DynAGreen: Hierarchical Dynamic Energy Efficient Task Assignment for Wireless Healthcare Systems

Priti Aghera[1], Dilip Krishnaswamy[21], Tajana Rosing[1]

[1] Department of Computer Science and Engineering, Univ of California San Diego, CA 92093, USA
[2]Qualcomm Research Center, 5775 Morehouse Drive, San Diego, CA 92121, USA

ABSTRACT

Wireless sensor-based healthcare systems consist of hierarchically organized components with varying energy and performance capabilities, such as sensors, local aggregators (e.g. cell phones) and servers. This paper proposes a distributed graph-based approach to partition tasks across these various components with the goal of optimizing the available energy. State of the art implementations assume that all the data is gathered and forwarded for computation to the servers. We show in this work that significant gains in energy efficiency can be obtained if some of the processing tasks are assigned to sensors and local data aggregators. Our *DynAGreen* algorithm takes the graph associated with the workload and successively partitions it between the server, cell phones and sensors such that the overall system energy utilization for computing and communication tasks is minimized. Our experiments show that the task assignment given by *DynAGreen* reduces the overall system energy by 30% with respect to an optimal static design time assignment when minor run time variations are considered.

1. INTRODUCTION

Emerging mobile healthcare systems consist of a heterogeneous set of devices with different computing and communications capabilities. In a typical sensor-based health-care system, patient information is continually sensed and sent to a backend server for analysis by healthcare professionals and/or automated healthcare applications at the server. One or more intermediate nodes, called local aggregators in this paper, can receive information from the sensors and then forward such information to a remote server node. To provide service continuity with patient mobility, we assume that a local aggregator has WWAN connectivity using a 3G/4G wireless protocol to reach the server. The local aggregator can typically be a mobile platform such as a phone or a PDA with an integrated WWAN modem.

Today in many instances the cost of computing locally is lower than communicating all the data to the back end [1], thus leading to a more complex task allocation across sensors, local aggregators and the back end server at run time. Such processing may be desirable if connectivity is weak or not available, so that alerts can be provided to the patient wherever they are, and regardless of the nature of the connectivity to the server. In this work, we propose dynamic runtime task assignment techniques for wireless healthcare systems with the goal of minimizing the overall energy consumed in the distributed system, while considering the computing and communication costs of tasks.

BodyNets '10, September 10-12, Corfu Island, Greece

Task assignment and scheduling onto multiple resources are classical problems in computing [2][3]. Stone in [3] proposes a multiprocessor task scheduling solution using the concept of min-cut to optimize inter-processor communication. There are a lot of variants of such scheduling problems, addressing different objectives. Many algorithms assume a system consisting of homogeneous and infinitely available resources, while others try to optimize latency, area, and the number of resources.

There is quite a bit of previous work on energy optimization in wireless sensor networks using efficient task assignment. Localized task mapping and task scheduling have been considered for WSNs in [4] and [5]. In [4], static EcoMapS algorithm is proposed for energy constrained applications in single-hop clustered WSNs with homogeneous nodes, to map and schedule communication and computation simultaneously. EcoMapS aims to schedule tasks with the minimum schedule length subject to energy consumption constraints. MTMS algorithm is proposed in [5] for task mapping and scheduling in multi-hop homogeneous WSNs with real time guarantees.

Our focus is to develop an algorithm that assigns tasks to the resources with a goal of minimizing the total energy consumption of the system while meeting the task dependency and communication constraints. Contrary to most prior work, our strategy is able to work with a heterogeneous set of resources with various performance and energy characteristics and different communication capabilities. In addition, our technique can adapt to changing system and workload characteristics at runtime, such as varying wireless channel conditions, processing load, network load, task arrival rates, and the available battery capacity of the resources.

We use an integer linear program as a baseline for comparison of our algorithm. The ILP solution provides optimal results for a known set of tasks assuming the workload and system characteristics do not change at run time. We show that in the static case optimal assignment of tasks to heterogeneous nodes in the system reduces the total energy consumption of the system by up to 22%. We have also designed a dynamic algorithm that adapts to changes at runtime. We implement our algorithms using Qualnet [7] and show that our dynamic scheduling technique is able to save additional 30 % energy over the ILP in dynamic conditions.

