Title
Energy-Efficient Task Assignment Framework for Wireless Sensor Networks

Authors
Heemin Park
Mani B. Srivastava

Publication Date
2003

Energy-Efficient Task Assignment Framework for Wireless Sensor Networks

Heemin Park
Networked and Embedded Systems Lab.
Department of Electrical Engineering
University of California, Los Angeles
Email: hmpark@ee.ucla.edu

Mani B. Srivastava
Networked and Embedded Systems Lab.
Department of Electrical Engineering
University of California, Los Angeles
Email: mbs@ee.ucla.edu

***Abstract*— Tasks for sensor networks often have constraints such as lifetime and latency. Performance of a sensor network like energy consumption and latency is largely affected by how the task is mapped to the nodes in network. This paper presents an energy-efficient task assignment and migration framework for sensor networks. With proposed framework, optimal task transformation and assignment is sought so as to minimize given cost function. Cost function reflects total energy consumption in a network, maximum energy consumption among nodes and maximum latency. Simulated annealing method is used to solve the task transformation and assignment problem. For run-time support, we developed distributed task migration method. While executing tasks in a node, if the remaining energy is less than threshold level, such tasks are migrated into neighbor node which is healthier. With demonstrative examples, we evaluate our task assignment framework and distributed task migration features through Sensorsim simulation which is an extension of ns-2 simulator. We demonstrate that our framework can handle node heterogeneity and various cost function requirements. The experimental results show that elaborate assignments can save more energy than simple assignment of aggregation functions and help in improving performance.**

I. Introduction

Technology advances in embedded system and wireless communication in recent years are making complex and diverse applications of sensor network possible. Wireless Ad-hoc Sensor Networks (WASN) can be applied to from monitoring application such as fire detection, traffic monitoring and wildlife habitat monitoring, to computation-based application like target detection and tracking, and acoustic signal processing which may require image processing and filtering operations. Computation-based applications tend to require various kind of tasks (e.g. sensing, filtering, image or speech processing, storing intermediate data, etc) and various resources (e.g. high-speed CPU, DSP, long-range radio, memory, high-capacity battery, etc) to accomplish the given tasks. Even monitoring application may require an aggregation (or fusion) task like average, maximum, minimum and summation. These aggregation functions can be performed at the basestation node (user node) after collecting sensed data from all the sensors. Otherwise, the aggregations can be done in distributed fashion by in-network processing. Therefore, WASN applications can be described as a set of tasks and dependencies among them. As WASN applications are becoming complex and require various resources it is becoming challenging research issue how to assign the tasks to sensor nodes in a network at *design-time* of energy-efficient sensor network. Because an application consists of a set of tasks and communications among them tasks assignments affect overall performance of the application. This is because the amount of communications and computations may be changed according to which node performing which task. Since sensor nodes can only afford a limited energy in battery in most cases and transmitting or receiving a bit wirelessly is much expensive than processing a bit in local CPU [14] it is much beneficial to process data in-network as much as possible. Energy-efficient task assignment method should seek solution that exploit the advantage of in-network processing. Moreover, current sensor network tends to be heterogeneous in node types thus it has tiered architecture. This means that nodes may have different power consumptions and different communication capabilities (e.g. radio range and rate). Task assignment becomes an issue when we are given a deployment of sensor network and tasks to be accomplished on the network and we are required to decide how the tasks should be mapped onto the network. The goal would be to maximize the lifetime of the network and/or minimize the latency which is the time required to complete one period of the entire job in the task. The energy consumed in the network is mainly spent for computation ad communication. The latency could be defined as the longest delay from sensors or leaf tasks to user node. There are many factors in determining assignment for each task Considering above requirements, the task assignment problem becomes an optimization problem of which cost function includes energy consumption, latency and given constraints. Also, node heterogeneity should be considered. Therefore, systematic approach for task assignment for sensor networks is required.

One of important characteristics of WASN is that the network changes dynamically over time. The network dynamics can happen when nodes move, some nodes die by depletion of battery and nodes change transmission power to control radio range. In any cases, current task assignment might be no longer the optimal. So, when there is dynamics in a network each node itself needs to decide if tasks inside a node need to be transfered to other nodes to form better solution than before. Since this task migration procedure is hard to be managed

in a central point (e.g. basestation), there should be some distributed mechanism to support task migration feature in *run-time* of sensor network. This is the case when subset of sensor nodes are being used for executing given task. After depleting energy in a node which participate in doing computation intensive job or long distance radio transmission, there would be still healthy neighbor nodes that have enough energy to continue doing the job. Then, if the tasks in nodes of low energy could migrate to other healthier node the lifetime of the network would be extended further. Since the situation of task migration happens in run-time, the process of task migration should be done in distributed manner.

In this paper, we present *Energy-Efficient Task Assignment Framework for Wireless Sensor Networks* which has two main features of 1) centralized task decomposition/transformation and assignment for design-time, and 2) distributed task migration for run-time support. The remainder of this paper is organized as follows. In the next section, we present related work and research issues on task assignment. Section III presents the framework and the problem formulations of our methods. In Section IV, we explain about approaches and implementations and we evaluate our framework with experimental results using illustrative examples in Section V. Then, we conclude in Section VI.

II. Related Work and Research Issues

A. Related Work

The assignment problem itself of task graph to heterogeneous multi-processors system has been investigated in the area of distributed system [5]. However, this does not take into account the communication between nodes much. The communication in sensor networks are much more costly and can be done in multi-hop fashion. There are many ways to optimize network by using control knobs such as dynamic voltage scaling, modulation scaling and radio power control. In [2], they provide a method of operator placement in a sensor network. However, they show only for simple application and their method do not change given task graph. In [6], they present solutions for task assignment and dynamic relocation of tasks. However, their method also do not change given task graph at all. In many cases of WASN applications, it is desirable to decompose and transform the given tasks to meet constraints and to get better solution. Most WASN topologies are multi-hop fashion. But, in [17], they assumed that all the nodes are reachable in single hop.

B. Research Issues

The energy-efficient task assignment can play an important role in design framework for sensor networks. This has brought up several related research issues which should be take into account.

