Int. J. Communications, Network and System Sciences, **2013, 6, 451-458**
http://dx.doi.org/10.4236/ijcns.2013.610047 Published Online October 2013 (http://www.scirp.org/journal/ijcns)

Decentralization of a Multi Data Source Distributed Processing System Using a Distributed Hash Table

Grzegorz Chmaj, Shahram Latifi
Department of Electrical and Computer Engineering, University of Nevada Las Vegas, Las Vegas, USA
Email: Grzegorz.Chmaj@unlv.edu, Shahram.Latifi@unlv.edu

Received September 19, 2013; revised October 10, 2013; accepted October 7, 2013

Copyright © 2013 Grzegorz Chmaj, Shahram Latifi. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

ABSTRACT

A distributed processing system (DPS) contains many autonomous nodes, which contribute their own computing power. DPS is considered a unified logical structure, operating in a distributed manner; the processing tasks are divided into fragments and assigned to various nodes for processing. That type of operation requires and involves a great deal of communication. We propose to use the decentralized approach, based on a distributed hash table, to reduce the communication overhead and remove the server unit, thus avoiding having a single point of failure in the system. This paper proposes a mathematical model and algorithms that are implemented in a dedicated experimental system. Using the decentralized approach, this study demonstrates the efficient operation of a decentralized system which results in a reduced energy emission.

Keywords: Data Transmission; Distributed Processing; Distributed Hash Table; Energy Dissipation

1. Introduction

Distributed Processing Systems (DPS) are used to process data over multiple nodes that are not located in one geographical location. The structure of a DPS usually consists of the central element, which performs the control tasks; the intercommunication structure (IS); and machines that join the structure in order to contribute with their computation resources. This type of organization is easy to implement and manage; however, it introduces the problem of a single point of error: the central element, without which system is unable to operate. The same issues were addressed for peer-to-peer media sharing systems, which at some point became the target of intellectual properties agencies. Centralized systems, such as Napster, could be closed easily because the only thing required was to shut down the central element. The design of decentralized systems, such as Gnutella, showed that this approach was very challenging, as broadcasting participants with system messages flooded the system and made it inoperable. Further, the data-locating aspect was found to be problematic; as in the case of decentralized system, there was no central server that could be simply asked for data location.

DPS suffer from the same problems as do their centralized counterparts, additionally, in DPS, all nodes are active system elements and perform vital functions (what introduces additional problems). Therefore, the system management in DPS has to be reliable and efficient. A distributed hash table (DHT) is an approach that allows data to be located in the fully distributed system, based on the data's identification number or content. DHTs are reliable and self-managed, and are often used in peer-to-peer systems, such as Bit Torrent. We propose to use a DHT for a DPS containing many data sources. In such multi-data-source system, each system participant offers data from its data sources.

Data at data sources change over time, so data have to be fetched at the time it is needed. The example of such a system is the structure of Unmanned Aerial Vehicles (UAVs), where each UAV provides geographic and environment data to other UAVs. A set of UAVs may be separated from the command center, so the use of a distributed control provides the independency. By using a DHT, high reliability and a maximum level of decentralization can be achieved.

2. Motivation and Contribution

Many applications of DPS are not critical; thus, a design involving a single point of failure (the central element) is sufficient. Often, self-organization and energy consumption are not a concern, especially if the system is based on stationary nodes. However, a trend of mobilization can be observed in such a system in which nodes operate using battery power. More and more applications require high reliability, the absence of single point of failure, and the ability to self-reorganize, in case any system element becomes disconnected. The main motivations for this study were:

- *Create mathematical foundation* for all data transmission aspects described, using expressions and formulas. The complete description is strictly clear; many of the system's properties can be formulated as lemmas and theorems, and proved mathematically. Further extension of such description can be easily verified for compatibility, relying on mathematical properties and using new proofs.
- *Easy implementation.* The mathematical description proposed is easy to implement in both the simulation and real systems, often using the expressions in the code directly.
- *A decentralized system*, with an architecture that will be resistant to attacks and random disturbances, and will not have any single point of failure.
- *Self-organization.* The goal is to make a system that is able to reorganize in case of any structural change, such as any of its components joining or leaving.
- *Algorithms.* The complete system description also requires rules of operation, which are defined in this study in the form of algorithms that are uniform for all system participants. This makes the system easier to manage.
- *Energy consumption.* The solutions provided to the systems, including mobile devices, include an aspect of energy minimization.

