CHAPTER 3

TIME VARIANT-PEER NODE HETEROGENEOUS PEER-TO-PEER NETWORKS USING DYNAMIC WARPING ALGORITHM

3.1 INTRODUCTION

In the current internet scenario many virtual servers and mirror servers are utilized to maintain load balance of the heterogeneous P2P networks. One of the existing works developed a balancing system based on three metrics. They were probability distribution of peer abilities, loads of virtual servers, and incomplete data for global peers. On the other hand state balancing method recognizing peer node capacities are difficult a) as the demand and weight of the peer differ from time to time which requires to be addressed b) In account of this, the peer node persistence needs to be managed with its CPU cycles for processing the client demands c) Load diversion of peer neighbors supply to the difficulty of corresponding peer nodes on serving their vital demand d) At the lower dimension of load balancing, various data format increases the processing time of the peer servers

To overcome the issues, a new plan is made to build Time Variant-Peer Node Heterogeneous Data Processing Scheme for efficient load balancing in distributed heterogeneous peer networks. Peer node time variant capacity is measured using Dynamic Time Warping algorithm (DTWA) to evaluate the magnitude of load-demand balance factors of peer servers. With
the resultant load-demand factor obtained from DTWA, peer server processing cycle requirements are identified using Duty Cycle Data Appropriation (DCDA) Technique. Load diversions are made to peer server with Node Selection Strategy based on DCDA rank representation. The heterogeneous data nature of the demand requested by the peer servers is implicitly identified by a priori of data format load levels. These are matched to current data format demand of respective nodes and their effects on load balancing the peer servers are calculated. Experimental performances are evaluated with the Heterogeneous peer networks data extracted from the large internet service providers. In addition to this simulations are carried out to show the effectiveness of proposed work with bench mark data sets from UCI (University of California, Irvine) Repository with performance metrics such as load distribution rate, node heterogeneity and performance rate with respect to load balancing.

3.2 LOAD BALANCING IN P2P NETWORKS

Client server computing systems employs two logical parts namely a server that offers services and a client that requests services of the servers. Jointly, the two form an absolute computing system with a very discrete separation of dependability. More precisely, client /server computes associates of two or more strands of implementation using a consumer/producer association. Clients provide as the customers in a client/server system. That is, they construct requests to servers for services or information and then use the response to perform the idea of the client. The server acts the role of the producer, satisfying data or service requests made of it by clients.
Figure 3.1 describes the architecture of client-server. Client-server is an eminent and reliable server of a data source. Clients request data from server. Very successful model in client server are WWW (HTTP), FTP, Web services, etc. But the limitation of client server is the Scalability which is tough to attain, delivers a single point of breakdown, needs management, and idle resources at the network connection.

![Figure 3.1 Architecture of Client-Server](image)

P2P systems attempt to solve these client server limitations. P2P computing is the allocation of computer devices and resources by direct substitute between systems. These devices and resources comprise the swap of services, executing cycles, memory loads, and disk storage for information. P2P forecasting takes merits of existing computing energy, memory storage and networking links, permitting users to influence their combined power to the advantage of all.
Figure 3.2 describes the architecture of P2P. All nodes participating in the network are both clients and servers and no centralized data source.

Figure 3.2 Architecture of P2P

The P2P system is capable of providing better data and consumes power. Any node is capable of initiating a connection. The characteristics of P2P network are elaborated as follows. Clients are also servers and routers unlike client-server. Nodes donate content, storage, memory, CPU. Nodes are autonomous or independent i.e., no organizational verification. Network is dynamic or self motivated i.e., nodes enter and leave the network often. Nodes cooperate with each other straightforwardly and not through recognized servers. Nodes have broadly changeable capabilities. The benefits of using P2P network are

1. Efficient use of resources and devices
   i. Idle bandwidth, memory storage, processing energy at the boundary of the network

2. Scalability
   i. Consumers of resources also contribute resources
   ii. Aggregate resources raise physically with consumption
3. Reliability
   i. Replicas
   ii. Geographic sharing
   iii. No single point of failure

4. Ease of management
   i. Nodes self organize
   ii. No necessitate to deploy servers to assure demand
   iii. Built-in fault tolerance, replication, and load balancing

The structured P2P is the second generation P2P network overlay. The structured P2P are Self-organizing, Load balance and Fault-tolerant. Scalable ensures on numbers of hops to react a query which is the key difference with unstructured P2P systems. The structured P2P is based on a distributed hash table interface.