How to describe a task?: The first issue is about the way of describing the task. The expression power of the task description would affect the complexity of task assignment system. There are two issues in describing tasks. One is about description of functionality of the task. One simple way of describing task is to use data flow graph (DFG) as in digital signal processing or high-level synthesis problems [7]. This task description issue is related much to the notion of model of computation [1]. The other issue is to specify the distribution (or locations) of sensors which will participate in the task. In many cases, user is only interested in collective information from a subset of sensors rather than entire set of sensors. Then, user do not need to specify and even know which sensors are involved in doing the job. For example, user may specify the task as ”find average temperature from 20 uniformly geographically distributed sensors among 100 sensors”. This might be intended to save energy consumption with trade-off between energy consumption and accuracy. Then, the task has specification of the sensor distribution and does not have specific sensor locations. There should be some method to map from the specification of sensor distribution to sensor locations. Mapping from high-level specification can be applied to coverage problems in sensor networks such as [8], [9] which provide analysis metrics to determine if desired coverage is obtained or not by the given deployment of sensor nodes. High-level specification of distribution of sensors and mapping to task description will be investigated for the future work.

However, in this paper, graph representation for task description is assumed. A node represents task which is to be performed inside a sensor node and a edge represents data transportation between tasks. The task would be one of *sensing*, *aggregation functions* and *user node*.

Task assignment: Task assignment problem to sensor nodes itself can be transformed into a traditional task assignment problem in distributed system [5] or placement problem in VLSI computer-aided design (CAD) field. But, in sensor networks, it is an challenging topic to accommodate the communication aspect at the same time. In wireless communication, the links between nodes are not static. By changing the radio range and transmission power, the communication links are changed. Thus, there is more optimization choices in wireless networks.

How to decompose and transform the given task: Some of tasks are decomposable and transformable. For instance, four-input MAX operation can be transformed into three two-input MAX operations and a form of binary tree. The transformed task must have same functionality, too. So, how to deal with the transformation of task and assignment at the same time is not a straightforward problem.

Time or frequency slot assignment: If time division multiple access (TDMA) or frequency division multiple access (FDMA) medium-access control (MAC) is adopted and the nodes are time synchronized, then appropriate time or frequency slot assignment might help to reduce overall latency. The basic idea should be to allocate more slots to the link which needs larger bandwidth to minimize the longest transportation delay.

Scheduling of tasks: Sensor network is a distributed and parallel system. So, optimal schedule of each tasks to minimize maximum latency and energy consumption needs to be sought.

No consideration on schedule of tasks or naive schedules may cause redundant communications and computations. This results in unnecessary collisions in wireless communications and spending more energy.

Energy consumption estimation and integration with control knobs: Energy consumption in a node varies according to the parameters of control knobs like voltage scaling, modulation scaling, transmission power control and shutdown scheme [11], [13], [16]. Thus, estimation of energy consumption should be done in accordance with available control knobs.

III. Task Assignment Framework

A. Framework of Task Assignment System

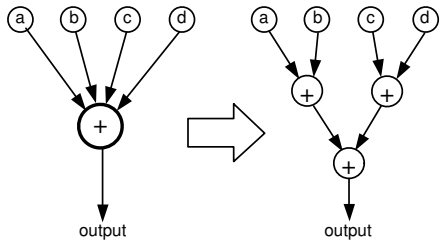
Because of the complexity of the entire problem, the system is divided into three phases. In Fig. 1, three phases and examples are shown. The first step is to parse task graph, then to decompose and transform the given task and assign them to sensor nodes so as to optimize given cost function. Then, the next step is to schedule the task in order to minimize collisions while communicating. Once task assignment and schedules are obtained, the sensor network can perform its task. The third step is for run-time support. If a node reaches a low energy state which means it will die soon the tasks in the node of low-energy state need to be transfered to neighbor nodes which have healtier condition. In Fig. 1, an example of input instance of the problem and assignment results are also shown. Fig. 1 (a) is an example deployment of a sensor network. The link represents that two nodes are within radio range. The given task, Fig. 1 (b), is to get a maximum value from four sensors and the task description can be represented as a graph. So, the problem is to obtain a task assignment like in Fig. 1 (c). Nodes of solid circle means that tasks are assigned to those nodes. The MAX operator is decomposed and transformed into two MAX operators and mapped onto two sensor nodes.

In third phase, it shows the case that when one of nodes at which MAX operator is assigned depletes energy in battery the MAX task migrates to neighbor and continues the task.

We have developed a centralized method for task assignment and scheduling for design-time support. For run-time support, distributed task migration method has been developed and implemented in simulation environment.

B. Problem Definition

The energy-efficient task assignment problem can be defined as followings. A task graph $G = (V, E)$ and a deployment of sensor network S are given. V is a set of tasks and E is a set of communications (u, v) from task u to task v. Also, the energy model for communication and computation for each node type, and cost function are given. The problem is *Find a task transformation T and task assignment A for the given deployment of a sensor network and task description such that the total cost is minimized and constraints are met*. Once the task transformation and assignment are obtained, radio range assignment, schedule of tasks can be determined. The transformation T is a kind of synthesis from decomposable task to transformed tree. If the function of a task is commutative and associative such as summation, maximum, minimum and average, then the task can be transformed into a form of tree to minimize the total energy consumption. This is because the energy consumption of computation is much less than that of communication with nodes in distance wirelessly and there are more chances of optimization with transformed tree. For example, let us assume we need to assign four input add operator as shown in Fig. 2 (a). If the sensor nodes are deployed in wide area it would be better of placing many aggregation points like Fig. 2 (b) rather than aggregating data at one point. The energy consumption for transmission increases exponentially by the path loss exponent factor which is dependent on propagation environment. So, multi-hop with short-range transmission is bettern than single hop with long-range transmission. This can be done by splitting an operator into operators with smaller number of inputs. The task which cannot be transformed into structure of tasks with more than one operator is called *atomic* task in this paper.



