Monetary Cost and Energy Use Optimization in Divisible Load Processing

Mequanint A. Moges,
Leonardo A. Ramirez,
Carlos Gamboa and
Thomas G. Robertazzi[1]
Department of Electrical and
Computer Engr.
Stony Brook University
Stony Brook, NY 11794
e-mail:
(mmoges,lramirez,gcarlos,tom)@ece.sunysb.edu

Abstract — **In this paper divisible load scheduling theory is used to examine monetary cost and energy use optimization in a single level tree network. The problem is to find an optimal sequence for the distribution of the load to each processor that will minimize the monetary cost or energy consumption of the network. Optimization results from four different algorithms and their associated computation times are presented. Monetary cost sensitivity analysis results are also provided.**

I. Introduction

The introduction of wireless sensor networks has attracted significant attention due to their capability to integrate sensing, communicating and processing technologies. They may consist of hundreds to thousands of nodes that are linked via short range ad hoc radio connections to forward data in a multi hop mode of operation. Each node has some sensing capability and computational power and operates in an unattended mode. There are many potential scientific and civilian applications of wireless sensor networks including geophysical, environmental and planetary exploration, vehicle tracking [1], glucose level monitoring and retinal prosthesis, habitat monitoring [2] and robots.

The creation of wireless sensor networks involves a need for efficient allocation of sensor loads to processors and links. For this requirement divisible load theory offers tractable performance analysis of systems incorporating sensing, communication and computation aspects, as in parallel and distributed systems. Divisible load theory has been intensively studied over the past decade or so and an important problem involves finding an optimal arrangement of loads to be partitioned and sequenced from the root (load originating node) to the other processors while trying to obtain minimum processing time or cost of such a partitioned sequence.

A key feature that distinguishes wireless sensor networks from traditional distributed systems is their need for energy efficiency. Many nodes in the emerging sensor systems will be untethered, having only finite energy reserves from a battery. All communication, even passive listening, will have a significant effect on these reserves. Load distribution policies for sensor networks must therefore also be mindful of energy that they consume besides the minimum processing time. That is, in sensor networks, existing performance optimization methods will need to be extended and combined in new ways in order to provide service that meets the needs of applications with the minimum possible energy consumption.

In this paper, we study the monetary cost and energy optimization of a single level tree network by using various swapping techniques. In changing the sequence of load distribution to processors one can see that the load assignment to each of the child processors will be different. The objective of this will be to find the sequencing of load distribution that result in a minimal monetary cost or energy use. We also investigate the empirical computational complexity of each algorithm in terms of the time needed to find the optimum sequence. Moreover, this paper considers the effect of link and processor variation on the total monetary cost.

The organization of this paper is as follows. Section II gives an overview of related works concerning minimum cost load distribution sequencing. In section III, the system model used is discussed. The mathematical analysis of the monetary cost and energy use in single level tree networks is presented in section IV. In section V, the various algorithms developed are described. Optimization results from the proposed algorithms are presented in section VI. Section VII presents the results obtained from the sensitivity analysis of the total monetary cost in terms of changes in the processor and link speed. Finally the conclusion is given in section $VIII$.

II. Related Work

The problem of minimizing monetary cost or the energy consumption of all wireless sensor networks has received an increasing amount of attention from researchers in universities, government and industry. Monetary cost optimization is of interest for metacomputing and future computer utilities. In the case of minimizing the monetary cost, Robertazzi, Sohn, Charcranoon and Luryi proposed a linear model where the processing and transmission costs were proportional to the size of the data being processed or transmitted. In their work, if one considers only processor costs and one has a load being distributed sequentially from a controller to processors in a bus network, one can show [3][4] that cost is minimized if load to the processors is distributed in order of increasing $c_i\omega_i$. Here c_i is the cost proportionality constant of the i^{th} processor and ω_i is the inverse computing speed of the i^{th} processor. If link costs are also included, heuristic algorithms can be developed to find efficient solutions [5].