2. SYSTEM MODEL & ILP

Figure 1 shows the architecture of a typical wireless distributed healthcare system consisting of three main components: Sensors, Local Aggregators (*LA*s – one per BAN), and a Backend Server

(*BE*). Sensors form the Body Area Network (*BAN*) and send data to the *LA* wirelessly. The *LA* aggregates data collected by sensors and may process the data before sending it to a centralized *BE* for analysis. *LA* typically contains multiple wireless technologies such as Bluetooth, Zigbee, WLAN, and WWAN. The *BE* receives information sent by the *LA,* processes and stores the information in a database which can be accessed for further analysis.

Figure 1: Mobile Healthcare System Architecture

Tasks of a wireless healthcare application are modeled as a Directed Acyclic Graph (DAG) $G = (T, E)$ where node set T represents set of n sensing/processing tasks, $\{T: i= 1,2,..n\}$ and E is set of edges that represent communication tasks between nodes, $\{E: i= 1,2,..n\}$. Each $Eij \in E$ represents a precedence relation between tasks $Ti,Tj \in T$. The data produced by task Ti has to be communicated to Tj before Tj can start its processing. The weights Wij on edge $Eij \in E$ represent the amount of data transmitted from task Ti to Tj.

ILP: We formulate Integer Linear Program (ILP) to obtain the task assignment for maximizing the lifetime of a wireless healthcare system. This static solution acts as a baseline of comparison for dynamic task assignment algorithm, and also provides a static allocation for systems with a priori known characteristics. A DAG containing a set T of n tasks and a set R of m (heterogeneous) resources is given. For each resource, the total energy consumed by a specific task assignment running on a resource r is given by E_r. The goal of our ILP is to minimize the total energy consumption of the entire system consisting of m resources subject to task precedence and allocation (i.e., a task having to run on a particular resource) constraints. The ILP considers processing and communication costs to determine the optimal assignment.

3. *DynAGreen*: DYNAMIC ALGORITHM

Since solving the ILP takes minutes of computational time, we design a fast and energy-efficient task assignment algorithm called *DynAGreen* (Dynamic task Assignment for a Green solution) which completes in miliseconds. Due to its computational efficiency DynAGreen algorithm is run periodically on LA and utilizes current energy costs of all tasks to find energy-efficient task assignment.

In [3] Harold Stone applied Ford-Fulkerson algorithm to a problem of efficiently assigning program modules in a two processor computing system to minimize the collective cost of computation time and interprocessor communication time. In this paper we have adopted Stone's method to find an assignment of tasks to wireless health system components that minimizes the total energy consumption of the distributed wireless system consisting of many different types of nodes.

Proposed algorithm is designed to be run on LA and it can be run periodically (e.g. every 15 minutes) or based on specific channel condition triggers (e.g. 20% increase in LA tx power). Sensors report their monitored communication cost to LA. Algorithm has two major steps. First step is called *BAN-BE partitioning* in which we find tasks that should be assigned on BE. We treat Sensors and LA as a single processing node BAN and BE as second processing node. Similar to Stone's method we create a flow graph from given task graph. Communication and computation energies associated with tasks are used as cost functions for determining weights of the edges in the flow graph. We then find minimum weight cutset of the flow graph and assign tasks in BE partition to BE. In second step of the algorithm *Sensors-LA partitioning* we partition the remaining set of tasks in Sensors and LA partition. In this step, given set of sensors in the system are represented as a single Sensors processing node and LA as a second processing node. We then apply Stone's mincut method for task partitioning. Tasks in LA partition as assigned on LA and tasks in Sensors partition are assigned to their source sensors. Following sections describes these steps in details.

