90095
Department of Electrical and Computer Engineering [‡]
University of California, Davis
Davis, CA 95616
{rjafari[†], majid[‡‡]}@cs.ucla.edu, nhyduke[††]@ucla.edu, soheil[‡]@ece.ucdavis.edu

***Abstract*— Tiny embedded systems have not been an ideal outfit for high performance computing due to their constrained resources. Limitations in processing power, battery life, communication bandwidth and memory constrain the applicability of existing complex medical/biological analysis algorithms to such platforms. Electrocardiogram (ECG) analysis resembles such algorithm. In this paper, we address the issue of partitioning an ECG analysis algorithm while the wireless communication power consumption is minimized. Considering the orientation of the ECG leads, we devise a technique to perform preprocessing and pattern recognition locally on small embedded systems attached to the leads. The features detected in pattern recognition phase are considered for classification. Ideally, if the features detected for each heart beat reside in a single processing node, the transmission will be unnecessary. Otherwise, to perform classification, the features must be gathered on a local node and thus, the communication is inevitable. We perform such feature grouping by modeling the problem with a hypergraph and applying partitioning schemes. This yields a significant power saving in wireless communication. Furthermore, we utilize dynamic reconfiguration by software module migration. This technique with respect to partitioning enhances the overall power saving in such systems. Moreover, it adaptively alters the system configuration in various environments and on different patients. We evaluate the effectiveness of our proposed techniques on MIT/BIH benchmarks.**

I. Introduction

The electrocardiogram (ECG) is the record of variation of bioelectric potential with respect to time as the human heart beats. Due to its ease of use and non-invasiveness, ECG play an important role in patient monitoring and diagnosis. Multichannel electrocardiogram (ECG) data provide cardiologists with essential information to diagnose heart disease in a patient. Our primary objective is to develop an ambulatory ECG analysis algorithm with real-time diagnosis functions for wearable computers. ECG analysis algorithms has always been a very difficult task in the realization of computer aided ECG diagnosis. Implementation of such algorithms become even harder for small and mobile embedded systems that meet the given latency requirements while minimizing overall energy dissipation for the system. Distributed embedded systems are now being successfully deployed in various wearable computers. Distributed architectures have been developed for cooperative detection, scalable data transport, and other capabilities and services. However, the complexity of algorithms running on the resource constraint systems has introduced a new set of challenges associated with resource constrained devices and their energy concerns. These obstacles may dramatically reduce the effectiveness of embedded distributed algorithms. Thus, a new distributed, embedded, computing attribute, dynamically reconfigurable, must be developed and provided to such systems. Reconfiguration capability may adaptively alter the system configuration to accommodate the objectives and meet the constraints for highly dynamic systems.

II. Related Work

Several "wearable" technologies exist to continually monitor patient's vital signs, utilizing low cost, well-established disposable sensors such as blood oxygen finger clips and electrocardiogram electrodes.The Smart Shirt from Sensatex [1] is a wearable health monitoring device that integrates a number of sensory devices onto the Wearable Motherboard from Georgia Tech [2]. Several other technologies have been introduced by MIT called MIThril [3], e-Textile from Carnegie Mellon University [4], Wearable e-Textile from Virginia Tech [5], CustoMed [6] and RFab-Vest [7] from UCLA. The Lifeguard project being conducted at Stanford University is a physiological monitoring system comprised of physiological sensors (ECG/Respiration electrodes, Pulse Oximeter, Blood Pressure Monitor, Temperature probe), a wearable device with built-in accelerometers (CPOD), and a base station (Pocket PC). The CPOD acquires and logs the physiological parameters measured by the sensors [8]. The Assisted Cognition Project conducted at the University of Washingtons Department of Computer Science is exploring the use of AI systems to support and enhance the independence and quality of life of Alzheimers patients. Assisted Cognition systems use ubiquitous computing and artificial intelligence technology to replace some of the memory and problem-solving abilities that have been lost by an Alzheimers patient [9]. Nevertheless, none of the above projects/systems supports

Proceedings of the Fourth Annual IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOMW'06)
0-7695-2520-2/06 $20.00 © 2006 IEEE