(a) Sum Operator (b) Transformed Tree

Fig. 2. An Example of Task Transformation

Task assignment A is a mapping between each task and a node at which it would be running. The number of combinations of possible task assignments increases exponentially with respect to the number of sensor nodes. The criteria to choose an optimal assignment is to minimize the total cost by given cost function. If an application or sensor nodes have constraints such as geographical constraint when placing tasks or program memory limitation in a node, then the cost function should consider the constraints not to violate the constraints.

IV. Approaches and Implementations

For the first phase which is task decomposition/transformation and assignment, simulated annealing framework [4] is adopted. Simulated annealing is a well-known iterative optimization method for combinatorial optimization problem [4]. The reason of choosing simulated annealing method is that it is easy to implement and it can accommodate various forms of changes in results such as transformation, assignments, and etc. To implement simulated annealing framework, four items are required to be prepared. Those are initial solution, cost function, neighborhood solution, temperature scheduling. Basically, the initial solution is depending on the task graph description. However, for the simple task graph, initial assignment is sought using simple placement method (explained in experimental results). Factor of 0.95 is used to reduce

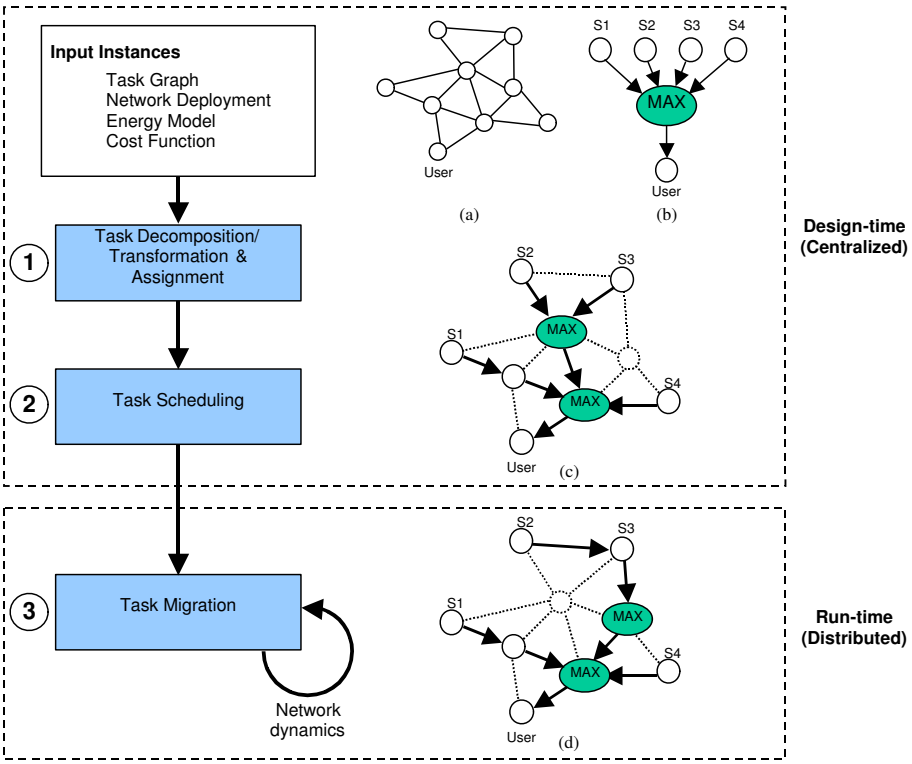


Fig. 1. Framework of Task Assignment System

temperature for next step. Cost function and neighborhood solution are explained in following sections.

A. Cost Function

The cost function is used in determining which is better result when choosing combinations of task assignments from search space. The cost function should be defined according to the application which user is interested in. However, in this paper, we present general metric which can accommodate various trade-offs. We defined the cost function as follows.

$$\begin{aligned} \textit{Total Cost} &= \sum_{\forall i} E_{comp}(i) \\ &+ \sum_{\forall (u,v)} E_{comm}(u,v) \\ &+ w_1 * \max(L(n)) \\ &+ w_2 * \max(E(n)) \\ &+ Penalty \end{aligned}$$

where,
i: task i in intermediate task graph
(u, v): transportation of data from task u to task v
n: sensor node n
w_1: weight for latency
w_2: weight for maximum energy consumption

$$Penalty = \begin{cases} \infty & \text{if there is constraint violation} \\ 0 & \text{otherwise} \end{cases}$$

The total cost is summation of energy consumption for communication and computation for all tasks, and weighted summation of maximum latency and maximum energy consumption among all nodes. In general, the life time of a network is the duration of alive time of the node dies first. So, the energy consumption of the node that consumes the most needs to be minimized. Assigning large value to w_2 can force to minimize $max(E(n))$. If only $max(E(n))$ is supposed to be minimized, then there would be many nodes which consume just less energy than $max(E(n))$. Therefore, it is desirable to include total energy consumption in cost function. By adjusting w_1 and w_2, user can change cost function to satisfy the requirements of application. In addition to these, $Penalty$ is added to avoid constraint violations. For instance, if many tasks are assigned to one node and the node does not have enough memory to accommodate all the tasks, the $Penalty$ value will be ∞.

All nodes are assumed to be time synchronized and have control knobs for shutdown and transmission power control are available. This means that a node will sleep when it is idle. Also, the transmission power can be controlled so as to minimize communication power. Another assumption is that program code for task is mobile code and can be ported to all the types of node in network like Java virtual machine, and it takes same CPU cycles to complete the task. Energy consumption for computation for task i, $E_{comp}(i)$ can be calculated as followings.

$$E_{comp}(i) = di(i) \times \frac{N(i)}{F(A(i))} \times I(A(i))$$

where,
$di(i)$: number of edges coming to task i, in-degree
$A(i)$: sensor node at which task i is currently assigned
$N(i)$: number of CPU cycles required to execute task i
$F(n)$: frequency of CPU in node n
$I(n)$: active CPU current in node n