3.2.1 Structured P2P Overlay Networks based Distributed Hash Tables (DHTs) for Load Balancing

Load balancing is a crucial issue for the well-organized process of P2P networks. Distributed Hash Tables (DHTs) one of the structured P2P overlay networks plan information to the P2P network depending upon a reliable hashing function. Such tracking for data sharing has an intrinsic load balance troubles. Thus, a load balancing mechanism is a crucial section of a structured P2P overlay network for better working. The rapid growth of P2P systems has caused issues in load balancing due to their characteristics by large scale, node heterogeneity, resource energy, and closeness. A well-organized load balancing method should be flexible enough and should provide with the mechanism to cope up with these features.
Structured P2P systems based on the DHT mechanism shows a potential intend for resource distribution on a large-scale and on peak of which many applications are intended such as information sharing, circulated file systems, real time flowing, and shared processing. In Structured P2P systems, each data item is traced to an exclusive identifier (ID) strained from an identifier space. The identifier space is divided among the nodes so that each node is dependable for a section of the identifier space, called region, and loading all the objects that are traced into its region.

One essential challenge in the DHT design is to balance the load across the nodes in the system. Most P2P systems that offer a DHT concept share out objects among peer nodes by selecting arbitrary identifiers for the objects. In the case of a uniform system where all nodes have the similar ability, Dynamic Hash Table (DHTs) show an $O(\log n)$ (with $n$ participants balance to load one at a time) imbalance factor. In addition, a P2P system is extremely varied, i.e. they probably comprise of peers that account from old desktops following modem lines to dominant servers associated to the Internet through high-bandwidth lines. The imbalance is considerably raises as the heterogeneity of the system increases.

The challenges faced by P2P system is overcome by two modules of solutions. Solutions in the first module use the idea of Virtual Servers (VS). Each physical node concretes with one or more virtual servers with arbitrary ID’s that behaves as peers in the DHT. In the case of a homogeneous system, preserving $O(\log n)$ (in space complexity) virtual servers for each physical node minimizes the load imbalance to a stable aspect. To face heterogeneity, each node gathers a number of virtual servers relative to its ability. Unfortunately, virtual servers acquire a major cost as a node with $x$ virtual servers which preserves $x$ sets of overlay connections. Generally
\[ x = O(\log n.) \] (in space complexity), which causes an asymptotic increase in overhead.

The next second module of solutions uses just a unique identifier (ID) per node. But, all such results relocate the identifiers (IDS) to preserve the load balance as nodes enter and leave the system. This leads to a high overhead as it engages broadcasting objects and updating overlay connection. Additionally, no solutions face the heterogeneity straightforwardly, though they could be shared with the virtual server methods. Also, for growth of next-generation internet infrastructure, application layer P2P networks (Chen & Tsai 2008) are measured to be insignificant along with the load balancing policies.

Thus the problem of load balancing in such P2P systems is addressed in time variant-peer node heterogeneous P2P networks using dynamic warping algorithm. The objective of building load-balancing algorithms is explored in dynamic warping algorithm such a way that it uses the notion of virtual servers. In addition, the iterative algorithmic approach for space exploration of solution is assumed to determine a finest relocation of servers for load balancing. The time variant-peer node heterogeneous P2P networks using dynamic warping algorithm utilizes reallocation algorithm.

### 3.2.2 Problems in Structured P2P Overlay Networks

The structure peer-to-peer on reliable hashing and histogram global load balancing based in (Vu et al 2009) assumed that nodes were aware of most of the other nodes in the system, building it unreasonable to level to huge number of nodes. Quite the reverse, each Chord node requires routing details about only some other nodes. Since the routing table is shared, a node decides the hash function by relating with a few other nodes. In the constant state, in an N-node system, each node preserves details only about \( O(\log N) \)
(meaning that the running time grows in accord to the size of the node) affects the other nodes, and all finds through $O(\log N)$ information to other nodes. Chord keeps its routing details as nodes merge and exit the network with high probability leading to $O(\log_2 N)$ (where $N$ is the total number of nodes in the network) messages.