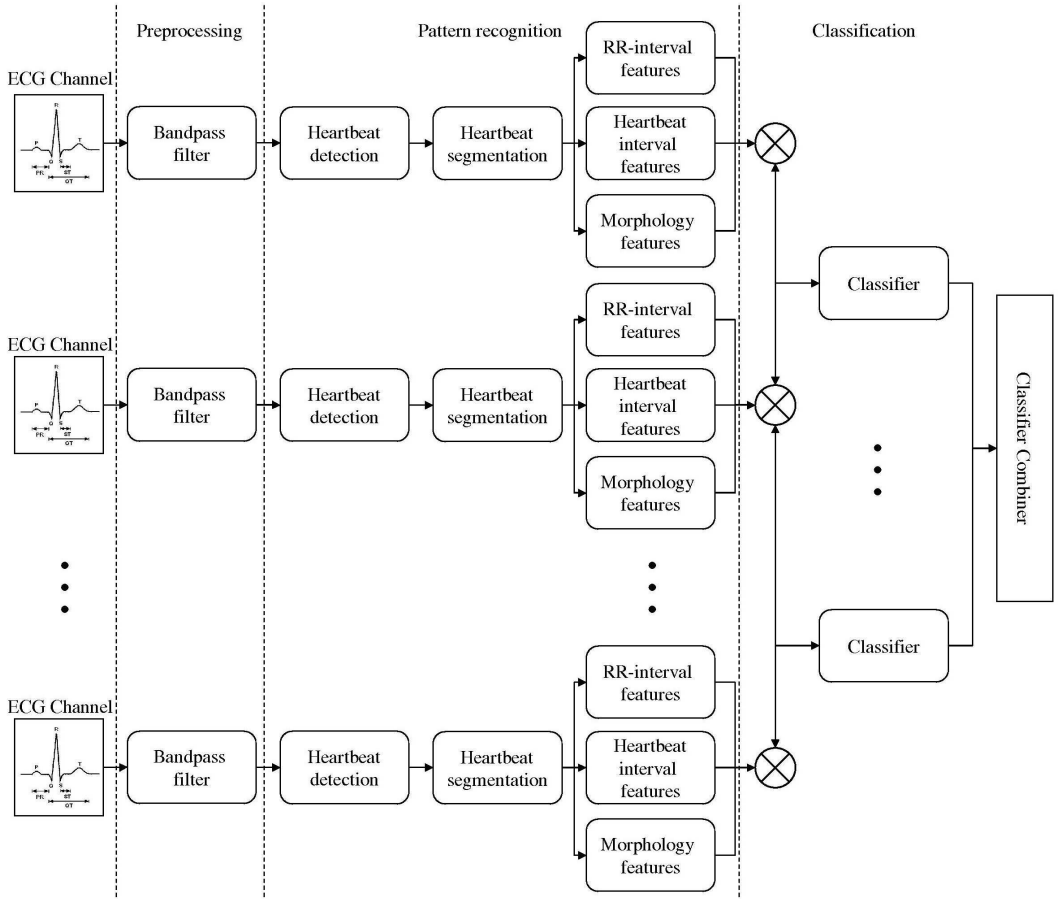

Fig. 1. ECG Analysis Schematic

the concept of scalability and adapting complex processing algorithms.

III. Automated Feature Set Detection

Given the goal of classifying objects based on their attributes, the functionality of an automated pattern recognition system can be divided into two basic tasks: the description task generates attributes of an object using feature extraction techniques, and the classification task assigns a group label to the object based on those attributes with a classifier.

There are two different approaches for implementing a pattern recognition system: statistical and structural. Each approach utilizes different schemes within the description and classification tasks which incorporates a pattern recognition system. Statistical pattern recognition [10][11] concludes from statistical decision theory to discriminate among data from different groups based upon quantitative features of the data. The quantitative nature of statistical pattern recognition, however, makes it difficult to discriminate among groups based on the morphological (i.e., shape-based or structural) subpatterns and their interrelationships embedded within the data. This limitation provided the impetus for the development of a structural approach to pattern recognition.

