OPTIMAL ALLOCATION OF COMPUTATION IN AN IOT NETWORK

by

Abhimanyu Chopra
A Thesis
Submitted to the
Graduate Faculty
of
George Mason University
In Partial fulfillment of
The Requirements for the Degree
of
Master of Science
Computer Engineering

Committee:

_____	Dr. Houman Homayoun, Thesis Director
_____	Dr. Avesta Sasan, Committee Member
_____	Dr. Cameron Nowzari, Committee Member
_____	Dr. Monson Hayes, Chairman, Department of Electrical and Computer Engineering
_____	Dr. Kenneth S. Ball, Dean, Volgenau School of Engineering
Date: _____	Spring 2017 George Mason University Fairfax, VA

Optimal Allocation of Computation in an IoT Network

A thesis submitted in partial fulfillment of the requirements for the degree of Master of Science at George Mason University

By

Abhimanyu Chopra
Bachelor of Technology
Maharshi Dayanand University, 2011

Director: Dr. Houman Homayoun, Professor
Department of Electrical and Computer Engineering

Spring 2017
George Mason University
Fairfax, VA

Dedication

This thesis work is dedicated to my parents Sanjeev & Madhu Chopra, their hardwork throughout my educational life has been a driving force for me. My sisters, Smriti and Khushi for their unconditional love. I also dedicate this work to my wife, Richa who has been a constant source of support and encouragement during the challenges of graduate school and life. This accomplishment would not have been possible without them.

Acknowledgments

I would like to express my sincere gratitude to my thesis advisor Dr. Houman Homayoun for his belief in me when I didn't believe in myself, for his patience, motivation, immense knowledge and continuous support throughout this research work.
I would also like to thank the rest of my thesis committee, Dr. Avesta Sasan and Dr. Cameron Nowzari for their support and insightful comments during the writing of this thesis.
Taking this opportunity I would like to thank Nisarg Patel, my fellow lab mate from the Green Computing and Heterogeneous Architectures (GOAL) Laboratory for the sleepless nights working on our research and all the fun we had in the past two years.

Table of Contents

Page

List of Tables vi
List of Figures vii
Abstract 0
1 Introduction 1
2 System Model & Problem Definition 4
3 Modeling & Formulation 7
3.1 Minimize Overall Energy Consumption (MOEC) 7
3.2 Maximize Minimum Energy Left on Any Node (MME) 8
3.3 Delay Deadline(DD) 9
4 Experimental Setup 12
4.1 Methodology for gathering real platform parameters 12
5 Observations 14
5.1 Minimize Overall Energy Consumption (MOEC) 14
5.1.1 Homogeneous Network 14
5.1.2 Heterogeneous Network 15
5.2 Maximize Minimum Energy Left on Any Node (MME) 16
5.2.1 Homogeneous Network 17
5.2.2 Heterogeneous Network 18
5.3 Delay Deadline (DD) 22
5.3.1 Homogeneous Network 22
5.3.2 Heterogeneous Network 23
5.4 Solver Overhead 32
6 Future Work and Concluding Remarks 34
Bibliography 35

List of Tables

Table		Page
2.1	Components of Energy Cost at a Node	5
2.2	Components of Delay at a Node	5
2.3	Platform Parameters & their description	6
3.1	Problem formulation for minimizing overall energy consumption	7
3.2	Maximize minimum energy left on any node	9
3.3	Minimize energy with deadline constraint	10
4.1	Real value of platform parameters	13
5.1	Proportionality of platform parameters on node's energy weight (A_i)	15
5.2	Task allocation strategy in a homogeneous IoT network with MME objective	17
5.3	Proportionality of platform parameters on node's delay weight (B_i)	23
5.4	Task allocation strategy in a heterogeneous network - DD objective	27

List of Figures

Figure		Page
2.1	IoT Network	4
5.1	Node's Energy Weight Comparison in a Homogeneous Network	14
5.2	Comparison of available energy on nodes	18
5.3	Comparison of available energy on nodes	19
5.4	Variation in energy levels of all nodes across the network	20
5.5	Assignment of computation work during the simulation	20
5.6	Available energy on nodes during MME simulation	21
5.7	Available energy on nodes during MME simulation / Time 0s-100s	22
5.8	Energy weight (A_i) for each node in the network	23
5.9	Delay weight (B_i) for each node in the network	24
5.10	Delay weight (A_i) for each node in the sample network	25
5.11	Delay weight (B_i) for each node in the sample network	26

Abstract

OPTIMAL ALLOCATION OF COMPUTATION IN AN IOT NETWORK

Abhimanyu Chopra

George Mason University, 2017

Thesis Director: Dr. Houman Homayoun

Internet of things (IoT) is being developed for a wide range of applications from home automation and personal fitness, to smart cities. With the extensive growth in adaptation of IoT devices, comes the uncoordinated and substandard designs aimed at promptly making products available to the end consumer. This substandard approach restricts the growth of IoT networks in the near future and necessitates studies to understand requirements of an efficient design. A particular area where IoT applications have grown significantly is the surveillance and monitoring. Applications of IoT in this domain are mainly relying on distributed sensors, each equipped with a battery, capable of collecting images, reprocessing images, and communicating the raw or processed images to the nearest node until it reaches the base station for decision making. In such an IoT network where processing can be distributed over the network, the important research question is how much of data each node should process and how much they should communicate to a given objective. This work answers this question and provides a deeper understanding of energy to delay(performance) trade off with different target metrics.

Chapter 1: Introduction

Internet of Things(IoT) is a system of devices, which has the ability to sense, compute and take some kind of action on the data like communicate the data, move an actuator etc. The data can be in the form of sound, image, temperature etc., based on geographic location and system application. Consumer-end products are quickly moving from traditional to network-enabled devices like Apple Watch and Fitbit. These devices have the ability to sense the user's heart rate, track number of steps and generate various reminders including those corresponding to water consumption, walking, competing with friends etc. Moreover, they consolidate data between different devices and provide the user with an overall view. Devices like the Nest have the ability to sense and control the temperature, lighting, air quality in the consumer's house and alert the user in case of abnormal conditions, or alternatively take steps to modify conditions based on pre-defined configurations. IoT has found its applications in numerous fields. One example is the deployment of IoT enabled-devices for object detection in large areas. For instance in forest, IoT devices are utilized to track animals, and near the international border they are used to check enemy infiltration. In smart cities ecosystem, IoT devices helps monitoring traffic or even pinpointing crimes. In all of these examples of IoT enabled ecosystems, IoT devices are interconnected and should operate in an energy-efficient manner in a resource constrained environment. The proliferation of IoT enabled devices in our homes, work space and every day life results in large amounts of data, which needs to be captured, computed, communicated and appropriately reacted upon in an energy-efficient and timely manner. There are several parameters influencing the trade-off between energy-efficiency and performance in an IoT systems, including the technology of computing platform of IoT devices (e.g. Intel Edison, Raspberry Pi, Arduino), the wireless technology interconnecting these devices (BLE, ZigBee, WiFi, Cellular), and the application these device are running for data processing (Image processing, DSP).

The battery capacity of an IoT device is usually very small. In fact, it is sometimes harvested using solar panels. Unlike other embedded systems, IoT devices can not afford to recharge their batteries frequently and are required to work for days or months on a single charge. This requires energy-efficient solutions to enhance their lifetime. An IoT network is also constrained by timing requirements when deployed for real-time applications, enunciating energy-efficient solutions while meeting performance requirements. As a result, energy-efficiency and performance are required to be simultaneously optimized in such IoT networks.

New platforms, algorithms and network protocols have been proposed and studied in the IoT space to minimize energy consumption. Lazarescu and Mihai(1) have presented a functional design and implementation of a complete wireless sensor network platform that can be used for a range of long-term environmental monitoring IoT applications. Serra et al.(2) introduced an energy-scheduling method that minimizes energy consumption cost based on the current energy price and user comfort constraints. Edalat et al(3) and, Yu and Prasanna (4) have proposed energy-aware allocation of tasks to increase network lifetime but allocated the task as a single unit rather than distributing the computation over the network.