The steady hashing mechanism allocates the keys to corresponding nodes as elaborated below. The identifiers are sequenced using an identifier with round modulo $2^r$. Next, the key is allocated to the first node of the identifier if it is equivalent to or tracks $x$ in the identifier space which is called as the successor node of key $X$, denoted by $\text{successor}(x)$. The identifiers are denoted as rounded numbers from 0 to $2^r - 1$, then $\text{successor}(x)$ is the first node starting from $x$. Figure 3.3 shows an identifier round with $r=3$ with round consisting of three nodes: 0, 1, and 3.

Successor of identifier 1 is node 1, so key 1 would be placed at node 1. At the same time, key 2 would be placed at node 3, and key 6 at node 0. Steady hashing is build to let nodes come in and go away the network with least interruption. To preserve the steady hashing mapping when a node $n$ merges the network, definite keys allocated prior to $n$’s successor now become allocated to $n$. If the node departs the network, all of its allocated keys are reallocated to $n$’s successor. No other alterations in task of keys to nodes happen. In the instance, if a node were to merge with identifier 7, it would incarcerate the key with identifier 6 from the node with identifier 0.
Figure 3.3 describes the identifier round consisting of the three nodes 0, 1 and 3. In this example, key 1 is placed at node 1, key 2 at node 3, and key 6 at node 0.

![Diagram of identifier circle consisting of nodes 0, 1, and 3 with successor relationships labeled as Successor (1) = 1, Successor (2) = 3, and Successor (6) = 0.]

**Figure 3.3 An Identifier Circle Consisting of the Three Nodes 0, 1, and 3**

Chord make simpler the construction of P2P systems and applications based on it by tackling complex troubles as load balance, scalability, Decentralization, availability, flexible naming. In the execution, the Chord software obtains the form of files to be associated with the client and server applications. The application communicates with Chord in two main ways.

First, Chord offers a key algorithm that gives up the IP address of the node dependable for the key. The initial work concentrates on key for identifying the IP address. Secondly, the Chord appliance on each node identifies the request of variations in the group of keys that the node is accountable for. This permits the application software to and for instance, shift resultant values to their new address when a new node merges. Both the manner the application interacts with the chord.
Figure 3.4 shows a possible three-layered software structure for a cooperative mirror system. The maximum layer would offer a file like edge to users, including user-friendly identification and authentication. This file system layer strength executes named directories and files, tracing operations on them to lower-level block stages. The next layer, i.e. the block storage layer, would execute the block operations. It would pay attention of memory storage, caching, and duplication of blocks. The block storage layer would use Chord to recognize the node accountable for loading a block and then converse to the block storage server on that node to interpret or write the block.

At its sensitivity, Chord offers rapid shared computation of a hash function tracing keys to nodes in charge for them. It uses steady hashing which has various high-quality properties. With high chance, the hash function balances load i.e., all nodes obtain approximately the same number of keys. Also with high chance, when an $N^{th}$ node links or go away the network, only an $O(1/N)$ fraction of the keys are stimulated to a different place, this is obviously the smallest amount essential to sustain a balanced load.
3.3 TIME VARIANT-PEER NODE HETEROGENEOUS P2P NETWORKS USING DYNAMIC WARPING ALGORITHM

Time Variant-Peer Node Heterogeneous Data Processing scheme is a method in which the peer node moves out of the network for the efficient load balancing in distributed heterogeneous peer networks. It is necessary to denote the liveliness of the peer node in the network. To monitor the time of the peer node in the heterogeneous P2P network, DTWA is used.

The DTWA is used to evaluate the dynamic time variant capacity of peer node to evaluate the magnitude of load-demand balance factors of peer nodes. The DTWA efficiently identifies the liveliness of the peer node in the network which varies in time and speed. The peer node analyses depend on the basic verification of the dynamic time so the peer nodes are checked before distributing.

