

**PARALLEL DATA DECOMPOSITION SOLUTION FOR  
THE P-MEDIAN PROBLEM**

By  
**Sobhi Ibrahim Al-Shekha**

Supervisor  
**Dr. Ahmed Sharieh**

Co-Supervisor  
**Dr. Moh'd Belal Al-Zoubi**

**This Thesis was Submitted in Partial Fulfillment of the  
Requirements for the Master's Degree in Computer Science**

**Faculty of Graduate Studies  
The University of Jordan**

August 2005

**This Thesis (Parallel Data Decomposition Solution for The P-Median Problem) was successfully defended and approved on August 25-2005.**

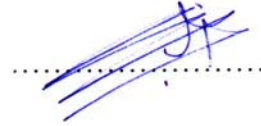
**Examination Committee**

Dr. Ahmad Abdel-Aziz Sharieh, Chairman  
Assoc. Prof. of Parallel Processing

**Signature**



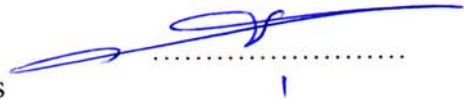
Dr. Moh'd Belal Al-Zoubi, Co-Supervisor  
Assist. Prof. of GIS and Computer Graphics



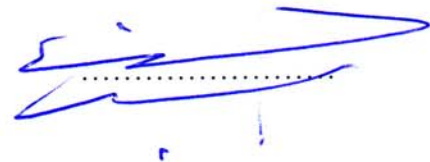
Dr. Sami I. Serhan, Member  
Assist. Prof. of Computer Design



Dr. Imad Khaled Moh'd Salah, Member  
Assist. Prof. of Complex Systems & Networks



Dr. Mohammed Mahafzah, Member  
Assoc. Prof. of Parallel Processing  
(Mu'tah University)



**الجامعة الأردنية****نموذج التفويض**

أنا صبحي إبراهيم الشبيخة، أفوض الجامعة الأردنية بتزويد نسخ من رسالتي للمكتبات أو المؤسسات أو الهيئات أو الأشخاص عند طلبها.

التوقيع:

التاريخ:

**The University of Jordan  
Authorization Form**

I, Sobhi Ibrahim Al-Shekha, authorize the University of Jordan to supply copies of my Thesis to libraries or establishments or individuals on request.

Signature:

Date:

**DEDICATION**

- *To my loving wife, Naryman, who has always been with me in all my problems, and my daughter, Sara.*
- *To my father, who has always given me his unlimited and unconditional love and care, and whose support has always granted me with the strong will to succeed.*
- *To my loving mother, whose presence has always been my source of motivation.*
- *To my brother and sisters: Mohammad, Kaltoon, and Eman.*

## ACKNOWLEDGEMENT

I would like to express my sincere gratitude and deepest appreciation to my advisor, Dr. Ahmed Sharieh, and my Co-Supervisor, Dr. Moh'd Belal Al-Zoubi, for their substantial inspiration and technical guidance throughout this study.

I would also thank the members of the committee for their support, help and valuable advice in writing this thesis.

Great thanks to my friends and to all who have contributed in bringing out this work, making it a success.

## LIST OF CONTENTS

<b>SUBJECT</b>	<b>PAGE</b>
<b>COMMITTEE DECISION</b>	<b>II</b>
<b>AUTHORIZATION</b>	<b>III</b>
<b>DEDICATION</b>	<b>IV</b>
<b>ACKNOWLEDGEMNET</b>	<b>V</b>
<b>LIST OF CONTENTS</b>	<b>VI</b>
<b>LIST OF TABLES</b>	<b>VIII</b>
<b>LIST OF FIGURES</b>	<b>X</b>
<b>LIST OF SYMBOLS</b>	<b>XIII</b>
<b>ABSTRACT</b>	<b>XIV</b>
<b>INTRODUCTION</b>	<b>1</b>
<b>1. Introduction</b>	<b>1</b>
<b>2. Parallel decomposition</b>	<b>2</b>
<b>2.1. Parallel data decomposition</b>	<b>3</b>
<b>2.2. Parallel task decomposition</b>	<b>5</b>
<b>2.3. Speedup (S), Efficiency (E)</b>	<b>6</b>
<b>3. Objectives</b>	<b>6</b>
<b>4. Samples of the study</b>	<b>6</b>
<b>5. Methodology</b>	<b>7</b>
<b>6. Structure of the study</b>	<b>7</b>
<b>REVIEW OF LITERATURE</b>	<b>9</b>
<b>1. Previous research</b>	<b>9</b>
<b>1.1 Heuristic search</b>	<b>9</b>
<b>1.2 Variable neighborhood search</b>	<b>10</b>
<b>1.2.1 Greedy method</b>	<b>11</b>
<b>1.2.2 Vertex substitution local search</b>	<b>11</b>
<b>1.3 Approximation algorithm for facility location problem</b>	<b>11</b>
<b>1.3.1 The uncapacitated facility location problem</b>	<b>12</b>
<b>1.3.2 The capacitated facility location problem</b>	<b>12</b>
<b>2. The studies of the p-median problem</b>	<b>13</b>
<b>3. Factors that effects on results</b>	<b>14</b>
<b>3.1 Hardware factors</b>	<b>14</b>
<b>3.2 Number of computers in the network</b>	<b>17</b>
<b>3.1 Software factors</b>	<b>17</b>
<b>3.3.1 Size of data set</b>	<b>17</b>
<b>3.3.2 Number of facilities</b>	<b>17</b>

<b>THE PARALLEL SOLUTION</b>	<b>18</b>
1. The parallel solution	18
2. The sequential p-median algorithm	20
3. Parallel p-median algorithm	20
4. Complexity analysis	25
5. Explanation of the p-median algorithm	26
4.1 Applying the p-median algorithm on one workstation	26
4.2 Applying the parallel p-median algorithm on four workstations	28
<b>RESULT AND ANALYSIS</b>	<b>29</b>
1. Introduction	29
2. Sequential p-median algorithm	30
3. Parallel p-median algorithm	34
3.1 Using tow processors	34
1.2 Using four processors with grid decomposition	36
1.3 Using four processors with vertical decomposition	38
1.4 Using four processors with horizontal decomposition	40
1.5 Nine processors with grid data decomposition	42
4. Comparison among decomposition types	44
5. Comparison between sequential p-median algorithm and parallel p-median algorithm	58
6. Number of facilities is changed on fixed size of data	72
7. Applying the sequential p-median algorithm and parallel p-median algorithm on the Baboon image data set	79
7.1 Apply the sequential p-median algorithm	80
7.2 Apply the parallel p-median algorithm	83
7.3 Comparison results	85
8. Comparison with other works	86
8.1 Computational experiments	87
<b>CONCLUSIONS AND DECOMENTATIONS</b>	<b>91</b>
1. Summary	91
2. Conclusions	91
3. Future research	92
<b>REFERENCES</b>	<b>93</b>
<b>ABSTRACT (IN ARABIC)</b>	<b>96</b>

## LIST OF TABLES

	<b>Title</b>	<b>Page</b>
<b>Table 1</b>	The coordinates of the first 20 nodes for the map	26
<b>Table 2</b>	Some of possible permutations and its objective functions for data in Table3.1.	27
<b>Table 3</b>	Computing the F on 4 computers for the data in Table3.1.	29
<b>Table 4</b>	Changing the time and objective function while changing the size of data and fixed numbers of P for one processor	34
<b>Table 5</b>	Changing the time and objective function while changing the size of data and fixed numbers of P for 2 processors	37
<b>Table 6</b>	Changing the time and objective function while changing the size of data and fixed numbers of P for 4 processors (grid decomposition)	39
<b>Table 7</b>	Changing the time and objective function while changing the size of data and fixed numbers of P for 4 processors (vertical decomposition)	41
<b>Table 8</b>	Changing the time and objective function while changing the size of data and fixed numbers of P for 4 processors (horizontal decomposition).	43
<b>Table 9</b>	The time and objective function while changing the size of data and fixed numbers of P for 9 processors (grid decomposition).	45
<b>Table 10</b>	Time, speedup, efficiency for grid, horizontal, and vertical decompositions where P=1.	47
<b>Table 11</b>	Time, speedup, efficiency for grid, horizontal, and vertical decompositions where P=2.	49
<b>Table 12</b>	Time, speedup, efficiency for grid, horizontal, and vertical decompositions where P=3.	51
<b>Table 13</b>	Time, speedup, efficiency for grid, horizontal, and vertical decompositions where P=4.	53
<b>Table 14</b>	Time, speedup, efficiency for grid, horizontal, and vertical decompositions where P=5.	55



<b>Table 15</b>	Time, speedup, efficiency for grid, horizontal, and vertical decompositions where $P=6$ .	57
<b>Table 16</b>	Time, speedup, efficiency for grid, horizontal, and vertical decompositions where $P=7,8,9,10$ .	59
<b>Table 17</b>	Comparing the times, speedup, and efficiencies for one, tow, four, and nine processors where $P=1$ .	61
<b>Table 18</b>	The times, speedup, and efficiencies for one, tow, four, and nine processors where $P=2$ .	63
<b>Table 19</b>	The times, speedup, and efficiencies for one, tow, four, and nine processors where $P=3$ .	65
<b>Table 20</b>	The times, speedup, and efficiencies for one, tow, four, and nine processors where $P=4$ .	67
<b>Table 21</b>	The times, speedup, and efficiencies for one, tow, four, and nine processors where $P=5$ .	69
<b>Table 22</b>	The times, speedup, and efficiencies for one, tow, four, and nine processors where $P=6$ .	71
<b>Table 23</b>	The times, speedup, and efficiencies for one, tow, four, and nine processors where $P=7,8,9,10$ .	73
<b>Table 24</b>	Time to compute F for p-median problem on one processor.	74
<b>Table 25</b>	Time to compute F for p-median problem on two processors.	76
<b>Table 26</b>	Time to compute F for p-median problem on four processors.	77
<b>Table 27</b>	Comparing the times, speedup, and efficiencies for one, tow, and four processors.	79
<b>Table 28</b>	Time to compute F for p-median problem on one processor for the Baboon image.	82
<b>Table 29</b>	Time to compute F for p-median problem on two processors.	85
<b>Table 30</b>	Synchronous Parallel Scatter Search and Grid Decomposition Search.	90

<b>Table 31</b>	Replication Combination Scatter Search and Grid Decomposition Search.	90
<b>Table 32</b>	Replication Parallel Scatter Search and Grid Decomposition Search.	91

## LIST OF FIGURES

	<b>Title</b>	<b>Page</b>
<b>Figure 1</b>	Vertical data decomposition.	3
<b>Figure 2</b>	Horizontal data decomposition.	4
<b>Figure 3</b>	Grid data decomposition.	4
<b>Figure 4</b>	Decomposition of dense matrix-vector multiplication into $n$ tasks, where $n$ is the number of rows in the matrix. The portions of the matrix and the input and output vectors accessed by Task $i$ are highlighted.	5
<b>Figure 5</b>	Generates the possible permutations.	19
<b>Figure 6</b>	The sequential algorithm to solve the $p$ -median problem.	20
<b>Figure 7</b>	The parallel $p$ -median algorithm on horizontal decomposition.	22
<b>Figure 8</b>	The parallel $p$ -median algorithm on vertical decomposition.	22
<b>Figure 9</b>	The parallel $p$ -median algorithm on grid decomposition.	23
<b>Figure 10</b>	The $p$ -median $(n, p)$ algorithm.	24
<b>Figure 11</b>	The distribution of points in the grid decomposition.	28
<b>Figure 12</b>	Sample arcs of the logical network of the Lose Angeles quad [2].	32
<b>Figure 13</b>	Relation between the number of facilities ( $P$ ) and the time ( $T$ ) where number of computer =1	35
<b>Figure 14</b>	Relation between the number of facilities ( $p$ ) and the time ( $T$ ) where number of computer=2 (horizontal decomposition).	38
<b>Figure 15</b>	Relation between the number of	

	facilities (p) and the time (T) where number of computer=4 (grid decomposition).	40
<b>Figure 16</b>	Relation between the number of facilities (p) and the time (T) where number of computer=4 (vertical decomposition).	42
<b>Figure 17</b>	Relation between the number of facilities (p) and the time (T) where number of computer=4 (horizontal decomposition).	44
<b>Figure 18</b>	Relation between the time (T) and the number of facilities (p) where number of computer=9 (grid decomposition).	46
<b>Figure 19</b>	The speedup versus the size of data when using the three data decomposition on four computers.	48
<b>Figure 20</b>	The efficiency is change with the size of data on 4 computers.	48
<b>Figure 21</b>	The speedup versus the size of data.	50
<b>Figure 22</b>	The efficiency against the size of data and number of computers is 4.	50
<b>Figure 23</b>	The speedup against the size of data.	52
<b>Figure 24</b>	The efficiency versus the size of data.	52
<b>Figure 25</b>	The speedup against the size of data.	54
<b>Figure 26</b>	The efficiency with the size of data on 4 computers.	54
<b>Figure 27</b>	The speedup against the size of data.	56
<b>Figure 28</b>	The efficiency versus the size of data.	56
<b>Figure 29</b>	The speedup against the size of data.	57

<b>Figure 30</b>	The efficiency versus the size of data.	58
<b>Figure 31</b>	The speedup the size of data for P=1.	62
<b>Figure 32</b>	The efficiency with the size of data for P=1.	62
<b>Figure 33</b>	The speedup with the size of data, while P=2.	64
<b>Figure 34</b>	The efficiency with the size of data for P=2	64
<b>Figure 35</b>	The speedup with the size of data and P=3.	66
<b>Figure 36</b>	The efficiency against the size of data and P=3.	66
<b>Figure 37</b>	The speedup against the size of data and P=4.	68
<b>Figure 38</b>	The efficiency versus the size of data and P=4.	68
<b>Figure 39</b>	The speedup versus the size of data and P=5.	70
<b>Figure 40</b>	The efficiency versus the size of data and P=5.	70
<b>Figure 41</b>	The speedup versus the size of data, P=6.	72
<b>Figure 42</b>	The efficiency versus with the size of data, P=6.	72
<b>Figure 43</b>	Time versus the number of facilities for one PC.	75
<b>Figure 44</b>	F versus the number of facilities on one processor	75
<b>Figure 45</b>	Time versus the number of facilities for 2 processors.	76
<b>Figure 46</b>	Time versus the number of facilities for 2 processors.	77
<b>Figure 47</b>	Time for computing F on 1, 2, and 4 processors for P=1,2,...,39.	78
<b>Figure 48</b>	The speedup versus the number of facilities and N=40.	80
<b>Figure 49</b>	The efficiency versus the number of facilities for N=40.	80

<b>Figure 50</b>	The Baboon image.	81
<b>Figure 51</b>	Time versus the number of facilities for one PC.	83
<b>Figure 52</b>	The objective function F versus the number of facilities N for one PC.	84
<b>Figure 53</b>	Time versus the number of facilities for two PCs.	86
<b>Figure 54</b>	The highlighted Baboon image.	86
<b>Figure 55</b>	Time versus the number of facilities for one PC and two PCs.	87

## LIST OF SYMBOLES

$N$	Size of the data set.
$P$	Number of facilities needed from the data set.
$F, f$	The objective function.
$S$	The speedup.
$E$	The efficiency.
$m$	The number of computers.
$I$	set of candidate facility locations.
$J$	set of customers.
$w_j$	demand for the service of customer $j$ .
$d_{ij}$	Minimum distance between customer $j$ and candidate location $i$ .
$y_i$	Decision variable $\{0,1\}$ according to whether facility location $i$ is established or not.
$x_{ij}$	Fraction of customer $j$ 's demand supplied from facility $i$ .
UTP	Unshielded Twisted Pair Cables.
TCP/IP	Transmission Control Protocol/Internet Protocol.
SPSS	Synchronous Parallel Scatter Search.
RCSS	Replication Combination Scatter Search.
RPSS	Replication Parallel Scatter Search.

# PARALLEL DATA DECOMPOSITION SOLUTION FOR THE P-MEDIAN PROBLEM

By  
**Sobhi Ibrahim Al-Shekha**

Supervisor  
**Dr. Ahmed Sharieh**

Co-Supervisor  
**Dr. Moh'd Belal Al-Zoubi**

## ABSTRACT

Solving the p-median problem on one computer (sequentially) is costly in its time and space and unsolvable for applications of large size. For applications of large size, the sequential algorithm goes to trashing and is appearing inefficient. This thesis provides an alternative algorithm to solve the p-median problem using parallel computers.

In this study, we present a new method for solving the p-median problem. This method depends on dividing the data set of the p-median problem into smaller parts, vertically, horizontally or grid. This decomposition of data is independent. These parts are distributed on a number of computers that have the same characteristics. They work simultaneously, where each one solve the p-median problem for the same all possible locations of facilities, but on part of data set.

The parallel algorithm was implemented and tested on available data set and computers. The samples of the study of two data set types. The first type was generated randomly with different sizes. The second type was taken from real applications. Some of these are: the maximum covering location problem, Baboon image, and Lenna image. The objective function



(F) of finding P was computed. The time of execution, speedup, and efficiency were computed. The input (independent variables) factors are the number of computers, size of input data sets, and the values of P. The major intermediate variables are the size of the memory of the computers and the speed of the CPU of the used computers. The responses (dependent variables) were the values of F, speedup, and efficiency.

It was found that one computer cannot continue the calculations when the size of data set is more than 300 nodes for one facility location, 80 nodes for 5 facilities, and 40 nodes for 9 facilities. When apply the p-median algorithm on more than one computer, we can increase the size of data set, and we get better time than one computer. For examples, for 2 computers, we can apply the algorithm on 520nodes for size of data set where number of facilities is one, 100 nodes for 5 facilities, and 60 nodes for 9 and 10 facilities. For 9 computers, we can apply the algorithm on 935 nodes for size of data set where number of facilities is one, 120 nodes for 7 and 8 and 9 facilities, and 100 nodes for 10 facilities.

## INTRODUCTION

### 1. Problem statements

The p-median problem is one of the most well-studied location-allocation problems, in which the aim is to partition a given set of points into clusters so that the points within a cluster are relatively close with respect to some measure (Scaparra M.P, and Scutella M.G. 2001). For the metric p-median problem, we are given n points in a metric space. We select p of these to be cluster centers, and then assign each point to its closest selected center. If point j is assigned to a center point i, the cost incurred is proportional to the distance between point i and point j. The goal is to select the p centers that minimize the sum of the assignment costs.

The p-median problem can be formulated as follows:

$I = \{1, \dots, n\}$ : Set of candidate facility locations.

$J = \{1, \dots, m\}$ : Set of customers.

$w_j$  : Demand for the service of customer j.

$d_{ij}$  : Minimum distance between customer j and candidate location i.

$y_i$ : Decision variable  $\{0,1\}$  according to whether facility location i is established or not.

$x_{ij}$ : Fraction of customer j's demand supplied from facility i.

The goal of the p-median problem is to find the minimum value of  $f$  in (1.1).

$$f = \sum_{j \in J} \sum_{i \in I} w_j d_{ij} x_{ij} \quad (1.1)$$

$$\text{Subject to } \sum_{i \in I} x_{ij} = 1 \quad \forall j \in J \quad (1.2)$$

$$y_i - x_{ij} \geq 0 \quad \forall i \in I, j \in J \quad (1.3)$$

$$\sum_{i \in I} y_i = k \quad (1.4)$$

$$x_{ij} \in \{0,1\} \quad \forall i \in I, j \in J \quad (1.5)$$

$$y_i \in \{0,1\} \quad \forall i \in I \quad (1.6)$$

There are two cases in the p-median problem: an uncapacitated case and a capacitated case. In the uncapacitated case, each facility can serve an unlimited number of clients. In the capacitated case, each facility can serve, for example, at most U clients. In this research, the uncapacitated case will be studied.

Solving the p-median problem on one computer (sequentially) is costly in its time and space. Some applications need huge space and execution time. This study is an alternative to solve the p-median problem using parallel computers. In this research, we will implement the computation of  $f$  sequentially for a possible data set and in a parallel environment for some with a larger data set it is impossible it in one computer. This environment is a lab consisting of a number of computers. The data decomposition will be applied on the input data, which would make this study different from other for it combines the concepts of p-median and parallel decomposition. Several values of p will be investigated, and their effects on values of  $f$  and execution time will be measured. The speedup (S) and the efficiency (E) will be computed to solve the p-median problem on several computers.

## 2. Parallel decomposition

The decomposition is the process of dividing a computation into smaller parts, some or all of which may potentially be executed in parallel. There are several decomposition techniques for achieving concurrency, these techniques are classified as recursive decomposition, data decomposition, exploratory decomposition, and speculative decomposition ( Silberschatz A *et al.*, 2002 ). The data decomposition will be investigated in this study to solve the p-median problem.

There are two types of parallel decomposition: the parallel data decomposition, and the parallel task decomposition.

## 2.1 Parallel data decomposition

The data set must be decomposed under one of the following three methods: the horizontal data decomposition, the vertical data decomposition, and the grid data decomposition.