The contribution of this work is to create complete system assumptions and ideas as well as a comprehensive description in terms of digital-data transmission theory. These elements are used to build an operating system for simulation and further research. To summarize, this study proposes novel ideas for a multi-data-source DPS that includes the use of DHT, and formulates an information theory based on the mathematics that describes the system. Algorithms, designed according to the proposed theory, included the minimization of energy consumption to make mobile systems more efficient. The solutions were implemented and tested, providing the experimentation results to be described in this paper.

3. Literature Overview

Distributed processing systems are the subject of wide-ranging research in the literature. The communication layer is considered mostly as overlay network, hiding the packet and lower layers that are not essential, in this case. A survey of overlay network and related management issues was described in [1]. As the overlay networks are often used for distributed systems, some work can be found that describes overlay networks that have a special kind of abstract document [2]. In addition, overlay networks provide the separation of networking issues in the experimentation simulators. The only layer modeled can be the overlay; the lower layers do not have to be modeled in terms of implementation details. However, they have to be described correctly in terms of the parameters seen from the overlay.

For this study, a universal communication simulator platform was built based on the experience and research results from our previous work in this area [3,4]. The DHT was chosen as the decentralization technique because since its introduction, the DHT approach has been widely used in many applications. It is a promising approach for decentralizing systems [5,6], and many authors have implement this approach to achieve the system decentralization. Authors of [7] implemented DHT, together with a Common Object Request Broker Architecture (CORBA) standard, in order to allow the overlay-network-based communication system to conduct event management. This approach can be compared to the communication platform using and Interconnection Structure (IS) that was developed for this study, used as a base to build the experimental system. However, the CORBA standard was not used in this study.

Nodes in the system proposed in this study may contain data sources that are attached (they can represent sensors, external devices, or other type of unit providing digital data). This idea also has been found in structures widely known as wireless sensor networks [8], which usually contain massive sensor sets that are available for the clients. The scale of these systems affects the efficiency of the network as well as energy efficiency. Attempts have been made to provide a DHT-based approach to wireless sensor networks; for example, the Pastry algorithm was used to design the data sharing system for the P2P sensor structure by authors of [9]. The researchers used DHT Pastry to provide the current sensor data; however, their PAST also can access historical sensor values. The applications of distributed systems using DHT include vehicular ad hoc network [10], peer-to-peer data sharing networks [11], web caching [12], instant messaging, and many others.

4. System Description

4.1. General Definitions

This section contains a comprehensive description of the system developed in this study, based on mathematical formulas. A distributed processing system contains V nodes: $v, w = 1, 2, \cdots, V$, which are connected using an interconnection structure (IS) through which nodes are able to transmit data. Each node in the system has the same uniform structure, shown in **Figure 1** and contains the following logical elements: DHT routing table, set of data sources, fz container, state indicator, network connection and two queues: *messages in* and *messages out*. Both queues are using FIFO mechanism and algorithms for dropping outdated elements. For the sake of simplicity in this paper 'processing' is interpreted as computing, although it can also mean processing the control functions, gathering real-time data, etc. There are Z computational tasks in the system: $z = 1, 2, \cdots, Z$. These tasks are issued to the system by nodes. The node that enables a task for computation is called the *task owner*, and is denoted by $e_{zv} = 1$ (0 otherwise). Tasks contain massive amounts of digital data. The key idea of distributed processing is to process these kinds of tasks by using many computational units. Thus, each task is divided into blocks: $b = 1, 2, \cdots, B$, and binary variable $c_{zb} = 1$ denotes that block b belongs to task z (0 otherwise). The computation of block b produces the result $r = 1, 2, \cdots, B$, and r belongs to task z when $c_{zr} = 1$ (0 otherwise). The structure of a task is presented in **Figure 2**, and the structure of a block is shown in **Figure 3**.

Many distributed processing systems are defined to divide tasks into uniform sized chunks. However, this kind of approach often does not match the real applications; thus, we propose to use variable-sized division. The size of each block b from task z is described by value of h_{zb}; also, h_{zr} is defined for result r. Each node is characterized by its processing power p_v, which represents the node's ability to quickly finish assigned computations.

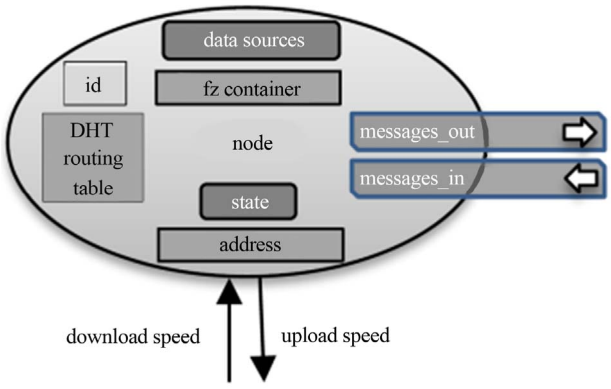

Figure 1. Node structure.