On the other hand, the problem of minimizing energy use involves minimizing communication between processors since this is the dominant operation in energy. Representative work

[1]This work was supported by NSF Grant CCR-99-12331.

includes that on aggregating data at nodes to shorten subsequent transmissions [6][7], the SPIN family of protocols which involve negotiation and communication between neighboring nodes to minimize data transmission and energy use [8][9] and probabilistic routing traffic to spread load across network nodes and prolong stored energy [10]. It is clear that the uneven characteristics of wireless sensor networks functionality require a re-thinking of integrated distributed processing algorithms and protocols. The study in this paper is focused on developing two functions: monetary cost and energy use in terms of parameters that are used in divisible load theory. Without loss of generality a linear model of these two function can be developed and tested for optimization.

In the following section, a system model that will be used throughout the paper is described.

III. System Model

In this section, some rules for scheduling in divisible load theory are described along with some notations and definitions. As mentioned earlier the network topology discussed in this study is the single level tree (*star*) network. It will be assumed that the total processing load is arbitrarily divisible into fractions of loads to be assigned to each processor over a network. The root processor where the total processing load originates, keeps some processing load for itself and sends out the rest of the load to the remaining processors over the network. It is assumed also that solution reporting time (back to the load originating node) is negligible compared to load distribution time and so is neglected. However, solution reporting time can be naturally modeled for divisible loads when necessary.

A Notations and Definitions:

α_i : The fraction of load that is assigned to processor i by the load originating processor.

ω_i : A constant that is inversely proportional to the computation speed of processor i in the network.

z_i : A constant that is inversely proportional to the speed of link i in the network.

T_{cp} : Computation intensity constant. This is the time it takes the i^{th} processor to process the entire load when $\omega_i = 1$. The entire load can be processed on the i^{th} processor in time $\omega_i T_{cp}$.

T_{cm} : Communication intensity constant. This is the time it takes to transmit the entire processing load over link l_i when $z_i = 1$. The entire load can be transmitted over the i^{th} link in time $z_i T_{cm}$.

One convention that is followed in this study is that the load originating at the root processor is assumed to be normalized to be a unit load.

IV. Monetary cost or Energy Use Analysis in Single level Tree Network

Consider a single level tree network with $N+1$ processors and N links as shown in Fig. 1. The root processor P_0, where the load originates, is assumed to be equipped with a front-end processor and partitions a total processing load into $N+1$ fractions, keeps its own fraction α_0, and distributes the other fractions α_1, α_2 , ... , α_N to the child processors P_1, P_2, ... , P_N respectively and sequentially. The sequencing of the network can be shown by the following ordered link-processor set as:

$$\theta = P_0, (l_1, P_1), (l_2, P_2) \ldots . (l_N, P_N).$$

From this ordered set, one can assume that the root processor will first assign a fraction of load α_1 to P_1, then α_2 to P_2 and so on. Any consecutive link-processor pairs $(l_i, P_i), (l_{i+1}, P_{i+1})$ may not necessarily be physically adjacent in the network. By changing the order of the adjacent link-processor one can see that the load assignment to each of the child processors will be different. As mentioned earlier, the objective of this paper will be to find the sequencing of load distribution that results in a minimal monetary cost or energy use of the network.

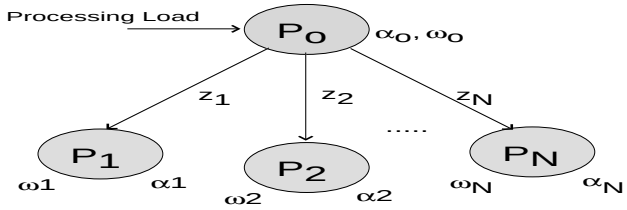

Figure 1: Single level tree network.