3.1 BAN-BE Partitioning:

We use a DAG for activity detection application as an example to explain our algorithm in detail. This system includes three accelerometers which are worn on hand and two legs to determine whether a person is sitting walking or running. We create a flow graph $G'=(T', E')$ shown in Figure 2 from given task graph $G=(T, E)$. Where $T'=T \cup \{BAN,BE\}$ and $E' = E \cup E_{BAN} \cup E_{BE}$. E_{BAN} is a set of edges from *BAN* node to each node in T with their weights equal to the computation cost of running given task on *BE*. We set computational cost of running a task on *BE* is 0 since it doesn't affect battery life of sensor or *LA*. Hence if given task is not bound to sensors or *LA*, we set the weight to 0. For example edges from *BAN* to tasks 3, 4, 5, 6 and 7 have weight of 0 in Figure 2. In case if a task is bound to specific sensor or LA, we want to make sure that edge between *BAN* and task doesn't become part of minimum cutset. This is ensured by assigning weight of ∞ to such an edge. Since in our example tasks 0, 1, and 2 are bound to their respective sensors, weights of edges from *BAN* to tasks 0, 1, and 2 are set to ∞.

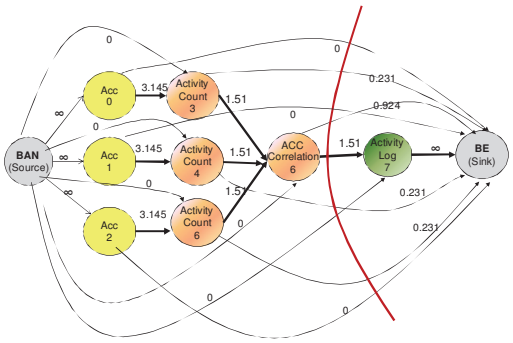

Figure 2: BAN-BE partitioning

All the edges from tasks to *BE* have computation cost of running them on LA. For example, edge from task 3 to *BE* has energy cost of execution task 3 on LA. For tasks that are bound to either sensors or LA, we want to ensure that edge from that task to BE is part of minimum cutset and hence we set it to 0. If a task is bound to BE we want to ensure that an edge from given

task to BE is not part of minimum cutset and hence we set the weight to ∞ as in case of *Activity Log* task in this example.

Weights of edges between the tasks represent communication cost. For example weight of the edge between task 0 and 3 represent amount of energy required to send output of task 0 from *LA* to *BE*. In our specific example communication cost of sending input of task 3 to BE (3.145) is quite high compared to summation of communication cost of sending its output to BE (1.51) and computation cost of doing task on LA(0.231). Figure 2 shows minimum weight cut set for the example task set and this provides energy cost optimal assignment for *BAN-BE* partition as per [3]. As a result of this, task 7 is assigned to *BE*.

3.2 Sensor-LA Partitioning:

In this step we focus on tasks in *BAN* partition and partition the remaining tasks into *Sensors-LA* partitions. We construct a new flowgraph *G" = (T", E")* as shown in Figure 3 with new nodes (*Sensors* and *LA*) added and tasks assigned to *BE* removed. All the sensors are collectively represented by a single node *Sensors* in new graph. For all the tasks whose successors are assigned on *BE*, we create new task nodes which act as transmission tasks to *BE*. In our example graph, we add *ACC Correlation Proxy* task 6' because its successor is assigned on *BE*. This task is added to factor in the cost to communicate task 6's output to *LA* in case 6' gets assigned to a sensor. The weight on the edges between task nodes is the communication power for the *Sensors-LA* wireless link. We set the weight of the edges between *Sensors* and task nodes and edges between task nodes and *LA* based on the computational power required to execute the task on *LA* and sensors respectively. Similar to *BAN-BE* partitioning we use the technique of assigning computational weights of 0 and ∞ to ensure specific assignment restriction. We also bind a processing task to *LA* if it receives input from multiple nodes. For example, in our task graph task 6 receives input from three different nodes. Since in our model sensors do not communicate with each, we need to assign such a task on *LA*. We achieve this by setting computation cost of running such a task on *LA* to 0 and running it on a sensor to ∞.