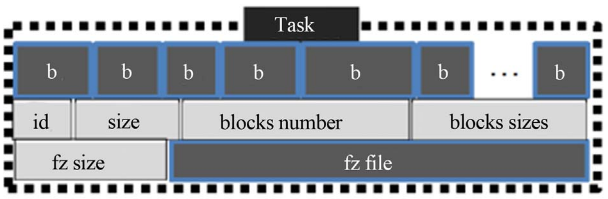

Figure 2. Task structure.

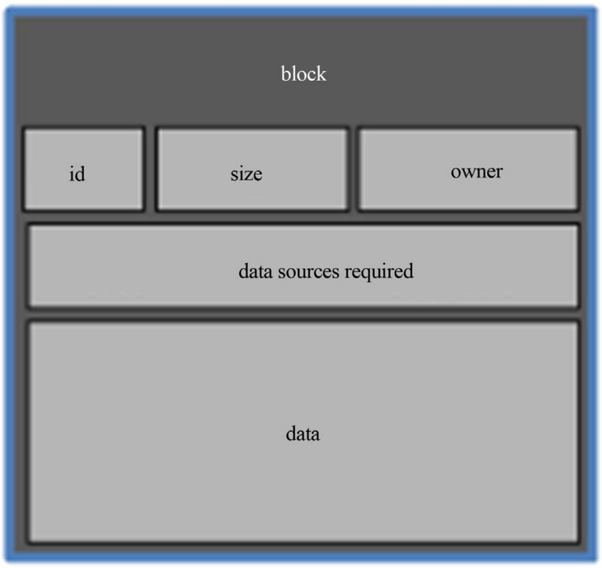

Figure 3. Blockstructure.

Besides the task binary data in the form of blocks, each task has an associated base file, *fz*. This file contains all the definitions and common data used for blocks processing. File *fz* is required to be present on the node that wants to process blocks from task *z*. The node acting as the task owner designates another node, called the *task manager*, which handles the *fz* file.

This makes management of the DHT easier for all computing nodes, as described further in this section. A base file *fz* of size h_{zf} is sent from the task owner to the task manager before computation starts for the blocks.

The system contains S data sources $s = 1, 2, \cdots, S$, which are physically present on the nodes. The presence of data sources on the nodes is denoted by $a_{vs} = 1$ (0 otherwise). One node can have zero $(\Sigma_s a_{vs} = 0)$, one, or many data sources. Each block b has its own requirements regarding data from data sources. Each requirement has to be satisfied before the start of block computation. The requirement for data source s made by block b belonging to task z is denoted by $q_{zbs} = 1$ (0 otherwise). There are no rules about requiring data from sources: block may require zero sources $(\Sigma_s q_{zbs} = 0)$, one or many $(\Sigma_s q_{zbs} \geq 1)$.

In order to precisely describe the system, we divide the system's operating time span as the set of time slots $t = 1, 2, \cdots, T$. This is the approach commonly used for modeling systems, and allows the assignment of any event to a precise moment in time:

- Block b belonging to task z is computed at node v during slot t: $x_{zbvt} = 1$ (0 otherwise).
- The base-data file for task z is present on node v at time t: $f_{zvt} = 1$ (0 otherwise).
- Block b belonging to task z is transmitted from node v to node w in slot t: $y_{zbwvt} = 1$ (0 otherwise).

- The result r belonging to task z is transmitted from node v to node w in slot t: $y_{zrwvt}=1$ (0 otherwise).
- The message m is transmitted from node v to node w in slot t: $y'_{mwvt}=1$ (0 otherwise).
- The data file is transmitted from node v to node w in slot t: $y''_{fzwvt}=1$ (0 otherwise).

Timing properties are used to define time-related properties of the system. Each node has an assigned upload speed u_v and download speed d_v, measured in kB/slot. These speeds characterize the network parameters of the link connecting the node to the network. As the nodes communicate with each other, the data transmission between two nodes is performed at the lower of the two speeds. The transmission time of block b belonging to task z from task owner w to node v is $\max\left(\left\lceil\frac{h_{zb}}{d_v}\right\rceil,\left\lceil\frac{h_{zb}}{u_w}\right\rceil\right)$.