A Link-Processor Monetary Cost

In a single level tree network, the link-processor monetary cost includes the cost of communication between the root processor and each child processor and the corresponding cost needed to process the fraction of load. That is,

$$C_i = C_i^{\text{l}} z_i T_{cm} + C_i^{\text{p}} \omega_i T_{cp} \tag{1}$$

where C_i is the total link-processor cost per unit load of the i^{th} processor, C_i^{l} and C_i^{p} are the communication cost per second and the computing cost per second of utilizing the i^{th} link and processor, respectively.

B Total Monetary Cost

On the other hand the total monetary cost includes the sum of individual monetary cost of each of the processor in the network. In this case, one can write the expression for the total monetary cost as:

$$C_{total} = \alpha_0 C_0 + \alpha_1 C_1 + \alpha_2 C_2 + ... + \alpha_N C_N \tag{2}$$

C Link-Processor Energy Use

The link-processor energy use for processing a fraction of load at any processor is defined as the energy consumed during transmission, reception and processing of the underlying fraction of load. The energy is defined in terms of the power dissipated during the time in which the processor is busy serving the assigned fraction of load. In other words, the individual processor energy consumption depends on the

fraction of load assigned to it, which in turn is determined by the sequence of load distribution. The following notations and definitions are defined in this section.

P_{t_0} : The transmission power of the load originating processor.

P_{r_i} : The reception power of the i^{th} receiving processor.

P_{p_i} : The processing power of the i^{th} processor.

E_{t_0} : The transmission energy needed by the load originating processor. We have,

$$E_{t_0} = P_{t_0}(z_1 + z_2 + ... + z_N)T_{cm}.$$

E_{r_i} : The energy needed to receive the entire load by the i^{th} receiving processor. Also we have,

$$E_{r_i} = P_{r_i} z_i T_{cm}.$$

E_{p_i} : The energy needed to process the entire load by the i^{th} processor. Similarly,

$$E_{p_i} = P_{p_i} \omega_i T_{cp}.$$

D Total Energy Use

Total energy use is the energy consumed by the network to process an entire load. Assuming a linear system, without loss of generality, this total energy is simply the sum of all individual link-processor energy consumptions discussed above. Since the objective of this part of the study is to determine the sequencing of load distribution for efficient energy use of the network, the main performance metric for optimization will be the total energy use to process the entire load.
Now for the root processor, one can write the following energy consumption as:

$$\begin{aligned} E_0 &= (P_{t_0} z_1 T_{cm} + P_{t_0} z_2 T_{cm} + ... + P_{t_0} z_N T_{cm}) \\ &+ P_{p_0} \omega_0 T_{cp} \\ &= E_{t_0} + E_{p_0} \end{aligned} \quad (3)$$

Similarly for the child processors, one can write:

$$\begin{aligned} E_i &= P_{r_i} z_i T_{cm} + P_{p_i} \omega_i T_{cp} \\ &= E_{r_i} + E_{p_i} \end{aligned} \quad (4)$$

E_i : The total energy consumed to communicate and process the entire load by the i^{th} processor.

$\alpha_i E_i$: The energy consumed to communicate and process the assigned fraction of load α_i by the i^{th} processor.

Now the total energy consumed by the whole network can be written as the summation of individual energy consumptions as :

$$\begin{aligned} E_{total} &= \alpha_0 E_{p_0} + (\alpha_1 + \alpha_2 + ... + \alpha_N) E_{t_0} \\ &+ \alpha_1 E_1 + \alpha_2 E_2 + ... + \alpha_N E_N \end{aligned} \quad (5)$$

The above equation shows that the total energy is composed of the energy consumed to process α_0 by P_0, the energy consumed to transmit α_1, α_2, up to α_N from the root processor to each of the child processors and the energy consumed to receive and process each of the α_i's by the child processors.

V. Proposed Algorithm

In this section, the description of the various swapping algorithms used and tested in this study is provided.