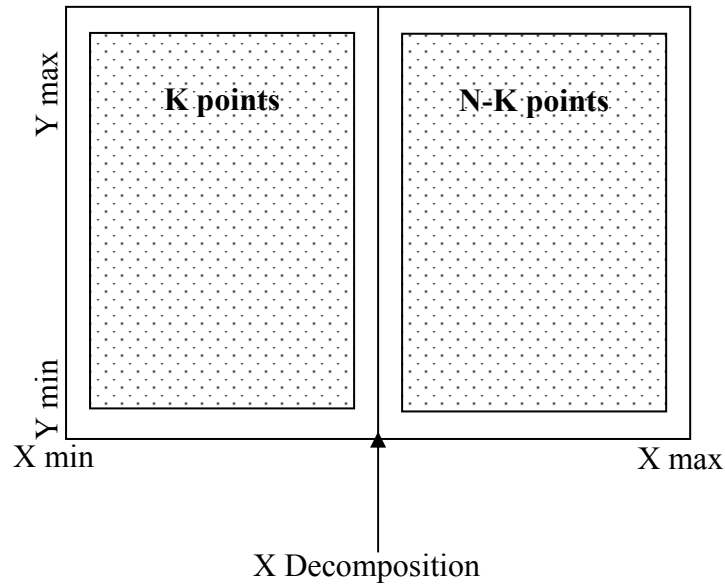


Figure 1: Vertical data decomposition.

In vertical data decomposition, the data is divided into smaller parts vertically for a metric space, as illustrated in Figure 1. The data is decomposed into two parts: one with  $K$  points and the other one with  $N-K$  points. The space consists of  $N$  points.

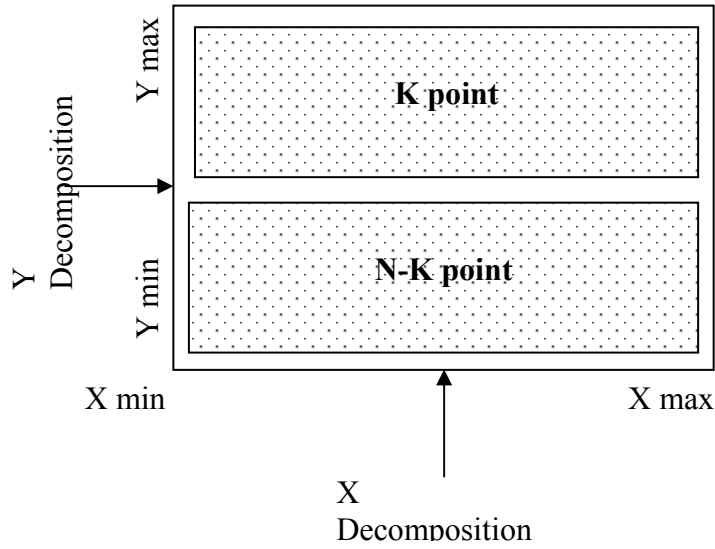


Figure 2: Horizontal data decomposition.

In horizontal data decomposition, the data is divided into smaller parts horizontally for a metric space. Figure 2 manifests the horizontal decomposition into two parts.

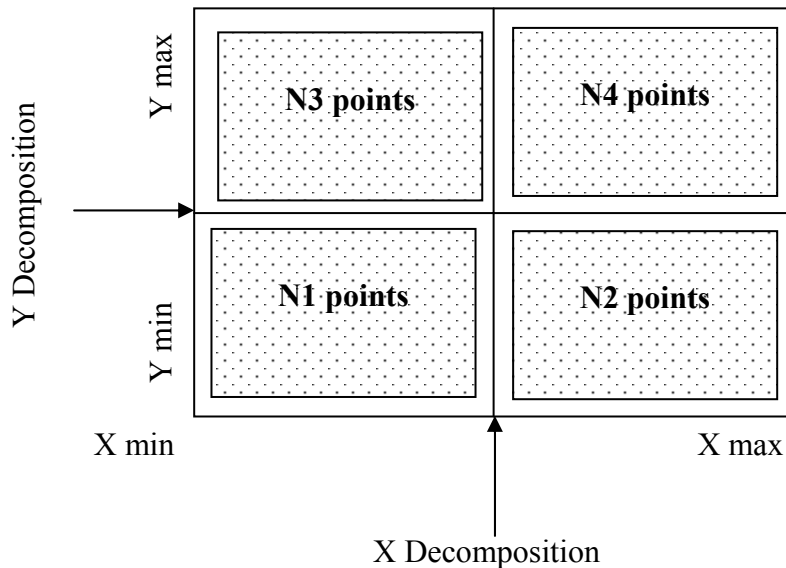


Figure 3: Grid data decomposition.

In grid data decomposition the data is divided into smaller parts horizontally and vertically for a metric space. Figure 3 manifests the grid decomposition into four parts.

## 2.2 Parallel task decomposition

Tasks are programmer-defined units of computation into which the main computation is subdivided by means of decompositions ( Silberschatz A *et al.*, 2002 ). The simultaneous execution of multiple tasks is the key to reduce the time required to solve the entire problem. Tasks can be of arbitrary size, but once defined, they are regarded as indivisible units of computation. The tasks into which a problem is decomposed may not all be of the same size.

Consider the Dense matrix-vector multiplication for example; it is the multiplication of a dense  $n$  by  $n$  matrix  $A$  with a vector  $b$  to yield another vector  $y$ . The  $i$ th element  $y[i]$  of the product vector is the dot-product of the  $i$ th row of  $A$  with the input vector  $b$ ; i.e.,

$$y[i] = \sum_{j=1}^n A[i, j].b[j].$$

As shown in Figure4, the computation of each  $y[i]$  can be regarded as a task.

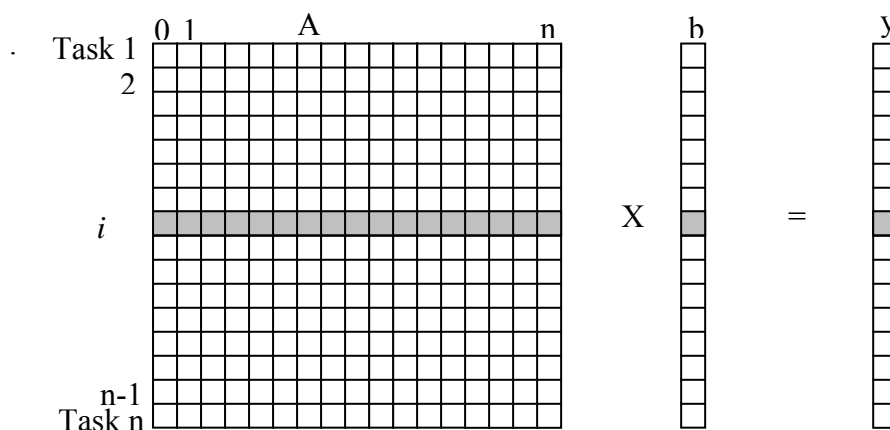


Figure 4: Decomposition of dense matrix-vector multiplication into  $n$  tasks, where  $n$  is the number of rows in the matrix. The portions of the matrix and the input and output vectors accessed by Task  $i$  are highlighted.

### 2.3 Speedup (S) and efficiency (E)

There are two parameters which must be calculated in the parallel decomposition to see how the decomposition is more useful. These parameters are the speedup and the efficiency. The speedup can be defined in formula (1.7):

$$S = \frac{T1}{T2} \quad (1.7)$$

The  $T1$  is the time needed to solve the problem without decomposition, and  $T2$  is the time needed to solve the problem after decomposition.

Theoretically, the speedup  $S$  must be smaller than or equal to  $N$ , where  $N$  is the number of parts related from the decomposition process.

The efficiency can be defined in formula (1.8):

$$E = \frac{S}{N} \times 100 \quad (1.8)$$

### 3. Objectives

The objectives of this research are to explain the implementation for the three types of data decomposition on a network of similar computers with windows 2000 operating a professional system. This is to minimize the run time and decrease the needed storage in the memories when applying the p-median algorithm. The F, S, and E will be computed on several computers for different values of P.

### 4. Samples of the study

The sample of the study is of two data set types. The first type has been generated randomly with different sizes, whereas the second type has been taken from real applications. Some of these are the maximal covering location problem, Baboon image, and Lina image.

The input (independent) factors are the number of computers, the sizes of input data sets, and the values of p. The major intermediate variables are the size of the memory of the

computers and the speed of the CPU. The responses (dependent) factors will be the values of  $f$ , speedup, and efficiency.

## 5. Methodology:

In this study, the past and current proposed technique to solve the p-median will be reviewed. We have designed a p-median algorithm which depends on swapping among all possible solutions, and then choosing the best solution. Then, this solution must be the exact solution, and the search is called Exact Search.

The designated algorithm for solving the p-median problem is implemented on one computer using an application program. This application program is programmed using Delphi programming language, and then concluding the performance at this stage. The next step is designing a parallel algorithm with a data decomposition technique and a data parallelism model. The p-median algorithm was implemented to run on multiple computers. The final step is comparing the performance between the two stages. Consequently, one can conclude which one is the best and when it can be used.

## 6. Structure of the study

This thesis includes the following chapters:

Chapter 1: introduces the problem of the study.

Chapter 2 consists of:

- Review of Literatures: includes of a general overview of several p-median methods and its solutions.
- Problem Formulation: includes the analysis of the p-median problem, factors that affect the problem, and the factors that are used in the study.

Chapter 3 consists of:



- The Parallel Solution: Introduces the sequential p-median algorithm and the parallel p-median algorithm, with an example to explain how they work.
- Complexity Analysis

Chapter4 Results and Analysis: Presents the experiments applied in this study with discussion of the results, showing the difference among the three methods of the parallel data decomposition, and comparing between the sequential and parallel p-median algorithm.

Chapter5 Conclusions and recommendations: includes summaries the work and recommendations.

## REVIEW OF LITERATURE

### 1. Previous research:

The analysis of facility location problems has represented an attractive field of research since the beginning of the century. The very first location model, due to Alfred Weber, appeared in 1909 and dominated the literature for many years hence. One of the most studied problems in locational analysis is the so called p-median problem.

#### 1.1 Heuristic search:

Hansen and Mladenovic (1992) provide good heuristic solutions for large multisource weber problems. This is done by solving related p-median problems in which potential locations of the facilities are users' locations, and then solving weber problems for the sets of users of each facility.

1. Define and solve a p-Median problem (PM) with the same users and demands as (MW) and the set of users' locations of (MW) as potential sites for locating facilities. Let  $(t_i^*) (z_{ij}^*)$  be the optimal solution.
2. Let  $I = \{i | t_i^* = 1\}$  and  $C_i = \{j | z_{ij}^* = 1\}$  for  $i \in I$ . For each  $i \in I$ , solve the Weber problem with user's set  $C_i$ . Let  $(x_i^*, y_i^*)$  be the optimal solution and  $f_i^*$  its value.
3. A heuristic solution for (MW) is given by  $\{(x_i^*, y_i^*) | i \in I\}$  and  $(z_{ij}^*)$  with value

$$\sum_{i \in I} f_i^* .$$

## 1.2 Variable neighborhood search

Mladenovic et al. (2000) present a basic Variable Neighborhood Search and two Tabu Search heuristics for the k-Center problem without triangle inequality.

Both proposed methods use the vertex substitution neighborhood structure. Classical heuristic in the literature usually exploit the close relationship between the p-center and another NP-hard problem called the dominating set problem. Given any graph  $G=(V,E)$ , where  $V=\{v_1, v_2, v_n\}$  is a set of n locations for facilities, and  $U=\{u_1, u_2, \dots, u_m\}$  is a set of m users, a complete graph  $G=(V,E)$  is defined with  $|E|=m.n$ . The p-center problem is to find a subset  $X \subseteq V$  of size p such that

$$f(x) = \max_{u_i \in U} \{ \min_{v_j \in X} d(u_i, v_j) \}$$

A dominating set S of G is a subset of V such that every vertex in V-S is adjacent in G to a vertex in S. The problem is to find a dominating set S with the minimum cardinality. For a given solution  $X = \{v_{j1}, \dots, v_{jp}\}$ , for the p-center problem, there obviously exists an edge  $(v_i, v_{jk})$  such that  $d(v_i, v_{jk}) = f(x)$ .

We can delete all links of the initial problem whose distances are larger than  $f(x)$ , and then X is the minimum dominating set in such a graph. If X is an optimal solution, then the corresponding graph with all edges whose lengths are less than or equal to  $f(x)$  is called a bottleneck graph.

There are two of such based on search method (Hansen P, and Mladenovic' N. 1997). Those are Greedy method and Vertex Substitution Local Search.

### 1.2 .1 Greedy method

With the greedy method, a first facility is located in order to minimize the maximum cost, (i.e., a 1-center problem is first solved). Facilities are then added one by one until the number  $p$  is reached; each time the location which most reduces the total cost is selected (Hansen P, and Mladenovie' N. 1997).

### 1.2.2 Vertex substitution local search

We denote by  $X = \{v_{j1}, \dots, v_{jp}\}$  a feasible solution of the  $p$ -center problem. A usual way to define a set of neighbors of  $X$  (noted  $N(x)$ ) is to replace in turn each facility belonging to the solution by each one out of it. Thus, the cardinality of  $N(x)$  is obviously  $p.(n-p)$ .

This neighborhood is known as 1-interchange or vertex substitution neighborhood. The local search heuristic that uses it finds the best solution  $X' \in N(x)$ , if  $f(X') < f(X)$  the move is made ( $X \leftarrow X'$ ), a new neighborhood is defined and the process is repeated. Otherwise, the procedure stops in a so-called local minimum (Hansen P, and Mladenovie N. 1999).

### 1.3 Approximation algorithm for facility location problem:

Shmoys et al. (2000) present new approximation algorithms for several facility location problems. In each facility location problem in this study, there is a set of locations at which we may build a facility (such as a warehouse), where the cost of building at location  $i$  is  $f_i$ . Furthermore, there is a set of client locations (such as stores) that require to be serviced by a facility. The objective is to determine a set of locations at which facilities can be opened so as to minimize the total facility and assignment costs.

### 1.3.1 The uncapacitated facility location problem

Let  $N = \{1, \dots, n\}$  be a set of locations, and distances between them  $C_{ij}, i, j = 1, \dots, n$ . There is a subset  $F \subseteq N$  of locations at which we may open a facility and a subset  $D \subseteq N$  of locations that must be assigned to some open facility. For each location  $j \in D$ , there is a positive integral demand  $d_j$  that must be shipped to its assigned location. For each location  $i \in F$ , the non-negative cost of opening a facility at  $i$  is  $f_i$ . The cost of assigning location  $i$  to an open facility at  $j$  is  $C_{ij}$  per unit of demand shipped. It is assumed that these costs are non-negative, symmetric, and satisfy the triangle inequality: that is,  $C_{ij} = C_{ji}$  for all  $i, j \in N$ , and  $C_{ij} + C_{jk} \geq C_{ik}$  for all  $i, j, k \in N$ . The problem is to find a feasible assignment of each location in  $D$  to an open facility so as to minimize the total cost incurred (Charikar M *et al.*, 2005).

### 1.3.2 The capacitated facility location problem

The case in which each facility can assign to serve a total demand that is at most  $u$ , where  $u$  is a positive integer. In the capacitated case, the situation is somewhat more complicated. First of all, there are two variants of the problem, depending on whether each location's demand must be assigned to only one facility, or the demand may be fractionally split among more than one (completely) open facility (Charikar M *et al.*, 2005).

The algorithm is based on rounding an optimal solution to its linear programming relaxation. This linear programming relaxation is identical to the one used in the uncapacitated case, except for that you must explicitly require that:

$$0 \leq y_i \leq 1 \quad \text{For each } i \in F,$$

and impose capacity constraints

$$\sum_{j \in D} d_j x_{ij} \leq u y_i \quad \text{For each } i \in F,$$

It is not possible to design an approximation algorithm for the capacitated problem based solely on this linear programming relaxation, since the ratio between its integers and fractional optimal is unbounded.

## 2. The studies of the p-median problem:

Several analytical studies have been performed in recent years relating to the problem of p-median. One of the most remarkable results in the study of p-median problems is due to Hakimi (Chhajed D, and Lowe T.J. 1992). Hakimi proved that the search for the set of p optimal locations for the facilities can be limited to the node set of the graph instead of the infinite number of points that lie on the links. This important result made it to study the problem in a discrete space rather than in the more complex continuous setting.

Many variations of the classical p-median model are derived from different choices for some element features and/or for the kind of relations among some basic components. One variation of the facility features can produce Media Shortest Path Problems (MSPP) in the case of facilities presenting specialized shapes, such as trees. Mobile facility location problems, for example, deal with the location of mobile facilities that travel in the space and stop at several points where users can receive services. Also, a network version of the p-median problem which includes interaction

among new facilities exists, (i.e. the p-median problem with Mutual Communication (PMMC)).

Very extensive is also the study of variations of the classical model arising from the uncertainty underlying some customer and location features (Chhajed D, and Lowe T.J. 1992). P-median problems dealing with uncertainty are usually referred to as stochastic network p-medians. Uncertain parameters can involve customer features such as demands, customer–location relations such as travel time, and customer-facility relations such as server availability.

Spatial distribution of candidate locations can differentiate among p-median problems. A special case is the one in which the underlying metric space is tree.

Solutions to p-median problems maximize the consumer accessibility to server facilities, since access is usually strictly related to distance. Consequently, this model is applicable in those location contexts where maximizing the consumer access to supply centers is a major objective and reasonably assuming that consumers visit the nearest facility. This is likely to be the case for convenience stores, fast food outlets, and services such as banks and post offices. More generally, minimum objectives are especially appropriate in the context of facility construction for the delivery of non-emergency services. However, this criterion tends to favor customers, which are clustered together to the detriment of customers who are spatially dispersed. This flaw induced some authors to question the adequacy of minimum objective to those and must be guaranteed to all clients.

### 3. Factors affecting results:

Several factors affect the results of the p-median algorithm. These factors are divided into two categories: the hardware factors and the software factors.

#### 3.1 Hardware factors

These factors are independent on the algorithm of the p-median problem. They depend on the characteristics of the hardware and how we implement the algorithm on it. The results of the p-median algorithm are affected by several characters, which are the CPU speed, the cache memory of the CPU, the size of used memory, and the speed of the bus on the main board.

These factors affect the time, speedup, and efficiency of the gated results.

Allocating instructions to actual physical locations in RAM can be done at three different times in the life cycle of a program

1. Compile Time: Is done by the compiler and is not that adaptable. The operating system simply loads the program into those locations.
2. Load Time: Is done by the operating system as it creates the new process, and is reasonably adaptable.
3. Execute Time: While the program is being executed, the hardware performs the address binding (this is called virtual addressing).

Virtual memory is a technique that allows the execution of processes that may not be completely in physical memory. One major advantage of this scheme is that programs can be larger than physical memory. Further, virtual memory abstracts main memory into an extremely large, uniform array of storage, separating logical memory as viewed by the user from physical memory. This technique frees programmers from the concerns of memory-storage limitations. Virtual memory also allows processes to easily share files and address spaces, and it provides an efficient mechanism for process creation (Silberschatz A. *et al.*, 2002).

In the virtual addressing, the process does not have to be contiguous in (RAM). There are two algorithms which allow the operating system to implement this. In both of these schemes, the memory of a process is broken up into blocks. In paging, the blocks are all the same size, they are called pages. In segmentation, the blocks can be of different sizes; they are called segments.



### 3.2 Number of computers in the network

The p-median algorithm is applied on one computer and multiple computers. Which work under the same circumstances and have the same characters (CPU, RAM, Bus speed).

Computers can be connected together in a network of windows 2000 professional operating system. The number of computers in the network can be increased to consolidate the speedup and efficiency of the p-median algorithm.

### 3.1 Software factors:

#### 3.4.1 Size of data set:

As the data set becomes larger, the amount of calculation increases. The computer running the p-median algorithm becomes incapable of continuing these calculations, which would definitely affect the speedup and efficiency of the algorithm and increase the time needed to perform all operations.

#### 3.4.2 Number of facilities:

The speedup is also affected by the number of facilities and their best location in the space of all possible locations. Increasing the number of facilities in the data set lead to increasing the time needed to find the best solution.

We have implemented the p-median algorithm in a lab that consists of a number of similar computers that are connected together in a network of windows 2000 professional operating system have the same characters, and work under the same circumstances.

## PARALLEL SOLUTION

### 1. Introduction:

When you apply the p-median algorithm on one computer, all mathematic calculations are done on the physical memory using paging technique. When the physical memory is full, the calculations continue on the virtual memory. At a specific input data set of large size, all memories become full, then one computer cannot perform additional operations. Consequently, the p-median algorithm is found to be inefficient when the input data set is large.

To solving this problem, we use the data decomposition technique. Which would divide the data set into smaller parts, then give every computer in the network one part of data to do calculations, and collect all results to get the exact solution. All computers in the network are running in synchronization. The two algorithms, p-median algorithm, and parallel p-median algorithm that employ data decomposition technique are followed.

### 2. The sequential p-median algorithm:

#### Initialization:

Denote the set of all possible locations  $g(1...n)$  with  $N(g)$ . Denote the set of permutation facilities  $p \{1, \dots, k\}$  with  $U(k)$ . Let the first  $p$  of them represent an initial solution. Denote the set of objective functions at each permutation  $\{1, \dots, p\}$  with  $F(k)$ ,  $k=1, \dots, p$ . The following is the algorithm to solve the p-median problem sequentially. Note Algorithm Generate  $p(n)$  generates the possible permutations, as illustrated in Figure 5.