Block size is given for each b, thus transmission time can be used to estimate the network delay. Given the size of block h_{zb} and the processing power p_v, the time of the computation for block b on node v can be determined: $\left\lceil\frac{h_{zb}}{p_v}\right\rceil$.

The DPS proposed in this study uses several messages:

- $m1$: request of a block: size: h_{m1}
- $m2$: request of a data source value:size: h_{m2}
- $m3$: data source value: size: h_{m3}
- $m4$: request of a file-base:size: h_{m4}

The time required to transmit message m from node w to node v can be formulated as $\max\left(\left\lceil\frac{h_{zb}}{p_v}\right\rceil,\left\lceil\frac{h_m}{u_w}\right\rceil\right)$. Messages are sent using data containers (**Figure 4**), which serve as the wrapper and are the basic unit of communication in IS.

4.2. Constraints Characteristics

The system's operation is limited by several constraints. Some of them provide the simple properties, and easily can be modified to include more complicated assumptions. We propose that one block can be computed on one node only. This assumes reliability such in a way that if the

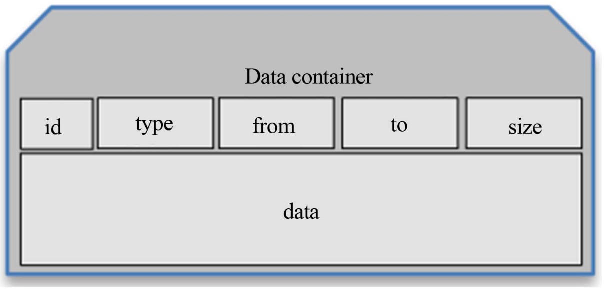

Figure 4. Data container.

block was started for computation, then the node finished the processing before possible quit. This constraint can be formulated as $\Sigma_v\Sigma_t x_{zbvt}=\left\lceil\frac{h_{zb}}{p_v}\right\rceil$ $z=1,2,\cdots,Z$, $b=1,2,\cdots,B$. Also, the node can process one block at the time: $\Sigma_z\Sigma_b\Sigma_v\Sigma_t x_{zbvt}\le 1$. There is no possibility that the node starts to process a block that is partially fetched. The block must be fully downloaded before its computation starts: $\Sigma_w y_{zbwvt}$ + $x_{zbvt'}$ $\le$ 1, $z=1,2,\cdots,Z$; $b=1,2,\cdots,B$; $t,t'=1,2,\cdots,T$; $t<t'$, $v=1,2,\cdots,V$. As stated earlier, node v can compute the block b belonging to task z only if it possesses the base-file fz related to task z: $f_{zvt}-x_{zbvt}\ge 0$; $z=1,2,\cdots,Z$; $v=1,2,\cdots,V$; $b=1,2,\cdots,B$; $t=1,2,\cdots,T$. When the computation is finished, then the result is sent back to the task owner:

$$\sum_r\frac{\sum_t y_{zrwvt}}{\max\left(\left\lceil\frac{h_{zr}}{d_v}\right\rceil,\left\lceil\frac{h_{zr}}{u_w}\right\rceil\right)}=\sum_b\frac{\sum_t x_{zbwt}}{\left\lceil\frac{h_{zb}}{p_w}\right\rceil}$$

where $w,v=1,2,\cdots,V$; $w\ne v$; $z=1,2,\cdots,Z$

The above description mathematically defines the system proposed in this paper. This form clearly describes all the assumptions and properties. Formulas stated above were directly used in the implementation of the experimental system, its properties, structure, and algorithms.

4.3. Energy Emission

In this system the energy used for system operation is taken into consideration. The proposed algorithms shall not only provide the decentralized structure, but also minimize the energy used. The energy components are:

- Energy required to transmit (send and receive) 1 kB of data between nodes v and w: E_{1vw}.
- Energy required to completely route message $m2$ between nodes v and w: E_{2vw}.
- Energy required to transmit (send and receive) message $m1$, $m3$, or $m4$ from a data source or from node v to w: E_{3vw}.
- Energy required to compute the SHA1 on node v: E_{4v}.