A Adjacent swap algorithm

This algorithm approves swaps in the position in the load distribution sequence between logically adjacent nodes if the swap yields to an improvement of the monetary cost or energy function. The swapping process repeats again and again until the cost of the improved load partition is less than a certain predefined percentage, which in this case is selected to be from 5 to 20 percent of the initial value of the cost.

At the start of the subprogram random values are chosen for the parameters ω_i, z_i, T_{cp}, T_{cm} ${C_i}^{\text{p}}$, and ${C_i}^{\text{l}}$. Then using these parameters the corresponding load shares, $\alpha_i's$, that should be assigned to each processor are obtained. The algorithm assumes that these load shares will be transmitted sequentially to the processors. The fraction of load assignments are then used to find the logically corresponding monetary cost or energy use functions.

On the next iteration, two logically adjacent nodes will temporarily be swapped and the algorithm checks for any improvement in the monetary cost or energy function. The swapping will be approved if the new value of the monetary cost or energy function is less than the previous value. Otherwise the swapping will not be approved and the process continues to the next available set of adjacent nodes.

When a predefined minimum threshold value of the monetary cost or energy use function is reached the algorithm stops. Here it should be recalled that the computation time needed to reach the given threshold value increases as the number of nodes in the network is increased. In a previous study [11] the algorithm was used and it was known to be convergent to up to a maximum of 22 nodes. In this paper, the tests were derived to up to a maximum of 50 nodes with total convergence.

B Best Swap Algorithm

The *best swap algorithm* in each iteration checks all the possible swaps in the network (N Combined 2) and selects the best swap that leads to the best improvement. This algorithm has therefore the least number of approved swaps, however, with the increase of number of nodes, in this case more that 20 nodes, the algorithm was found to be very time consuming.

C Random Swap Algorithm

As can be implied from its name, this algorithm picks two nodes randomly and swaps them if and only if such swap yields an improvement of the monetary cost or energy function. In this case, due to the random nature of the algorithm a timeout was implemented to terminate the algorithm.

D First Swap Algorithm

This algorithm uses the first possible swap if such a swap yields an improvement in the monetary cost or energy function and keeps on checking all the possible swaps until a threshold value mentioned earlier is reached in the cost improvement.

VI. Simulation Results

In this section, simulation results showing monetary cost or energy use optimization are presented. As mentioned earlier, the order of link-processor load distribution is "swapped" in order to find a distribution sequence that results in an improved cost from the previous sequence. This process will repeat until the final threshold value of improvement is reached. In this case the only parameter that changes value will be the fraction of load, α_i, that is assigned to each processor in the network. Any runs resulting in excessive delays, which can happen occasionally due to random swapping, were set to stop by using a timeout. The algorithms were tested for different number of nodes starting from 10 up to 50 nodes.

Fig. 2 shows the monetary cost minimizations and the corresponding number of swaps by the four algorithms. In this figure it can be seen that the number of swaps needed to reach the specified threshold improvement is different for each case. This is because each algorithm uses a different strategy for swapping. The result showed that all algorithms improved the monetary cost function in some cases obtaining a very good result of 60 percent or more improvement. In some cases like, the adjacent swapping, even though, it was found to be much faster than the other three algorithms the improvement was not as good as the others when the number of nodes is increased. On the other hand these three algorithms were found to be robust with the increase of number of processors but with a clear disadvantage of computing time as it is shown in table 1. Similarly Figures 3 and 4 show monetary cost optimization but for different threshold values of 10 and 15 percent respectively. As expected these plots show less improvement in cost but faster computation time. This observation can lead to a conclusion that depending on the application of a specific system there will be a tradeoff between the speed and required cost saving.

Processors	Timeout	First Swap	Adj. Swap	Best Swap	Random Swap
10	10	2	1	1	16
20	40	39	10	29	40
40	3000	1633	25	1491	3000
50	5000	3517	285	4646	5000
50	6000	5035	123	4744	6000

Table 1: Comparison of Computation Time needed by each algorithm.