Structural pattern recognition [12][13] relies on syntactic grammars to discriminate among data from different groups based upon the morphological interrelationships (or interconnections) present within the data. Structural pattern recognition systems have proven to be effective to image data as well as time-series data.

We have ported an accurate ECG processing algorithm based on structural pattern recognition onto our processing units (dot-motes) [14]. The algorithm consists of three stages: preprocessing, pattern recognition and classification. We perform the preprocessing and pattern recognition locally and within close proximity to the ECG leads. The preprocessing includes filtering while the pattern recognition includes heartbeat detection (through the QRS complex detection), segmentation as well as feature extraction. Once the features are extracted, they will be processed for classification.

The filtering is performed by finite impulse response (FIR) filters with cut-off frequencies of 5-200 Hz. The heartbeat detection is implemented with a QRS detector based on the algorithm of Pan and Tompkins [15] with some improvements that employs slope information. The scheme proposed by

Proceedings of the Fourth Annual IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOMW'06)
0-7695-2520-2/06 $20.00 © 2006 IEEE

Laguna et al [16] was used to extract the fiducial points. Consequently, features relating to heartbeat intervals and ECG morphology were calculated for each heartbeat. The list of features is included in Table I and are based on [17] with minor additions.

We extracted a total of 17 features from the ECG signals,

Group label	Features
RR Intervals	Pre-RR interval
	Post-RR interval
	Average-RR interval
	Local Average-RR interval
Heartbeat Intervals	QRS duration
	T wave duration
	PR duration
	ST duration
	QT duration
Morphology 1A	ECG Morphology (10 samples) between Q and S
Morphology 1B	Normalized ECG Morphology (10 samples) between Q and S
Morphology 2A	ECG Morphology (10 samples) between S and T wave offset
Morphology 2B	Normalized ECG Morphology (10 samples) between S and T wave offset
Morphology 3A	ECG Morphology (9 samples) between R-50ms and R+100ms
Morphology 3B	Normalized ECG Morphology (9 samples) between R-50ms and R+100ms
Morphology 4A	ECG Morphology (8 samples) between R+150ms and R+500ms
Morphology 4B	Normalized ECG Morphology (8 samples) between R+150ms and R+500ms

TABLE I

FEATURES CATEGORIZED BY GROUPS THAT CONSIDERED IN THE STUDY

and each derives from one of the groups in Table I. Our objective is to minimize the communication among processing nodes before the classification phase, therefore, this study does not investigate the problem of classification and we did not implement a classifier for our platform. However, any classifier suitable for suitable for small embedded systems may be deployed.

IV. SOFTWARE PROFILING

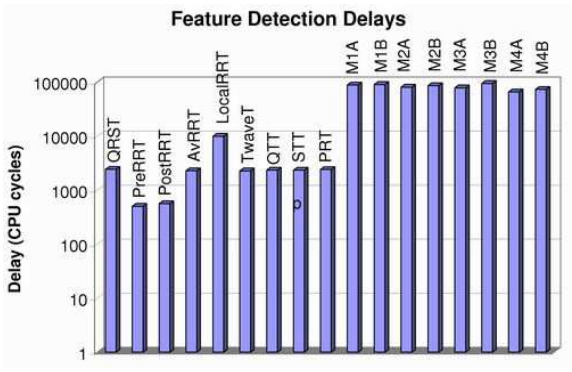

Fig. 2. ECG Feature Detection Delays Extracted in Profiling Phase

To measure the execution delay of our heartbeat detection and feature extraction system, we used Avrora [18], a microcontroller simulator framework developed at UCLA. For our experiments, we implemented a program monitor on Avrora that generates a control flow graph (CFG) while measuring the execution frequency and delay of each basic block. Function delay is measured as the duration when execution enters a function to when execution exits. Calls to other functions are accounted for, while interrupts are not. Function delays measured are represented in Figure 2.