The overall energy used by node v can be described in a following way:

$$E_v=\Sigma_v\Big($$

$$\sum_z\sum_n\frac{\sum_t y''_{fznvt}}{\max\left(\left\lceil\frac{h_{zf}}{d_v}\right\rceil,\left\lceil\frac{h_{zf}}{u_w}\right\rceil\right)}\left(E_{4v}+E_{2vw}+E_{1wv}h_{zf}+\right. \quad (1)$$

$$\sum_z\sum_b\frac{\sum_t x_{zbvt}}{\left\lceil\frac{h_{zb}}{p_v}\right\rceil}\left(E_{3vn}+E_{1nv}h_{zb}+\right. \quad (2)$$

$$\sum\nolimits_{s} q_{zbs} * \left(E_{4v} + E_{2vn1} + E_{3n1v}\right)\Big)\Big) + \tag{3}$$

$$\sum\nolimits_{n2} \sum\nolimits_{t} y'_{m2n2vt} * E_{3vn2} + \tag{4}$$

$$\sum\nolimits_{n2} \sum\nolimits_{t} y'_{m2n2vt} * E_{1vn2} h_{zf} \Big) \tag{5}$$

where:

w—the task manager node,

n—the task owner,

$n1$—the data source owner,

$n2$—the requesting node.

The components state: (1) is the energy used for downloading fz base-file, including computing SHA1 function and routing the message in DHT; (2) is the energy used to fetch block b; (3) expresses the energy required to fetch data from all data sources that are required by block b; (4) energy required to send the data source value to requesting node; (5) energy emitted to send base-file fz to requesting node.

Nodes that are task issuers also must include the following:

$$\sum\nolimits_{z} e_{zv} \Big($$

$$E_{4v} + \tag{6}$$

$$E_{1vw} h_{zf} + \tag{7}$$

$$\sum\nolimits_{b} \sum\nolimits_{n2} c_{zb} \frac{\sum\nolimits_{t} y'_{m1n2vt}}{\max\left(\left\lceil \frac{h_{m1}}{d_v} \right\rceil, \left\lceil \frac{h_{m1}}{u_{n2}} \right\rceil\right)} * E_{3n2v} + \tag{8}$$

$$\sum\nolimits_{b} \sum\nolimits_{n2} c_{zb} \frac{\sum\nolimits_{t} y_{zbvn2t}}{\max\left(\left\lceil \frac{h_{zb}}{d_{n2}} \right\rceil, \left\lceil \frac{h_{zb}}{u_v} \right\rceil\right)} * E_{1vn2} * h_{zb} + \tag{9}$$

$$\sum\nolimits_{r} \sum\nolimits_{n2} c_{zr} \frac{\sum\nolimits_{t} y_{zrn2vt}}{\max\left(\left\lceil \frac{h_{zr}}{d_v} \right\rceil, \left\lceil \frac{h_{zr}}{u_{n2}} \right\rceil\right)} * E_{1vn2} * h_{zr} \Bigg) \tag{10}$$

The components meanings are: (6) the computation of SHA1 function of the task manager to handle fz file. (7) states the upload of base-file fz to task manager. (8) is the receiving of requests for blocks (messages of type $m1$). (9) denotes the energy used for sending all blocks from the task, and (10) describes the energy emitted to receive B results belonging to task z. The whole expression denotes the energy required to handle all tasks that are issued by node v. The values of particular energy consumption elements, used for experiments, were estimated using available datasheets and electronic simulation software.

The definition of the system allows many other metrics to be easily added. In this study we focus on the energy emission, while others such as queuing issues, latency, network routing, and many other parameters are considered as a future work.

4.4. DHT Properties

The main goal of the system is to use the distributed hash table (DHT) to decentralize the whole organization. The following assumptions were formulated to define the DHT operation. SHA1 is the function used to generate hashes. The hash length is 40 bits, but this can be changed, if needed. The base-file fz is the DHT resource; it's hash is the *key* in DHT, and is generated based on the task name, $\text{DHT}_k = sha1(task\_name)$.

Data sources are the resources in DHT as well. DHT keys related to data sources are computed based on the data source's name, $\text{DHT}_k = sha1(name_s)$. To identify the node in the DHT structure, we propose to use its physical address as the argument for the SHA1 function to compute the *id* in DHT, $\text{DHT}_{id} = sha1(addr_v)$. The node v associated with DHT_{id} is able to provide the following data to other nodes: a) the current value of the embedded data source and b) the base-files fz that are present on node v.

To get the data from a data source, the requesting node has to compute the DHT key based on data source's name and further locate the data source by DHT algorithm. Following the DHT rules, the data source having the key equal to DHT_k will be located on the node of $\text{DHT}_{id} = \text{DHT}_k$. If there is no node with such DHT_{id}, the data is then placed on the node having the next higher DHT_{id} (the closest one in terms of DHT organization). The search of the DHT location is performed using chord.

Figure 5 shows the general organization scheme of IS. Pool is the logical unit containing all data containers being transmitted. IS uses DHT for localizing the resources in the network. DHT unit uses several internal functions, among which the most important are presented in **Figure 5**.