The term $\frac{N(i)}{F(A(i))}$ means execution time of task i. Thus, term $\frac{N(i)}{F(A(i))} \times I(A(i))$ represents the energy consumption of computation for task i at the sensor node $A(i)$. Because a node cannot receive data from more than one sender at a time in wireless networks, in order to receive multiple data from senders the task should be performed multiple times. So, $di(i)$ is multiplied. Likewise, energy consumption for communication is:

$$E_{comm}(u,v) = Output\_Size(u) * E_{path}(A(u), A(v)).$$

where,
(u,v): transportation of data from task u to task v
$Output\_Size(u)$: size of output data from task u.
$E_{path}(a,b)$: energy consumption to transport 1 bit from node a to node b

Energy consumption to transport data between two nodes is dependent on the underlying routing protocol which is implemented in network layer in sensor nodes. To adopt a specific routing protocol, it is enough to modify $E_{path}(a,b)$ function according to the value of expected energy consumption to transport data over routes from node a to b using that specific routing protocol. Although our method does not restrict the underlying routing protocol, we assume that the routes of the minimal energy consumption are available when assigning tasks. The minimal energy consumption and minimal energy routes for all the pair of nodes are obtained by Dijkstra shortest path algorithm [3] in the beginning of assignment program. For the nodes which are within radio ranges, following general equation is used to calculated energy consumption for communication.

$$E_{path}(a,b) = \alpha + \beta * D(a,b)^{\gamma}$$

Here, $D(a,b)$ is the Euclidean distance between node a and b. The terms α and β are dependent on the radio hardware used in sensor nodes. The term γ represents path loss coefficient which is normally from 2 to 4.

Since various type of node can be specified with the cost function we provide, heterogeneous nodes and tiered network architecture can be applied to our task assignment framework.

B. Neighborhood Solutions

Simulated annealing is an iterative improvement method which adopts neighborhood of current solution based on the probability. The probability is determined according to current annealing temperature. We adopted four kinds of neighborhood solutions that can cover task assignment and decomposition/transformation at the same time. Those are MOVE_TASK, SPLIT_TASK, MOVE_EDGE and MERGE_TASK. Three of SPLIT_TASK, MOVE_EDGE and MERGE_TASK are for task decomposition/transformation. The four neighborhood solutions are depicted in Fig. 3. In this paper, the tasks are said decomposable and transformable if the tasks are commutative and associative. Short explanations of four neighborhood solutions are following.

MOVE_TASK: This is basic movement and it is to move the task from one to other node.

SPLIT_TASK: If the task is decomposable and transformable, then it can be split into two same operations. The task to be split has a child task that has same operator.

MOVE_EDGE: If two tasks are decomposable and two tasks are split from one original task, then the edge can move from one to the other as shown in Fig. 3 (c).

MERGE_TASK: If there were excess of redundant tasks, then it would be better of merging them. Of course those two tasks should be decomposable and transformable and two should have been split from same original tasks.

```
If (energy level < Threshold) begin
    for each neighbor n begin
        Send REQUEST_COST packet to node n
         with the information of in/out tasks
    end

    Collect REPLY_COST packets
    Decide the best neighbor based on costs
    Transfer tasks to the chosen neighbor
end
```

(a) For the node of low energy level

```
If received REQUEST_COST packet begin
    If the packet does not come from
       the node of low energy level
    begin
        reply REPLY_COST packet
        Exit
    end

    for each in/out task u begin
        Send REQUEST_COST packet to node u
    end

    Collect REPLY_COST packets
    Sum all the collect cost
    Reply REPLY_COST packet to originator
      with summation of the costs
end
```

(b) For the node that received REQUEST_COST packet

Fig. 4. Task Migration Algorithm

C. Distributed Task Migration Algorithm

Distributed task migration algorithm is implemented in simulation environment. Basically, in our method, the node that has low energy level has responsibility to migrate to healtier node before it dies. To guarantee for the node to have time to migrate before it dies, a node starts searching neighbors for transferring task when it reaches a certain low energy level (e.g. 5% remaining energy). Pseudo code of the algorithm is shown in Fig. 4. As shown in Fig. 4 (a), the node that reached low energy level initiates task migration process by sending REQUEST_COST packets to its neighbors. Neighbors will

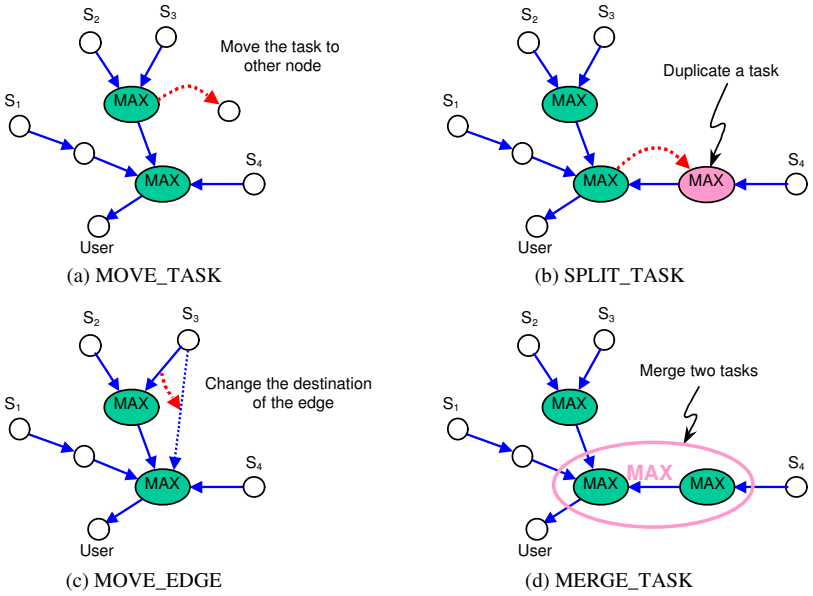


Fig. 3. Neighborhood Solutions for Simulated Annealing

respond with the cost if the tasks migrate to that node. After collecting all the cost values from neighbors, the node can determine which neighbor node is the best to transfer tasks. If the neighbors receive REQUEST_COST packet, they calculate cost of transferring tasks to those nodes by requesting cost value to incoming and outgoing tasks of the originator task. Thus, determining the node for task migration is done in two phases.