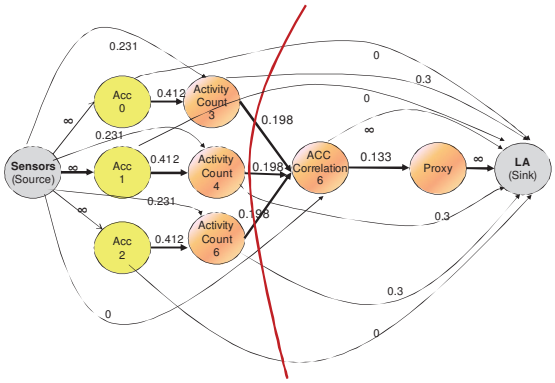

Figure 3: Sensors-LA partitioning

Figure 3 shows minimum weight cut set for *Sensor-LA* partition. All tasks in *LA* partitions are assigned to *LA*. Note that if a task and task's proxy both are in the *LA* partition, we ignore the proxy task since it is only required if its corresponding task is not assigned on *LA*. In *Sensor* partition, we assign the given task to same sensor node as its source sensing task. For example, for task 3, task 0 is its source sensing task and hence task 3 will be assigned to a sensor to which task 0 is bound to.

We used the implementation of mincut/maxflow algorithm proposed in [9]. The computational time of ILP is significantly higher (in mins) than *DynAGreen*'s execution time (milliseconds) as shown in Table II. ILP execution time increases exponentially with a number of tasks in the task graph as indicated by Table II. Because of this low computational overhead compared to ILP, our approach enables frequent execution to address dynamically changing system parameters.

4. EXPERIMENTAL RESULTS

4.1 Experimental Setup:

We used four main task graphs for our experiments which are described in Table I. Each task has an arrival rate and number of instructions to be executed and produces a number of bytes as output each time it executes. Tasks in Table I, mainly represent tasks in preventive healthcare systems such as PALMS [8]. We use Qualnet [7] a state of the art discrete event wireless network simulator.

Table I: EXPERIMENTAL WORKLOAD

ID	Task graph	# of sensor	# of task	Application
A	HR + Activity	2	6	ECG sensor detects heart rate per minute. While accelerometer keeps track of activity
B	Activity Detect	3	8	Detects a person's activity using three accelerometers.
C	GPS + HR + Activity	4	17	Correlates heart rate and activity of a person based on GPS data as context
D	All-Vital	5	20	Log all vital signs like heart rate, blood pressure, activity in addition to location

Sensor nodes are modeled as MicaZ nodes with Zigbee radio. The local aggregator is modeled as a UMTS-UE (User Equipment i.e. Handset) with an additional Zigbee radio interface, a 900mAh battery and CPU speed of 400MHz. The handset is connected to the Backend Server via the UMTS network and with sensors via Zigbee radio. Tasks assigned to the resources sends data to next resource over UDP if any of its successor tasks is not assigned on the same resource. We implemented logic for periodic task execution, data transmissions, radio link parameter measurement and reporting, task assignment control messaging and execution of the *DynAGreen* algorithm at the application layer. We created an ILP for each task graph and used an open source ILP solver called lp_solve [6] to get the optimal task assignments for each of them.

Table II: Comparison between All-on-BE vs. ILP & DynAGreen

TG	Energy Savings for ILP	Energy Savings for *DynAGreen*	Exec Time ILP (sec)	Exec Time *DynAGreen* (sec)
A	17.96%	17.96%	0.49	0.0002
B	21.08%	21.08%	5.92	0.0002
C	22.31%	22.31%	144.13	0.0004
D	20.98%	20.98%	574.12	0.0005

4.2 Static Conditions

Table II shows the percentage improvement in the total energy consumption based on the task assignments for A,B, C and D Task Graphs (TG) as determined by the ILP and the *DynAGreen* algorithm with respect to a commonly used assignment All-On-BE, where all data is streamed to the backend server for processing. In these experiments, system parameters such as execution time and arrival rate of the tasks are constant and *LA* and sensors are stationary. For task graph "All-Vital", which has higher number of processing tasks, the ILP performs ~21% better than All-On-BE. These results shows that task assignment can significantly impact the system's lifetime. *DynAGreen* algorithm gives the same assignment as provided by ILP and thus obtains very close to optimal solutions.

4.3 DynAGreen Adaptability

Next, we simulate the changes to various runtime characteristics and demonstrate the capability of our dynamic algorithm to change the task assignments on the fly to optimize the energy consumption with varying system characteristics. We compare the total energy savings between the dynamic task assignment given by *DynAGreen* and the static assignment given by *DynAGreen* during initial setup to emphasize the fact that dynamic task assignment is necessary for a greener solution in wireless healthcare systems.