Dynamic voltage scaling (5; 6) has been proposed (7; 8) to be used in IoT devices where the CPU works at lower frequencies to decrease the power consumption by regulating the voltage supplied to the system or shutting down in-active modules.

Rodoplu et al.(14) build a minimum energy communication network using a position based algorithm aiming to construct a topology consisting of lowest energy paths to transmit from any wireless sensor in a network to the sink node using the concept of relay transmission.

Many works (15) (16) (17) (18) (19) (20) (21) (22) have taken the optimal routing approach to minimize the communication energy cost. The approach is to minimize the energy consumed to reach the destination or sink node, which would minimize the energy consumed per packet or task. If all traffic is routed through the same path, batteries of

such nodes would run out quickly and that of other nodes would remain intact.

Optimal task/resource allocation has been extensively studied for wireless sensor networks. In (23) node's residual energy is taken into account and a centralized task allocation algorithm is proposed to improve the network lifetime. The task execution time or delay is taken into account in (24) and the same problem is analyzed.In (25) the authors provide a framework that distributes tasks to different nodes in a network by means of a distributed optimization algorithm based on gossip communication to increase network lifetime.

In this work, we study a common case IoT network where nodes are interconnected, collecting data, and in a cooperative manner processing and communicating the data through the network to a base station for decision making. Such common IoT network can be found in smart cities for surveillance or in environmental applications for monitoring the airborne quality, water quality, radiation, and many other environment indicators. We study IoT networks with various optimization objectives and constraint metrics to address energy-efficiency and performance requirements by dividing and efficiently allocating computation of workload data over the entire IoT network. The important research question that is raised in such IoT network is, how much data each node should communicate and how much locally compute for a given optimization goal. The allocation is optimized by analyzing the immediate platform-specific parameters such as computation energy, transmit energy, receive energy, available energy in each node and distance from the base station as well as reduction (or alternatively compression) of data based on the running application as it travels through the network and being processed by various nodes. The problem is categorized into a LP (Linear Programming) problem based on the optimization objectives and is solved using Scip Optimization Suite(9) and Symphony (10) from Coin-OR. This approach not only minimizes the overall energy consumption of the network, but also provides a balanced performance-energy solution with increased network lifetime. The results provide network designer with information to create an efficient IoT network based on application-specific requirements.

Chapter 2: System Model & Problem Definition

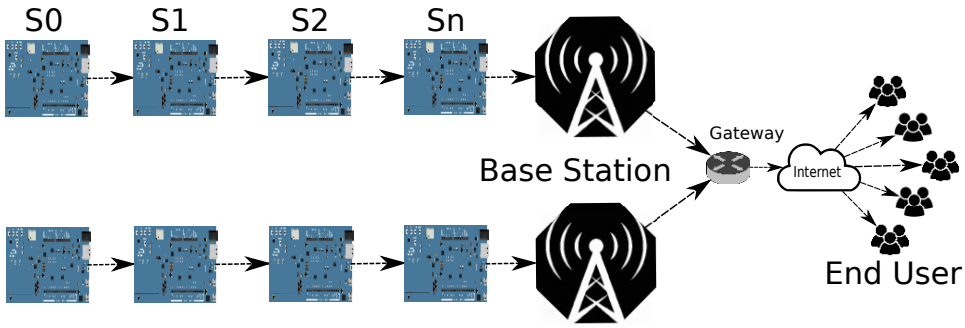


Figure 2.1: IoT Network

We consider a set of n sensor nodes, S = $\{S_i : i = 0, 1, 2, ..., n\}$, where S_0 is the source or generator of data and S_n is the sink and the only node connected to a base station. Each node powered with a small battery and is only capable of communicating with its immediate neighbors, i.e S_i can only communicate with S_{i+1} & S_{i-1}. The task for the sensor nodes is to compute and communicate data generated at S_0 to the base station. The task incurs an energy and delay cost on each sensor node.

The energy cost at each node N_i, has two components as defined in Table 2.1 viz. (1) Computation, which is a product of computation cost per bit P_i and the number of bits computed at that node x_i and (2) Communication, the sum of transmission and receive cost. Transmission and receive cost are defined as a product of the number of bits transmitted/received and the transmission cost T_i or receive cost R_i per bit respectively. At S_0, the receive cost is zero as it does not receive data from any other node.

The delay cost D_i for a node is defined by it's two components, Computation delay and Transmission delay, Table 2.2. When a node transmits data, the next node receives the data at the same time. Hence, we do not take into account any receive delay as both

Table 2.1: Components of Energy Cost at a Node

Energy Cost Component	**Description**
Computation	Computation Cost per Bit * Bits Computed
Communication	Transmission Cost + Receive Cost
Transmission	Transmission Cost per Bit * Bits Transmitted
Receive	Receive Cost per Bit * Bits Received

Table 2.2: Components of Delay at a Node

Delay Component	**Description**
Computation	Computation Delay per Bit * Bits Computed
Transmission	Bits Transmitted/Transmission Rate

transmission and receive happen simultaneously. Computation delay is the product of Delay due to Computation per bit D_{Pi} and the number of bits computed x_i and the Transmission delay is product of number of bits transmitted and the inverse of Transmission rate D_{Ti}. The platform specific parameters defined here are listed in Table 2.3.

When a node computes part of the total data size α, the size of the computed data reduces based on the compression factor. C_i is defined as 1-Compression factor. For e.g. if 100 bits are computed on a node with C_i of 0.1, the size of the computed data is only 10 bits. The node then transmits all the computed data along with the uncomputed data to its immediate neighbor towards the sink node. If there is any remaining uncomputed data when it reaches the sink node, all of it must be computed before being communicated to the base station. Such a IoT network is commonly used for surveillance, traffic control, and environmental monitoring, where sensors are collecting and cooperatively processing and

transmitting the data to the base for decision making.

We study this type of common IoT network to find out the optimal assignment of computation vs communication work with various objectives such as meeting a deadline, or energy-efficiency.

Table 2.3: Platform Parameters & their description

Parameter	**Description**	**Unit**
N_i	Energy Cost Incurred at node i	J
D_i	Delay Cost Incurred at node i	s
x_i	Data allocated to be computed at node i	bits
P_i	Computation Cost per Bit at node i	J/bit
T_i	Transmission Cost per bit at node i	J/bit
R_i	Receive Cost per Bit at node i	J/bit
C_i	(1 - Compression Factor) at node i	-
E_i	Energy Level at node i	J
D_{Pi}	Delay due to Computation per bit at node i	s/bit
D_{Ti}	Transmission rate at node i	bit/s
α	Data Size	bit
δ	Delay Deadline	s

Chapter 3: Modeling & Formulation

For optimization purposes, we define our model for various metrics and formulate the problem into its objective and constraints.

3.1 Minimize Overall Energy Consumption (MOEC)

Minimize Overall Energy (MOEC) objective is to minimize the overall energy consumption over the entire network, thus the sum of energy consumption throughout all the nodes in the network should be the least.