Figures 5 and 6 both show the result of the energy use minimization by the four algorithms. In this case the total energy consumption has to be determined as a function of all parameters, including the power parameters, besides the load fraction. For the sake of completeness the values of transmission power, reception power and processing power are assumed to take arbitrary values. Figure 5 shows the case where the communication power (transmission power and reception power) is larger than the processing power. On the other hand Figure 6 shows the case where the communication power is assumed to be smaller than the processing power. Based on the above assumptions, it was found that all of the four algorithms can achieve a predefined threshold level of energy improvement.

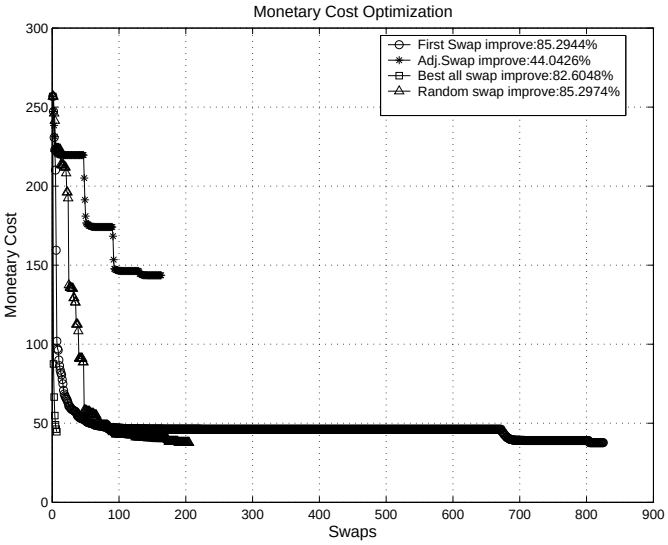

Figure 2: Monetary Cost Improvement with 5 percent Threshold.

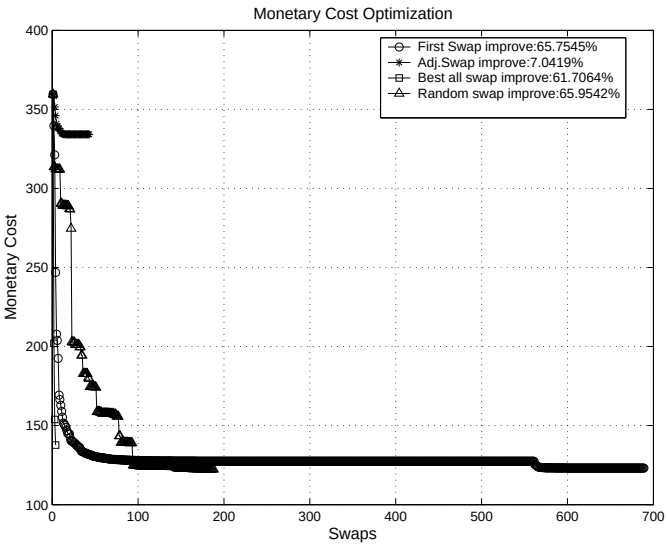

Figure 3: Monetary Cost Improvement with 10 percent Threshold.

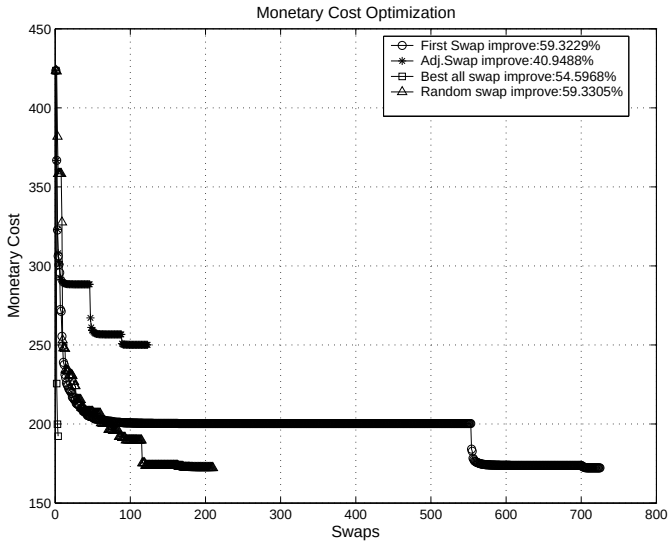

Figure 4: Monetary Cost Improvement with 15 percent Threshold.