**Algorithm Generate  $p(n)$** 

Initial with permutation  $p$ .

For each facility  $k$  ( $k = 1, \dots, p$ ) do the following

For each possible location  $g$  ( $g = 1, \dots, n - p$ ), in  $(N - U)$ , do the following

Replace  $k$  by  $g$

Generate the new permutation  $p$

End for  $g$

End for  $k$

Figure 5: Generates the possible permutations.

The Algorithm p-median  $(n, p)$  computes the objective function  $f(F(k))$  as shown in Figure 6.

**Algorithm p-median  $(n, p)$** 

Generate a new permutation by the **Generate Algorithm**

For each permutation  $p$  do the following

For each user  $i$  ( $i = 1, \dots, n - p$ ) do the following

(\* For users without facilities\*)

For each facility  $k$  ( $k = 1, \dots, p$ ) do the following

Calculate the distance  $d_{ki}$  between user  $i$  and facility  $p$ ;

. If ( $d_{ki} \leq d_{\min}$ ) then

$d_{\min} = d_{ki}$

. Endif

End for  $k$

End for  $i$

Calculate  $F(k)$  the summation of all  $d_{\min}$ ;

End for  $p$

Find the minimum value of  $F(k)$  and the permutation  $p$  generated at this value; you will then get the optimal function, which is  $F(k)_{opt}$

Figure 6: The sequential algorithm to solve the p-median problem.

### 3. Parallel p-median algorithm:

Each computer in the network where the parallel p-median algorithm must be implemented has the complete size of the data set, and has a sequence number, which is either  $H$  for horizontal decomposition,  $V$  for vertical decomposition, or  $GR$  for grid decomposition.

Consider the following initialization for the parallel p-median algorithms.

#### Initialization:

Denote the set of all possible locations  $g(1, \dots, n)$  with  $N(g)$

Denote the set of permutation facilities  $p \{1, \dots, k\}$  with  $U(k)$ . Let the first  $p$  represent an initial solution.

Denote the set of objective functions at each permutation with  $F(k)$ ,  $k = 1, \dots, p$ .

Denote each set of data divided horizontally  $g(1, \dots, l)$  with  $h(g)$ , and  $h$  is the number of sets in horizontal decomposition.

Denote each set of data divided vertically  $g(1, \dots, m)$  with  $v(g)$ , and  $v$  is the number of sets in vertical decomposition.

Denote each set of data divided as grid  $g(1, \dots, n)$  with  $Gr(g)$ , and  $Gr$  is the number of sets in grid decomposition.

Denote the boundaries of each set vertically  $v(g)$  with  $\{x_{\max}, x_{\min}\}$ .

Denote the boundaries of each set horizontally  $h(g)$  with  $\{y_{\max}, y_{\min}\}$ .

Denote the boundaries of all data set  $N(g)$  with  $\{X_{\max}, X_{\min}\}$ ,  $\{Y_{\max}, Y_{\min}\}$ .

Use the algorithm illustrated in Figure 3.1 to generate a permutation for algorithms in Figure 7, Figure 8, and Figure 9.

**Algorithm horizontal data decomposition**  $(N, h)$ 

Initial with  $x_{\min} = X_{\min}$ ,  $x_{\max} = X_{\max}$ , and  $y_{\min} = Y_{\min}$ ,  $y_{\max} = (Y_{\max} - Y_{\min})/h$ , and  $H = 1$ .

(\* Where  $H$  is the sequence number of computer\*)

While  $H \leq h$  do the following

For each set  $h(g)$  do the following

For each node in  $N(g)$  do

If  $(y_{\max} < y < y_{\min})$  then

$h(g) = h(g) + g$

End for  $N(g)$ .

Apply the **Algorithm p-median**  $(n, p)$  for computer with sequence number  $H$ , and on the data set  $h(g)$ .

The boundaries for the next set are

$y_{\min+1} = y_{\max}$ ,  $y_{\max+1} = y_{\max} + (Y_{\max} - Y_{\min})/h$ .

$x_{\min} = X_{\min}$ ,  $x_{\max} = X_{\max}$

$H = H + 1$ ;

End for  $h(g)$ .

End while.

Figure 7: The parallel p-median algorithm on horizontal decomposition.

**Algorithm vertical data decomposition**  $(N, v)$ 

Initial with  $y_{\min} = Y_{\min}$ ,  $y_{\max} = Y_{\max}$ , and  $x_{\min} = X_{\min}$ ,  $x_{\max} = (X_{\max} - X_{\min})/v$ , and  $V = 1$ .

(\* Where  $V$  is the sequence number of computer\*)

While  $V \leq v$  do the following

For each set  $v(g)$  do the following

For each node in  $N(g)$  do

If  $(x_{\max} < x < x_{\min})$  then

$v(g) = v(g) + g$

End for  $N(g)$ .

Apply the **Algorithm p-median**  $(n, p)$  for computer with the sequence number  $V$ , and on the data set  $v(g)$ .

The boundaries for the next set are

$x_{\min+1} = x_{\max}$ ,  $x_{\max+1} = x_{\max} + (X_{\max} - X_{\min})/v$ .

$y_{\min} = Y_{\min}$ ,  $y_{\max} = Y_{\max}$

$V = V + 1$ ;

End for  $v(g)$ .

Figure 8: The parallel p-median algorithm on vertical decomposition.

### Algorithm grid data decomposition ( $N, h$ )

Initial with  $x_{\min} = X_{\min}$ ,  $x_{\max} = (X_{\max} - X_{\min})/v$ , and  $y_{\min} = Y_{\min}$ ,  $y_{\max} = (Y_{\max} - Y_{\min})/h$ , and  $GR = 1$ .

(\* Where  $GR$  is the sequence number of computer\*)

While  $GR \leq Gr$  do the following  
For each set  $Gr(g)$  do the following

For each node in  $N(g)$  do

If  $(y_{\max} < y > y_{\min})$  and  $(x_{\max} < x > x_{\min})$  then

$Gr(g) = Gr(g) + g$

End for  $N(g)$ .

Apply the **Algorithm p-median** ( $n, p$ ) for computer with sequence number  $GR$ , and on the data set  $Gr(g)$ .

The boundaries for the next set are

While  $x_{\max} \leq X_{\max}$  do the following

$x_{\min+1} = x_{\max}$ ,  $x_{\max+1} = x_{\max} + (X_{\max} - X_{\min})/v$

$y_{\min} = Y_{\min}$ ,  $y_{\max} = Y_{\max}$

End while.

While  $y_{\max} \leq Y_{\max}$  do the following

$y_{\min+1} = y_{\max}$ ,  $y_{\max+1} = y_{\max} + (Y_{\max} - Y_{\min})/h$

$x_{\min} = X_{\min}$ ,  $x_{\max} = X_{\max}$

$GR = GR + 1$ ;

End while.

End for  $Gr(g)$ .

Figure 9: The parallel p-median algorithm on grid decomposition.

### Algorithm p-median ( $n, p$ )

Generate a new permutation by the **Generate Algorithm**

For each permutation  $p$  do the following

For each user  $i$  in the set  $S$  do the following

(\* Where  $S$  might be  $h(g)$  or  $v(g)$  or  $Gr(g)$ \*)

For each facility  $k$  ( $k = 1, \dots, p$ ) do the following

Calculate the distance  $d_{ki}$  between user  $i$  and facility  $p$ ;

. If  $(d_{ki} \leq d_{\min})$  then

$d_{\min} = d_{ki}$

. Endif

End for  $k$

End for  $i$

Calculate  $F(k)$  the summation of all  $d_{\min}$ ;

End for  $p$ .

Figure 10: the p-median ( $n, p$ ) algorithm.

All computers are running in the synchronization mode. Each one sends the table of extracted objective functions to the server, then the server sums all the tables given from computers in the network. The server finds the minimum value of  $F(k)$  and the permutation  $p$  that was generated at this value, and then we get the optimal function that is  $F(k)_{opt}$ .

The sequence p-median algorithm and the parallel p-median algorithm are implemented and tested on the data sets and on the available computers. The number of  $p$  has been computed. The time of execution, speedup, and efficiency have been computed.

#### 4. Complexity analysis

When we aim to develop or use an algorithm on large problems, it is important to understand how long the algorithm might take to run. The time for most algorithms depends on the amount of data or size of the problem. In order to analyze an algorithm, we try to find a relationship showing how the time needed for the algorithm depends on the amount of data. This is called the "complexity" of the algorithm. A simple algorithm may have a high complexity, whereas an algorithm which is very complex in its organization sometimes pays off by having a lower complexity in the sense of time needed for computation (Computer Science 150/Core 142 Contemporary Issues in Computer Science).

In our work, the complexity of the sequential p-median algorithm can be derived as follows.

The complexity of generating permutations  $R$  can be derived from the algorithm shown in Figure 3.1 as illustrated in Equation 3.1.

$$C = \sum_{k=1}^P \sum_{g=1}^{N-P} R = R(N - P)(P) \quad (3.1)$$

Then the complexity of the sequential p-median algorithm can be derived from Equation 3.2 and Equation 3.3.

$$\sum_{k=1}^P \sum_{g=1}^{N-P} \sum_{i=1}^{N-P} \sum_{k=1}^P cd_{ki} = P.(N-P).(N-P).Pc \quad (3.2)$$

$$C_s(N, P) = P^2.(N^2 - 2N.P + P^2) \cong O(P^2N^2) \quad (3.3)$$

Where N is the size of data, P is the number of facilities,  $d_{ki}$  is the distance between user i and facility P, and c is a time cost for computing  $d_{ki}$ .

The complexity of the parallel algorithm can be derived as follows:

$$T = \text{Cost of (Computation + Communication)} \quad (3.4)$$

Here, we can assume that the cost of communication is constant K for one machine.

The cost of computation for the parallel algorithm can be derived as illustrated in Equation 3.5.

$$\sum_{k=1}^P \sum_{g=1}^{\frac{N-P}{m}} \sum_{i=1}^{\frac{N-P}{m}} \sum_{k=1}^P cd_{ki} = P.\left(\frac{N}{m} - P\right).\left(\frac{N}{m} - P\right).P.c \quad (3.5)$$

For fixed m and a cost of K, then:

$$C_p(N, P, m) = P^2.\left(\frac{N^2}{m^2} - 2\frac{N}{m}.P + P^2\right) + K.m \quad (3.6)$$

Thus, the speedup will be as shown in Equation 3.7.

$$S = \frac{C_s}{C_p} \quad (3.7)$$

## 5. Explanation of the p-median algorithms:

The following example illustrates the steps of the p-median algorithm in sequential and in parallel.

The data set for this example was extracted from the maximum covering location problem (Berman O *et al.*, 1990)..



Table 1: The coordinates of the first 20 nodes for the map.

Node Number (i)	X coordinate	Y coordinate
1	108	21
2	144	21
3	120	28
4	126	28
5	156	28
6	174	28
7	192	28
8	204	28
9	228	28
10	126	42

### 5.1 Applying the p-median algorithm on one computer:

For a fixed number of nodes  $n=10$  and the number of facilities  $p=4$ , give the following objective functions shown in Table 2.

Table 2: Some possible permutations and their objective functions for data in Table 1.

Number of Permutation	Set of Possible Locations for Facilities	Objective Function
1	[1,2,3,4]	297.10
3	[5,2,3,4]	241.67
3	[6,2,3,4]	205.90
4	[7,2,3,4]	187.60
5	[8,2,3,4]	199.50
6	[9,2,3,4]	248.19
7	[10,2,3,4]	297.10
8	[1,5,3,4]	257.78
9	[1,6,3,4]	227.20
10	[1,7,3,4]	221.20
11	[1,8,3,4]	233.20
12	[1,9,3,4]	299.20
13	[1,10,3,4]	383.20
14	[1,2,5,4]	260.41
15	[1,2,6,4]	224.63
16	[1,2,7,4]	206.33
17	[1,2,8,4]	218.23
18	[1,2,9,4]	266.92
19	[1,2,10,4]	310.53
20	[1,2,3,5]	250.80

Minimum  
objective  
function

For the permutations in Table 2, the optimal solution (permutation) is the permutation number 4: The set of possible locations for facilities is [7,2,3,4], and the value of the objective function is 187.60.

#### 4.2 Applying the parallel p-median algorithm on four computers:

Figure 11 shows the grid data decomposition for the data in Table3.1.

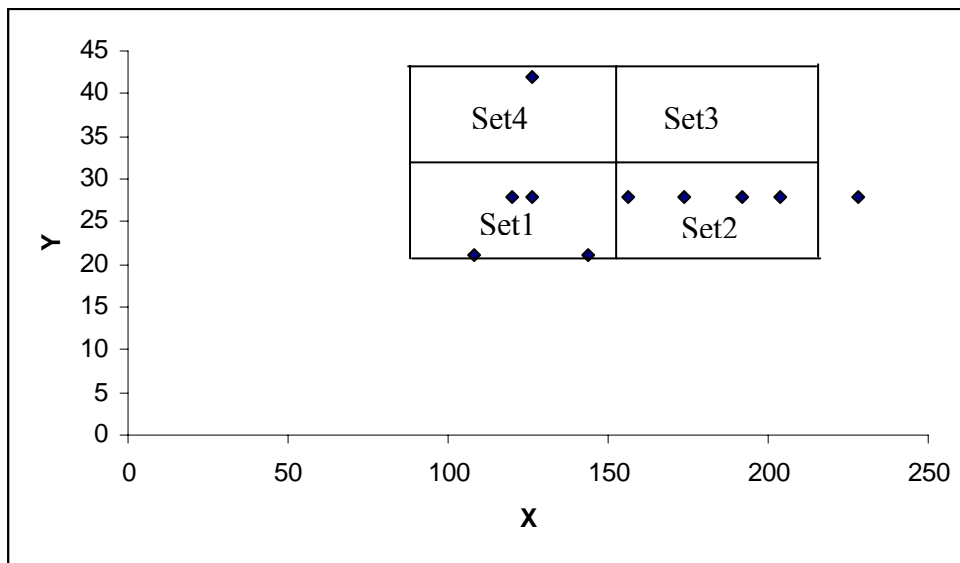


Figure 11: The distribution of points in the grid decomposition.

In this decomposition, set1 on computer1 has 5 points, set2 on computer2 has 4 points, set3 on computer3 has 0 points, and set4 on computer4 has 1 point.

The time needed to find the best solution in the parallel p-median algorithm is the time needed by the computer that take the largest time, plus the time of connection and the time of processing results from all computers in the server (see Table 3).

The decomposition in Figure 11 leads to a load balancing problem. This problem is outside the scope of this research.

Table 3: Computing the F on 4 computers for the data in Table 1.

Number of Permutation	Set of Possible Locations for Facilities	Objective Function for Computer Number 1 F1	Objective Function for Computer Number 2 F2	Objective Function for Computer Number 3 F3	Objective Function for Computer Number 4 F4	Objective Function for All Computers (F=F1+F2+F3+F4)
1	[1,2,3,4]	59.09	224.01	0.00	14.00	297.10
3	[5,2,3,4]	53.67	174.00	0.00	14.00	241.67
3	[6,2,3,4]	59.09	132.80	0.00	14.00	205.90
4	[7,2,3,4]	59.09	114.50	0.00	14.00	187.60
5	[8,2,3,4]	59.09	126.40	0.00	14.00	199.50
6	[9,2,3,4]	59.09	175.09	0.00	14.00	248.19
7	[10,2,3,4]	59.09	224.01	0.00	14.00	297.10
8	[1,5,3,4]	69.78	174.00	0.00	14.00	257.78
9	[1,6,3,4]	63.20	150.00	0.00	14.00	227.20
10	[1,7,3,4]	75.20	132.00	0.00	14.00	221.20
11	[1,8,3,4]	75.20	144.00	0.00	14.00	233.20
12	[1,9,3,4]	75.20	210.00	0.00	14.00	299.20
13	[1,10,3,4]	75.20	294.00	0.00	14.00	383.20
14	[1,2,5,4]	72.41	174.00	0.00	14.00	260.41
15	[1,2,6,4]	77.83	132.80	0.00	14.00	224.63
16	[1,2,7,4]	77.83	114.50	0.00	14.00	206.33
17	[1,2,8,4]	77.83	126.40	0.00	14.00	218.23
18	[1,2,9,4]	77.83	175.095	0.00	14.00	266.92
19	[1,2,10,4]	72.51	224.01	0.00	14.00	310.53
20	[1,2,3,5]	61.56	174.00	0.00	15.23	250.80

Minimum objective function

We found that the summation of the objective functions given by each computer when using the parallel data decomposition (parallel p-median algorithm) is the same when using the sequential p-median algorithm. Consequently, the same result can be computed, but the large size of data would result in less amount of run time.

Chapter4 will present results when the parallel and sequential algorithms are on different datasets.

## RESULTS AND ANALYSIS

### 1. Introduction:

In this research, we apply the sequential p-median algorithm and the parallel p-median algorithm on available data sets in a lab that consists of homogeneous computers. The results are followed in the next tables. These results are discussed for several states. The speedup and efficiency have been calculated and discussed. The comparison between the sequential and the parallel algorithms has been studied. The time and the size of data set and all factors that affect the performance of the algorithms have been studied.

In the experiments:

- a- The number of facilities is fixed and the size of data is change.
- b- The number of facilities is changed on fixed size of data.

This chapter includes:

1. Sequential p-median algorithm.
2. Results of parallel p-median algorithm.
  - i. On two computers (horizontal decomposition).
  - ii. On four computers grid decomposition (the data set was generated randomly with different sizes).
  - iii. On four computers vertical decomposition (the data set was generated randomly with different sizes).
  - iv. On four computers horizontal decomposition (the data set was generated randomly with different sizes).
  - v. On nine computers grid decomposition (the data set is the maximal covering location problem).

## 2. Sequential p-median algorithm:

Consider the data for the maximal covering location problem. In this, the number of facilities is assumed to be fixed and the size of data to be changed.

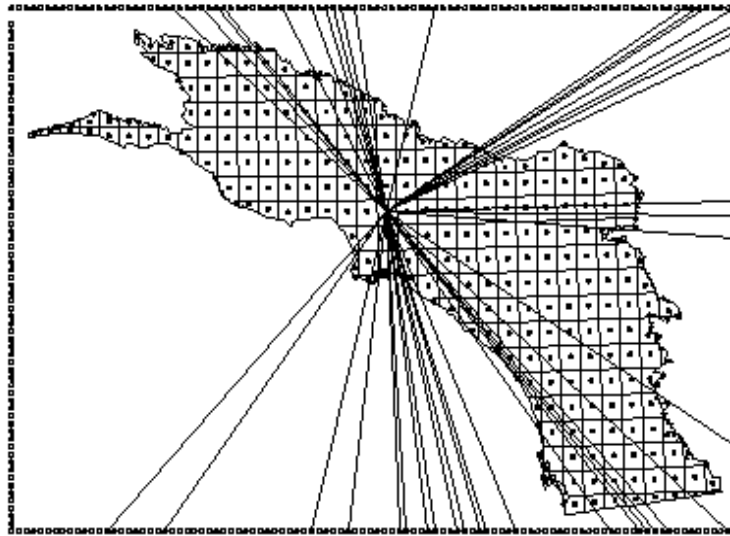


Figure 12: Sample arcs of the logical network of the Los Angeles quad.

Figure 12 shows the logical network between site nodes and nodes created to integrate the reserve selection model into an existing GIS location-allocation module (Berman O *et al.*, 1990). An application program was implemented in Delphi programming language is used to process this picture. The coordinates for every node in the map were extracted.

Applying the sequential p-median algorithm on one computer gave the results in Table 4.

Table 4: Changing the time and objective function while changing the size of data and fixed numbers of P for one computer.

N Number of All Nodes	P = 1		P = 2		P = 3		P = 4		P = 5	
	T Time in Second	F Objective Function	T Time in Second	F Objective Function	T Time in Second	F Objective Function	T Time in Second	F Objective Function	T Time in Second	F Objective Function
20	0.5	١٠٠٨٧٩,٦٢	0.5	٦٨٧,٧٣	0.75	٥٨٩,٨٨	1	٥٦٠,٥٦	1	٤٩٦,٧٣
40	1	١٠٢٢٠٦,٤٩	1	١٨٠٣,٦٥	1	١٦٧٨,٣٢	2	١٦٤٩,٠٠	2	١٥١٠,٠٧
60	1	١٠٤٣٧١,٦٤	2	٣٦٨٩,٣٠	2	٣٣٩٠,٦٧	4	٣٣٦٠,٣٠	42	٣١٥٨,٣٧
80	2	١٠٦٣٩٣,١٨	2	٥٥٧١,٤٥	4	٥١١٩,٥٥	76	٥٠٨٧,٤٩	265	٤٩١٦,٧٩
100	2	١١٠٦٤٦,٦١	3	٨٣٢٧,٦٩	51	٧٧٤٤,٧٦	293	٧٧١١,٣٧	Cannot be continue	-
120	3	١١٤٨٢٦,٩٩	5	١١٢٤٨,٩٧	92	١٠٦٢١,٤٤	Cannot be continue	-	-	-
140	3	١١٨٤٧٨,٥١	19	١٣٥١٥,٩٦	463	١٢٨٤٧,٠٢	-	-	-	-
160	3	١٢١٩٠٩,٠٩	76	١٥٩٠١,٥٠	Cannot be continue	-	-	-	-	-
180	3	١٢٥٣٣٠,١٤	215	١٧٨٦٣,٠٩	-	-	-	-	-	-
200	4	١٢٩٢٣٤,٥٥	480	٢٠٥١٣,٩١						
220	7	١٣٣٣٨٦,٥٨	715	٢٢٨٦٨,٧٩						
240	9	١٣٧٦٩٦,٩٠	Cannot be continue	-						
260	13	١٤٢٤٥٧,٤٦	-	-						
280	56	١٤٧٦٢٣,١١	-	-						
300	86	١٥٣١١٨,٣٤	-	-						

Table 4: Continue.