Table 3.1: Problem formulation for minimizing overall energy consumption

MOEC problem constraints			
Minimize	$\sum_{i=0}^{n}$	N_i	(a)
Subject to	$\sum_{i=0}^{n}$	$x_i = \alpha$	(b)

Table 3.1 shows the parameters employed for the linear programming formulation. Expression(a) is the objective of the problem. $\sum_{i=0}^{n} N_i$ is the sum of the energy consumed on each node in the system, which takes into account the energy used to compute and communicate data to the next node. It is defined as:

$$N_i = \overbrace{P_i x_i}^{Computation\ Energy\ Cost} + \overbrace{T_i(\alpha - \sum_{j=0}^{i} x_j + \sum_{j=0}^{i} C_j x_j)}^{Transmit\ Energy\ Cost} + \overbrace{R_i(\alpha - \sum_{j=0}^{i-1} x_j + \sum_{j=0}^{i-1} C_j x_j)}^{Receive\ Energy\ Cost} \quad (3.1)$$

When simplified in terms of x_i, $\sum_{i=0}^{n} \quad N_i$ can be represented as,

$$\sum_{i=0}^{n} A_i x_i + \beta \tag{3.2}$$

where, A_i is the weight/coefficient of x_i towards the total energy consumption and ,

$$A_i = P_i + \{\sum_{j=i}^{n-1} T_j(C_i - 1)\} + T_n * C_i + \sum_{j=i+1}^{n} R_j(C_i - 1) \tag{3.3}$$

and constant β is,

$$\beta = \alpha * (\sum_{j=0}^{n-1} T_j + \sum_{j=1}^{n} R_j) \tag{3.4}$$

It should be noted that A_i defined as a node's energy weight takes both communication as well as computation cost into account.

Expression(b) in Table 3.1 is the constraint, where x_i is the amount of data assigned to be computed on node i and their sum should be equal to the data size α, to make sure that all data is computed within the network.

3.2 Maximize Minimum Energy Left on Any Node (MME)

The MME's objective is to maximize the minimum energy left on any node. This objective is correlated with the network lifetime. The network is considered to be defunct when any node's energy depth reaches zero. This problem is formulated in two steps as shown Table 3.2, where we calculate a number of optimal solutions in step 1 and reduce the search area further in step 2.

Table 3.2: Maximize minimum energy left on any node

MME problem constraints			
Step 1	Minimize	N_m	(c)
	Subject to	$\forall\ i \in\ \{0,..,n\}, i \neq m:\ E_i - N_i \geq E_m$	(d)
		$\sum_{i=0}^{n}\ \ x_i = \alpha$	(e)
Step 2	Minimize	$\sum_{i=0}^{n}\ \ N_i$	(f)
	Subject to	$N_m \leq$ Optimal Solution from Step 1	(g)
		$\forall\ i \in\ \{0,..,n\}:\ E_i - N_i \geq E_m$	(h)
		$\sum_{i=0}^{n}\ \ x_i = \alpha$	(i)

We consider a sensor node S_m having the least energy level E_m among all the nodes in the network. To maximize E_m, we formulate the objective such that the solution consumes the least energy on S_m to complete the task i.e., Minimize N_m, expression (c). It is essential to check if the objective unnecessarily penalize other nodes in the network and hence expression(d) restricts the energy consumption on all other nodes such that their energy level do not fall below E_m . Expression(e) makes sure that all data is computed within the network. It should be noted that we derive many optimal solutions in step 1, each having different overall energy consumption. An additional constraint (i.e., g) derived from step 1 is utilized in step 2 to minimize the energy consumption on node N_m and subsequently minimize the overall energy consumption to find the solution for our bi-objective problem, from the pool of optimal solutions.

3.3 Delay Deadline(DD)

The Delay Deadline(DD) objective is defined to observe the effect of computation allocation on energy and delay for real-time applications in an IoT network. The objective of the metric is to minimize the overall energy consumption while meeting the performance or deadline

requirement of the task. The deadline δ in expression (k) may be a user defined constraint or a quality of service assurance by the network which restricts the end to end delay in the network.

Table 3.3: Minimize energy with deadline constraint

DD problem constraints			
Minimize	$\sum_{i=0}^{n}$	N_i	(j)
Subject to	$\sum_{i=0}^{n}$	$D_i \leq \delta$	(k)
	$\sum_{i=0}^{n}$	$x_i = \alpha$	(l)

To observe the delay associated with the network, we study various sources of delay on a node. As discussed in section 2, the delay on each node is defined to be the sum of the computation and transmission delay. We restrict ourselves from indulging into queuing theory and assume queuing delay to be zero. Since the distances between two neighbor nodes is small in an IoT network, the propagation delay is assumed to be zero.

Thus, The delay on each node is formulated as follows:

$$D_i = \overbrace{D_{Pi}x_i}^{Computation\ Delay} + \overbrace{(\alpha - \sum_{j=0}^{i} x_j + \sum_{j=0}^{i} C_j x_j)/D_{Ti}}^{Transmission\ Delay} \tag{3.5}$$

$\sum_{i=0}^{n} D_i$ can be simplified in terms of x_i and be represented as,

$$\sum_{i=0}^{n} B_i x_i + \lambda \tag{3.6}$$

where, B_i is the weight/coefficient of x_i towards the total delay and ,

$$B_i = D_{Pi} + \{\sum_{j=i}^{n-1}(C_i - 1)/D_{Tj}\} + C_i/D_{Tn} \tag{3.7}$$

and constant λ is,

$$\lambda = \alpha * (\sum_{j=0}^{n-1} T_j) \tag{3.8}$$

It should be noted that B_i defined as a node's delay weight takes both communication as well as computation delay cost into account.

Chapter 4: Experimental Setup

The experimental setup used in this work is discussed in this section. A simulator is developed in the C-language, which uses platform and network parameters and synthesizes the problem into a solver specific format based on the required objective. We use two solvers to obtain the optimal solution to the problem generated by the simulator, Symphony(10) and Scip optimization suite(9). We perform image feature detection using Harris Corner and Canny Edge detection from the OpenCV Suite(11) on NVIDIA Jetson TK1, Intel Galileo and Intel Edison boards to obtain real world values of platform parameters. These are low power embedded platform which are designed for a wide range of low-end to high-end IoT application processing. The power consumption of all three platforms is measured using Watts up PRO power meter (12) and the timing is measured using the CPU timing functions available under the GNU C library. The communication cost used is measured at 0.00525uW/bit for Wi-Fi when transmitting a 40 Mbps User Datagram Protocol (UDP) payload, 0.153 uW/bit for BLE and 185.9 uW/bit for Zigbee in our experiments (13).

4.1 Methodology for gathering real platform parameters

Three different IoT platforms were selected to be used in our experiments. The platforms power are measured using the Watts up PRO power meter which provides energy consumption per second. When no task is running on the platforms, the reading from the power meter is considered as the platform's idle power consumption. Two image feature detection tasks are executed on each platform. The first task is executed five times with a delay intervals of 120 seconds, and contains nine hundred iterations of Harris corner detection on a 2560x1600 image. The 120 second delay intervals between each task is necessary to allow the platform to stabilize to its idle energy consumption. The idle energy consumption

is subtracted from the reading observed when the task is executed to obtain the dynamic power consumed by the task. The acquired value is divided by the size of total processed data in bits to obtain the energy consumption per bit. The task's start and finish time-stamp is used calculate the total time in seconds required to complete the task. The value is then divided by the total size of data processed to calculate the delay per bit for each platform. Same experiments are then performed for canny edge detection algorithm. Result for these experiments is shown in table 4.1.

Table 4.1: Real value of platform parameters

Algorithm / Platform	**Harris Corner**		**Canny Edge**	
	J/bit	s/bit	J/bit	s/bit
Nvidia Jetson	3.87E-008	1.16E-008	2.02E-008	6.21E-009
Nvidia Jetson Low Power	2.47E-008	2.36E-008	1.01E-008	1.31E-008
Intel Galileo	7.89E-008	4.32E-007	2.58E-008	1.30E-007
Intel Edison	1.91E-008	4.71E-008	6.63E-009	2.80E-008

Chapter 5: Observations

In this section we use our simulation data to discuss the observations regarding the three optimization objectives.

5.1 Minimize Overall Energy Consumption (MOEC)

5.1.1 Homogeneous Network

For MOEC objective in a homogeneous network, the optimal solution is to compute all the data at the first node. The first node always has the least energy weight in the network as shown in figure 5.1. This solution attempts to reduce the communication overhead of data and is optimal as computation cost for all nodes in the network is the same, whether we compute at a single node or distributed the computation to several nodes.