V. TARGET ARCHITECTURE MODEL

Networked sensor nodes containing small, often battery-powered embedded computers can densely sample phenomena that were previously difficult or costly to observe. Sensor nodes can be placed anywhere on a patients body [6] [7]. Due to the mobility of such systems, wireless sensor networks are expected to be both autonomous and long-lived, surviving environmental hardships while conserving energy as much as possible. It is well-known that the amount of energy consumed for a single wireless communication of one bit can be many orders of magnitude greater than the energy required for a single local computation. Thus, we focus our analysis to the energy used for wireless communication. In our model, since all the node are placed within close proximity of each other, we assume they communicate directly and multi-hop communication is not required. Therefore, the total energy consumed for in-network communication is:

$$\varepsilon(n) = b(n) \times e(n) \quad (1)$$

where $b(n)$ is the number of packets transmitted and $e(n)$ is the average amount of energy required to transmit one packet. In our design, we consider Collision Free Model (CFM) which simplifies the programming by abstracting all details of low level channel contention and packet collision away from the algorithm designers. By abstracting reliable communication as an atomic operation, programming based on CFM bears resemblance to existing algorithm design in parallel and distributed computation. CFM does not really capture the impact of packet collision that distinguishes wireless communication from wired communication, which makes performance analysis under CFM not very accurate. However, for the sake of simplicity, we consider CFM in our design.

VI. DYNAMIC RECONFIGURATION

Our target operating system is SOS, a new operating system for mote-class sensor nodes that takes a more dynamic point on the design spectrum [19]. SOS consists of dynamically-loaded modules and a common kernel, which implements messaging, dynamic memory, and module loading and unloading, among other services. Dynamic reconfigurability is one of our primary assumptions. In the domain of embedded computing, reconfigurability is the ability to modify the software on individual nodes of a network after the network has been deployed and initialized. This provides the ability to incrementally update the sensor network after it is deployed, add new software modules to a sensor node or remove unused software modules when

they are no longer needed. The growing tensions between large hard to update networks and complex applications with incremental patches has made reconfigurability an issue that can no longer be ignored. SOS will support the a mechanism that enables over the air reprogramming of the sensor nodes. Using this methods, software modules may be modified, added or removed.

VII. Feature Set Partitioning

A hypergraph is a generalization of a graph, where the set of edges is replaced by a set of hyperedges. A hyperedge extends the notion of an edge by allowing more than two vertices to be connected by a hyperedge. Formally, a hypergraph $H = (V, E^h)$ is defined as a set of vertices V and a set of hyperedges E^h, where each hyperedge is a subset of the vertex set V [20], and the size a hyperedge is the cardinality of this subset. Let w_i denote the weight of vertex $v_i \in V$. A K-way vertex partition $\Pi = \{V_1, V_2, \ldots, V_k\}$ of H is said to be balanced if each V_i satisfies the following equation:

$$W_k \leq W_{avg}(1+\epsilon), \text{for} \quad k = 1, 2, \ldots, K \tag{2}$$

where,

$$W_k = \sum_{v_i \in V_k} w_i \tag{3}$$

$$W_{avg} = (\sum_{v_i \in V} w_i)/K \tag{4}$$

In a partition of H, a hyperedge that has at least one vertex in a partition is said to *connect* that partition. Connectivity set Λ_j of a hyperedge e_j is defined as the set of partitions connected by e_j . Connectivity $\lambda_j = |\Lambda_j|$ of a hyperedge e_j denotes the number of partitions connected by e_j . A hyperedge h_j is said to be cut (external) if it connects more than one partitions (i.e. $\lambda_j > 1$), and uncut (internal) otherwise (i.e. $\lambda_j = 1$). Therefore, cut-size is defined as follows:

$$\text{cutsize}(\Pi) = \sum_{e_j \in E^h} (\lambda_j - 1) \tag{5}$$

Hence, the cut-size is equal to the number of cut nets. The hypergraph partitioning is defined as dividing it into two or more parts such that the cut-size is minimized, while a given balance criterion among the partition weights is achieved. The hypergraph partitioning problem is known to be NP-hard [21]. During the software partitioning, it is quite important to be able to divide the system specification into clusters so that the inter-cluster (inter-mote) connections are minimized. Hypergraphs can be used to naturally represent feature extraction algorithms. The vertices of the hypergraph are modeled as features and their weights represent the computational time required for features detection, and the hyperedges resembles the number of times a set of features are triggered simultaneously. Partitioning such graph such that the cut-size is minimized while the partitions are balanced can reduce the communication that is required among various processing units for classification phase (The vision is that all features selected must be classified on a local node, thus, in the events where selected features reside on distributed nodes, inter-node communication is inevitable). A high quality hypergraph partitioning algorithm greatly affects the feasibility, quality, and the cost of the resulting system.