N Number of All Nodes	P = 6		P = 7		P = 8		P = 9		P = 10	
	T Time in Second	F Objective Function	T Time in Second	F Objective Function	T Time in Second	F Objective Function	T Time in Second	F Objective Function	T Time in Second	F Objective Function
20	1	٤٣١,٥٦	1	٣٨٣,٦٦	1	٣٥٦,٥٣	2	٣٣٠,٨١	2	٣٢٤,٣٨
40	3	١٣٩٦,٧٧	4	١٣١٦,٠٦	47	١٢٧٩,٠٨	62	١٢٤٣,٩٠	118	١٢٣١,٣٦
60	141	٣٠٠٢,٠٥	340	٢٩٠٣,٧٧	612	٢٨٥٠,٢٠	Cannot be continue	-	Cannot be continue	-
80	Cannot be continue	-	Cannot be continue	-	Cannot be continue	-	-	-	-	-



Table 4 shows the times and the values of the objective function for testing the sequential p-median algorithm where  $P=1,2,3\dots 10$ . The number of nodes varies from 20 to 300. For one facility, the algorithm continue executing for  $N=20,40,60,\dots,300$ . For 2 facilities, the algorithm continue executing for  $N=20,40,60,\dots,220$ , the algorithm fails to execute for  $N=240$  and above. It is clear as the values of  $P$  increases the values of  $N$  used in computation must be decreased in order to handle the computations.

The time of execution increases rapidly if  $P$  changes from 1 to 10. For example when  $N=60$  the time are 1, 2, 4, 42, 141, 340, and 612 for  $P=1,2,3,\dots,8$ , respectively. The values of the objective functions  $F$  decrease as the number of facilities  $P$  increases for all fixed values of  $N$ .

Figure 13 shows how the time changes when the size of data  $N$  and numbers of facilities  $P$  change.

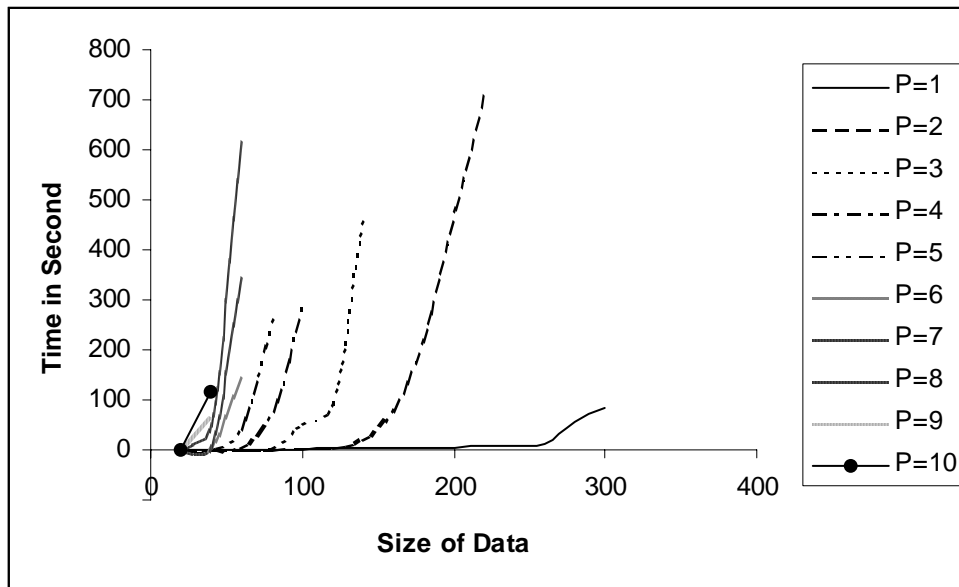


Figure 13: Relation between the number of facilities ( $P$ ) and the time ( $T$ ) where number of computer =1.

### 3. Parallel p-median algorithms:

#### 3.1 Two computers:

Table 5 shows the Results on two computers, with horizontal decomposition, to compute F for the problem in Figure 12:

Table 5: Changing the time and objective function while changing the size of data and fixed numbers of P for 2 computers.

	P = 1	P = 2	P = 3	P = 4	P = 5	P = 6	P = 7	P = 8	P = 9	P = 10
N Number of All Nodes	T Time in Second	T Time in Second	T Time in Second	T Time in Second	T Time in Second	T Time in Second	T Time in Second	T Time in Second	T Time in Second	T Time in Second
20	0.1	0.1	0.5	0.5	1	1	1	1	1	1
40	0.3	0.3	1	2	2	2	3	4	6	37
60	0.3	0.3	2	2	3	6	71	175	330	425
80	0.5	2	3	4	62	132	360	651	Cannot be continue	Cannot be continue
100	1	3	6	88	349	Cannot be continue	Cannot be continue	Cannot be continue		
120	2	3	49	263	Cannot be continue					
140	2	4	98	520						
160	2	6	196	Cannot be continue						
180	2	6	198							
200	3	10	430							
220	3	140	Cannot be continue							
240	4	193								
260	5	476								
280	8	930								
300	12	Cannot be continue								

Using two computers decrease the execution time to find F for a fixed number of N. See for example, the time to compute F for 60 nodes and P=8 on one computer and on two computers. The size of data that can be tested on two computers is larger than on one computer.

Figure 14 shows how the time is changed when the size of data N and the number of facilities P are changed.

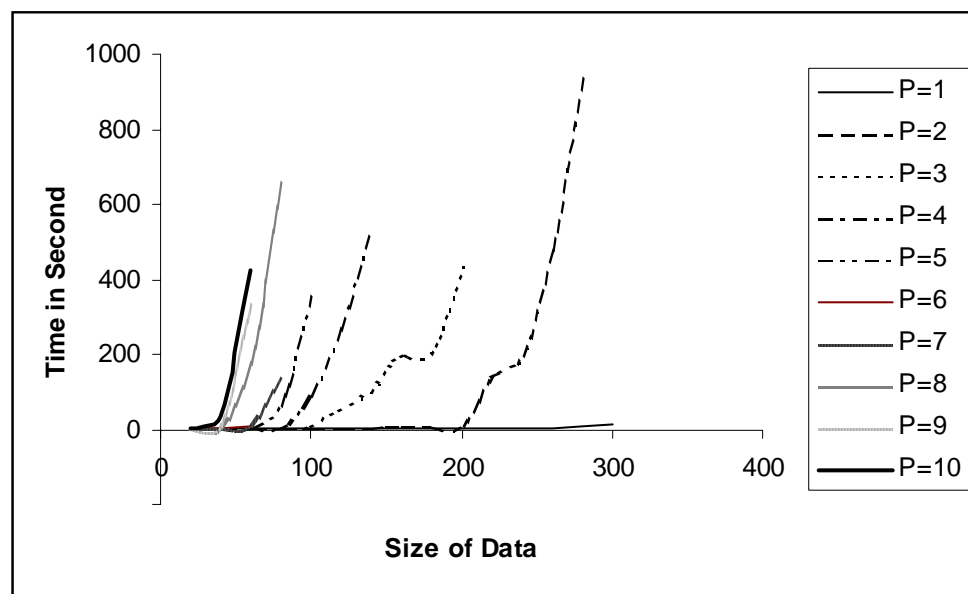


Figure 14: Relation between the number of facilities (p) and the time (T) where number of computer=2 (horizontal decomposition).

Note that the values of the objective functions computed by the parallel p-median algorithm, for the same data set, is equal to same values computed by the sequential p-median algorithm.

### 3.2 Using four computers with grid decomposition:

Table 6 shows results when four computers on grid decomposition were used. This sample of data was generated randomly by an application program implemented in Delphi programming language.

Table 6: Changing the time and objective function while changing the size of data and fixed numbers of P for 4 computers (grid decomposition).

	P = 1	P = 2	P = 3	P = 4	P = 5	P = 6	P = 7	P = 8	P = 9	P = 10
N Number of All Nodes	T Time in Second	T Time in Second	T Time in Second	T Time in Second	T Time in Second	T Time in Second	T Time in Second	T Time in Second	T Time in Second	T Time in Second
20	1	1	1	1	1	1	1	1	1	1
40	1	1	1	1	2	2	2	2	3	3
60	1	11	1	2	2	3	5	8	63	118
80	1	2	2	2	4	27	75	142	390	540
100	1	2	3	5	86	192	480	Cannot be continue	Cannot be continue	Cannot be continue
120	1	2	3	14	120	400	Cannot be continue			
140	1	3	6	108	392	Cannot be continue				
160	2	3	50	330	Cannot be continue					
180	2	4	129	764						
200	2	7	269	Cannot be continue						
220	3	44	535							
240	3	65	Cannot be continue							
260	3	96								
280	3	173								
300	4	270								

Using four computers with grid decomposition decreases the execution time to find  $F$  for a fixed number of  $N$ . See for example, the time to compute  $F$  for 60 nodes and  $P=8$  on one computer and on four computers. The size of data that can be tested on two computers is larger than on one computer

Figure 15 shows how the time changes while size of data  $N$  and number of facilities  $P$  change.

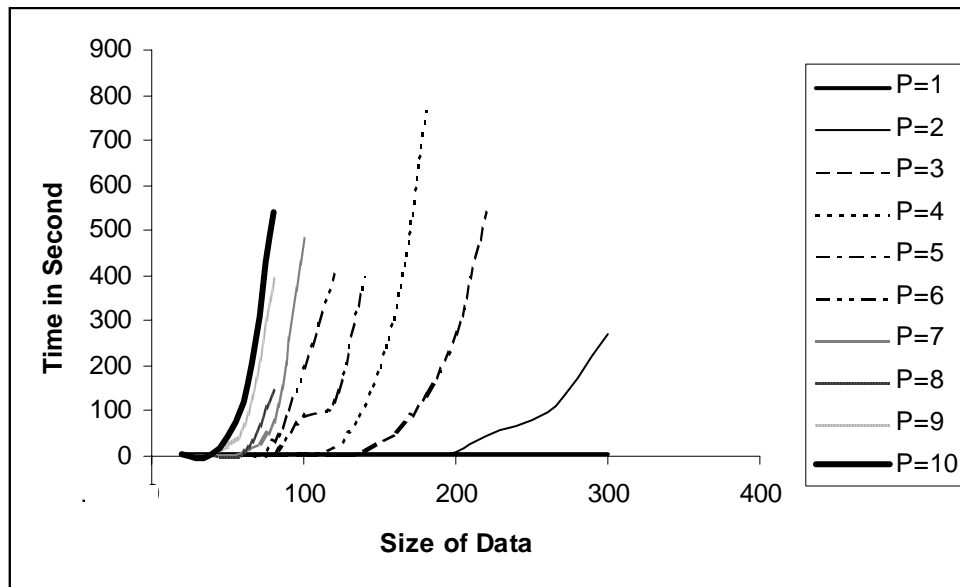


Figure 15: Relation between the time ( $T$ ) and the number of facilities ( $p$ ) where the number of computer=4 (grid decomposition).

### 3.3 Using four computers with vertical decomposition:

Table 7 shows results when four computers on vertical decomposition were used.

The data was generated randomly.

Table 7: The time and objective function while changing the size of data and fixed numbers of P for 4 computers(vertical decomposition).

	P = 1	P = 2	P = 3	P = 4	P = 5	P = 6	P = 7	P = 8	P = 9	P = 10
N Number of All Nodes	T Time in Second	T Time in Second	T Time in Second	T Time in Second	T Time in Second	T Time in Second	T Time in Second	T Time in Second	T Time in Second	T Time in Second
20	1	1	1	1	1	1	1	1	1	1
40	1	1	1	1	2	3	3	3	4	4
60	1	1	2	2	3	3	5	10	18	29
80	1	1	2	3	4	8	30	45	264	Cannot be continue
100	1	2	3	5	20	59	322	Cannot be continue	Cannot be continue	
120	1	2	4	29	138	480	Cannot be continue			
140	2	3	7	112	348	Cannot be continue				
160	2	4	31	196	Cannot be continue					
180	2	4	71	296						
200	3	5	75	Cannot be continue						
220	3	6	160							
240	3	9	Cannot be continue							
260	3	12								
280	3	12								
300	4	20								

Using four computers with vertical decomposition decrease the execution time to find F for a fixed number of N. The time to compute F for 60 nodes and P=8 on one computer is 612 seconds, and on four computers the time is 10 seconds. The size of data that can be tested on four computers is larger than on one computer

The Figure 16 shows how the time is changes as the size of data N and number of facilities P is changed.

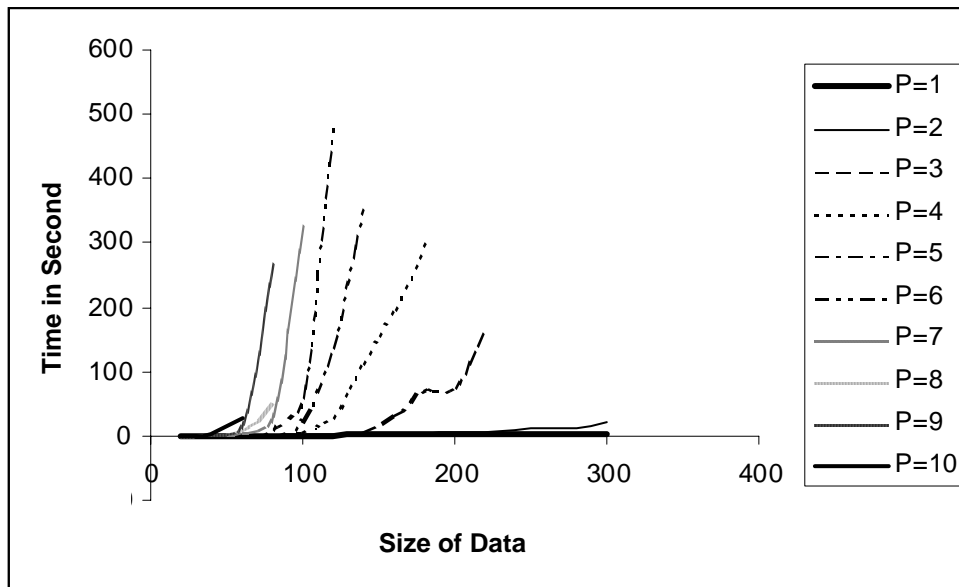


Figure 16: Relation between the time (T) and the number of facilities (p) where the number of computer=4 (vertical decomposition).

### 3.4 Using four computers with horizontal decomposition:

Table 8 shows results on four computers (vertical decomposition).



Table 8: The time and objective function while changing the size of data and fixed numbers of P for 4 computers(horizontal decomposition).

	P = 1	P = 2	P = 3	P = 4	P = 5	P = 6	P = 7	P = 8	P = 9	P = 10
N Number of All Nodes	T Time in Second	T Time in Second	T Time in Second	T Time in Second	T Time in Second	T Time in Second	T Time in Second	T Time in Second	T Time in Second	T Time in Second
20	1	1	1	1	1	1	1	1	1	1
40	1	1	1	1	2	3	1	2	2	3
60	1	1	1	2	3	4	5	9	62	136
80	1	1	2	3	5	11	112	237	229	Cannot be continue
100	1	2	3	5	12	96	282	Cannot be continue	Cannot be continue	
120	1	3	5	52	210	620	Cannot be continue			
140	2	4	7	66	278	Cannot be continue				
160	2	4	9	72	Cannot be continue					
180	2	4	14	210						
200	2	5	120	Cannot be continue						
220	3	8	169							
240	3	18	Cannot be continue							
260	3	20								
280	4	107								
300	4	110								

Using four computers with horizontal decomposition decrease the execution time to find F for a fixed number of N. For example, the time to compute F for 60 nodes and P=8 on one computer is 612 seconds, and on four computers the time is 9 seconds. The size of data that can be tested on four computers is larger than on one computer. Figure 17: shows how the time changes while the N and P are changed.

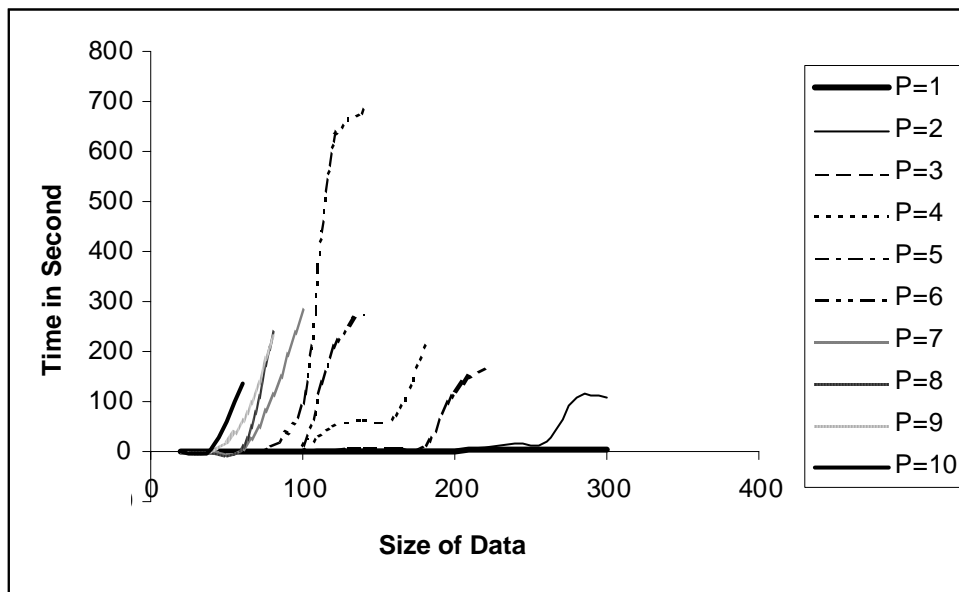


Figure 17: Relation between the time (T) and the number of facilities (p) where the number of computer=4 (horizontal decomposition).

### 3.5 Nine computers with grid data decomposition:

Table 9 shows the results on nine computers (grid Decomposition) to compute F for the problem in Figure4.1:

Table 9: The time and objective function while changing the size of data and fixed numbers of P for 9 computers (grid decomposition).

	P = 1	P = 2	P = 3	P = 4	P = 5	P = 6	P = 7	P = 8	P = 9	P = 10
N Number of All Nodes	T Time in Second	T Time in Second	T Time in Second	T Time in Second	T Time in Second	T Time in Second	T Time in Second	T Time in Second	T Time in Second	T Time in Second
20	1	1	1	1	1	1	1	1	1	1
40	1	1	1	1	1	2	2	2	2	2
60	1	1	1	1	2	2	2	3	4	6
80	1	1	2	3	3	5	21	79	138	280
100	1	1	2	3	4	39	95	230	390	780
120	1	2	2	3	7	82	190	406	819	Cannot be continue
140	1	2	3	45	124	404	Cannot be continue	Cannot be continue	Cannot be continue	
160	1	2	4	66	261	701				
180	1	3	4	69	265	750				
200	1	3	6	102	380	Cannot be continue				
220	2	3	54	256	Cannot be continue					
240	2	5	134	701						
260	2	8	341	Cannot be continue						
280	3	11	Cannot be continue							
300	3	58								

Using nine computers with horizontal decomposition decrease the execution time to find F for a fixed number of N. For example, the time to compute F for 60 nodes and P=8 on one computer is 612 seconds, and on nine computers the time is 3 seconds. The size of data that can be tested on tow computers is larger than on one computer. Figure 18: shows how the time changes while the N and P are changed.