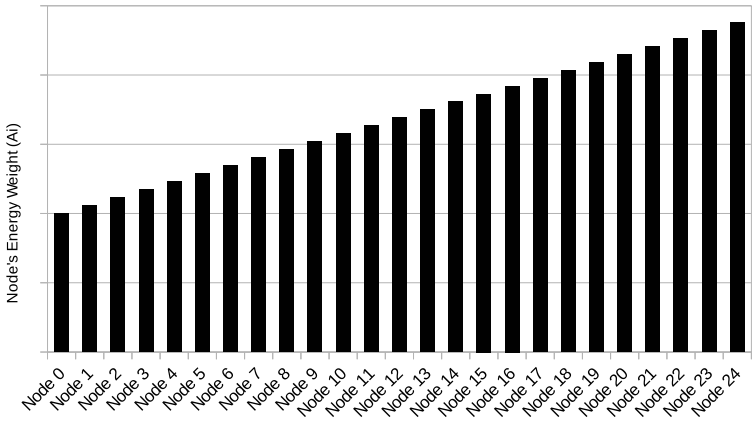


Figure 5.1: Node's Energy Weight Comparison in a Homogeneous Network

5.1.2 Heterogeneous Network

For a heterogeneous network, the results show that all computation work should be assigned to the node having the least energy weight (A_i), to reduce the total power consumption. It is important to note that the node weight defined as A_i, take both communication as well as computation cost into account. It is observed that, an increase in the computation cost (P_i) results in an increase in the weight/coefficient of the node. An increase in transmission cost T_i, receive cost R_i & compression factor negatively affects the weight.[Table 5.1].

Table 5.1: Proportionality of platform parameters on node's energy weight (A_i)

Platform Parameter	**Proportionality**
Computation cost per bit	Proportional
Transmission cost per bit	Inversely Proportional
Receive cost per bit	Inversely Proportional
Compression Factor	Inversely Proportional

Based on the network configuration and platform specifications, in special cases, the lowest node weights correspond to two or more nodes. In these cases, we can distribute the computation work on the nodes with equal energy weights in any bias and minimize the total energy consumption of the system.

To show that with minimizing the total power consumption as our objective, carrying out the entire computation on one node is always the best solution, we use a counter argument.

Proof. The overall energy consumption of the system when simplified in terms of x_i is given by:

$$\sum_{i=0}^{n} A_i x_i + C$$

Let us assume that two nodes (i.e. Y and Z) have large coefficients and node M has the

least coefficient in the above equation. We assume that the overall energy consumption of the system is least when work is assigned to node Y and node Z. Thus,

$$A_Y x_Y + A_Z x_Z + C \leq A_M x_M + C \quad ; \quad x_Y + x_Z = x_M$$

But,

$$A_M \leq A_Y \quad | \quad A_M \leq A_Z$$

Multiplying both sides of the two equations with x_Y and x_Z respectively, we get:

$$A_M x_Y \leq A_Y x_Y \quad | \quad A_M x_Z \leq A_Z x_Z$$

Adding both the equations, replacing $x_Y + x_Z$ with x_M and adding constant C on both the sides,

$$A_M x_M + C \leq A_Y x_Y + A_Z x_Z + C$$

which contradicts our assumption. □

Thus carrying out all the computation on node M would be the most efficient solution given the MOEC objective.

5.2 Maximize Minimum Energy Left on Any Node (MME)

We observe that when the battery life-time constraint is added to our optimization objective, the best solution is to distribute the computation work among several nodes. This task distribution leads to increased overall energy consumption of the network but allows a longer network lifetime. With the MOEC metric, since only one of the nodes is carrying the entire computation load, it is stressed with its energy depleting at a faster rate than other nodes in the network. The network's lifetime depletes before the available energy on all nodes is utilized. In MME, our solver provides an energy-aware solution which leads to maximum utilization of energy on each node.

5.2.1 Homogeneous Network

Table 5.2: Task allocation strategy in a homogeneous IoT network with MME objective

Step 1	Search node S_{min} with minimum energy E_{min}
Step 2	Find least energy weight node S_{Amin0} earlier in the sequence than S_{min}
Step 3	Allocate maximum percentage of data $x_{S_{Amin0}}$ such that available energy remain above E_{min} after processing.
Step 4	If $\sum_{i=0}^{n} x_{S_{Li}} < \alpha$ Find next least weight node S_{Li+1} Go to Step 3 Else,
Step 5	Finish.

As discussed in section 5.1.1, the first node has the least energy weight towards total power consumption in a homogeneous network. If the first node is not the one having the least energy available in the system, then all data is initially processed on it. The allocation requires a change, if processing all data on the node results in its energy to go below the minimum energy in the network. Allocation of computation is then done such that the energy level stays above the minimum energy and the rest of the computation is allocated to the next available node with the least weight. Table 5.2 shows this allocation strategy. The allocation is done such that all data is computed before reaching the node with the least energy level. This minimizes the energy consumption on the node with minimum energy S_{min} by reducing the communication overhead by reducing the size of the workload after computation on previous node. In case none of the previous node can be used for allocation due to energy level restriction, the node S_{min} or the one after can be used based on their energy weights. It is noted that when the cost of computation is small, noticeably

less than the cost of communication, the rate at which the battery of the first node depletes is becoming small such that the data computation may never be offloaded to the next node.

5.2.2 Heterogeneous Network

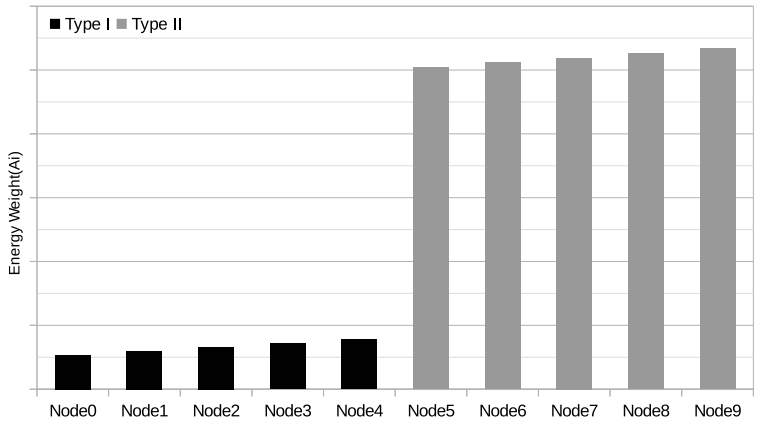


Figure 5.2: Comparison of available energy on nodes

In a heterogeneous network, the computation allocation follows a similar strategy to a homogeneous network.

To better understand the distribution of computation in a heterogeneous network we provide a simple example. The energy weights of the first 10 nodes are shown in figure 5.2. In figure 5.3, the y-axis represent the available energy on each node and the x-axis represent each nodes in the network. To account for heterogeneity in the network we divide nodes into two groups. Node0-Node4(Type I) and Node5-Node24(Type II) are two groups of nodes. Type I has a lower computation and communication energy cost (weight) than Type II and initially have equal available energy at 10800 J(Fully charged AA batteries). Node 5 is an exception with the least available energy in the network at some random value, 10400 J. MME 0 is the initial state of the network. MME 20 is the available battery on each node after twenty tasks are computed on the network and MME 100 is the observations after one

hundred tasks have been computed on the network. We notice that the depletion of energy on nodes is balanced by the solution and the average variation in energy levels of all nodes in the network is reducing as more task is being processed and the solution attempts to avoid reducing the energy of the battery on the node(s) with minimum energy Figure 5.4.