So the peer node, before sharing its virtual nodes with other node, actually checks the DT (Dynamic Time) of the neighbor peer node first using DTWA in the heterogeneous P2P network. After analyzing it, the virtual nodes are shared for maintaining the packet data to equalize the load imbalance factor.

Peer node time variant capacity is measured using DTWA to evaluate the magnitude of load-demand balance factors of peer nodes. With the resultant load-demand factor obtained from DTWA, peer node processing cycle requirements are identified using Duty Cycle Data Appropriation (DCDA) technique. DCDA outperforms other solution in terms of the cost overhead on virtual nodes and load imbalance factor. Time Variant-Peer Node
Heterogeneous Data Processing Scheme efficiently balances load in distributed heterogeneous peer networks.

### 3.3.1 Reallocation Algorithm of Virtual Server to Peer Node

Imagine that the total hash space offered by a DHT is [0, 1], and each virtual server in the DHT has a distinctive ID chosen separately and consistently at arbitrary from the space [0, 1]. Let P be the set of participating peers, and S be the set of virtual servers in the DHT. Let the set of virtual servers be denoted by $S_i$. Each peer $i$ belongs to P determines the load $L_i$, which is indicated by $T_i$, that it should recognize, where $T_i = \sim A \times C_i + C$, where $\sim A$ is an estimation of expected load per unit with a capacity ‘C’, as in Equation (3.1) i.e.,

$$\sum_{s \in S} L_s / \sum C_i \quad (3.1)$$

Where $C$ is pre-defined system parameter.

If the present overall load of $i$ is greater than $T_i$ (i.e., $i$ is overloaded), then $i$ drifts its $S$ to other peers $i$, else it does nothing but remains to accept the drifted virtual servers $S$. If peer $i$ is overloaded, $i$ choose virtual servers $S$ for relocation, such that 1) $i$ turns under loaded, and 2) the total movement cost, $MC$, is minimized due to the reallocation.

If peer $i$ is under loaded, then $i$ is demanded to accept a drifted virtual server $S_i$, and $i$ admits such a virtual server if the added load due to the virtual server will not overload itself; otherwise, $i$ rejects such virtual server.
Algorithm (Reallocation (i))

Input: P, S, i, T

Output: allocation of S_i to P_i

Switch (Load (i)) do

Case > T_i

U_i <> 0

While Load (i) > T_i and S_i <> U_i do

S ← arg min { l_v | v ∈ S_i, U_i }  

Break;

Case < T_i

While Load (i) < T_i do

Receive s to host

S_i ← S_i U \{s\}

Break;

End

But the stability of peer node in P2P network is unrecognizable. Since there is a chance of peer node to move out of the network, it is necessary to denote the time of the peer node i.e., (liveliness of the peer node in the network). To monitor the time of the peer node in the heterogeneous P2P network, the Dynamic Time Warping algorithm is used.

3.3.2 Dynamic Time Warping Algorithm

The DTWA is used to evaluate the dynamic time variant capacity of peer node to evaluate the magnitude of load-demand balance factors of peer servers. The architecture diagram of time variant-peer node heterogeneous P2P networks using dynamic warping algorithm is shown in Figure 3.5.
Figure 3.5 describes the architecture diagram of time variant-peer node heterogeneous P2P networks using dynamic warping algorithm. If load of the nodes increases, reallocation algorithm is performed with allocated
virtual server (VS) for each node, followed by DTWA execution. The DTWA for evaluating the time variant capacity of peer node is described below:

\[
\text{Int DTWA time variant capacity (C (p1), C (p2)…. C (pn), int w)}
\]

\[
\{
\text{ Declare DTWA(C(p1), C(p2)….C(pn))}
\]

\[
\text{Int w: MAX-time (C (p_i)), i, j}
\]

\[
\{
\text{ For i = 1 to n}
\text{ Evaluate w for C (p_i)}
\text{ End For}
\]

\[
\text{For j = 1 to n}
\text{ Do}
\text{ Algorithm (Reallocation (j))}
\text{ While (w for (C (p_j)))}
\text{ End Do}
\text{ End For}
\]

The DTWA efficiently identifies the liveliness of the peer node in the network which may vary in time or speed. So the peer node, before sharing its virtual servers with other nodes, will check the DT (Dynamic Time) of the neighbor peer node first using DTWA in the heterogeneous P2P network. After analyzing it, the virtual servers are shared for maintaining the packet data to equalize the load imbalance factor.
3.4 EXPERIMENTAL EVALUATION

The experiments were run on an Intel P-IV machine with 2 GB memory and 3 GHz dual processor CPU. The proposed time variant peer heterogeneous P2P network using DTWA is compared with an existing Load Balance with Imperfect Information in Structured Peer-to-Peer Systems (Hsiao et al. 2011).