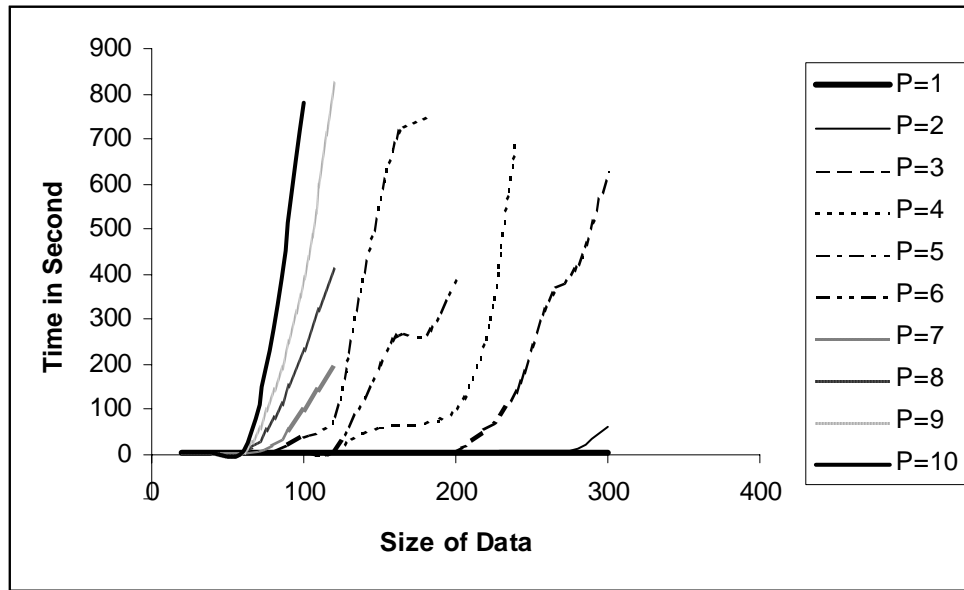


Figure 18: Relation between the time (T) and the number of facilities (p) where number of computer=9 (grid decomposition).

#### 4. Comparison among decomposition types:

The following tables and figures show which type of decomposition is better.

Numbers in bold has no theoretical values, for example when  $N=260$  the speedup must be less than 4, and the efficiency must be less than 100%, but here the speedup is 4.33 and the efficiency is 108.33%. This is a super speedup. This results when the sequential algorithm starts thrashing when it works on data set with large size.

Table 10: Time, speedup, efficiency for grid, horizontal, and vertical decompositions where P=1 and number of computers is 4.

Number of Nodes	Time Needed in Grid Decomposition	Time Needed in Horizontal Decomposition	Time Needed in Vertical Decomposition	S(Speed Up) for Grid Decomposition	S(Speed Up) for Horizontal Decomposition	S(Speed Up) for Vertical Decomposition	E(efficiency) for Grid Decomposition	E(efficiency) for Horizontal Decomposition	E(efficiency) for Vertical Decomposition
20	1	1	1	1.00	1.00	1.00	25.00	25.00	25.00
40	1	1	1	1.00	1.00	1.00	25.00	25.00	25.00
60	1	1	1	1.00	1.00	1.00	25.00	25.00	25.00
80	1	1	1	2.00	2.00	2.00	50.00	50.00	50.00
100	1	1	1	2.00	2.00	2.00	50.00	50.00	50.00
120	1	1	1	3.00	3.00	3.00	75.00	75.00	75.00
140	1	2	2	3.00	1.50	1.50	75.00	37.50	37.50
160	2	2	2	1.50	1.50	1.50	37.50	37.50	37.50
180	2	2	2	1.50	1.50	1.50	37.50	37.50	37.50
200	2	2	3	2.00	2.00	1.33	50.00	50.00	33.33
220	3	3	3	2.33	2.33	2.33	58.33	58.33	58.33
240	3	3	3	3.00	3.00	3.00	75.00	75.00	75.00
260	3	3	3	<b>4.33</b>	<b>4.33</b>	<b>4.33</b>	<b>108.33</b>	<b>108.33</b>	<b>108.33</b>
280	3	4	3	<b>18.67</b>	<b>14.00</b>	<b>18.67</b>	<b>466.67</b>	<b>350.00</b>	<b>466.67</b>
300	4	4	4	<b>21.50</b>	<b>21.50</b>	<b>21.50</b>	<b>537.50</b>	<b>537.50</b>	<b>537.50</b>

Figure 19 shows how the speedup changes with the size of data, while changing the type of decomposition. The number of facilities is 1.

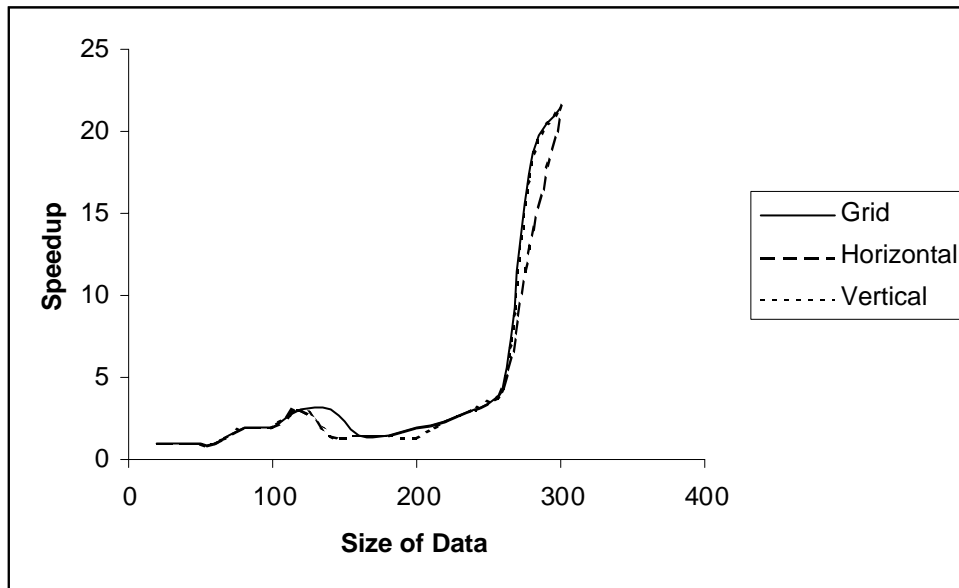


Figure 19: The speedup versus the size of data when using the three data decomposition on four computers.

Figure 20 shows how the efficiency changes with the size of data for different data decomposition, where number of facilities is 1 and number of computers is 4.

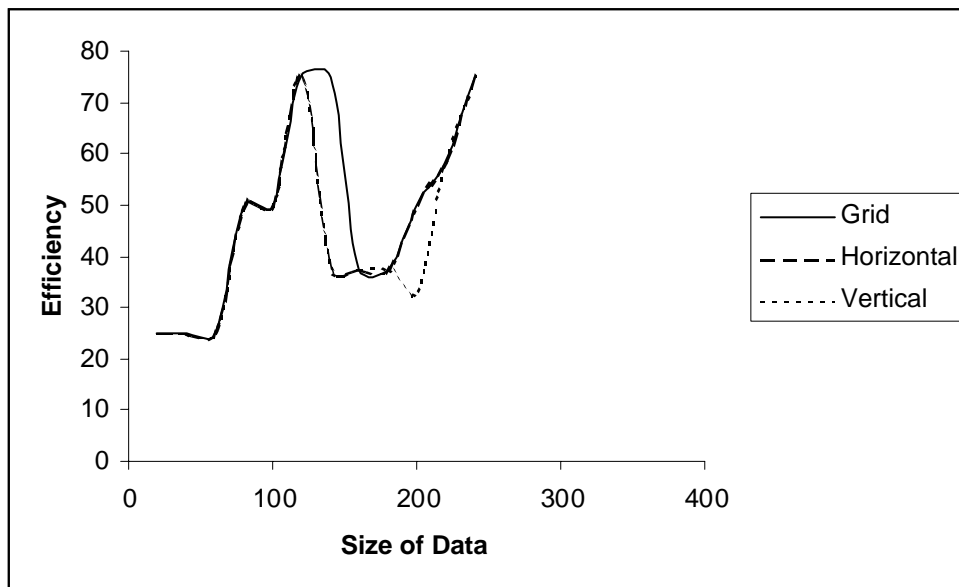


Figure 20: The efficiency is change with the size of data on 4 computers.

In general, for  $P=1$ , we find that the grid decomposition has better speedup and efficiency than the horizontal and the vertical decomposition.

Table 11: Time, speedup, efficiency for grid, horizontal, and vertical decompositions where P=2 and number of computers is 4.

Number of Nodes	Time Needed in Grid Decomposition	Time Needed in Horizontal Decomposition	Time Needed in Vertical Decomposition	S(Speed Up) for Grid Decomposition	S(Speed Up) for Horizontal Decomposition	S(Speed Up) for Vertical Decomposition	E(efficiency) for Grid Decomposition	E(efficiency) for Horizontal Decomposition	E(efficiency) for Vertical Decomposition
20	1	1	1	1.00	1.00	1.00	25.00	25.00	25.00
40	1	1	1	1.00	1.00	1.00	25.00	25.00	25.00
60	1	1	1	2.00	2.00	2.00	50.00	50.00	50.00
80	2	1	1	1.00	2.00	2.00	25.00	50.00	50.00
100	2	2	2	1.50	1.50	1.50	37.50	37.50	37.50
120	2	3	2	2.50	1.67	2.50	62.50	41.67	62.50
140	3	4	3	<b>6.33</b>	<b>4.75</b>	<b>6.33</b>	<b>158.33</b>	<b>118.75</b>	<b>158.33</b>
160	3	4	4	<b>25.33</b>	<b>19.00</b>	<b>19.00</b>	<b>633.33</b>	<b>475.00</b>	<b>475.00</b>
180	4	4	4	<b>53.75</b>	<b>53.75</b>	<b>53.75</b>	<b>1343.75</b>	<b>1343.75</b>	<b>1343.75</b>
200	7	5	5	<b>68.57</b>	<b>96.00</b>	<b>96.00</b>	<b>1714.29</b>	<b>2400.00</b>	<b>2400.00</b>
220	44	8	6	<b>16.25</b>	<b>89.38</b>	<b>119.17</b>	<b>406.25</b>	<b>2234.38</b>	<b>2979.17</b>

Numbers in bold has no theoretical values, for example when N=140 the speedup must be less than 4, and the efficiency must be less than 100%, but here the speedup is 6.33 and the efficiency is 158.33%. The number of computers was 4 and the speedup must be less than or equal to 4 and the efficiency must be less than or equal to 100%.

Figure 21 shows how the speedup changes with the size of data and the change of the type of decomposition, where number of facilities is 2.

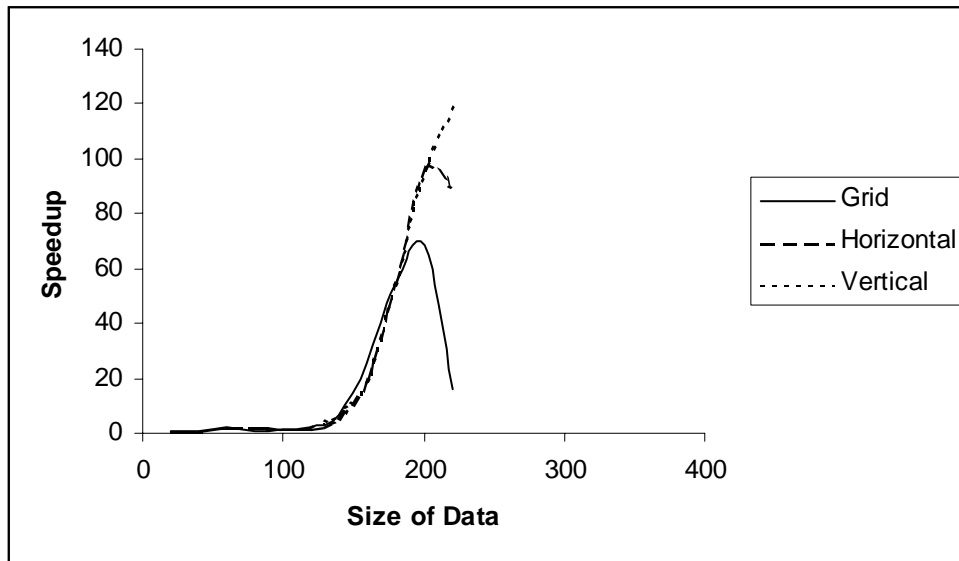


Figure 21: The speedup versus the size of data.

Figure 22 shows how the efficiency changes with the size of data and the change of the type of decomposition, where number of facilities is 2.

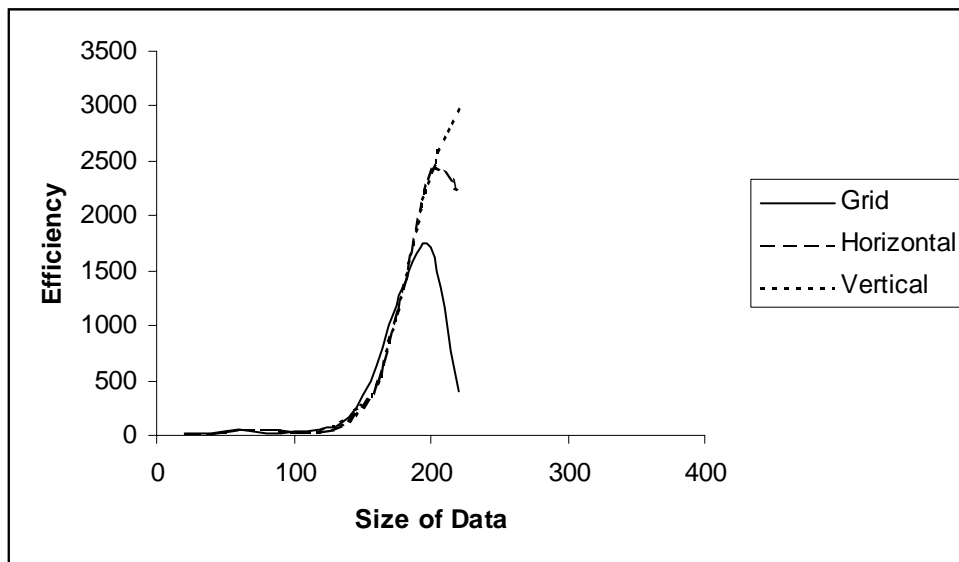


Figure 22: The efficiency against the size of data and number of computers is 4.

Here, we find that the vertical decomposition is the better in the speedup and efficiency, then the vertical, then the grid.



Table 12: Time, speedup, efficiency for grid, horizontal, and vertical decompositions where P=3 on 4 computers.

Number of Nodes	Time Needed in Grid Decomposition	Time Needed in Horizontal Decomposition	Time Needed in Vertical Decomposition	S(Speed Up) for Grid Decomposition	S(Speed Up) for Horizontal Decomposition	S(Speed Up) for Vertical Decomposition	E(efficiency) for Grid Decomposition	E(efficiency) for Horizontal Decomposition	E(efficiency) for Vertical Decomposition
20	1	1	1	0.75	0.75	0.75	18.75	18.75	18.75
40	1	1	1	1.00	1.00	1.00	25.00	25.00	25.00
60	1	1	2	2.00	2.00	1.00	50.00	50.00	25.00
80	2	2	2	2.00	2.00	2.00	50.00	50.00	50.00
100	3	3	3	<b>17.00</b>	<b>17.00</b>	<b>17.00</b>	<b>425.00</b>	<b>425.00</b>	<b>425.00</b>
120	3	5	4	<b>30.67</b>	<b>18.40</b>	<b>23.00</b>	<b>766.67</b>	<b>460.00</b>	<b>575.00</b>
140	6	7	7	<b>77.17</b>	<b>66.14</b>	<b>66.14</b>	<b>1929.17</b>	<b>1653.57</b>	<b>1653.57</b>

Numbers in bold in Table 4.9, has no theoretical values. For example when N=100 the speedup must be less than 4, and the efficiency must be less than 100%, but here the speedup is 17.00 and the efficiency is 425.00%.

Figure 23 shows how the speedup changes with the size of data where number of facilities is 3 and number of computers is 4.

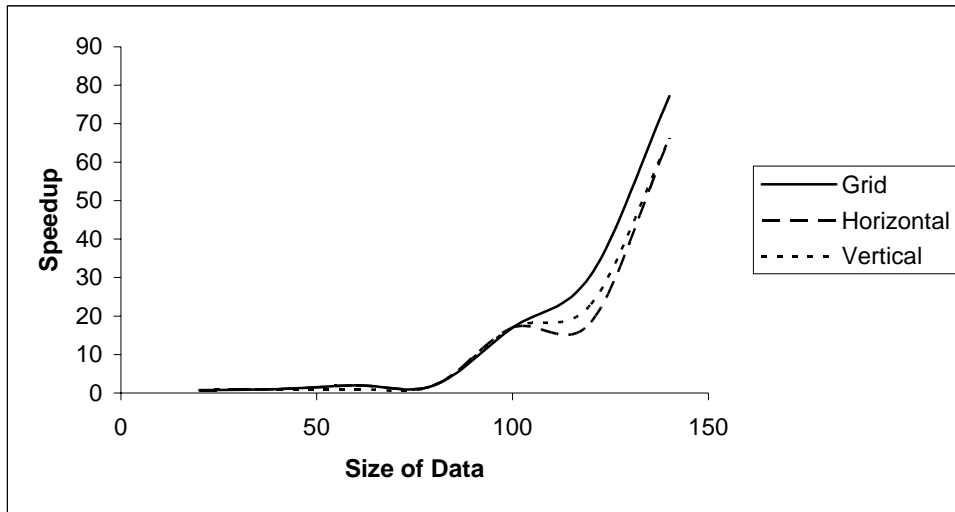


Figure 23: The speedup against the size of data.

Figure 24 shows how the efficiency changes with the size of data for the three types of decomposition, where number of facilities is 3.

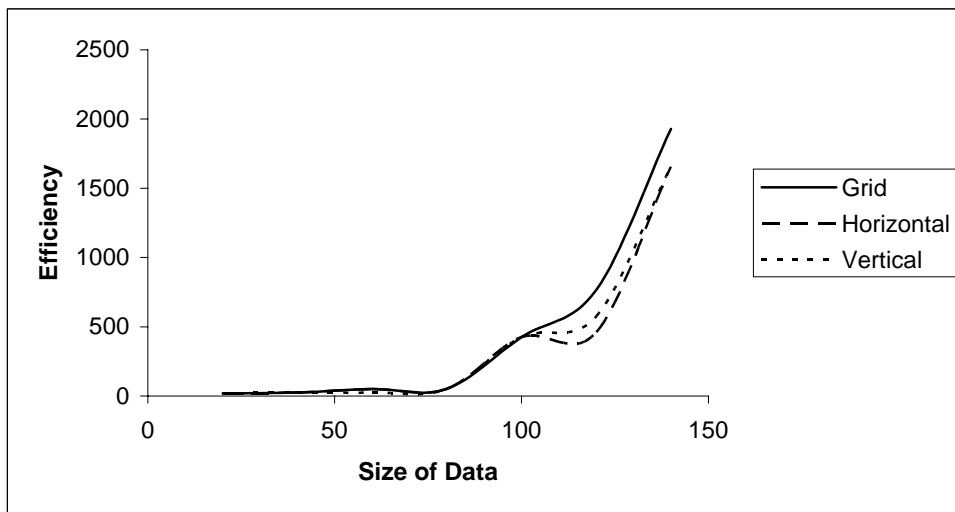


Figure 24: The efficiency versus the size of data.

In general, for  $P=3$ , we find that the grid decomposition has better speedup and efficiency than the horizontal and the vertical decomposition.

Table 13: Time, speedup, efficiency for grid, horizontal, and vertical decompositions where P=4 and number of computers is 4.

Number of Nodes	Time Needed in Grid Decomposition	Time Needed in Horizontal Decomposition	Time Needed in Vertical Decomposition	S(Speed Up) for Grid Decomposition	S(Speed Up) for Horizontal Decomposition	S(Speed Up) for Vertical Decomposition	E(efficiency) for Grid Decomposition	E(efficiency) for Horizontal Decomposition	E(efficiency) for Vertical Decomposition
20	1	1	1	1.00	1.00	1.00	25.00	25.00	25.00
40	1	1	1	2.00	2.00	2.00	50.00	50.00	50.00
60	2	2	2	2.00	2.00	2.00	50.00	50.00	50.00
80	2	3	3	<b>38.00</b>	<b>25.33</b>	<b>25.33</b>	<b>950.00</b>	<b>633.33</b>	<b>633.33</b>
100	5	5	5	<b>58.60</b>	<b>58.60</b>	<b>58.60</b>	<b>1465.00</b>	<b>1465.00</b>	<b>1465.00</b>

Numbers in bold has no theoretical values. For example when N=80, the speedup must be less than 4, and the efficiency must be less than 100%, but here the speedup is 38.00 and the efficiency is 950.00%.

Figure 25 shows how the speedup changes with the size of data for the three types of decomposition, where number of facilities is 4 and 4 computers.

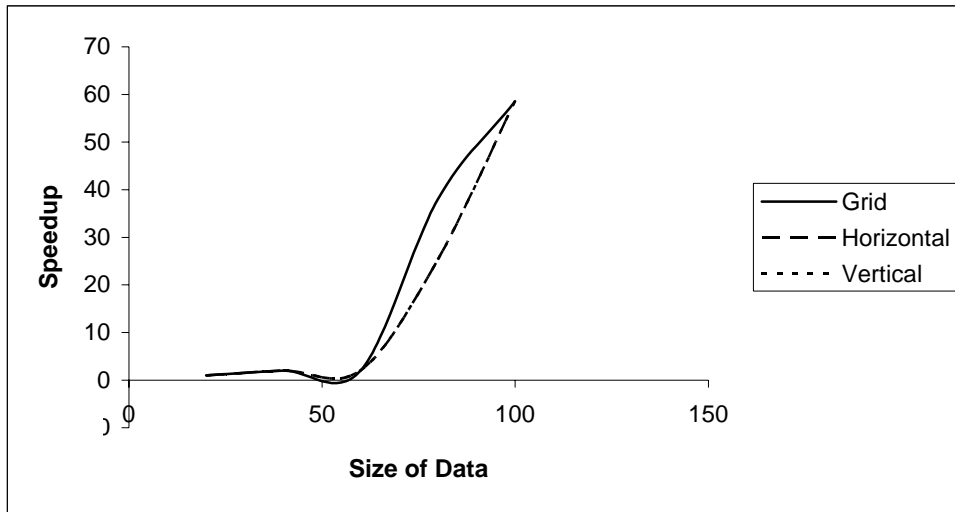


Figure 25: The speedup against the size of data.