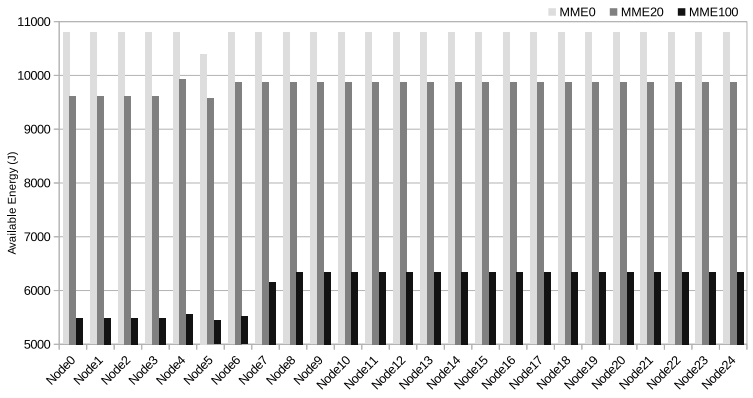


Figure 5.3: Comparison of available energy on nodes

We explain the allocation by using two figures, one which shows us the energy levels at different nodes after completion of each task and the other showing allocations per task.

Figure 5.5 shows assignment of computation work during the simulation time. Each column in the graph shows the allocation of computation suggested by the simulator. The y-axis is the percentage of work allocated on a node and the x-axis is the simulated time. Every time interval (1 second) a new task is injected into the network, computed at optimal locations and communicated to the base station. As expected, initially, the first node or the node with the least weight gets majority of the allocation to reduce the energy consumption and gradually the work is offloaded to the nodes with the least weight from the pool of available nodes to preserve the network's life-time. Figure 5.6 contains three subplots which show the available energy on each node during different time slot. Figure 5.7 shows the overall trend from time zero to the end of simulation where as 5.6a and 5.6b are for shorter time intervals for better presentation of trends and results. In figure 5.6a,

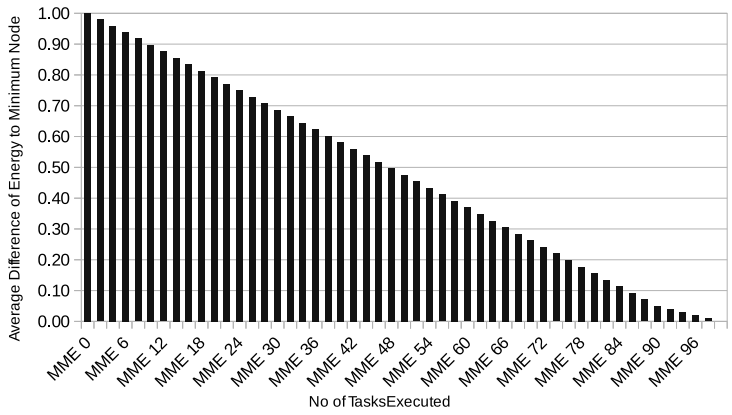


Figure 5.4: Variation in energy levels of all nodes across the network

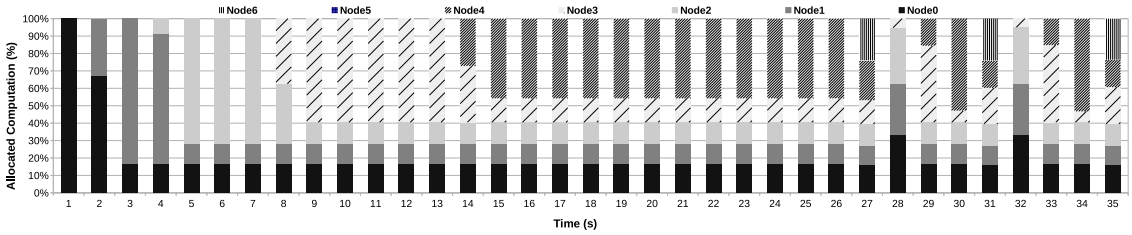
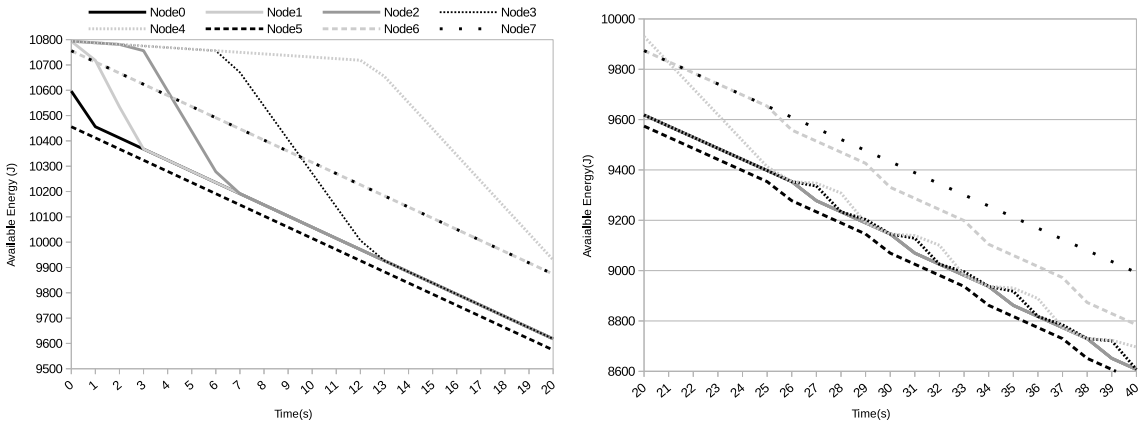


Figure 5.5: Assignment of computation work during the simulation

the offloading point can be identified as the point when the available energy on the node with majority of the allocation reaches close to the energy level of the node with least available energy in the network, i.e. Node 5 in this case. At this point, the node cannot keep consuming the same amount of energy and computation load needs to be offloaded to an alternate node to prevent its energy depleting below the least allowable level in the network, i.e. energy level of Node 5. Thus, after the initial allocation, alternate nodes are allocated with the majority of the computation, and the initial nodes are allocated a constant computation load such that their available energy stays above the least allowable



(a) Time 0s-20s

(b) Time 20s-40s

Figure 5.6: Available energy on nodes during MME simulation

level in the network. As the energy level on Node 4 reaches close to Node 5, we observe a slightly different behavior. Node 5 is skipped for computation allocation and computation is moved to Node 6. This causes the energy depletion rate of Node 5 to increase since a larger data size is communicated to the next node(Processed Data + Unprocessed data). Subsequently, when next time a new task is injected into the network, the gap between the energy level of Node0-Node4 to that of minimum battery node increases such that more data is allocated back to these nodes. This pattern occurs recursively till the energy level at Node 6 depletes to that of Node 5. Next, Node 7 is allocated some data and the pattern repeats with Node0-Node7. This pattern is clearly shown in figure 5.6b and figure 5.7.

An interesting observation from figure 5.6 is regarding the network life-time which can be inferred from the slope plotted between the available energy and time. The slope represents the depletion rate of available energy on a node. The slope for each node is proportional to the computation and communication cost of the node and can be used to understand whether the computation will be offloaded to the next efficient node with the least weight. If the slope of the node with minimum energy is smaller (negative) than the most efficient

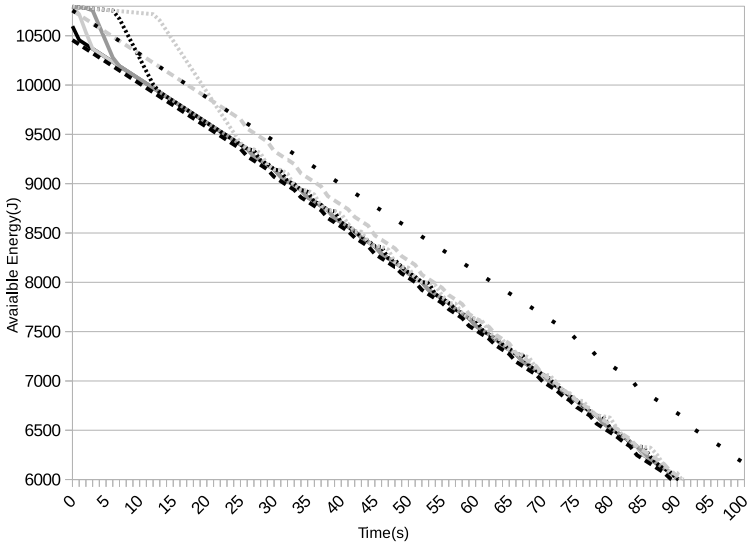


Figure 5.7: Available energy on nodes during MME simulation / Time 0s-100s

node, the computation will not be offloaded. The network lifetime increases as the slope positively increases towards zero and decreases when it moves in the other direction.