V. Experimental Results

We have developed C program for task decomposition/transformation and assignment under Redhad Linux 9.0 environment with gcc compiler. Then, we evaluate the framework and task migration method in distributed environment using Sensorsim [12]. Sensorsim is an enhanced version of the ns-2 network simulator [10]. The aggregation functions and distributed task migration algorithm are implemented inside the sensor application layer.

The sensor network used for experimental results is shown in Fig. 5. On 500 x 500 meter area, 100 nodes are randomly placed and maximum radio range is 100 m. The links between nodes represent that two nodes are within radio range each other. So, the communication is assumed to be symmetric.

The sensor node used in experiment is WINS node used in [15]. It equips 133MHz StrongARM S1100 processor and 100m-range radio. In both of task assignment program and Sensorsim simulation, We use linear battery model and same parameters as those in [15] (79.14 mA, 41.41 mA and 42.23mA for current value of radio transmission, reception and active CPU, respectively).

Fig. 6 is the parameters used in calculating energy consumption during task assignment. The term α includes the energy consumption in electronics part in radio and also can include the energy consumption of CPU when forwarding a packet.

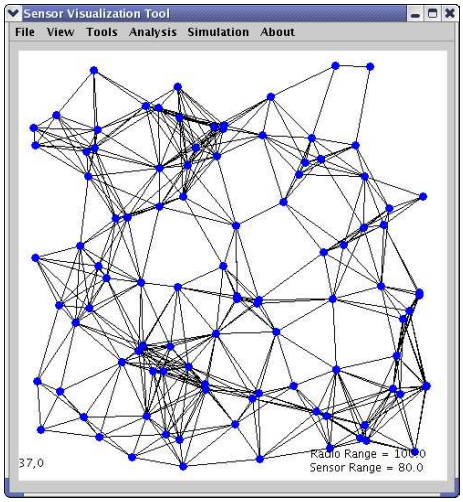


Fig. 5. A Sensor Network of 100 Nodes

α = 100 nJ/bit
β = 0.07 nJ/bit/m^2
γ = 2 (path loss coefficient)
Active CPU current in a node: 42.23 mA

Fig. 6. Parameters for Estimating Energy Consumption

A. Evaluation of Task Assignment

The first example is to get a maximum value of 20 sensors out of 100 deployed sensors at every 0.05 second. The task graph is shown in Fig. 7. The MAX operator is commutative and also associative. So it can be decomposed and transformed if needed. For experimental purpose, the location of sensors and user node are given and fixed. Because the sensor nodes are placed randomly, we just place 20 sensor to node 1 through

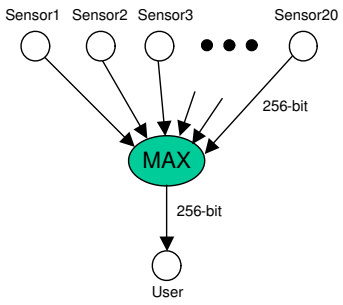


Fig. 7. Example 1: Task Graph for Getting Maximum of 20 Nodes

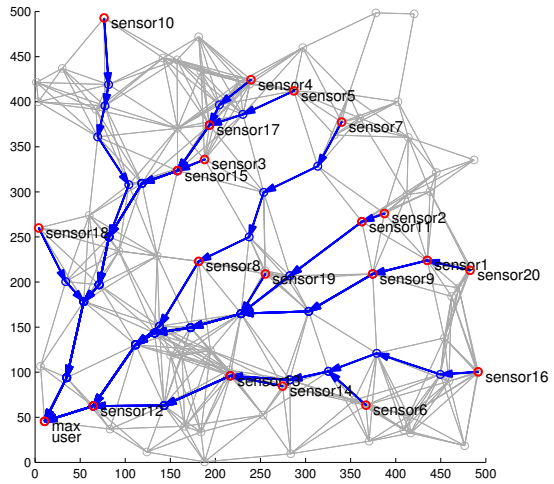


Fig. 8. Task Assignment Result of Example 1: No Task Assignments

20. The data size of sensors and output of MAX operation are 256 bits and number of cycles required for MAX operation is set to 1000. The MAX operations is assigned to the user node initially. As we mentioned earlier, the communication between two nodes is assumed to be done through minimal energy routes. So, if we directly map the given task onto the sensor network without any effort of task assignment, then we get assignment of Fig. 8. The user node is located at the left lower corner. The user node collects sensing data from each sensors and the MAX operation is done at user node. Obviously, this naive task assignment would spend much communication cost because every sensor node needs to send data to user node through long routes and there are heavy traffics as closer to the user node.

It is easy to notice that if we place the MAX operator at the node which has more than one input edges, then the traffic would be reduced. This is the traditional aggregation method. We adopted this assignment as an initial task assignment. Of course this is available only for simple two-level task graph like example 1. Fig. 9 shows the initial task assignment. Now, the MAX operators are assigned to every node which has more than one input edges or sensor nodes which are not leaf. However, this is not still the optimal task assignment for minimal energy consumption. In Fig. 10, the optimized task assignment by our framework is shown. This result is

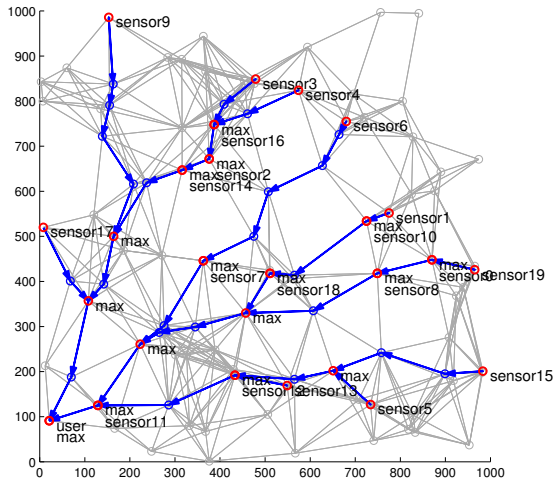


Fig. 9. Task Assignment Result of Example 1: Initial Assignments

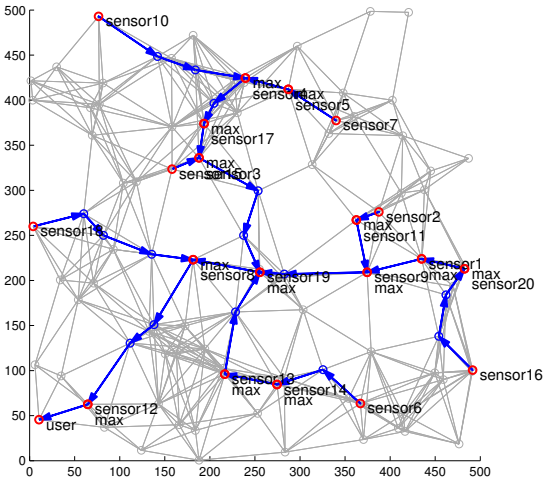


Fig. 10. Task Assignment Result of Example 1: Optimized Assignments (w_2=0)