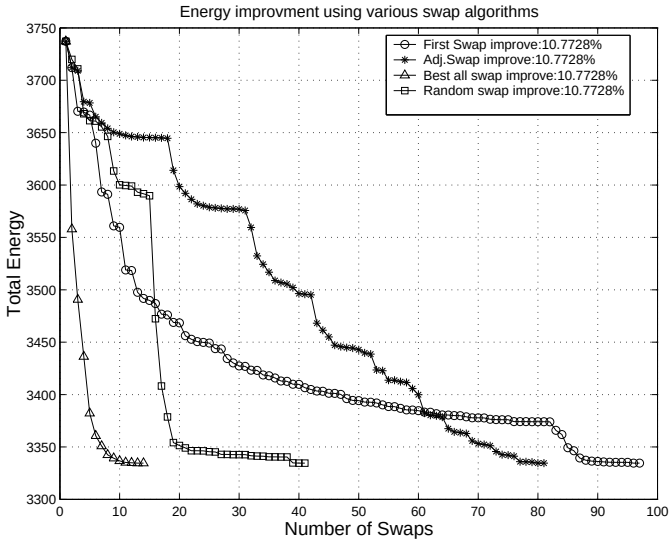

Figure 6: Energy Use Improvement when Communication Power < Processing Power.

In this specific examples considered, it can be seen that there is up to 66 percent energy use improvement.

To summarize, our results show that the monetary cost or energy use in a single level tree network can be improved significantly by changing the order of distribution of load fractions to each processor.

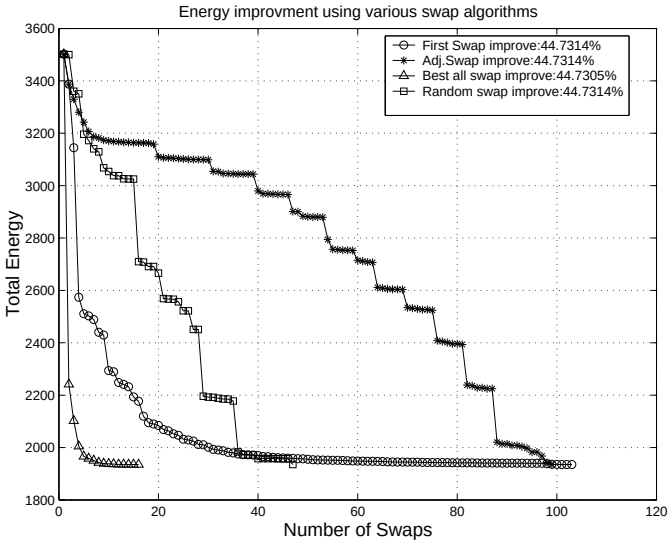

Figure 5: Energy Use Improvement when Communication Power > Processing Power.

VII. Monetary Cost Sensitivity Analysis

This section considers the effect of changes of processor speed and link speed on the total monetary cost in a single level tree network. To see this effect a sensitivity analysis was implemented for a homogeneous network of $N = 10$ processors. That is, the speed of all the processors is the same and all links have identical characteristics. The following notations and definitions are used in this section.

ω_{nom} : Nominal value of the processor speed to be compared.

z_{nom} : Nominal value of the link speed to be compared.

ω_{var} : Changed value of the processor speed to be compared.

z_{var} : Changed value of the processor speed to be compared.