We employed a hypergraph partitioning algorithm that is based on the multilevel paradigm. In the multilevel paradigm, a sequence of successively coarser hypergraphs is constructed. A bisection of the smallest hypergraph is computed and it is used to obtain a bisection of the original hypergraph by successively projecting and refining the bisection to the next level finer hypergraph. We have used hMETIS, a set of programs for partitioning hypergraphs implemented for PCs [22]. Same algorithm can be easily ported on a mobile computer such as a Pocket PC to facilitate dynamic reconfiguration. The vision is that the hypergraph information is collected real-time from the processing nodes of the wearable computer and the algorithm running on the motes are reconfigured accordingly. The number of partitions is determined as described below:

The deadline for performing preprocessing tasks as well as pattern recognition must be completed before the next heartbeat arrives. Let the heartbeat be N beats per minute (bpm). Therefore, the heartbeat period can be obtained from:

$$T_{heartbeat} = 60/N \tag{6}$$

Let the time required for preprocessing and pattern recognition be t_{pre} and t_{recog} respectively.

$$t_{pre} + t_{recog} < \delta \times (T_{heartbeat}) \quad \text{where} \quad \delta < 1 \tag{7}$$

The factor δ is selected to be 0.9 to ensure a margin that prevents from overloading the processing units. Therefore, the maximum cpu time that may be assigned to pattern recognition is $\delta \times (T_{heartbeat}) - t_{pre}$ where t_{pre} is fixed and can be computed during profiling stage. As described earlier, the weight on vertices represents the computational time for features. W_k is already outlined in Equation 4. Therefore, the following objective shall be accommodated:

$$\begin{array}{c} \text{Minimize} \quad K \\ s.t. \\ W_k < t_{recog} \quad \forall k = 1..K \end{array} \tag{8}$$

To determine the value of K, we cosider the total time required for pattern recognition on all features , T_{recog}, (extracted from profiling analysis). It is trivial that the lower bound on K can be obtained from the following Equation:

$$K = T_{recog}/t_{recog} \tag{9}$$

Once partitioning is performed based on the lower bound value of K, the solution may be imbalanced and violates the constraint described in Equation 8. In this case, K must be incremented and the features are re-partitioned until a feasible solution is determined.

Record #	# of queries exchanged with adaptive partitioning	# of queries exchanged with manual partitioning	Transmission power saving with adaptive partitioning (%)
100	1	62	6200.00
101	10	58	580.00
102	0	0	N/A
103	0	0	N/A
104	31	109	351.61
105	45	272	604.44
106	0	0	N/A
107	0	0	N/A
118	6	73	1216.67
119	0	0	N/A
200	6	129	2150.00
201	0	0	N/A
202	0	0	N/A
203	0	24	N/A
205	5	115	2300.00
207	44	715	1625.00
208	11	61	554.55
209	145	928	640.00
210	32	150	468.75
212	173	2609	1508.09
213	58	842	1451.72
214	49	559	1140.82
215	22	610	2772.73
217	0	0	N/A
219	0	0	N/A
Average	25.52	292.64	1146.71