obtained by setting w_1 and w_2 to zero. So, the objective of cost function was to minimize the total energy consumption in entire network. Other feasible objective is to minimize the energy consumption of the node that consumes energy the most among all nodes. This kind of cost function can be obtained by increasing w_2. In Fig. 11, we get other optimized assignment by setting w_2=10 by our framework. As you can find from those two results, the maximum length of wireless link in Fig. 11 is less than that of Fig. 10.

The estimation of energy consumptions of each case are compared in Table I. The energy consumption is estimated only for one sampling period and it is assumed that dynamic shutdown control knobs for CPU and radio, and radio transmission range control are available. We compare the estimated energy consumption for four cases: no assignment, initial assignment, two options for optimized assignment. By setting weight w_2 to 10, we can optimize the assignment to minimize the maximum energy consumption among nodes. We set w_1 to zero in this experiment. As you can see from the comparison,

TABLE I

COMPARISON OF ENERGY CONSUMPTION ESTIMATION FOR EXAMPLE 1

Optimization Option		No Assignment	Initial Assignment	Optimized Assignment	
				Minimize total	Minimize maximum
Energy Consumption [uJ]	Computation	6.4	10.5	10.2	10.2
	Communication	14224.6	6807.7	4883.0	5205.6
	Maximum	1184.4	205.4	178.3	147.8
	Total	14231.0	6818.2	4893.1	6693.2
Latency [ms]		23.0	23.0	35.8	46.1

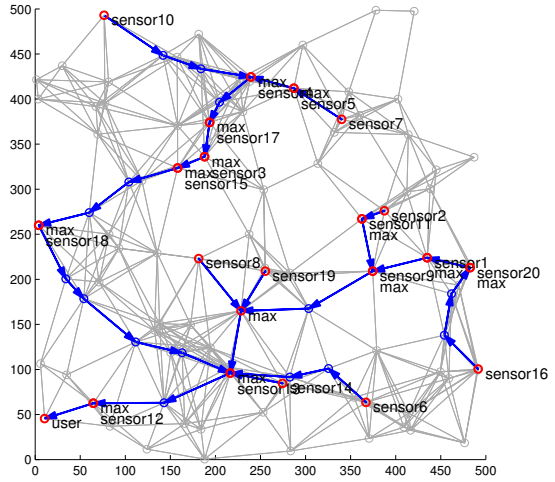


Fig. 11. Task Assignment Result of Example 1: Optimized Assignments (w_2=10)

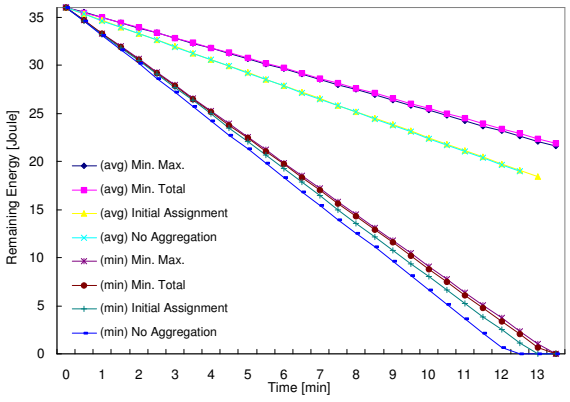


Fig. 12. Minimum and Average Energy in Nodes

the total energy consumption of optimal assignment ("Minimize total") is less than even that of the initial assignment by about 28.3%. The optimization has been done in trade-offs among energy consumption of computation, communication and latency. As shown in last column ("Minimize maximum"), sacrificing latency a little bit, we could reduce the maximum energy consumption among nodes by 28.1%.

To evaluate the task assignment results in networking environment, we simulated the assignment results using Sensorsim. Dynamic source routing (DSR) is used for the underlying routing protocol and IEEE 802.11 MAC is used. To save energy consumption, shutdown scheme is adopted for CPU in a node and radio transmission range is adjusted to save energy in radio hardware. However, the radio hardware is always turned on thus it consumes energy for reception continuously. To make experiment easy, we set the initial energy level as 36 Joule. Fig. 12 shows the minimum remaining energy level among all nodes and the average remaining energy level of all the nodes.

Table II shows the results of Sensorsim simulation for example 1. Although the lifetime of the network is extended by task assignment, the difference is not big. This is because we did not adopt dynamic shutdown scheme for radio in Sensorsim simulation. That means that the energy consumption of receiving packets was very large. However, we get promising results from Sensorsim simulation. Even though the specified sampling rate was 20Hz, we found that achieved sampling rates were very low without elaborate task assignment ("No Assignment" and "Initial Assignment"). With task assignment results by our framework, almost close to required sampling rates has been accomplished. The low sampling rates were caused due to heavy traffic and collisions.