5.3 Delay Deadline (DD)

For the delay deadline optimization objective, we study the effect of deadline requirement on the computation allocation. With this metric we are able to find out the computation allocation that provides least possible delay with the least energy consumption of the network. The allocation can be used to guarantee a minimum quality of service with minimized energy consumption.

5.3.1 Homogeneous Network

For a homogeneous network, the first node is the most efficient node in terms of energy consumption & delay. If all the data is processed on the first node, the overhead of communication is the least, as data size gets reduced by some compression C_i. As we compute data

away from the source node, the delay increases and reaches its maximum when the entire data is computed on the last node. It should be noted that the node's delay weight is proportional to the computational delay per bit and inversely proportional to the transmission rate and the compression factor, table 5.3.

Table 5.3: Proportionality of platform parameters on node's delay weight (B_i)

Platform Parameter	**Proportionality**
Computational delay per bit	Proportional
Transmission rate	Inversely Proportional
Compression Factor	Inversely Proportional

5.3.2 Heterogeneous Network

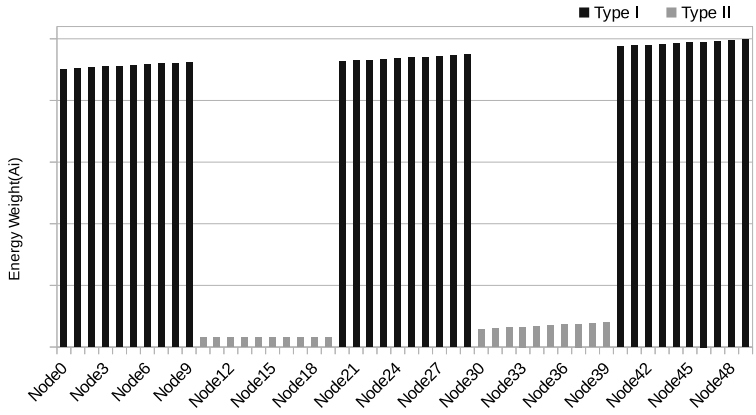


Figure 5.8: Energy weight (A_i) for each node in the network

In this section, we first look at a heterogeneous system made of two types of nodes placed alternatively in groups. We look at the A_i & B_i(introduced in section 3.2 and 3.3),

the energy weight and the delay weight of each node to understand allocation. In figure 5.8 when comparing identical type of groups, we notice that all nodes of a group closer to the source are always more efficient in terms of energy(first node of the group being the most efficient) than a group placed later in the network. The same trend can be seen in figure 5.9 for delay weights for each node. These observations can be used to find out candidate nodes for computation allocation in a network.

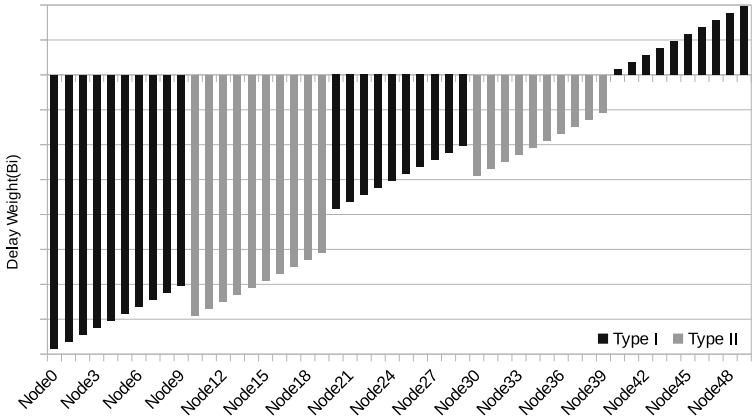


Figure 5.9: Delay weight (B_i) for each node in the network

In figure 5.8, Node 10 has the least weight towards energy consumption and is the best candidate to start our allocation strategy with. All subsequent nodes within the homogeneous group and of other identical groups in the network are not considered candidates as they will have a higher weight towards energy consumption as well as delay. All data can be allocated to be processed on the candidate node and checked for deadline adherence. This is the most efficient node in the network and network designers should make sure that most of their workload deadlines can be with this node itself. If the solution does not meet the deadline, the next candidate to offload data is searched. It should be noted that if the least weight node in terms of energy consumption and delay is the same and does not meet the deadline, it is impossible to find any other computation allocation that adheres to the

deadline. The next candidate is Node 0 in this example, as this is the node with the least energy weight from the left over nodes with an improved delay weight. The allocation is now iteratively increased on this new candidate and reduced on the previous one until the solution is in adherence with the required deadline. The optimal solution for computation allocation will always have at max two nodes sharing computation.

The complete allocation strategy is listed in table 5.4 and another example is used to explain it. We look at a sample heterogeneous network with energy weights and delay

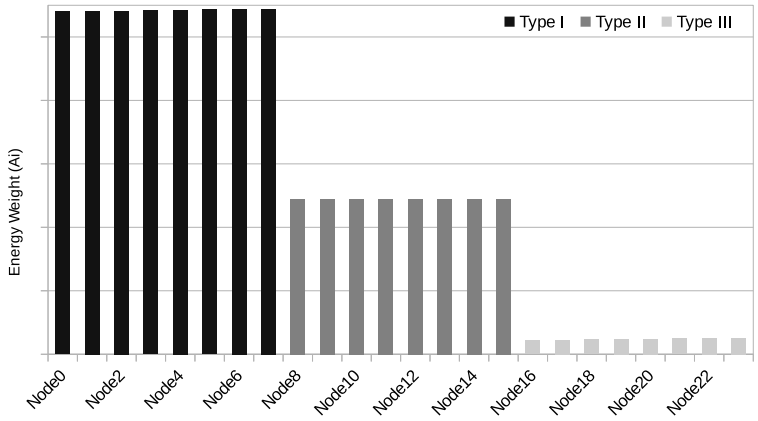


Figure 5.10: Delay weight (A_i) for each node in the sample network

weights of each node as shown in figure 5.10 and 5.11 respectively. Following the task allocation strategy, we first find the node with the least and second least energy weight in the network. In this example that node is Node 16 and Node 8. The next step is to allocate all the data on Node 16 and check if this solution adheres to the required deadline. If the deadline is met, we have the optimal solution for the defined deadline else we find all possible candidates for transfer of allocation to meet the deadline. The candidates in this example are Node 8 and Node 0 as they have much smaller delay weight B_i than Node 16. Here Node 9-Node 15 and Node 1-Node 7 are not candidates for transfer of allocation as the first node in a homogeneous group has the best energy and delay weight. With

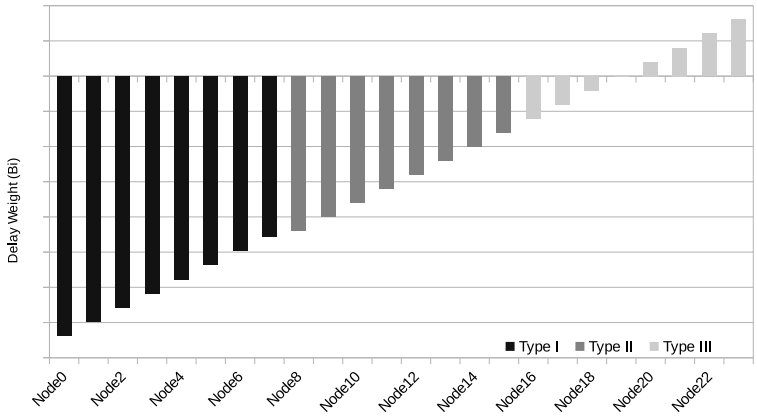


Figure 5.11: Delay weight (B_i) for each node in the sample network

extensive simulations we found out that we get optimal solutions with two or less nodes. Hence, we find out the allocation (Node16,Node8) and (Node16,Node0) separately such that the allocation meets the deadline. We compare the total energy consumption between the two and choose the minimum. It should also be noted that if there is any possible allocation such that the deadline is met with the top two candidates which are Node 16 and Node 8, the optimal solution can be found with the best candidate (Node16) and other candidates(Node8,Node0) viz. (Node16,Node8) and (Node16,Node0). If not, we have to look at other possible combinations like (Node8,Node0) and choose the one which provides the best energy consumption.