Figure 26 shows how the efficiency changes with the size of data, where number of facilities is 4.

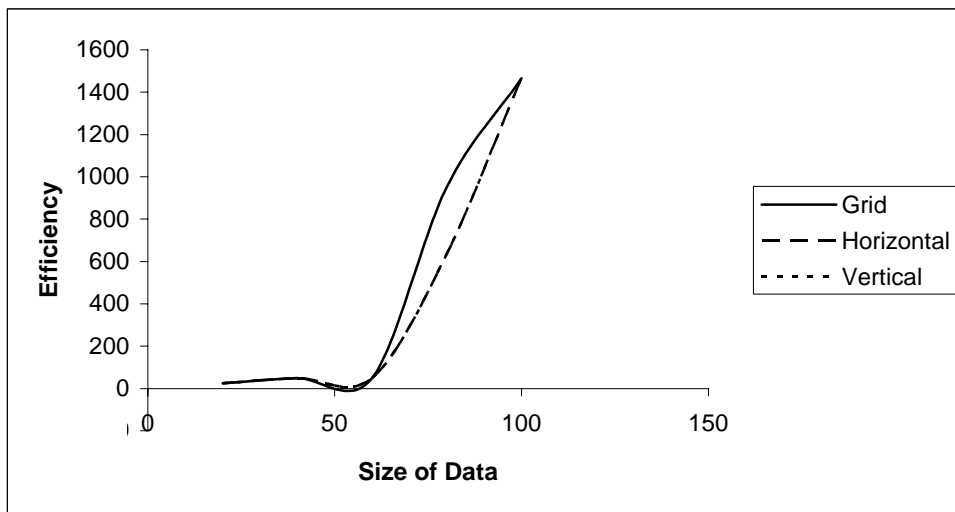


Figure 26: The efficiency with the size of data on 4 computers.

In general, for  $P=4$ , we find that the grid decomposition has better speedup and efficiency than the horizontal and the vertical decomposition.

Table 14: Time, speedup, efficiency for grid, horizontal, and vertical decompositions where P=5 on 4 computers.

Number of Nodes	Time Needed in Grid Decomposition	Time Needed in Horizontal Decomposition	Time Needed in Vertical Decomposition	S(Speed Up) for Grid Decomposition	S(Speed Up) for Horizontal Decomposition	S(Speed Up) for Vertical Decomposition	E(efficiency) for Grid Decomposition	E(efficiency) for Horizontal Decomposition	E(efficiency) for Vertical Decomposition
20	1	1	1	1.00	1.00	1.00	25.00	25.00	25.00
40	2	2	2	1.00	1.00	1.00	25.00	25.00	25.00
60	2	3	3	<b>21.00</b>	<b>14.00</b>	<b>14.00</b>	<b>525.00</b>	<b>350.00</b>	<b>350.00</b>
80	4	5	4	<b>66.25</b>	<b>53.00</b>	<b>66.25</b>	<b>1656.25</b>	<b>1325.00</b>	<b>1656.25</b>

Numbers in bold has no theoretical values. For example, when N=60, the speedup must be less than 4, and the efficiency must be less than 100%, but here the speedup is 21.00 and the efficiency is 525.00%.

Figure 27 shows how the speedup changes with the size of data, where number of facilities is 5 and number of computers is 4.

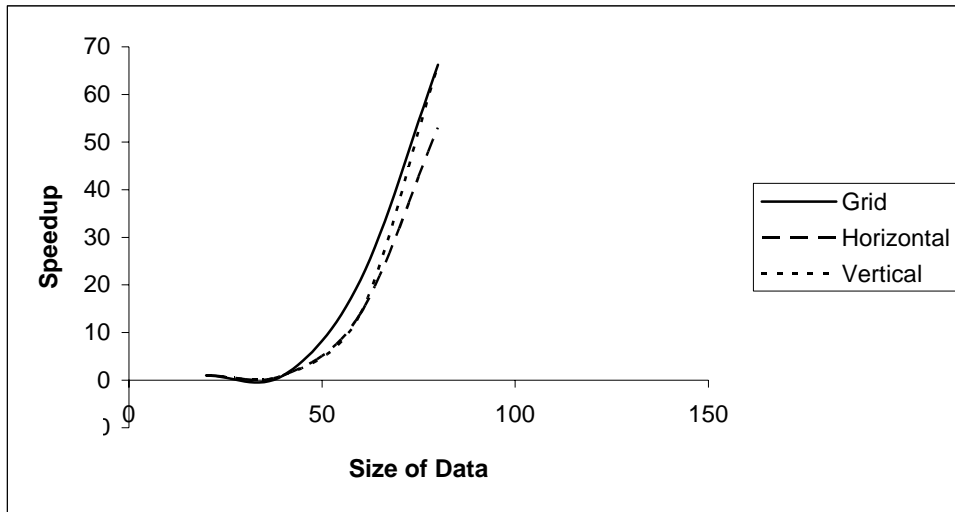


Figure 27: The speedup against the size of data.

Figure 28 shows how the efficiency changes with the size of data, where number of facilities is 5 and number of computers is 4.

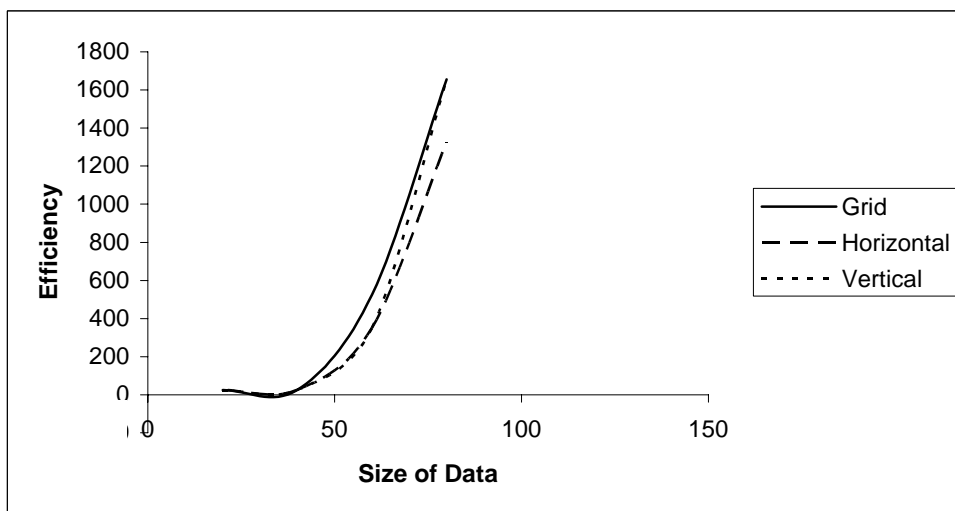


Figure 28: The efficiency versus the size of data.

In general, for  $P=5$ , we find that the grid decomposition has better speedup and efficiency than the horizontal and the vertical decomposition. This is because the better load balancing in grid decomposition than horizontal and vertical.

Table 15: Time, speedup, efficiency for grid, horizontal, and vertical decompositions where P=6 on 4 computers.

Number of Nodes	Time Needed in Grid Decomposition	Time Needed in Horizontal Decomposition	Time Needed in Vertical Decomposition	S(Speed Up) for Grid Decomposition	S(Speed Up) for Horizontal Decomposition	S(Speed Up) for Vertical Decomposition	E(efficiency) for Grid Decomposition	E(efficiency) for Horizontal Decomposition	E(efficiency) for Vertical Decomposition
20	1	1	1	1.00	1.00	1.00	25.00	25.00	25.00
40	2	3	3	1.50	1.00	1.00	37.50	25.00	25.00
60	3	4	3	<b>47.00</b>	<b>35.25</b>	<b>47.00</b>	<b>1175.00</b>	<b>881.25</b>	<b>1175.00</b>

Figure 29 shows how the speedup changes with the size of data, where number of facilities is 6.

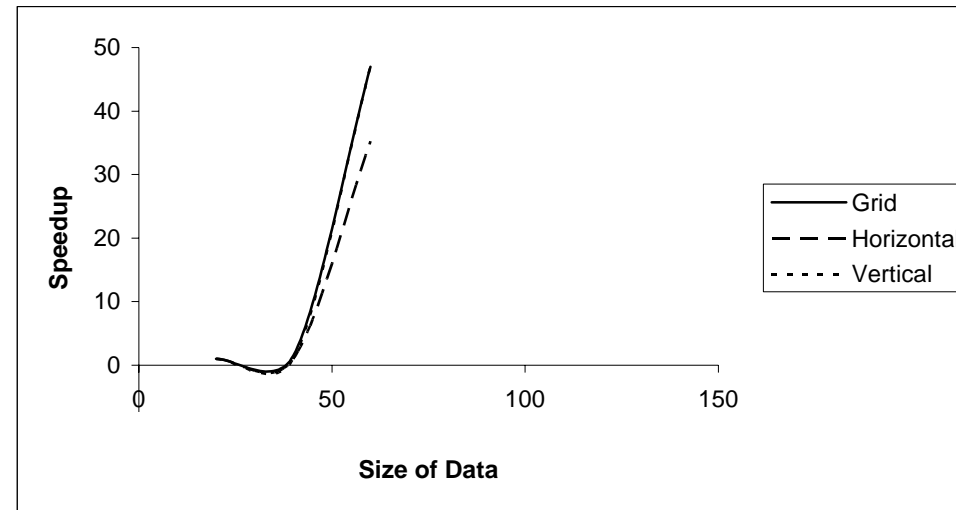


Figure 29: The speedup against the size of data.

Figure 30 shows how the efficiency changes with the size of data, where number of facilities is 6.

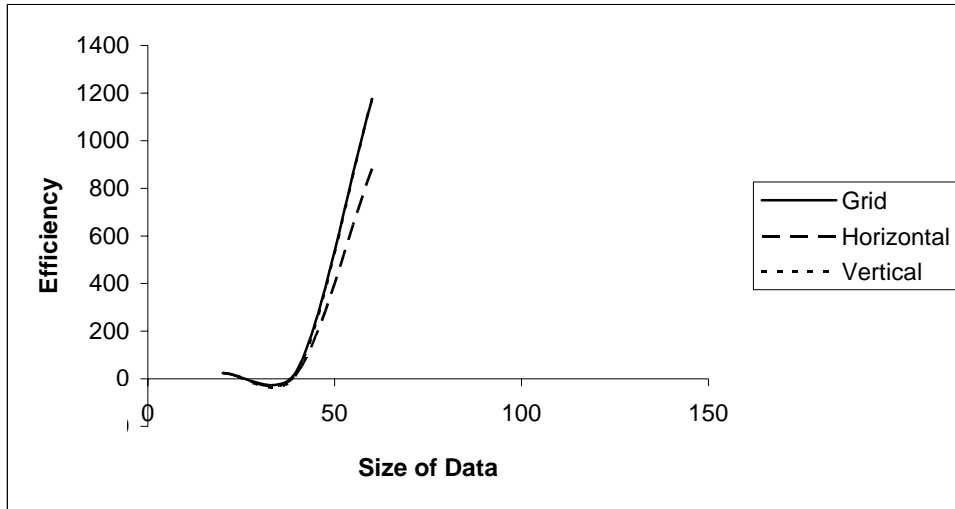


Figure 30: The efficiency versus the size of data.

In general, for  $P=6$ , we find that the grid decomposition has better speedup and efficiency than the horizontal and the vertical decomposition.



Table 16: Time, speedup, efficiency for grid, horizontal, and <sup>32</sup>vertical decompositions where P=7,8,9,10 on 4 computers.

Number of Facilities = 7									
Number of Nodes	Time Needed in Grid Decomposition	Time Needed in Horizontal Decomposition	Time Needed in Vertical Decomposition	S(Speed Up) for Grid Decomposition	S(Speed Up) for Horizontal Decomposition	S(Speed Up) for Vertical Decomposition	E(efficiency) for Grid Decomposition	E(efficiency) for Horizontal Decomposition	E(efficiency) for Vertical Decomposition
20	1	1	1	1.00	1.00	1.00	25.00	25.00	25.00
40	2	1	3	2.00	4.00	1.33	50.00	100.00	33.33
60	5	5	5	<b>68.00</b>	<b>68.00</b>	<b>68.00</b>	<b>1700.00</b>	<b>1700.00</b>	<b>1700.00</b>
Number of Facilities = 8									
Number of Nodes	Time Needed in Grid Decomposition	Time Needed in Horizontal Decomposition	Time Needed in Vertical Decomposition	S(Speed Up) for Grid Decomposition	S(Speed Up) for Horizontal Decomposition	S(Speed Up) for Vertical Decomposition	E(efficiency) for Grid Decomposition	E(efficiency) for Horizontal Decomposition	E(efficiency) for Vertical Decomposition
20	1	1	1	1.00	1.00	1.00	25.00	25.00	25.00
40	2	2	3	<b>23.50</b>	<b>23.50</b>	<b>15.67</b>	<b>587.50</b>	<b>587.50</b>	<b>391.67</b>
60	8	9	10	<b>76.50</b>	<b>68.00</b>	<b>61.20</b>	<b>1912.50</b>	<b>1700.00</b>	<b>1530.00</b>
Number of Facilities = 9									
Number of Nodes	Time Needed in Grid Decomposition	Time Needed in Horizontal Decomposition	Time Needed in Vertical Decomposition	S(Speed Up) for Grid Decomposition	S(Speed Up) for Horizontal Decomposition	S(Speed Up) for Vertical Decomposition	E(efficiency) for Grid Decomposition	E(efficiency) for Horizontal Decomposition	E(efficiency) for Vertical Decomposition
20	1	1	1	2.00	2.00	2.00	50.00	50.00	50.00
40	3	2	4	<b>20.67</b>	<b>31.00</b>	<b>15.50</b>	<b>516.67</b>	<b>775.00</b>	<b>387.50</b>
Number of Facilities = 10									
Number of Nodes	Time Needed in Grid Decomposition	Time Needed in Horizontal Decomposition	Time Needed in Vertical Decomposition	S(Speed Up) for Grid Decomposition	S(Speed Up) for Horizontal Decomposition	S(Speed Up) for Vertical Decomposition	E(efficiency) for Grid Decomposition	E(efficiency) for Horizontal Decomposition	E(efficiency) for Vertical Decomposition
20	1	1	1	2.00	2.00	2.00	50.00	50.00	50.00
40	3	3	4	<b>39.33</b>	<b>39.33</b>	<b>29.50</b>	<b>983.33</b>	<b>983.33</b>	<b>737.50</b>

In general, we found that the grid data decomposition is better in its speedup and efficiency. For example: when the size of data set is  $N=120$  and the number of facilities is  $P=3$ , the speedup (when using grid data decomposition) was 30.67. The efficiency was 766.67. The speedup in the horizontal data decomposition was 18.40 and the efficiency was 460. The speedup for the vertical data decomposition was 23 and the efficiency was 575. This different among results is repeated for all size of data set and for all values of the number of facilities. Then we can conclude that, the grid data decomposition is the better method for solving the p-median problem in parallel, and then the next is the vertical data decomposition, and then the horizontal data decomposition.

#### **5. Comparison between the sequential p-median algorithm and the parallel p-median algorithm:**

Found that the following tables and figures show the calculated efficiencies and speedup for tow computers, four computers, and nine computers.

Table 17: Comparing of the times, speedup, and efficiencies for one, tow, four, and nine computers, where P=1.

Number of Nodes	Time Needed in One PC	Time Needed in Two PCs	Time Needed in Four PCs	Time Needed in Nine PCs	S(Speed Up) for Two PCs	S(Speed Up) for Four PCs	S(Speed Up) for Nine PCs	E(efficiency) for Two PCs	E(efficiency) for Four PCs	E(efficiency) for Nine PCs
20	1	1	1	1	1.00	1.00	1.00	50.00	25.00	11.11
40	1	1	1	1	1.00	1.00	1.00	50.00	25.00	11.11
60	1	1	1	1	1.00	1.00	1.00	50.00	25.00	11.11
80	2	1	1	1	2.00	2.00	2.00	100.00	50.00	22.22
100	2	1	1	1	2.00	2.00	2.00	100.00	50.00	22.22
120	3	2	1	1	1.50	3.00	3.00	75.00	75.00	33.33
140	3	2	1	1	1.50	3.00	3.00	75.00	75.00	33.33
160	3	2	2	1	1.50	1.50	3.00	75.00	37.50	33.33
180	3	2	2	1	1.50	1.50	3.00	75.00	37.50	33.33
200	4	3	2	1	1.33	2.00	4.00	66.67	50.00	44.44
220	7	3	3	2	<b>2.33</b>	2.33	3.50	<b>116.67</b>	58.33	38.89
240	9	4	3	2	<b>2.25</b>	3.00	4.50	<b>112.50</b>	75.00	50.00
260	13	5	3	2	<b>2.60</b>	<b>4.33</b>	6.50	<b>130.00</b>	<b>108.33</b>	72.22
280	56	8	3	3	<b>7.00</b>	<b>18.67</b>	<b>18.67</b>	<b>350.00</b>	<b>466.67</b>	<b>207.41</b>
300	86	12	4	3	<b>7.17</b>	<b>21.50</b>	<b>28.67</b>	<b>358.33</b>	<b>537.50</b>	<b>318.52</b>

Figure 31 shows how the speedup changes with the size of data, for  $P=1$ .

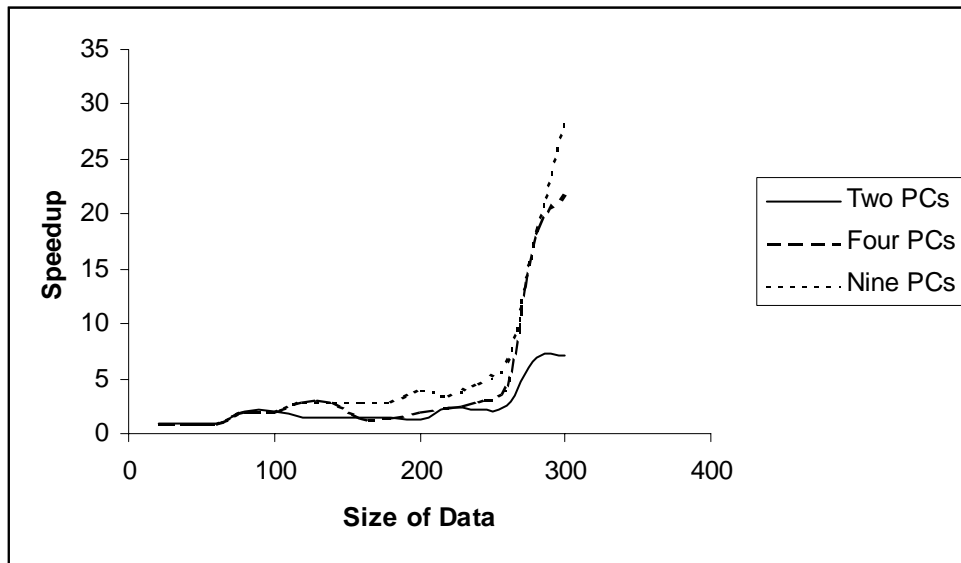


Figure 31: The speedup the size of data for  $P=1$ .

Figure 32 shows how the efficiency changes with the size of data, while changing the number of computers, and  $P=1$ .

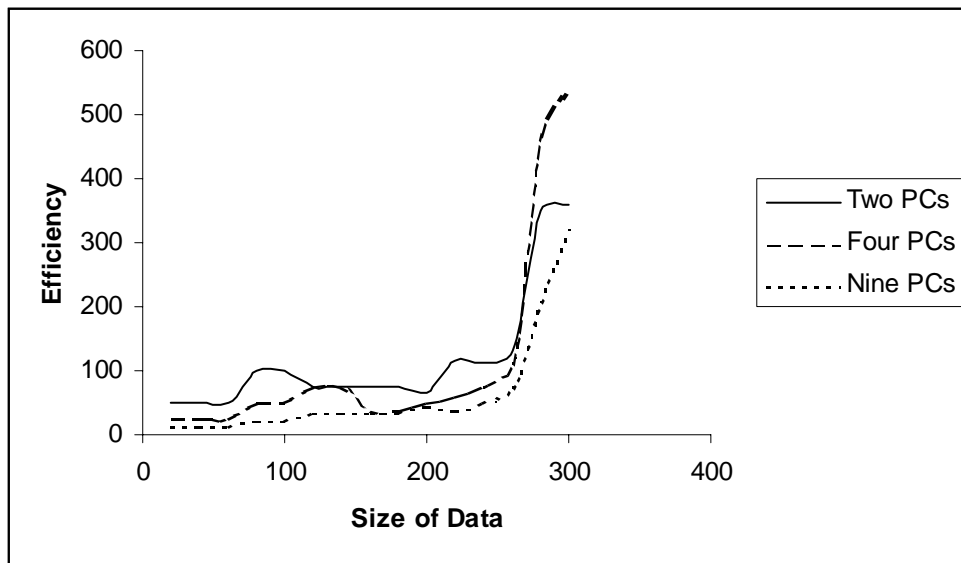


Figure 32: The efficiency with the size of data for  $P=1$ .