In this case two expressions for sensitivity $S(\omega)$ and $S(z)$ are used to study how variations in processor and/or link speed affect the total monetary cost.

(a) link speed fixed, processor speed varies

The sensitivity expression is given as:

$$S(\omega) = (C_{total}(\omega_{nom}) - C_{total}(\omega_{var}))/(\omega_{nom} - \omega_{var}) \qquad (6)$$

(b) link speed varies, processor speed fixed

The sensitivity expression is given as:

$$S(z) = (C_{total}(z_{nom}) - C_{total}(z_{var}))/(z_{nom} - z_{var}) \qquad (7)$$

Due to its effectiveness in terms of the number of swaps, the *best swap* algorithm was used to implement the sensitivity analysis. Using this algorithm as a starting point, random values were chosen for the costs of links and the processors. However, three sets of cost of processors and cost of links were constrained so that ${C_i}^1 < {C_i}^{\mathrm{P}}$, ${C_i}^1 = {C_i}^{\mathrm{P}}$ and ${C_i}^1 > {C_i}^{\mathrm{P}}$. Thus, three configurations of the network for those processor and link costs were studied.

In Fig. 7 the result of the computation of sensitivity for the case where z varies is shown. In this case the percentage change of link speed Δz is shown in the horizontal axis, whereas the change of the monetary cost given in (*eqn.*7) is shown in the vertical axis of this plot. Notice here that when ${C_i}^1 <= {C_i}^{\mathrm{P}}$ the total cost function has a decreasing behavior. However, when ${C_i}^1 > {C_i}^{\mathrm{P}}$ the total cost is increasing and this is reflected in the sensitivity axis.

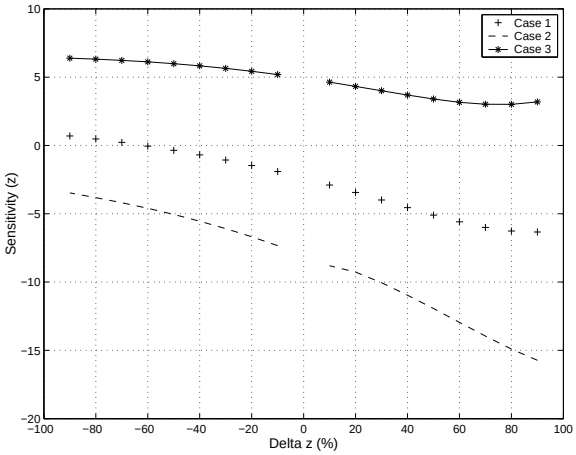

Figure 7: Sensitivity of Monetary cost to link speed change.

Fig. 8 on the other hand shows the result of the computation of sensitivity when the processor speed is varied. In this case the percentage change of processor speed $\Delta\omega$ is shown in the horizontal axis, whereas the change of the monetary cost given in ($eqn.6$) is shown in the vertical axis of this plot. As can be seen from this figure it is evident that for all the processor speed variations in the three cases (${C_i}^{\mathrm{l}} < {C_i}^{\mathrm{p}}$, ${C_i}^{\mathrm{l}} = {C_i}^{\mathrm{p}}$ and ${C_i}^{\mathrm{l}} > {C_i}^{\mathrm{p}}$) the cost of processors and link have positive values for the sensitivity axis.

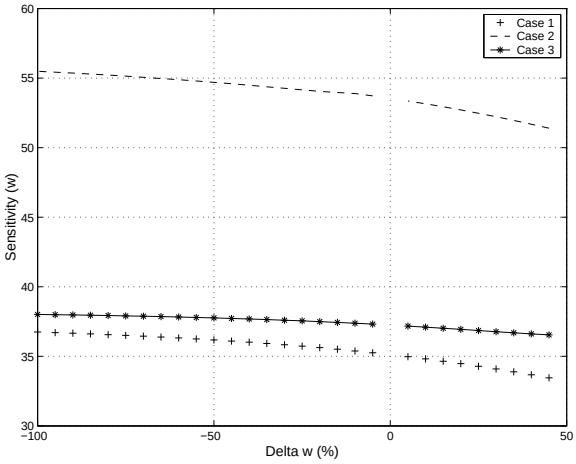

Figure 8: Sensitivity of Monetary cost to processor speed change.