B. Tiered Architecture Support

In this section, we demonstrate the capability of our framework dealing with heterogenenous nodes and tiered architecture. We changed node type of node 91, 28, 42 and 60 to basestation which has higher speed of CPU and long-range radio. Of course, those consume much more energy than normal sensor nodes. Those resources can be utilized if less latency is required in spite of spending more energy. If requirement of application is to have less latency then these resources should be explored to seek the best solution. To test this situation, we set w_1 as very big number to override other costs (e.g. total energy consumption). In Fig. 13, task assignment result for heterogeneous nodes and tiered architecture for minimum latency is shown. The basestation nodes are marked with "BS" in circles. Through above experimental results, it is shown that our framework finds a energy-efficient task assignment solution systematically for the tiered architecture with heterogeneous nodes for various requirements of cost

TABLE II
SENSORSIM SIMULATION RESULTS FOR EXAMPLE 1

Optimization Option	No Assignment	Initial Assignment	Optimized Assignment	
			Minimize total	Minimize maximum
Lifetime [sec]	734.1	774.4	795.2	802.7
Achieved Sampling Rate	0.12	12.13	19.94	19.23

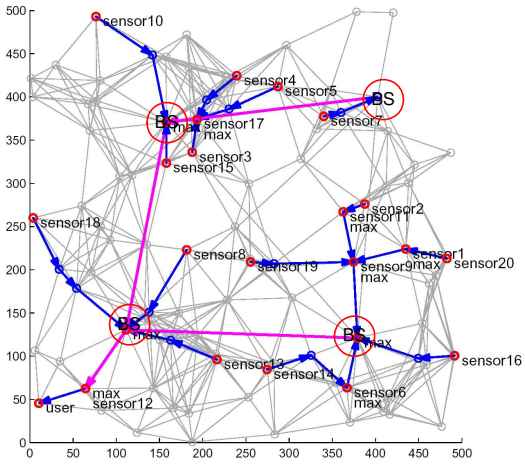


Fig. 13. Task Assignment Result of Example 1: With Base Stations

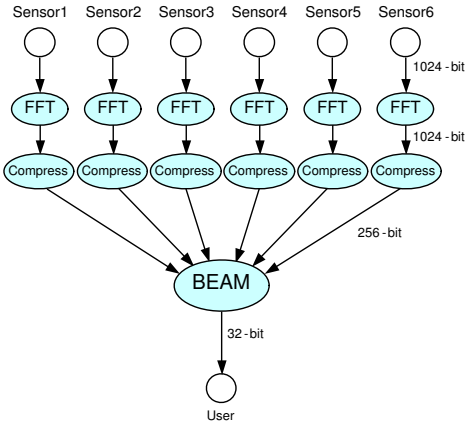


Fig. 14. Example 2: Task Graph of Beamforming

function.

C. Run-time Support: Distributed Task Migration

To demonstrate task migration feature, we made an artificial example that has atomic tasks and the output sizes are different according to the operations. The task is to calculate line of bearing (LOB) estimation from six sources. The task graph is shown in Fig. 14. We put compress task to reduce the output data size after FFT operation. Since beamforming operation (BEAM) and compress (COMP) task are computation intensive tasks the nodes at which BEAM and COMP are assigned would consume energy rapidly. The output data size of sensors

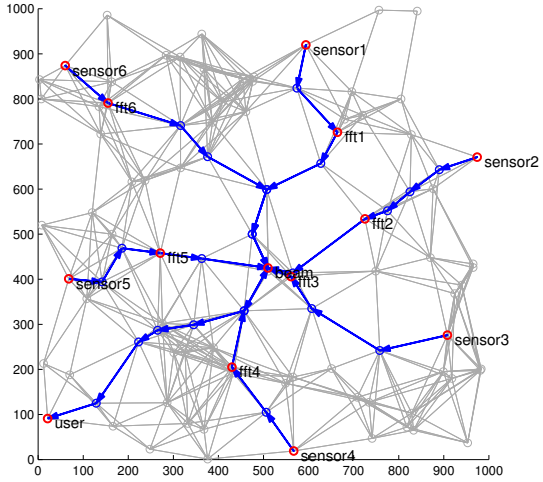


Fig. 15. Example 2: Beamforming. Optimized Assignments

and output of FFT operation are 1024 bits, and 256 bits and 32 bits after COMP and BEAM tasks, respectively. In this case, all the tasks are atomic. The sensors and user node has fixed locations. Because of the big size of the tasks, no more than two tasks can fit into a node. With our energy-efficient task assignment program, we get an optimized task assignment as shown in Fig. 15.

The distributed task migration algorithm in Fig. 4 is implemented in sensor application layer of Sensorsim. A node will start task migration process if the energy of battery remains less than 5%. For the initial energy, we set 360 Joule for each node. The one sample period of the application is 10 second. Likewise in previous simulation, DSR and IEEE 802.11 MAC are adopted. To save energy consumption, shutdown scheme is adopted for CPU. However, the transmission range is not adjusted because nodes should be able to communicate to any node for task migration. In Fig. 16, the migration paths of BEAM task and COMP task are depicted. The BEAM task is assigned to node 48 first and the battery in node 48 is depleted at 5833.9 second. With our distributed task migration method, at 5459.4 second, node 48 determines that migration to other node is required before its battery is entirely depleted. BEAM task migrates from node 48 through 71, 7, 18, 65 47 as battery reaches threshold level (5%). Since COMP task is also computation-intensive task COMP tasks also migrate to their neighbor. Thanks to task migration feature, user task can get information until 7792.8 second which is 34% longer time