The general diagram of proposed system is shown on above **Figure 6**. System contains many nodes, some of them can decide to take the role of task issuer and issue computational task into the system. Task is divided into blocks of various sizes and includes fz file containing all necessary constants and definitions required for blocks processing. Blocks are sent to other nodes for processing,

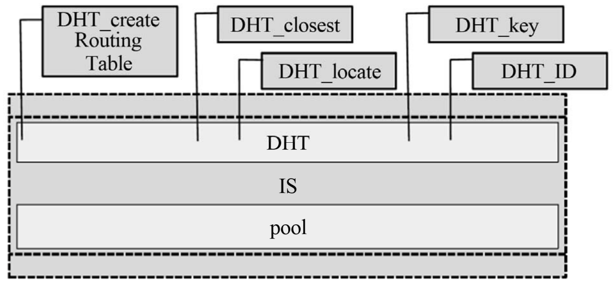

Figure 5. Structure of Distributed Hash Table (DHT).

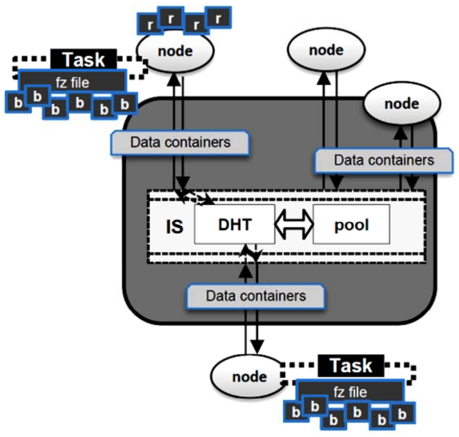

Figure 6. The overall schematic of the system.

each block may require the values of one or more data sources for processing. Data sources and other assets are located using DHT. Computation results are sent back to the task issuer, where they are combined into the final result. The interconnection structure IS, which provides communication, uses data containers to transmit all types of messages.

4.5. Proposed Algorithms

Along with the system definitions, constraints, energy aspects, and DHT-related properties, we propose the design of the universal operation algorithm, used for every node in our system. The universal for of the algorithm enables each node to act as task issuer, if needed. If a node operates as regular computing node, the task issuing part of the algorithm is not used.

5. Experimentation Results

The research was conducted using our experimentation communication platform, which allows defining various elements, such as message containers of many types, any network topology, the nodes' properties, their input/output queues, and communication rules. Experimentation communication platform and its extension for the purpose of this study were created in Ruby language. An additional DHT layer was added; and the related functions defined, such as *sha*1 computations, routing tables creation, routing of messages, and more. Further, energy consumption measurement factors were added. This resulted in a complete simulation system operating online, bringing its behavior much closer to those of real systems. The algorithms were coded such in a way that they are fully portable to physical nodes, such as PC machines connected over the Internet. The only element to be replaced, in that case, would be the IS layer with all the related and underlying layers.

To ensure the proper operation of the whole system, the first experiment involved analyzing log files regarding the traffic in the IS and DHT layers. To reach that goal, audit functions were implemented, among others, in a form of direct use of the parts of the mathematical model. This proved that a decentralized system structure was able to handle the distributed processing system. Each node contained on average 6.9 entries in DHT routing table, and the average number of hops during resource location was 3.2.

The second experiment included the implementation of a centralized protocol, where the central server was used instead of a DHT structure, known as the *Single Central Unit* (SCU). Due to the lack of a DHT, there is no way of determining the locations of system resources, such as data sources or blocks; therefore, the nodes have to communicate through the server. The energy consumption when using the SCU approach was defined the same way as for the DHT, and mainly was based on the energy consumption per kilobyte, defined according to the network topology. The energy consumption for sending a certain amount of kilobytes between two nodes in the system was the same for both the DHT-based and server-based approaches. In this way, it was possible to compare the energy emissions required by these two cases.

The test data used for the comparison was generated according to predefined requirements: network parameters, processing-power parameters, network transmission-energy consumption, and computation-energy emission. The requirements for parameters, including ranges from which final values were selected, were set to reflect real systems (e.g. u_v and d_v are reflecting network speeds available among Internet network). Number of nodes in the system was selected arbitrarily, network parameters were selected according to defined ranges, number of tasks and their sizes were set according to experimentation plan. Each system structure, containing all the parameters, together with tasks details states the experimentation load, so the experiment for each system + tasks definition can be repeated. The system studied in this paper includes the random delay which is present in distributed processing systems: message sent to destination node may be the subject of competition with a message coming from another node (exactly as in real networks). Due to FIFO nature of nodes' queues, the competition effect has the influence of system operation. To avoid uncertainty of the results, the experiment for each system-task (each point in the graphs) was repeated 50 times and resulting values were averaged.