Table 18: Comparison the times, speedup, and efficiencies for one, two, four, and nine computers, where P=2.

Number of Nodes	Time Needed in One PC	Time Needed in Two PCs	Time Needed in Four PCs	Time Needed in Nine PCs	S(Speed Up) for Two PCs	S(Speed Up) for Four PCs	S(Speed Up) for Nine PCs	E(efficiency) for Two PCs	E(efficiency) for Four PCs	E(efficiency) for Nine PCs
20	1	1	1	1	1.00	1.00	1.00	50.00	25.00	11.11
40	1	1	1	1	1.00	1.00	1.00	50.00	25.00	11.11
60	2	1	1	1	2.00	2.00	2.00	100.00	50.00	22.22
80	2	2	2	1	1.00	1.00	2.00	50.00	25.00	22.22
100	3	3	2	1	1.00	1.50	3.00	50.00	37.50	33.33
120	5	3	2	1	1.67	2.50	5.00	83.33	62.50	55.56
140	19	4	3	2	<b>4.75</b>	<b>6.33</b>	<b>9.50</b>	<b>237.50</b>	<b>158.33</b>	<b>105.56</b>
160	76	6	3	2	<b>12.67</b>	<b>25.33</b>	<b>38.00</b>	<b>633.33</b>	<b>633.33</b>	<b>422.22</b>
180	215	6	4	3	<b>35.83</b>	<b>53.75</b>	<b>71.67</b>	<b>1791.67</b>	<b>1343.75</b>	<b>796.30</b>
200	480	10	7	3	<b>48.00</b>	<b>68.57</b>	<b>160.00</b>	<b>2400.00</b>	<b>1714.29</b>	<b>1777.78</b>
220	715	140	44	3	<b>5.11</b>	<b>16.25</b>	<b>238.33</b>	<b>255.36</b>	<b>406.25</b>	<b>2648.15</b>

Figure 33 shows how the speedup changes with the size of data, while the change of the number of computers, where number of facilities is 2.

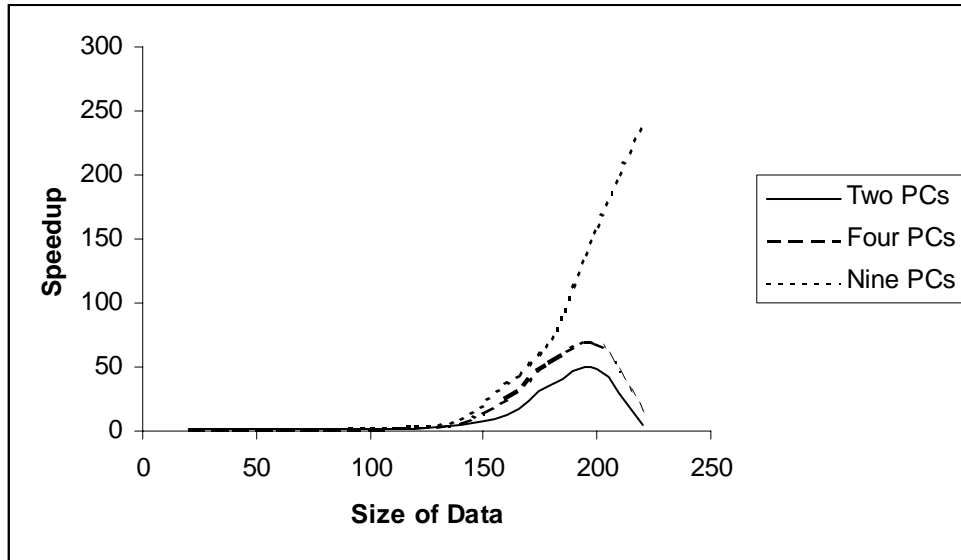


Figure 33: The speedup with the size of data, while  $P=2$ .

Figure 34 shows how the efficiency changes with the size of data, while the change of the number of computers, where number of facilities is 2.

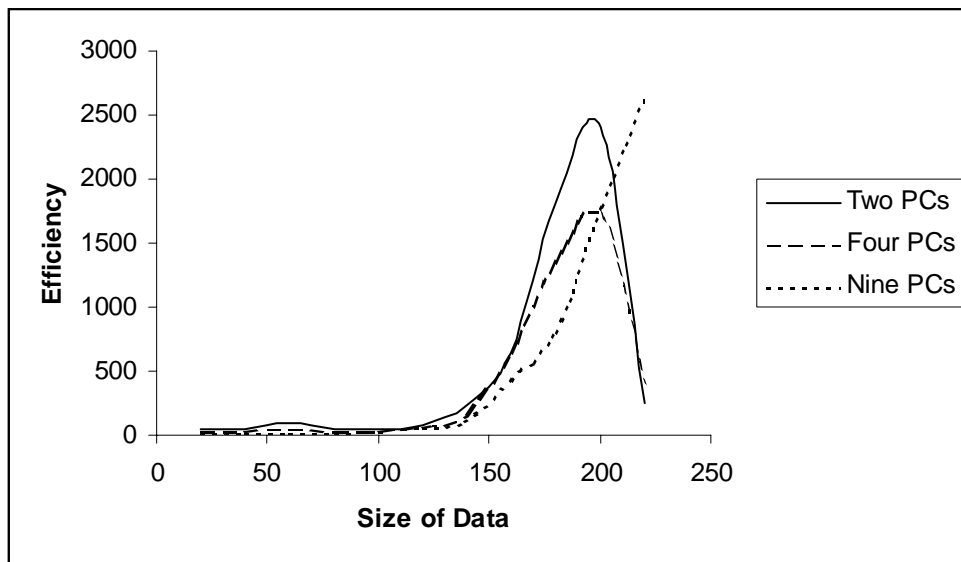


Figure 34: The efficiency with the size of data for  $P=2$ .

Table 19: The times, speedup, and efficiencies for one, four, and nine computers where  $P=3$ .

Number of Nodes	Time Needed in One PC	Time Needed in Two PCs	Time Needed in Four PCs	Time Needed in Nine PCs	S(Speed Up) for Two PCs	S(Speed Up) for Four PCs	S(Speed Up) for Nine PCs	E(efficiency) for Two PCs	E(efficiency) for Four PCs	E(efficiency) for Nine PCs
20	1	1	1	1	1.00	1.00	1.00	50.00	25.00	11.11
40	1	1	1	1	1.00	1.00	1.00	50.00	25.00	11.11
60	2	2	1	1	1.00	2.00	2.00	50.00	50.00	22.22
80	4	3	1.5	1	1.33	2.67	4.00	66.67	66.67	44.44
100	51	6	3	2	<b>8.50</b>	<b>17.00</b>	<b>25.50</b>	<b>425.00</b>	<b>425.00</b>	<b>283.33</b>
120	92	49	3	2	<b>1.88</b>	<b>30.67</b>	<b>46.00</b>	<b>93.88</b>	<b>766.67</b>	<b>511.11</b>

Figure 35 shows how the speedup changes with the size of data, while the change of the number of computers, for number of facilities is 3.

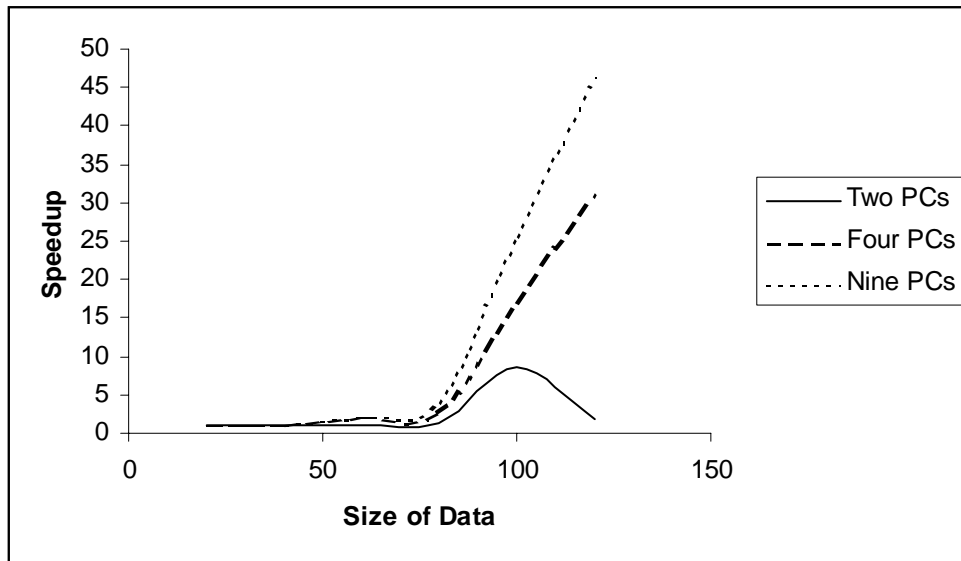


Figure 35: The speedup with the size of data and  $P=3$ .

Figure 36 shows how the efficiency changes with the size of data, while the change of the number of computers, for number of facilities is 3.

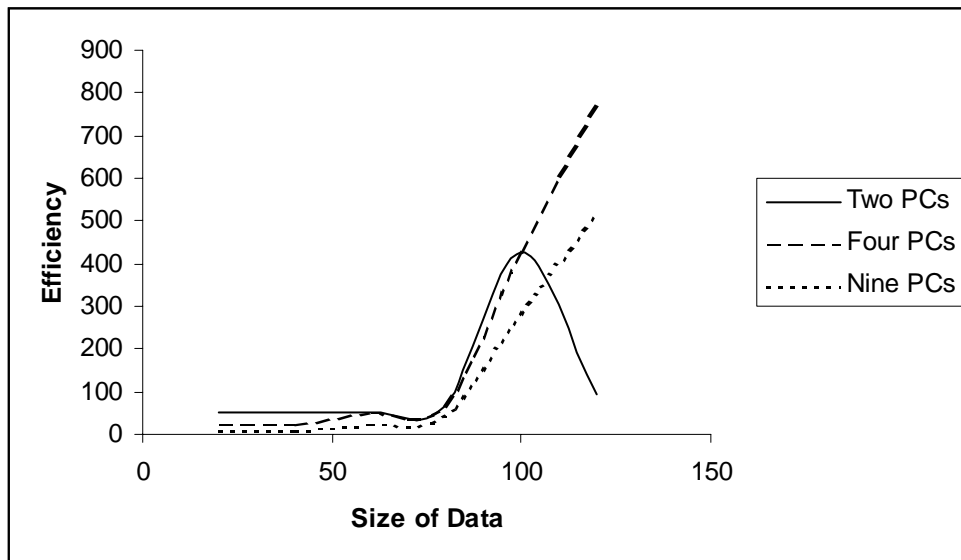


Figure 36: The efficiency against the size of data and  $P=3$ .



Table 20: The times, speedup, and efficiencies for one, two, four, and nine computers where P=4.

Number of Nodes	Time Needed in One PC	Time Needed in Two PCs	Time Needed in Four PCs	Time Needed in Nine PCs	S(Speed Up) for Two PCs	S(Speed Up) for Four PCs	S(Speed Up) for Nine PCs	E(efficiency) for Two PCs	E(efficiency) for Four PCs	E(efficiency) for Nine PCs
20	1	1	1	1	1.00	1.00	1.00	50.00	25.00	11.11
40	2	2	1	1	1.00	2.00	2.00	50.00	50.00	22.22
60	4	2.5	2	1	1.60	2.00	4.00	80.00	50.00	44.44
80	76	4	2	3	<b>19.00</b>	<b>38.00</b>	<b>25.33</b>	<b>950.00</b>	<b>950.00</b>	<b>281.48</b>
100	293	88	5	3	<b>3.33</b>	<b>58.60</b>	<b>97.67</b>	<b>166.48</b>	<b>1465.00</b>	<b>1085.19</b>

Figure 37 shows how the speedup changes with the size of data, while the change of the number of computers, for 4 facilities.

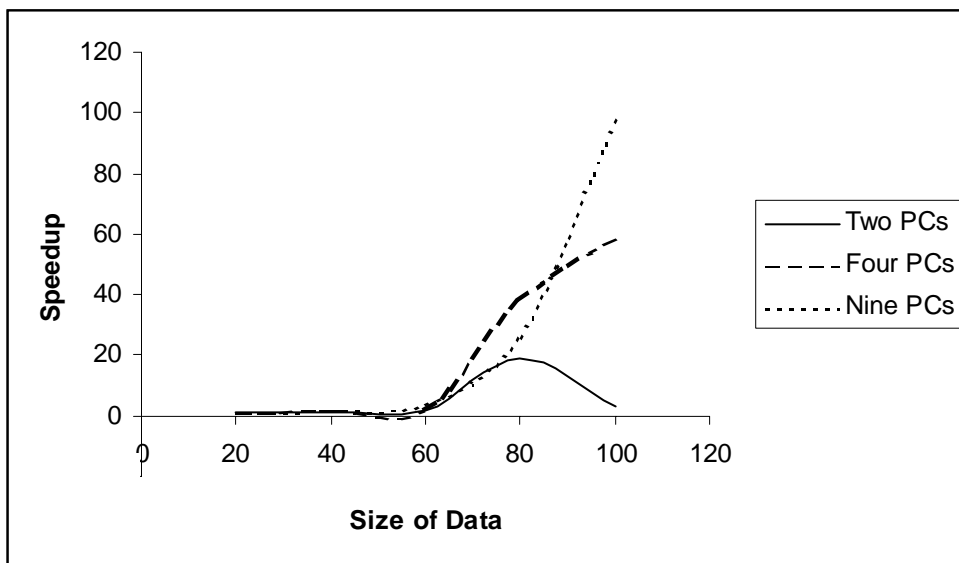


Figure 37: The speedup against the size of data and  $P=4$ .

Figure 38 shows how the efficiency changes with the size of data, while the change of the number of computers, for 4 facilities.

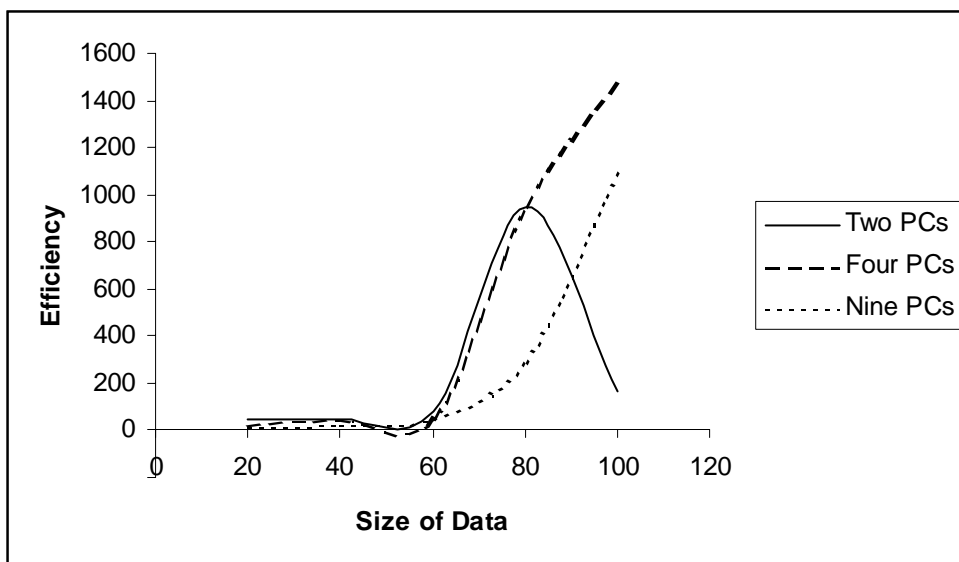


Figure 38: The efficiency versus the size of data and  $P=4$ .

Table 21: The times, speedup, and efficiencies for one, two, four, and nine computers where P=5.

Number of Nodes	Time Needed in One PC	Time Needed in Two PCs	Time Needed in Four PCs	Time Needed in Nine PCs	S(Speed Up) for Two PCs	S(Speed Up) for Four PCs	S(Speed Up) for Nine PCs	E(efficiency) for Two PCs	E(efficiency) for Four PCs	E(efficiency) for Nine PCs
20	1	1	1	1	1.00	1.00	1.00	50.00	25.00	11.11
40	2	2	2	1	1.00	1.00	2.00	50.00	25.00	22.22
60	42	3	2	2	<b>14.00</b>	<b>21.00</b>	<b>21.00</b>	<b>700.00</b>	<b>525.00</b>	<b>233.33</b>
80	265	62	4	3	<b>4.27</b>	<b>66.25</b>	<b>88.33</b>	<b>213.71</b>	<b>1656.25</b>	<b>981.48</b>

Figure 39 shows how the speedup changes with the size of data, while the change of the number of computers, where number of facilities is 5.

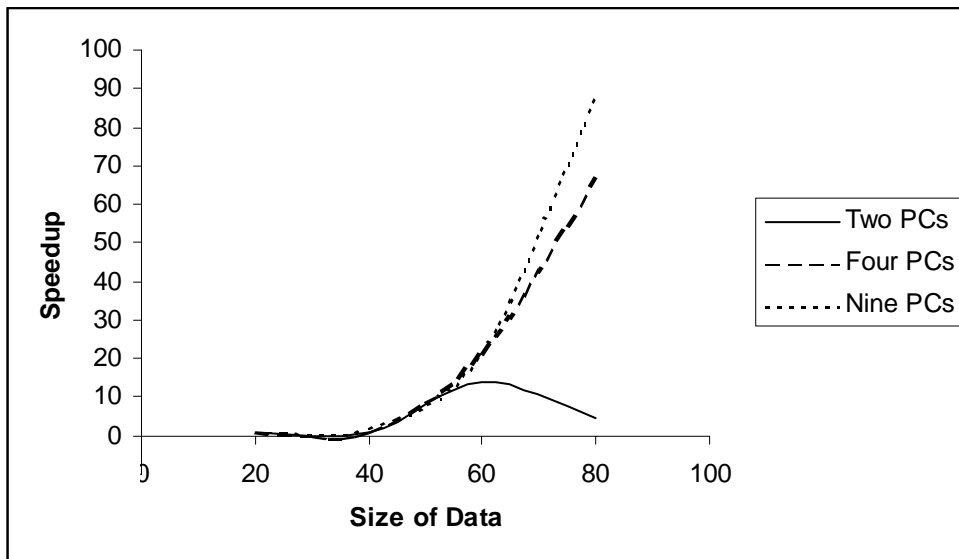


Figure 39: The speedup versus the size of data and  $P=5$ .

Figure 40 shows how the efficiency changes with the size of data, while the change of the number of computers, for 5 facilities.

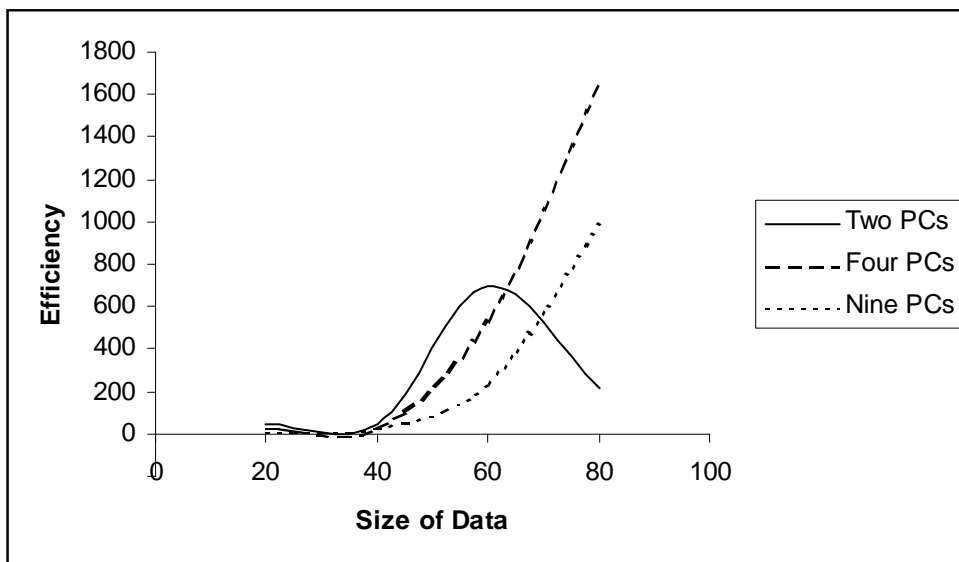


Figure 40: The efficiency versus the size of data and  $P=5$ .

Table 22: Comparing of the times. speedup. and efficiencies for one. tow. four. and nine computers where P=6.