The behavior of the sensitivity analysis for the total monetary analysis presented here is for specific set of values of parameters. That is, this is not a general result for all possible values of the parameters. This behavior is due to the complexity of the expression defined for α_n and its relationship with the total cost. Hence, the results obtained in this study show some of the representative curves of sensitivity of a single level tree network with specific network parameter values.

VIII. Conclusions

In this paper, the problem of load distribution sequencing for optimizing monetary cost or energy use in single level tree networks was presented. The monetary cost and energy use functions are functions of the fraction of loads that are assigned to each processor. Then by using four different swapping strategies a specified threshold level for improvement in cost or energy use was obtained. It was shown that by changing the order of load distribution in the network, better use of energy and a saving in monetary cost of the network can be obtained. This paper also gives directions for future research for the various network topologies used in divisible load theory.

References

[1] G. J. Pottie, W. J. Kaiser, "Wireless Integrated Network Sensors," *Communications of ACM*, vol. 43, no. 5, pp. 551–558, May 2000.

[2] A. Cerpa et al., "Habitat monitoring: Application driver for wireless communications technology," *2001 ACM SIGCOMM Workshop on Data Communications in Latin America and the Caribbean*, Costa Rica, April 2001.

[3] J. Sohn, T.G. Robertazzi and S. Luryi, "Optimizing Computing Costs using Divisible Load Analysis," *IEEE Transactions on Parallel and Distributed Systems*, vol. 9, no. 3, pp. 225–234, March 1998.

[4] J. Sohn, T.G. Robertazzi and S. Luryi, "Load Sharing Controller for Optimizing Monetary Cost," *US Patent 5,889,989*, March 1999.

[5] S. Charcranoon, T.G. Robertazzi and S. Luryi, "Parallel Processor Configuration Design with Processing/Transmission Costs," *IEEE Transactions on Computers.*, vol. 49, no. 9, pp. 987–991, Sept 2000.

[6] C. Intanagonwiwat, D. Estrin, R. Govindan and J. Heidemann, "Impact of Network Density on Data Aggregation in Wireless Sensor Networks," *Proceedings of the* 22^{nd} *International Conference on Distributed Computing Systems, ICDCS'02.*, pp. 414–415, 2002.

[7] B. Krishnamachari, D. Estrin and S. Wicker, "The Impact of Data Aggregation in Wireless Sensor Networks," 22^{nd} *International Conference on Distributed Computer Systems Workshop.*, pp. 575–578, 2002.

[8] W. Heinzelman, J. Kulik and H. Balakrishnan, "Adaptive Protocols for Information Dissemination in Wireless Sensor Networks," *Mobicom'99, Seattle, Washington.*, pp. 174–185, 1999.

[9] J. Kulik, W. Heinzelman, and H. Balakrishnan, "Negotiation-Based Protocols for Disseminating Information in Wireless Sensor Networks," *Wireless Sensor Networks.*, vol. 8, pp. 169–185, 2002.

[10] R. C. Shah and J. M. Rabaey Ra, "Energy Aware Routing for Low Energy Ad Hoc Sensor Networks," 3^{rd} *IEEE Wireless Communications and Networking Conference, Orlando, Florida.*, pp. 151–165, 2001.

[11] S. Charcranoon, T.G. Robertazzi and S. Luryi, "Cost Efficient Load Sequencing in Single-Level Tree Networks," *Proc. 1998 Conference on Information Sciences and Systems, Princeton University, Princeton. NJ*, March 1998.