First, the energy emitted was measured according to the total number of blocks present in the system. The number of block includes blocks from all the tasks present in the system. The structure containing $V = 50$ nodes demonstrated that the DHT approach required less energy to perform the assigned tasks than SCU (**Figure 7**).

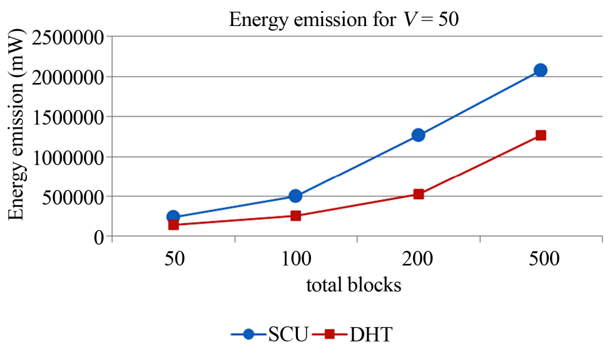

Figure 7. Energy emission for *V* = 50.

The difference increased slightly as the task size grew. However, in this case, the difference in energy stayed proportional to the increase in task size and stayed in the range 40% - 60%. The case presented in **Figure 8** (V = 500) shows that the energy consumption was similar for SCU and DHT for tasks size B = 500, and the difference increased slightly with a bigger task size present in the system, staying around 30% - 40%, and increased to 60% for B = 2500.

To compute a high number of blocks, the SCU approach exposed a significant increase as it started to suffer problems with single-unit congestion. The DHT kept the increase in the energy requirement proportional to the growth in task size.

The next experiment investigated the complexity of the data sources structure on energy emission. The data sources provided relatively small amounts of data, which could be perceived as measurements; thus, the data seen in **Figures 9** and **10** were averaged for each data point. The DHT structure demonstrated to be scalable and quite resistant to a growing number of nodes and the efficiency of DHT routing used (in the DHT, the average number of nodes taking part in a routing was small). The SCU showed the proportional relation between energy emission and the number of data sources present in the system as well as the complexity of data required from data sources in order to perform the block processing.

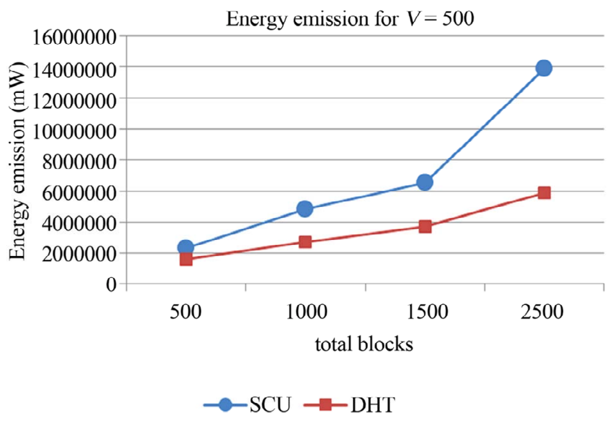

Figure 8. Energy emission for *V* = 500.

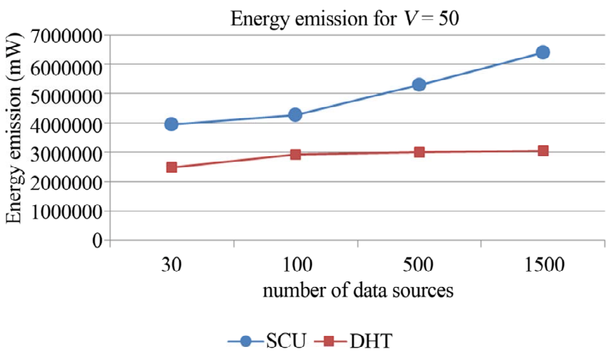

Figure 9. Energy emission for *V* = 50.

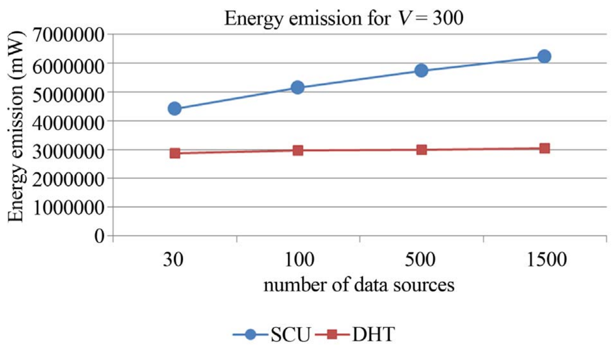

Figure 10. Energy emission for *V* = 300.