3.5 RESULTS AND DISCUSSION

Compared to an existing Load Balance with Imperfect Information in Structured Peer-to-Peer Systems (Hsiao et al. 2011), the proposed time variant peer heterogeneous P2P network using DTWA perform better in terms of (i) load distribution rate, (ii) node heterogeneity, and (iii) performance ratio. A table and graph below depicts the performance of both existing Load Balance with Imperfect Information in Structured Peer-to-Peer Systems and proposed time variant peer heterogeneous P2P network using DTWA.

3.5.1 Load Distribution Rate

The performance metrics used here for the improvement of heterogeneous P2P networks are Load imbalance factor describing the load distribution rate. The Load Imbalance factor for node $i \in N$ is termed as

$$
\text{Load Imbalance Factor} = \sum_{v_i} \frac{L_v}{C_i}
$$

(3.2)

For a given peer nodes, $N$, and the set of virtual servers, $v_i$, for each peer node where $I = 1, 2, ..., N$ for which each peer $I$ migrates a subset of its virtual servers, $V_i$ (where $vCv_i$), to other peers in relation to the time $C_i$. The MS is defined as

$$
S = U_{i\in N} V_i
$$

(3.3)
Using these two metrics, the performance of the proposed time variant peer heterogeneous P2P network using DTWA is evaluated to show the performance.

Table 3.1 illustrates the load distribution rate with respect to number of nodes. If the number of node increases, the load imbalance factor of the node remains constant at some stage in the proposed time variant peer heterogeneous P2P network using DTWA. Based on the Table 3.1, a graph is shown in Figure 3.6.

### Table 3.1 No. of Node vs. Load Distribution

<table>
<thead>
<tr>
<th>No. of Node</th>
<th>Load Distribution Rate (%)</th>
<th>Time Variant P2P Network using DTWA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Load Imbalance Factor</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>79</td>
<td>85</td>
</tr>
<tr>
<td>40</td>
<td>72</td>
<td>78</td>
</tr>
<tr>
<td>60</td>
<td>65</td>
<td>70</td>
</tr>
<tr>
<td>80</td>
<td>59</td>
<td>63</td>
</tr>
<tr>
<td>100</td>
<td>52</td>
<td>57</td>
</tr>
<tr>
<td>120</td>
<td>45</td>
<td>51</td>
</tr>
<tr>
<td>140</td>
<td>38</td>
<td>46</td>
</tr>
<tr>
<td>160</td>
<td>31</td>
<td>39</td>
</tr>
</tbody>
</table>
Figure 3.6 describes the Load distribution rate of nodes in the network. When compared to an existing Load Balance with Imperfect Information in Structured Peer-to-Peer Systems, the proposed one is good about 5-10% in maintaining the capacity of the node. The virtual servers are shared for maintaining the packet data to equalize the load imbalance factor in proposed time variant P2P network using DTWA.

![Graph of No. of Node vs. Load Distribution](image)

**Figure 3.6 No. of Node vs. Load Distribution**

### 3.5.2 Node Heterogeneity

Many resource factors of node heterogeneity, including process speed, storage and bandwidth decide the performance of methods. A table and graph below depicts the performance of node heterogeneity for both existing Load Balance with Imperfect Information in Structured Peer-to-Peer Systems (Hsiao et al 2011) and the proposed time variant peer heterogeneous P2P network using DTWA.
Table 3.2 describes the node heterogeneity based on the capacity of nodes. Based on the Table 3.2 a graph is shown in Figure 3.7.