Table 5.4: Task allocation strategy in a heterogeneous network - DD objective

Step 1	Find node S_m with least energy weight A_m
Step 2	Find node S_n with least energy weight s.t
	$A_n \geq A_m$ and $B_n < B_m$
Step 3	Allocate 100% data to be computed on S_m
Step 4	Check if solution meets deadline
	If True, finish.
	Else,
Step 5	Check if B_m is least in the system
	If True, Deadline cannot be met.
	Else,
Step 6	Find candidates for transfer of allocation
Step 7	For all candidates S_i, find (x_i, x_{mi}) such that,
	$B_i x_i + B_m x_{mi} \leq \delta$, where $x_i + x_{mi} = \alpha$
Step 8	If there exists no solution for $B_n x_n + B_m x_m \leq \delta$
	Store $Z \leftarrow Min(\forall i \in candidates : A_i x_i + A_m x_m)$
	$S_m = S_n$, Go to Step 2
	Else,
Step 9	$Min(\forall i \in candidates : A_i x_i + A_m x_m, Z)$, finish.

We now show that for the deadline delay objective two or less than two nodes will only provide an optimal solution, we use a counter argument.

Proof. Let us assume we have two solutions that adhere to the deadline δ, one where two nodes participate and the other where an extra node participate in the solution. Let us name these nodes, S_D, S_E, S_F with energy weights A_D, A_E, A_F.

We have the following assumptions:

$$A_D < A_E < A_F \tag{5.1}$$

$$\overbrace{x_D + x_E + x_F}^{3\ node\ solution} = \overbrace{x_G \quad + \quad x_H}^{2\ node\ solution} = \alpha \tag{5.2}$$

We look at three cases and assume that energy consumption is less when allocation is done on three nodes than on two nodes. We then prove that our assumption is incorrect.

Case I : When deadline is met with S_D, S_E

Case I.A : When moving from a 2 to 3 node solution some part of x_H is off loaded to the third node for computation

$$x_G = x_D \quad \& \quad x_H = x_E + x_F \tag{5.3}$$

Comparing energy consumption between the two solutions

$$A_D x_D + A_E x_E + A_F x_F + \beta < A_D x_G + A_E x_H + \beta$$

Using equation 5.3, we get

$$A_D x_D + A_E x_E + A_F x_F + \beta < A_D x_D + A_E x_E + A_E x_F + \beta$$

Removing equal terms on both sides, we get

$$A_F < A_E$$

Which contradicts are initial assumption in equation 5.1.

Case I.B : When moving from a 2 to 3 node solution some part of x_G is off loaded to the third node for computation

$$x_H = x_E \quad \& \quad x_G = x_D + x_F \tag{5.4}$$

Comparing energy consumption between the two solutions

$$A_D x_D + A_E x_E + A_F x_F + \beta < A_D x_G + A_E x_H + \beta$$

Using equation 5.4, we get

$$A_D x_D + A_E x_E + A_F x_F + \beta < A_D x_D + A_D x_F + A_E x_E + \beta$$

Removing equal terms on both sides, we get

$$A_F < A_D$$

Which contradicts are initial assumption in equation 5.1.

Case I.C : When moving from a 2 to 3 node solution some part of $x_G \& x_H$ is off loaded to the third node for computation

$$x_G > x_D \quad \& \quad x_H > x_E \tag{5.5}$$

Comparing energy consumption between the two solutions

$$A_D x_D + A_E x_E + A_F x_F + \beta < A_D x_G + A_E x_H + \beta$$

Using equation 5.2, we get

$$A_D x_D + A_E x_E + A_F x_F + \beta < A_D x_G + A_E(x_D + x_E + x_F - x_G) + \beta$$

Removing equal terms and moving remaining terms to L.H.S.

$$x_F(A_F - A_E) + (x_G - x_D)(A_E - x_D) < 0$$

But,

$$x_F(A_F - A_E) > 0, since \quad A_F > A_E \quad \& \quad x_F > 0$$

$$(A_E - x_D) > 0, since \quad A_E > A_D$$

$$(x_G - x_D) > 0, since \quad x_G > x_D$$

which contradicts our assumption.

Case II : When deadline can be met with S_E, S_F, but not S_D, S_E

In this case if the deadline can be met with S_D, S_E, S_F, it can also be met with S_D, S_F.

$$i.e \quad B_D > B_E > B_F \tag{5.6}$$

Comparing energy consumption of all three solutions, we assume the three node solution consumes less energy than a two node solution.

$$A_D x_D + A_E x_E + A_F x_F + \beta \leq A_D x_G + A_F x_H + \beta$$

Simplifying the equation and moving all terms to L.H.S

$$x_D(A_D - A_E) + (x_H - x_F)(A_E - x_F) \leq 0 \tag{5.7}$$

In the above equation,

$$x_D > 0$$

$$(A_D - A_E) < 0$$

$$(A_E - x_F) < 0$$

Hence, for the equation 5.7 to hold true, we have two cases

$$(x_H - x_F) \geq 0 \quad or \quad (x_H - x_F) < 0$$

We now prove that in case of $(x_H - x_F) \geq 0$ the deadline cannot be met, and for $(x_H - x_F) < 0$, S_E, S_F provide better energy consumption.

Case II.A : $(x_H - x_F) \geq 0$

Comparing delays between the two solution,

$$B_D x_D + B_E x_E + B_F x_F + \lambda \leq B_E x_G + B_F x_H + \lambda$$

Using equation 5.2 and simplifying,

$$x_D(B_D - B_E) + (x_H - x_F)(B_E - B_F) \leq 0 \tag{5.8}$$

But,

$$x_D > 0$$

$$(B_D - B_E) > 0$$

$$(B_E - B_F) > 0$$

$$(x_H - x_F) \geq 0$$

which contradicts equation 5.8.

Case II.B : $(x_H - x_F) < 0$ or $(x_F - x_H) > 0$

Comparing energy consumption between the two solutions.

$$A_D x_D + A_E x_E + A_F x_F + \beta \leq A_E x_G + A_F x_H + \beta$$

Simplifying the equation and moving all terms to L.H.S

$$x_E(A_E - A_D) + (x_F - x_H)(A_F - x_D) \leq 0 \tag{5.9}$$

But,

$$x_E > 0$$

$$(A_E - A_D) > 0$$

$$(A_F - x_D) > 0$$

$$(x_F - x_H) > 0$$

which contradicts equation 5.9.