6. Conclusions

The stated goals of the design of a decentralized approach were fulfilled. The mathematical statements describing the system brought the advantage of a clear form of description. Additionally, statements were used directly in the algorithms, system design, and auditing functions. The mathematical model was easy to extend with new constraints and elements.

The proposed algorithm was noted in the form of a pseudocode, which later was implemented directly in the experimental system. In order to build that system, the universal communications platform was extended with system-specific objects, properties, and functions. Further, it was equipped with an additional communication layer, which provided DHT functionality. The research results proved that the proposed solutions in providing the decentralized architecture fulfilled their roles because the decentralized system was fully functional and efficient in its operation. The comparison between DHT and SCU approaches showed that the decentralized approach is able to reduce energy emissions and avoid communication congestions that are present in the SCU system.

Future work includes extensive research of the DHT and other decentralized approaches and also the design of a universal decentralized communication system for distributed processing.

7. Acknowledgements

This work was supported in part by the Nevada EPSCoR program NSF award #EPS-IIA-1301726 and by the DOD-DTRA Grant/Award #: HDTRA1-12-1-0033.

REFERENCES

[1] J. Ding, I. Balasingham and P. Bouvry, "Management of Overlay Networks: A Survey," *Proceedings of* 3rd *International Conference on Mobile Ubiquitous Computing*, *Systems*, *Services and Technologies*, Sliema, 11-16 October 2009, pp. 249-255. http://dx.doi.org/10.1109/UBICOMM.2009.49

[2] D. Kasthurirathna and C. Keppetiyagama, "Architectural description based Overlay Networks," *Proceedings of International Conference on Advances in ICT for Emerging Regions*, Colombo, 1-2 September 2011, pp. 14-18.

[3] G. Chmaj and K. Walkowiak, "A P2P Computing System for Overlay Networks," *Future Generation Computer Systems*, Vol. 29, No. 1, 2013, pp. 242-249. http://dx.doi.org/10.1016/j.future.2010.11.009

[4] G. Chmaj and D. Zydek, "Software Development Approach for Discrete Simulators," *Proceedings of* 21st *International Conference on Systems Engineering*, Las Vegas, 16-18 August 2011, pp. 273-278.

[5] K. P. N. Puttaswamy and B. Y. Zhao, "A Case for Unstructured Distributed Hash Tables," *Proceedings of IEEE Global Internet Symposium*, Anchorage, 11 May 2007, pp. 7-12. http://dx.doi.org/10.1109/GI.2007.4301423

[6] Z. Zhang, "The Power of DHT as a Logical Space," *Proceedings of* 10*th IEEE International Workshop on Future Trends of Distributed Computing Systems*, Suzhou, 26-28 May 2004, pp. 325-331.

[7] F. Umer and A. Qayyum, "Architecture for Decentralized, Distributed Event Communication Mechanism through Overlay Network," *Proceedings of IEEE Symposium on Emerging Technologies*, Islamabad, 17-18 Sepember 2005, pp. 252-257.

[8] C. S. Raghavendra, K. M. Sivalingam and T. Znati (Editors), "Wireless Sensor Networks," Springer, 2004. http://dx.doi.org/10.1007/b117506

[9] W. Song, S. Kim, S. Seok and D. Choi, "Pastry Based Sensor Data Sharing," *Proceedings of* 18*th Internatonal-Conference on Computer Communications and Networks*, San Francisco, 3-6 August 2009, pp. 1-6.

[10] T. Delot, N. Mitton, S. Ilarri and T. Hien, "Decentralized Pull-Based Information Gathering in Vehicular Networks Using GeoVanet," *Proceedings of* 12*th International Conference on Mobile Data Management*, Lulea, 6-9 June 2011, pp. 174-183.

[11] J. Timpanaro, T. Cholez, I. Chrisment and O. Festor, "When KAD Meets BitTorrent—Building a Stronger P2P Network," *Proceedings of IEEE International Parallel & Distributed Processing Symposium*, Shanghai, 2011, pp. 1635-1642.

[12] E. Rosas, N. Hidalgo and M. Marin, "Two-Level Result Caching for Web Search Queries on Structured P2P Networks," *Proceedings of IEEE* 18*th International Conference on Parallel and Distributed Systems*, Singapore, 17-19 December 2012, pp. 221-228.