TABLE II

NUMBER OF QUERIES EXCHANGED AMONG THE PROCESSING UNITS : SAMPLING RATE = 360 SAMPLE/SEC

Record #	# of queries exchanged with adaptive partitioning	# of queries exchanged with manual partitioning	Transmission power saving with adaptive partitioning (%)
100	138	2218	1607.25
101	5	16	320
102	0	0	N/A
103	0	0	N/A
104	88	948	1077.27
105	34	884	2600
106	0	0	N/A
107	0	0	N/A
118	65	1170	1800
119	0	0	N/A
200	13	604	4646.15
201	0	0	N/A
202	0	0	N/A
203	5	289	5780
205	42	602	1433.33
207	43	1198	2786.05
208	165	1317	798.18
209	114	1497	1313.16
210	64	1008	1575
212	260	3245	1248.08
213	126	1653	1311.90
214	23	1202	5226.09
215	9	352	3911.11
217	0	0	N/A
219	0	0	N/A
Average	47.76	728.12	1524.54

TABLE III

NUMBER OF QUERIES EXCHANGED AMONG THE PROCESSING UNITS : SAMPLING RATE = 100 SAMPLE/SEC

VIII. EXPERIMENTAL ANALYSIS

This section presents various experimental analysis performed to exhibit the effectiveness of our technique. All the experiments were carried out with ECG signals from MIT-BIH Arrhythmia database. The MIT-BIH Arrhythmia database

contains 48 half-hour excerpts of two-channel ambulatory ECG recordings, obtained from 47 subjects studied by the BIH Arrhythmia Laboratory between 1975 and 1979. The recordings were digitized at 360 samples per second per channel with 11-bit resolution over a 10 mV range. We used 25 of 48 complete records freely available from PhysioNet [23]. Each MIT-BIT record has the recordings of two channels. However, we only used the first channel. The second channel was not used for the sake of simplicity.

We performed profiling analysis on the algorithm described in Section I using Avrora to compute the computational delay of feature detection modules. The ECG algorithm was ported both for dot-motes (SOS) and PCs. The algorithm for PC was written in C language. The simulation for feature and hypergraph extraction was done on PC. As for hypergraph partitioning, we utilized hMETIS. The MIT/BIH benchmarks were used with two sampling rates as illustrated in Tables II and III. The original sampling rate was 360 samples/sec while 100 samples/sec were acquired by downsampling data from benchmarks. In Table II and III, two scenarios for configuration was considered. In one scenario, features were adaptively assigned to processing units based on hypergraph partitioning. In the other scenario, the configuration was manually set and remained fixed through out the experiments. The lower bounds on the number of partitions were obtained from Equation 9 for each benchmark. Table II figures the number of queries exchanged in both scenarios. Considering that the experiments were carried out through the simulations, we were unable to measure the wireless power consumption. However, given the number of features we examined - seventeen, each query may be incorporated in a wireless packet of dot-motes (30 bytes). Therefore, taking into account Equation 1, the wireless power consumption is proportional to the number of queries exchanged. On average, the wireless power consumption was reduced by a factor of 11 using adaptive reconfiguration in the 360 samples/sec set of experiments. Likewise, the power was reduced by factors of 15 for sampling rate of 100. The wireless communication overhead for partitioning was negligible due to the small size of our hypergraphs and their slowly changing nature. The maximum number of reconfigurations was five while the maximum number of hyperedges was eight for half an hour duration of benchmarks. Therefore, its effect on the performance of the system was negligible.

IX. Conclusion

We proposed a technique for software partitioning in tiny embedded systems. Our target application was an ECG analysis algorithm which is generally classified as a complex biomedical application. We addressed the problem of mapping such application onto resource constrained embedded systems. Our main objective was to extend the lifetime of the system. This was achieved by reducing the power consumption due to wireless communication. We proved the effectiveness of our technique on various ECG excerpts form MIT/BIH benchmarks. On average the power consumption was reduced by a factor of 13.

References

[1] "Sensatex," *http://www.sensatex.com*.

[2] S. Park, K. Mackenzie, and S. Jayaraman, "The wearable motherboard: a framework for personalized mobile information processing (pmip)," in *Design Automation Conference, 2002. Proceedings. 39th*. ACM/IEEE, 2002, pp. 170–174.

[3] R. DeVaul, J. G. M. Sung, and A. Pentland, "Mithril 2003: applications and architecture," in *Wearable Computers, Seventh IEEE International Symposium on,*. IEEE, 2003, pp. 4–11.