Hence, we prove that allocation on two node will always consume less energy for a delay requirement than on a three node network. □

5.4 Solver Overhead

In this section We discuss the solver overhead. We assume that the source is running the solver, for every incoming task to the network, and the source node is aware of each node computing and communicating cost. This information has to be communicated once to the source node, and at the beginning of the first task injected into the network. Based on platform parameters of each node, the source node calculates and keeps track of their battery level. Hence, for the MME metric, the battery level information does not require to be sent after every new incoming task. Every time a new task is injected into the network,

the source node runs the solver once to obtain the optimal allocation solution. The solver has negligible overhead in comparison to the task studied in this work. For a small task, SD Frame of 922Kb runs on Nvidia Jetson with BLE($D_{T_i} = 1.00E{-}006$), the computation delay is 0.046s and communication delay at 70% compression rate is 2.26s. In a homogeneous system, assuming that all data is computed on the first node which is the most efficient, the total delay is 2.3s plus an additional 2.26s for every node it needs to communicate, to reach to the base station. In Comparison, the average solver run time is 0.002s and hence the total overhead of the solver is estimated to be around 0.08%. Including the communication cost even reduces the overhead further and below 0.01%. For larger tasks, the solver run time remains the same leading to an even lower solver overhead. If we look at the energy consumption of the solver, it requires 0.06J for MME metric on average for any data size whereas for a 10 node homogeneous network the total energy consumption of the network is 0.46J. It should be noted that the solver has many I/O operations to save results which when removed would minimize the overhead.

Chapter 6: Future Work and Concluding Remarks

Energy limits and performance requirements are a real issue for IoT networks. This work is an effort to thoroughly study and analyze IoT networks using three different optimization metrics. The metrics were formulated into an ILP(Integer Linear Programming) problem and the results show that when energy efficiency is the only objective, it makes sense to process all the data on a single node. When performance and network lifetime is a constraint, the results show that their is benefit to distribute the computation on nodes that comes in its path, as it travels towards the sink node. The observations can be used by network designers to understand the energy delay trade off within their network and appropriately design the network such that the most energy efficient node or least energy weight node can be identified and used to assure quality of service with low energy consumption. The results can also be used for appropriate platform selection and battery allocation in a network.

This work can be further used in interpreting the optimal allocation for different networks. For a completely connected network after we find the optimal paths, the optimal allocation can be done using this work. We can conclude that in a energy efficiency metric for a completely connected network the optimal solution would be to allocate computation on one node i.e the node with the least energy weight. The case specific allocation strategies can be efficiently coded and the optimal solution can be found without using a general purpose solver.

Bibliography

[1] M. T. Lazarescu, "Design of a wsn platform for long-term environmental monitoring for iot applications," *IEEE Journal on emerging and selected topics in circuits and systems*, vol. 3, no. 1, pp. 45–54, 2013.

[2] J. Serra, D. Pubill, A. Antonopoulos, and C. Verikoukis, "Smart hvac control in iot: Energy consumption minimization with user comfort constraints," *The Scientific World Journal*, vol. 2014, 2014.

[3] N. Edalat, W. Xiao, C.-K. Tham, E. Keikha, and L.-L. Ong, "A price-based adaptive task allocation for wireless sensor network," in *2009 IEEE 6th International Conference on Mobile Adhoc and Sensor Systems*. IEEE, 2009, pp. 888–893.

[4] Y. Yu and V. K. Prasanna, "Energy-balanced task allocation for collaborative processing in wireless sensor networks," *Mobile Networks and Applications*, vol. 10, no. 1-2, pp. 115–131, 2005.

[5] M. S. Krishnan, "Dynamic voltage scaling," Jun. 15 2010, uS Patent 7,739,531.

[6] J. Pouwelse, K. Langendoen, and H. Sips, "Dynamic voltage scaling on a low-power microprocessor," in *Proceedings of the 7th annual international conference on Mobile computing and networking*. ACM, 2001, pp. 251–259.

[7] H. Jayakumar, K. Lee, W. S. Lee, A. Raha, Y. Kim, and V. Raghunathan, "Powering the internet of things," in *Proceedings of the 2014 international symposium on Low power electronics and design*. ACM, 2014, pp. 375–380.

[8] C. Zhu, V. C. Leung, L. Shu, and E. C.-H. Ngai, "Green internet of things for smart world," *IEEE Access*, vol. 3, pp. 2151–2162, 2015.

[9] G. Gamrath, T. Fischer, T. Gally, A. M. Gleixner, G. Hendel, T. Koch, S. J. Maher, M. Miltenberger, B. Müller, M. E. Pfetsch, C. Puchert, D. Rehfeldt, S. Schenker, R. Schwarz, F. Serrano, Y. Shinano, S. Vigerske, D. Weninger, M. Winkler, J. T. Witt, and J. Witzig, "The scip optimization suite 3.2," ZIB, Takustr.7, 14195 Berlin, Tech. Rep. 15-60, 2016.

[10] T. K. Ralphs and M. Güzelsoy, "The symphony callable library for mixed integer programming," in *The next wave in computing, optimization, and decision technologies*. Springer, 2005, pp. 61–76.

[11] G. Bradski, "Opencv library," *Dr. Dobb's Journal of Software Tools*, 2000.

[12] E. E. Devices, "Watts up pro," 2009.

[13] P. Smith, "Comparing low-power wireless technologies," https://www.digikey.com/en/articles/techzone/2011/aug/comparing-low-power-wireless-technologies, 2011.

[14] V. Rodoplu and T. H. Meng, "Minimum energy mobile wireless networks," *IEEE Journal on selected areas in communications*, vol. 17, no. 8, pp. 1333–1344, 1999.

[15] D. Baker and A. Ephremides, "The architectural organization of a mobile radio network via a distributed algorithm," *IEEE Transactions on communications*, vol. 29, no. 11, pp. 1694–1701, 1981.

[16] A. Ephremides, J. E. Wieselthier, and D. J. Baker, "A design concept for reliable mobile radio networks with frequency hopping signaling," *Proceedings of the IEEE*, vol. 75, no. 1, pp. 56–73, 1987.

[17] V. Rodoplu and T. H. Meng, "Minimum energy mobile wireless networks," in *Communications, 1998. ICC 98. Conference Record. 1998 IEEE International Conference on*, vol. 3. IEEE, 1998, pp. 1633–1639.

[18] R. G. Gallager, P. A. Humblet, and P. M. Spira, *A distributed algorithm for minimum weight spanning trees*. Citeseer, 1979.

[19] S. Singh, M. Woo, and C. S. Raghavendra, "Power-aware routing in mobile ad hoc networks," in *Proceedings of the 4th annual ACM/IEEE international conference on Mobile computing and networking*. ACM, 1998, pp. 181–190.

[20] T. H. Meng and V. Rodoplu, "Distributed network protocols for wireless communication," in *Circuits and Systems, 1998. ISCAS'98. Proceedings of the 1998 IEEE International Symposium on*, vol. 4. IEEE, 1998, pp. 600–603.

[21] M. Ettus, "System capacity, latency, and power consumption in multihop-routed ss-cdma wireless networks," in *Radio and Wireless Conference, 1998. RAWCON 98. 1998 IEEE*. IEEE, 1998, pp. 55–58.

[22] T. J. Shepard, "Decentralized channel management in scalable multihop spread- spectrum packet radio networks," Ph.D. dissertation, Massachusetts Institute of Technology, 1995.

[23] V. Pilloni and L. Atzori, "Deployment of distributed applications in wireless sensor networks," *Sensors*, vol. 11, no. 8, pp. 7395–7419, 2011.

[24] J. Zhu, J. Li, and H. Gao, "Tasks allocation for real-time applications in heterogeneous sensor networks for energy minimization," in *Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, 2007. SNPD 2007. Eighth ACIS International Conference on*, vol. 2. IEEE, 2007, pp. 20–25.

[25] V. Pilloni, M. Franceschelli, L. Atzori, and A. Giua, "A decentralized lifetime maximization algorithm for distributed applications in wireless sensor networks," in *Communications (ICC), 2012 IEEE International Conference on.* IEEE, 2012, pp. 1372–1377.

Biography

Abhimanyu Chopra graduated from Summer Fields School(New Delhi, India) in 2007. He received his Bachelors of Technology in Electrical and Electronics Engineering in 2011 from Maharshi Dayanand University(Haryana, India). He worked at Precision Tech Enterprises for three years before joining the master's program at George Mason University where he worked under Dr. Houman Homayoun in the Green Computing and Heterogeneous Architectures (GOAL) Laboratory.