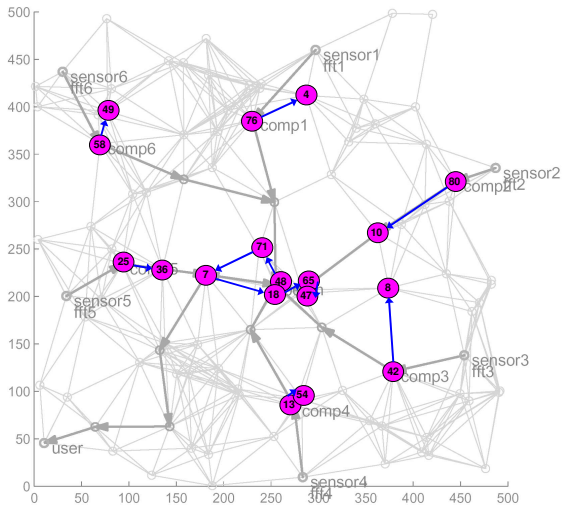


Fig. 16. Migration Paths of BEAM and COMP Task

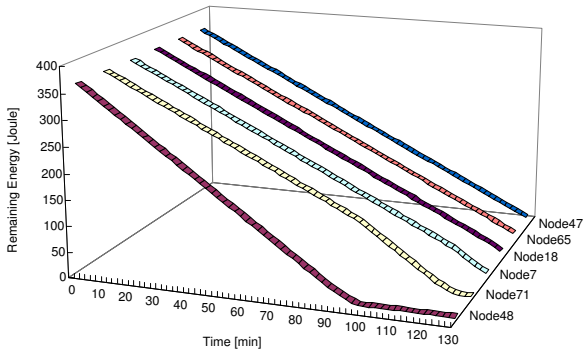


Fig. 17. Remaining Energy in Nodes for Task Migration of BEAM

than that without task migration feature. In Fig. 17, remaining energy of the nodes that BEAM task goes through are shown. As shown in Fig. 17, the battery in node 48 is consumed the first, then the battery in node 71 and node 7 and so on.

VI. Conclusions

In this paper, energy-efficient task assignment and distributed task migration for sensor networks has been presented. Because energy for communication is much expensive than computation and they are trade-offs each other, a certain systematic approach to minimize the total energy consumption has been needed. Presented method in this paper based on simulated annealing shows improvements in estimation by about 28 percent for total energy consumption and maximum energy consumption in a node. Also, through Sensorsim simulation, task assignment helps in increasing achieved sample rate to desired level. For run-time support, we have developed distributed task migration algorithm and implemented in simulation environment. Through the experimental results of network simulator, it is verified that our task migration algorithm works correctly and extends the lifetime of the network.

As we have mentioned in the section of research issues, task scheduling, time and frequency slot allocation, tighter integration with control knobs are remaining for the future work. To provide interaction with users, it would be nice if the framework is integrated with visual environment. Another challenging extension of this framework would be high-level description of sensor distribution. Instead of specifying locations of sensors, high-level specification such as coverage constraints, distributions, or areas for sensor node will be required for an important part of system-level design environment for sensor networks.

References

[1] A. Agrawal, A. Bakshi, J. Davis, B. Eames, A. Ledeczi, S. Mohanty, V. Mathur, S. Neema, G. Nordstrom, V. Prasanna, C. Raghavendra, M. Singh, *MILAN: A Model Based Integrated Simulation Framework for Design of Embedded Systems*, Workshop on Languages, Compilers, and Tools for Embedded Systems (LCTES 2001), Snowbird, Utah, June 2001.

[2] Boris Jan Bonfils and Philippe Bonnet, *Adaptive and Decentralized Operator Placement for In-Network Query Processing*, Information Processing in Sensor Networks (IPSN03), Palo Alto, CA, April 2003.

[3] T. H. Cormen, C. E. Leiserson, R. L. Rivest, C. Stein, *Introduction to Algorithms*, The MIT Press, McGraw-Hill, 2001.

[4] S. Devadas and A. R. Newton, *Algorithms for hardware allocation in data path synthesis*. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, July 1989, Vol.8, (no.7):768-81.

[5] M. Kafil and I. Ahmad, *Optimal task assignment in heterogeneous distributed computing systems*, Concurrency, IEEE [see also IEEE Parallel & Distributed Technology] , Volume: 6 Issue: 3, Jul-Sep 1998, Page(s): 42 -50.

[6] Rajnish Kumar, Matthews Wolenetz, Bikash Agarwalla, Phil Hutto and Kishore Ramachandran, *DFuse: A Framework for Distributed Data Fusion*, to be appeared in SenSys03, 2003.

[7] Giovanni De Micheli, *Synthesis and Optimization of Digital Circuits*, McGraw-Hill, 1994.

[8] Seapahn Meguerdichian, Farinaz Koushanfar, Miodrag Potkonjak and Mani Srivastava, *Coverage Problems in Wireless Ad-Hoc Sensor Networks*,IEEE Infocom 2001, Vol 3, pp. 1380-1387, April 2001.

[9] Seapahn Meguerdichian, Farinaz Koushanfar, Gang Qu and Miodrag Potkonjak, *Exposure In Wireless Ad Hoc Sensor Networks*, Procs. of 7th Annual International Conference on Mobile Computing and Networking (MobiCom '01), pp. 139-150, July 2001.

[10] *Ns-2 Simulator*, http://www.isi.edu/nsnam/ns, 2001.

[11] T. Pering, T. Burd, and R. Brodersen,*Dynamic Voltage Scaling and the Design of a Low-Power Microprocessor System*, In Power Driven Microarchitecture Workshop, attached to ISCA98, June 1998.

[12] S. Park, A. Savvides and M. B. Srivastava, *Simulating networks of wireless sensors*. Winter Simulation Conference, pp. 1330-1338, 2001.

[13] V. Raghunathan, S. Ganeriwal, C. Schurgers, M. B. Srivastava, *E2WFQ: An Energy Efficient Fair Scheduling Policy for Wireless Systems*, International Symposium on Low Power Electronics and Design (ISLPED'02), Monterey, CA, August 12-14, 2002.

[14] V. Raghunathan, C. Schurgers, S. Park, and M.B. Srivastava, *Energy-Aware Wireless Microsensor Networks*, IEEE Signal Processing Magazine, vol.19, no.2, March 2002, pp.40-50.

[15] A. Savvides, S. Park, and M. B. Srivastava, *On Modeling Networks of Wireless Micro Sensors*, poster session at SIGMETRICS 2001, Boston, MA, June 2001.

[16] C. Schurgers, O. Aberthorne, M. B. Srivastava, *Modulation Scaling for Energy Aware Communication Systems*, International Symposium on Low Power Electronics and Design (ISLPED'01), Huntington Beach, CA, pp. 96-99, August 6-7, 2001.

[17] Yang Yu and Vitor K. Prasanna, *Energy-Balanced Task Allocation for Collaborative Processing in Networked Embedded Systems*, ACM Conference of Language, Compilers, and Tools for Embedded Systems (LCTES), June 2003.