[4] D. Marculescu, R. Marculescu, and P. Khosla, "Challenges and opportunities in electronic textiles modeling and optimization," in *Design Automation Conference, 2002. Proceedings. 39th*. ACM/IEEE, 2002, pp. 175– 180.

[5] T. Martin, M. Jones, J. Edmison, and R. Shenoy, "Towards a design framework for wearable electronic textiles," in *Wearable Computers, Seventh IEEE International Symposium on,*. IEEE, 2003, pp. 190–199.

[6] R. Jafari, A. Encarnacao, A. Zahoory, F. Dabiri, H. Noshadi, and M. Sarrafzadeh, "Wireless sensor networks for health monitoring," in *MobiQuitous '05: Proceedings of the Second Annual International Conference on Mobile and Ubiquitous Systems*, 2005, pp. 479–1481.

[7] R. Jafari, F. Dabiri, P. Brisk, and M. Sarrafzadeh, "Adaptive and fault tolerant medical vest for life-critical medical monitoring," in *SAC '05: Proceedings of the 2005 ACM symposium on Applied computing*. New York, NY, USA: ACM Press, 2005, pp. 272–279.

[8] "Lifeguard monitoring system," *http://lifeguard.stanford.edu*.

[9] H. Kautz, O. Etzioni, D. Fox, and D. Weld, "Foundations of assisted cognition systems," University of Washington, Computer Science Department, Technical Report, Tech. Rep., 2003.

[10] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*, 2nd ed. John Wiley and Sons, Inc., 2000.

[11] A. K. Jain, R. P. W. Duin, and J. Mao, "Statistical pattern recognition: A review," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 1, pp. 4–37, 2000.

[12] A. Graps, "An introduction to wavelets," *IEEE Computational Sciences and Engineering*, vol. 2, no. 2, pp. 50–61, 1995. [Online]. Available: http://www.amara.com/IEEEwave/IEEEwavelet.html

[13] T. Pavlidis, *Structural Pattern Recognition*, ser. Springer Series in Electrophysics. Springer-Verlag, 1977, vol. 1.

[14] "Crossbow technology inc." *http://www.xbow.com*.

[15] J. Pan and W. J. Tompkins, "A real-time qrs detection algorithm," *IEEE Trans. Biomed. Eng*, vol. 32, no. 3, pp. 230–236, 1985.

[16] P. Laguna1, R. G. Mark, A. Goldberger, and G. B. Moody, "A database for evaluation of algorithms for measurement of qt and other waveform intervals in the ecg," pp. 673–676, 1997.

[17] P. de Chazal, M. ODwyer, and R. B. Reilly, "Automatic classification of heartbeats using ecg morphology and heartbeat interval features," *IEEE Transactions on Biomedical Engineering*, vol. 51, no. 7, pp. 1196–1206, 2004.

[18] B. L. Titzer, D. Lee, and J. Palsberg, "Avrora: Scalable sensor network simulation with precise timing," in *IPSN'05, Fourth International Conference on Information Processing in Sensor Networks*, 2005.

[19] C.-C. Han, R. Kumar, R. Shea, E. Kohler, and M. Srivastava, "A dynamic operating system for sensor nodes," in *MobiSys '05: Proceedings of the 3rd international conference on Mobile systems, applications, and services*. New York, NY, USA: ACM Press, 2005, pp. 163–176.

[20] S. Dutt and W. Deng, "A probability-based approach to vlsi circuit partitioning," in *DAC '96: Proceedings of the 33rd annual conference on Design automation*. New York, NY, USA: ACM Press, 1996, pp. 100–105.

[21] M. R. Garey and D. S. Johnson, *Computers and Instractability: A Guide to the Theory of NP–Completeness*. San Francisco, CA: W.H Freeman, 1979.

[22] G. Karypis, R. Aggarwal, V. Kumar, and S. Shekhar, "Multilevel hypergraph partitioning: application in vlsi domain," in *DAC '97: Proceedings of the 34th annual conference on Design automation*. New York, NY, USA: ACM Press, 1997, pp. 526–529.

[23] "Physiobank - physiologic signal archives for biomedical research," *http://www.physionet.org/physiobank/*.