**Table 3.2 Capacity of Nodes vs. Node Heterogeneity**

<table>
<thead>
<tr>
<th>Capacity of Nodes</th>
<th>Node Heterogeneity (%)</th>
<th>Time Variant P2P Network using DTWA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Load Imbalance Factor</td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>6</td>
<td>14</td>
</tr>
<tr>
<td>50</td>
<td>13</td>
<td>25</td>
</tr>
<tr>
<td>75</td>
<td>18</td>
<td>32</td>
</tr>
<tr>
<td>100</td>
<td>25</td>
<td>39</td>
</tr>
<tr>
<td>125</td>
<td>29</td>
<td>46</td>
</tr>
<tr>
<td>150</td>
<td>34</td>
<td>50</td>
</tr>
<tr>
<td>175</td>
<td>37</td>
<td>57</td>
</tr>
</tbody>
</table>

Figure 3.7 describes the node heterogeneity based on the capacity of nodes in the network. If the capacity of nodes increases, the node heterogeneity also increases in the proposed time variant peer heterogeneous network using DTWA. Compared to an existing Load Balance with Imperfect Information in Structured Peer-to-Peer Systems, the proposed time variant peer heterogeneous P2P network using DTWA provides better node heterogeneity of about 10-20%. As the proposed work one monitors the time of the peer node in heterogeneous P2P network, the Dynamic Time Warping algorithm is able to increase the process speed, storage and bandwidth.
3.5.3 Performance Rate

The performance rate of time variant P2P network using DTWA is decided based on the better load balancing efficiency. Load balancing ability of the system facilitates in superior P2P network system. Table 3.3 describes the performance rate based on dynamic peer node time. Based on the Table 3.3 a graph is shown in Figure 3.8.

Table 3.3 Dynamic Peer Node Time vs. Performance Rate

<table>
<thead>
<tr>
<th>Dynamic Peer Node Time</th>
<th>Performance Rate (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Load Imbalance Factor</td>
</tr>
<tr>
<td>0.2</td>
<td>73</td>
</tr>
<tr>
<td>0.4</td>
<td>69</td>
</tr>
<tr>
<td>0.6</td>
<td>63</td>
</tr>
<tr>
<td>0.8</td>
<td>58</td>
</tr>
<tr>
<td>1.0</td>
<td>56</td>
</tr>
<tr>
<td>1.2</td>
<td>51</td>
</tr>
<tr>
<td>1.4</td>
<td>47</td>
</tr>
</tbody>
</table>
Figure 3.8 describes the performance rate of peer nodes in the network based on dynamic time. If the dynamic peer node time increases, the performance rate of the peer node also increases for about in the proposed time variant peer heterogeneous P2P network using DTWA. When compared to an existing Load Balance with Imperfect Information in Structured Peer-to-Peer Systems, the proposed one performance is 20-25% high. As the DTWA efficiently identifies the liveliness of the peer node that shares its virtual servers with other node checking DT (Dynamic Time) of the neighbor peer node using DTWA in the heterogeneous P2P network resulting in better performance rate.

![Figure 3.8 Dynamic Peer Node Time vs. Performance Rate](image)

**Figure 3.8 Dynamic Peer Node Time vs. Performance Rate**

Finally, time variant peer heterogeneous P2P network using DTWA ensures the load balancing of a P2P network with better energy consumption. Experimental evaluation proved the better performance in terms of time variant peer heterogeneous P2P network using DTWA in terms of load distribution rate for about 5-10%, node heterogeneity around 10-20% and the performance rate based on nodes load balancing about 20-25% compared to existing Load Balance with Imperfect Information in Structured P2P Systems.
3.6 CONCLUSION

This paper presented a time variant peer heterogeneous P2P network using DTWA with virtual servers. The proposal is distinctive in representing the system condition with probability distributions. Not like previous solutions that frequently rely on universal knowledge of the system, each peer in our proposal, separately guess the probability distributions for the facility of contributing peers and the loads of virtual servers stranded on fractional knowledge of the system. With the estimated probability distributions, each node recognizes whether it is under loaded and then reallocates its loads if it is overloaded. The simulation results prove that the proposal performs well in contrast with the existing Load Balance with Imperfect Information in Structured P2P Systems and outperforms solution in terms of the node heterogeneity of virtual servers, load imbalance factor, and performance rate.