Wireless channel conditions change due to mobility of the person using the system, weather conditions, and data extra traffic. To simulate changes in link conditions we simulated a typical day of the user. In this experimental configuration we assume that the user's home is away from the base station and thus wireless link conditions are relatively poor with lower effective data rates when the user is at home. The initial assignment is computed with the link conditions observed at home. If the user is mobile 95% of the time and gets closer to the base station (eg. at work, hospital, school), the wireless link conditions improves. *DynAGreen* takes advantage of these improving link conditions and sends some of the processing tasks from *LA* to *BE*. To simulate this we assume that the user is at home at the start of the experiment and is equal distance from home and base station after one hour, goes near the base station in the next hour and finally reaches very close to the base station after three hours. The user stays near the base station for 4 hours and reaches back home in similar fashion. Since we maintain various link-related parameters for each mobile resource in the system, the *LA* detects these changes on the fly and updates the task assignment. Table III shows that we can increase the energy savings by as much as 21% using the *DynAGreen* algorithm to handle link condition changes due to the mobility of the *LA* as compared to initial assignment.

Dynamic load balancing is required in the system as the processing complexity of a task may change depending on amount of processing required on the data, or due to increased load on a resource in the system such as the *LA* (a phone serving as an *LA* may have to process other tasks). In our experiments we increased the execution time of processing tasks by a factor of three after 4 hours and set it back for the next 4 hours to simulate two levels of processing load. The *DynAGreen* algorithm detects this variation in load and re-computes new task assignment quickly. In Table III we compare the energy savings based on new task assignments obtained by *DynAGreen* algorithm with a previously-determined task assignment obtained by *DynAGreen* in initial condition. We see up to ~30% increase in energy savings when we use the *DynAGreen* algorithm. Similar gains can be observed when other task parameters such as task's output data and arrival rate are changed.

Table III: Energy savings achieved by *DynAGreen* over the initial assignment under dynamic link condition changes

	% Energy savings			
Dynamic Change	Taskgraph A	Taskgraph B	Taskgraph C	Taskgraph D
Wireless channel conditions	6.9	3.5	16.54	20.75
Task Complexity	12.67	7.7	24.69	29.9

5. CONCLUSION

In this paper, we have presented a dynamic algorithm *"DynAGreen"* whose objective it is to minimize the total energy consumption of a wireless healthcare system. Efficient task assignment plays a critical role in energy efficiency of the system. We implemented the *DynAGreen* algorithm in the Qualnet discrete event wireless simulation environment to simulate various healthcare system scenarios. We show that the task assignment generated by *DynAGreen* gives an identical task assignment as given by the ILP in static conditions. In dynamically changing scenarios, *DynAGreen* saves additional 30% energy over the static allocation.

6. ACKNOWLEDGEMENT

This work has been funded by NSF project, "Citisense" grant CNS-0932403, Center for networked systems (http://cns.ucsd.edu) grant CNS08TR and Qualcomm.

7. REFERENCES

[1] E. Jeannot, B. Knutsson, and M. Bjo¨rkman, "Adaptive Online Data Compression," Proc. IEEE Int'l Symp. High Performance Distributed Computing '02, July 2002.

[2] S. H. Bokhari, "Partitioning problems in parallel, pipelined, and distributed computing," IEEE.

[3] H. S. Stone, "Multiprocessor scheduling with the aid of network flow algorithms," IEEE Transactions on Software Engineering

[4] Yuan Tian Ekici, E. Ozguner, F, "Energy-constrained task mapping and scheduling in wireless sensor networks" in Mobile Adhoc and Sensor Systems Conference, 2005

[5] Y. Tian. Cross-layer collaborative in-network processing in multihop wireless sensor networks.IEEE Transactions on Mobile Computing.

[6] Lp_solve - http://lpsolve.sourceforge.net/5.5/

[7] "QualNet Simulator," Scalable Network Technologies, Inc., http://www.qualnet.com

[8] CWPHS Projects: http://cwphs.calit2.net/index.php?option=com_content&view=article&id=68&Itemid=79

[9] Yuri Boykov and Vladimir Kolmogorov, "An Experimental Comparison of Min-Cut/Max-Flow Algorithms for Energy Minimization in Vision." IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), September 2004.