Number of Nodes	Time Needed in One PC	Time Needed in Two PCs	Time Needed in Four PCs	Time Needed in Nine PCs	S(Speed Up) for Two PCs	S(Speed Up) for Four PCs	S(Speed Up) for Nine PCs	E(efficiency) for Two PCs	E(efficiency) for Four PCs	E(efficiency) for Nine PCs
20	1	1	1	1	1.00	1.00	1.00	50.00	25.00	11.11
40	3	2	2	2	1.50	1.50	1.50	75.00	37.50	16.67
60	141	6	3	2	<b>23.50</b>	<b>47.00</b>	<b>70.50</b>	<b>1175.00</b>	<b>1175.00</b>	<b>783.33</b>

Figure 41 shows how the speedup changes with the size of data, while the change of the number of computers, where number of facilities is 6.

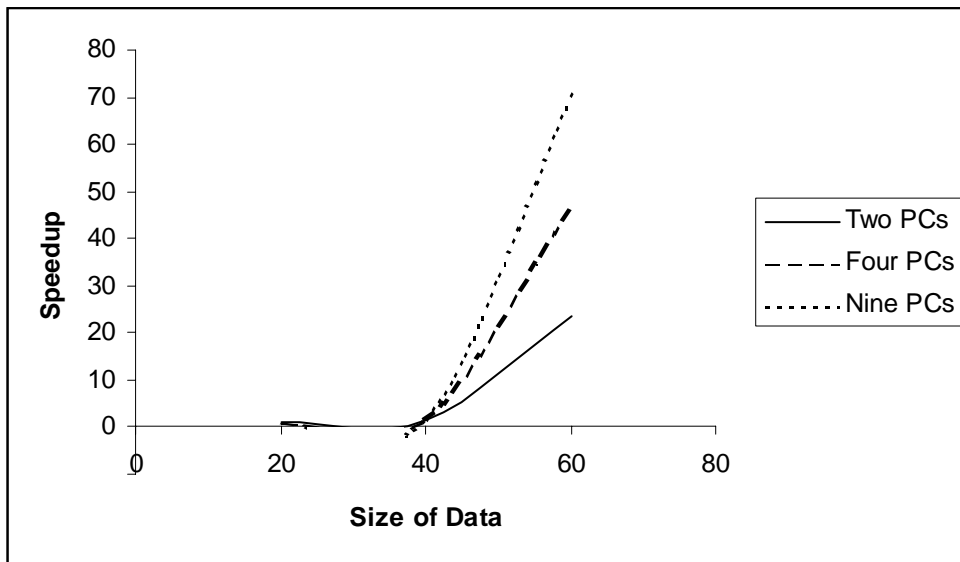


Figure 41: The speedup versus the size of data, P=6.

Figure 42 shows how the efficiency changes with the size of data, while the change of the number of computers, where number of facilities is 5.

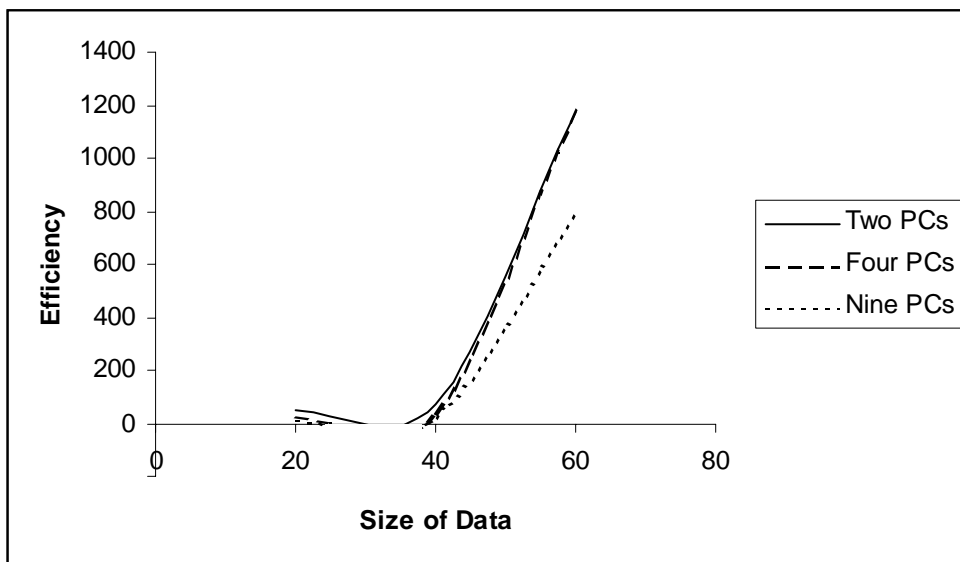


Figure 42: The efficiency versus with the size of data, P=6.

Table 23: Comparing the times, speedup, and efficiencies for one, two, four, and nine computers where P=7,8,9,10.

Number of Facilities = 7										
Number of Nodes	Time Needed in One PC	Time Needed in Two PCs	Time Needed in Four PCs	Time Needed in Nine PCs	S(Speed Up) for Two PCs	S(Speed Up) for Four PCs	S(Speed Up) for Nine PCs	E(efficiency) for Two PCs	E(efficiency) for Four PCs	E(efficiency) for Nine PCs
20	1	1	1	1	1.00	1.00	1.00	50.00	25.00	11.11
40	4	2	2	2	<b>2.00</b>	2.00	2.00	<b>100.00</b>	50.00	22.22
60	340	6	5	2	<b>56.67</b>	<b>68.00</b>	<b>170.00</b>	<b>2833.33</b>	<b>1700.00</b>	<b>1888.89</b>
Number of Facilities = 8										
Number of Nodes	Time Needed in One PC	Time Needed in Two PCs	Time Needed in Four PCs	Time Needed in Nine PCs	S(Speed Up) for Two PCs	S(Speed Up) for Four PCs	S(Speed Up) for Nine PCs	E(efficiency) for Two PCs	E(efficiency) for Four PCs	E(efficiency) for Nine PCs
20	1	1	1	1	1.00	1.00	1.00	50.00	25.00	11.11
40	47	4	2	2	<b>11.75</b>	<b>23.50</b>	<b>23.50</b>	<b>587.50</b>	<b>587.50</b>	<b>261.11</b>
60	612	175	8	3	<b>3.50</b>	<b>76.50</b>	<b>204.00</b>	<b>174.86</b>	<b>1912.50</b>	<b>2266.67</b>
Number of Facilities = 9										
Number of Nodes	Time Needed in One PC	Time Needed in Two PCs	Time Needed in Four PCs	Time Needed in Nine PCs	S(Speed Up) for Two PCs	S(Speed Up) for Four PCs	S(Speed Up) for Nine PCs	E(efficiency) for Two PCs	E(efficiency) for Four PCs	E(efficiency) for Nine PCs
20	2	1	1	1	<b>2.00</b>	2.00	2.00	<b>100.00</b>	50.00	22.22
40	62	6	3	2	<b>10.33</b>	<b>20.67</b>	<b>31.00</b>	<b>516.67</b>	<b>516.67</b>	<b>344.44</b>
Number of Facilities = 10										
Number of Nodes	Time Needed in One PC	Time Needed in Two PCs	Time Needed in Four PCs	Time Needed in Nine PCs	S(Speed Up) for Two PCs	S(Speed Up) for Four PCs	S(Speed Up) for Nine PCs	E(efficiency) for Two PCs	E(efficiency) for Four PCs	E(efficiency) for Nine PCs
20	2	1	1	1	<b>2.00</b>	2.00	2.00	<b>100.00</b>	50.00	22.22
40	118	37	3	2	<b>3.19</b>	<b>39.33</b>	<b>59.00</b>	<b>159.46</b>	<b>983.33</b>	<b>655.56</b>

### 6. Number of facilities is changed on fixed size of data:

Table 4.21 shows the results when we apply the sequential p-median algorithm on one workstation for fixed number of nodes (40) and change the number of facilities. These nodes are given from the Lina image data set (Mladenovic N *et al.*, 1995). The CPU speed was 550MHZ, cache memory was 512KB, and RAM was 128MB. The number of facilities is changed from 2 to 39 points.

Table 24: Time to compute F for p-median problem on one computer.

$P$	T(sec)	F
2	1	149.09
3	2	129.74
4	3	128.05
5	3	123.55
6	4	89.96
7	6	89.96
8	11	81.96
9	26	75.88
10	90	73.41
11	196	66.43
12	232	62.72
13	342	60.35
14	358	59.09
15	495	59.09
16-34	Cannot be calculated	Cannot be calculated
35	665	47.07
36	388	47.07
37	198	47.07
38	65	47.07
39	5	47.07



Figure 43 shows how the time changes during the change of the number of facilities for fixed size of data  $N=40$ .

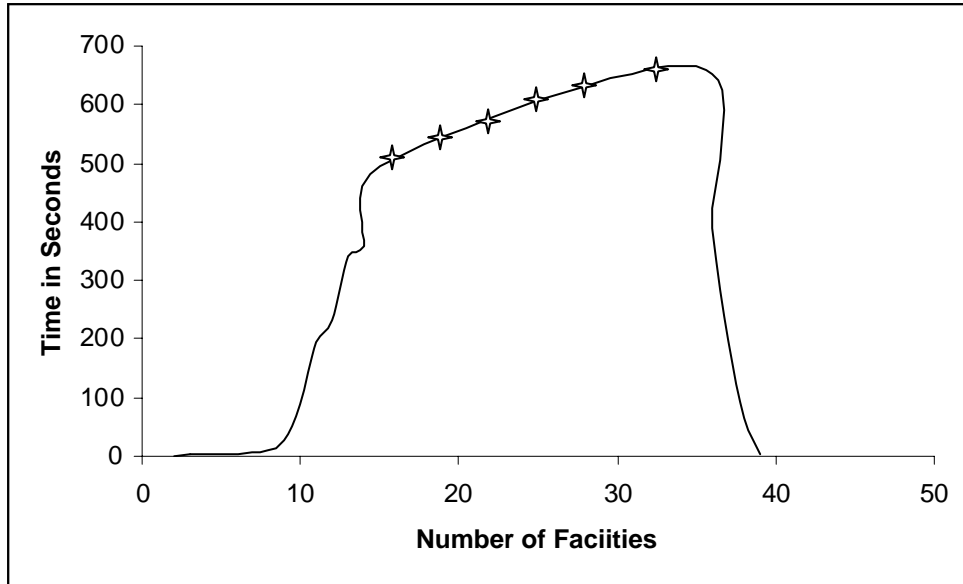


Figure 43: Time versus the number of facilities for one PC

Figure 44 shows how the objective function changes during the changing of the number of facilities for fixed size of data  $N=40$ . The values of  $F$  decrease as the  $P$  increases from 1 to 39.

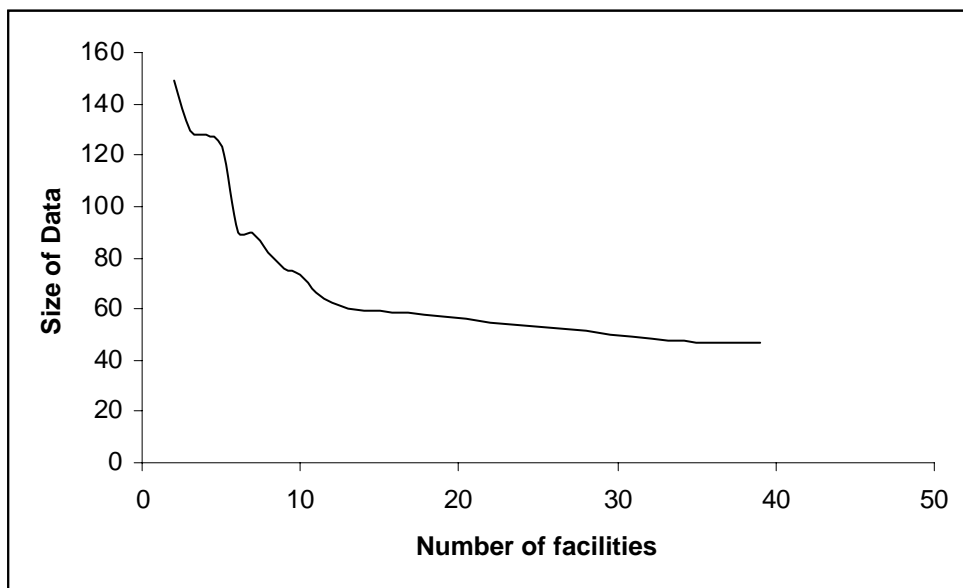


Figure 44:  $F$  versus the number of facilities on one computer.

The parallel p-median algorithm on two computers for fixed number of nodes was tested. The number of facilities was changed. Where, the type of decomposition is vertical. The results are shown in Table 25.

Table 25: Time to compute F for p-median problem on two computers.

P	2	3	4	5	6	7	8	9	10	11	12	13	14	15	Cannot be calculated	35	36	37	38	39
Tsec	1	1	1	2	2	3	4	4	9	17	53	91	125	209	Cannot be calculated	252	177	83	11	3

Figure 45 shows how the time changes during the change of the number of facilities P for fixed size of data N=40.

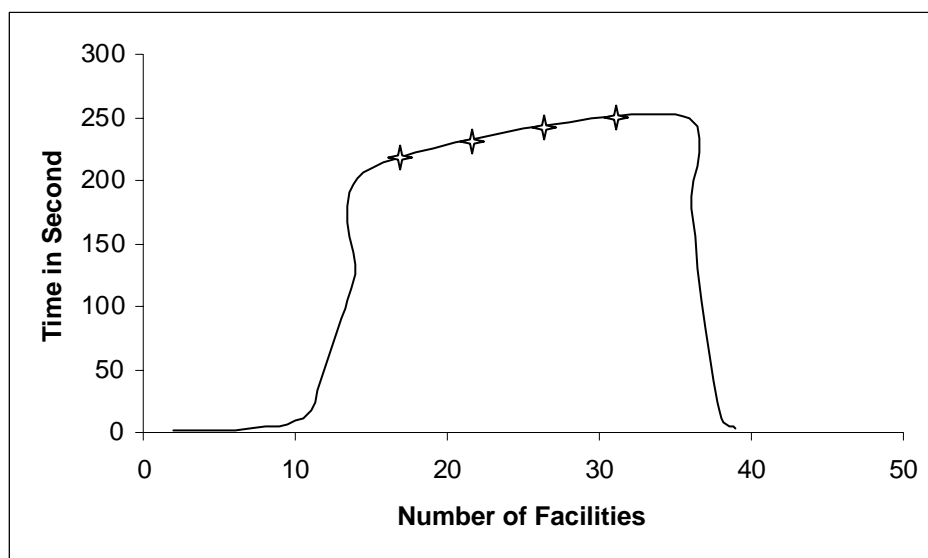


Figure 45: Time versus the number of facilities for 2 computers.

The parallel p-median algorithm on four computers for fixed number of nodes was tested. The number of facilities was changed. The type of decomposition is grid. The results are shown in Table 26.

Table 26: Time to compute F for p-median problem on two computers.

P	2	3	4	5	6	7	8	9	10	11	12	13	14	15	Cannot be calculated	35	36	37	38	39
Tsec	1	1	1	2	2	2	3	4	4	4	5	5	10	8	Cannot be calculated	14	7	4	3	2

Figure 46 shows how the time changes during the change of the number of facilities P for fixed size of data N=40.

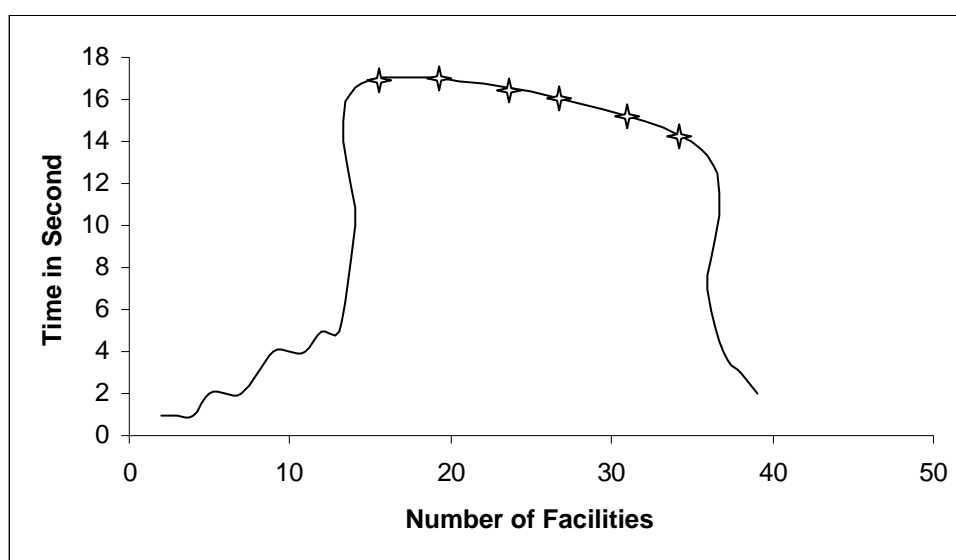


Figure 46: Time versus the number of facilities for 2 computers.

Figure 47 shows how the time decreases, while the number of computers is increased:

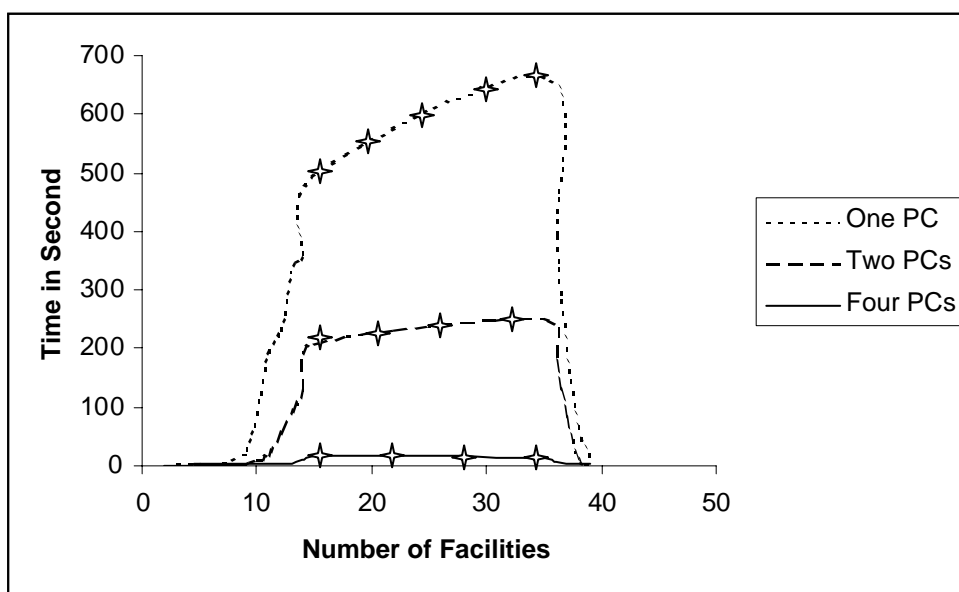


Figure 47: Time for computing F on 1, 2, and 4 computers for  $P=1,2,\dots,39$ .

Table 27: Comparing the times, speedup, and efficiencies for one, two, and four computers.

P	Tsec			Speedup		Efficiency		F
	1 PC	2 PCs	4 PCs	2 PCs	4 PCs	2 PCs	4 PCs	
2	1	1	1	1.00	1.00	50.00	25.00	149.09
3	2	1	1	2.00	2.00	100.00	50.00	129.74
4	3	1	1	<b>3.00</b>	3.00	<b>150.00</b>	75.00	128.05
5	3	2	2	1.50	1.50	75.00	37.50	123.55
6	4	2	2	<b>2.00</b>	2.00	100.00	50.00	89.96
7	6	3	2	<b>2.00</b>	3.00	100.00	75.00	89.96
8	11	4	3	<b>2.75</b>	3.67	<b>137.50</b>	91.67	81.96
9	26	4	4	<b>6.50</b>	<b>6.50</b>	<b>325.00</b>	<b>162.50</b>	75.88
10	90	9	4	<b>10.00</b>	<b>22.50</b>	<b>500.00</b>	<b>562.50</b>	73.41
11	196	17	4	<b>11.53</b>	<b>49.00</b>	<b>576.47</b>	<b>1225.00</b>	66.43
12	232	53	5	<b>4.38</b>	<b>46.40</b>	<b>218.87</b>	<b>1160.00</b>	62.72
13	342	91	5	<b>3.76</b>	<b>68.40</b>	<b>187.91</b>	<b>1710.00</b>	60.35
14	358	125	10	<b>2.86</b>	<b>35.80</b>	<b>143.20</b>	<b>895.00</b>	59.09
15	495	209	17	<b>2.37</b>	<b>29.12</b>	<b>118.42</b>	<b>727.94</b>	59.09
35	665	252	14	<b>2.64</b>	<b>47.50</b>	<b>131.94</b>	<b>1187.50</b>	47.07
36	388	177	7	<b>2.19</b>	<b>55.43</b>	<b>109.60</b>	<b>1385.71</b>	47.07
37	198	83	4	<b>2.39</b>	<b>49.50</b>	<b>119.28</b>	<b>1237.50</b>	47.07
38	65	11	3	<b>5.91</b>	<b>21.67</b>	<b>295.45</b>	<b>541.67</b>	47.07
39	5	3	2	1.67	2.50	83.33	62.50	47.07

Figure 48 shows how the speedup changes with the number of facilities while the change of the number of computers and size of data is 40.

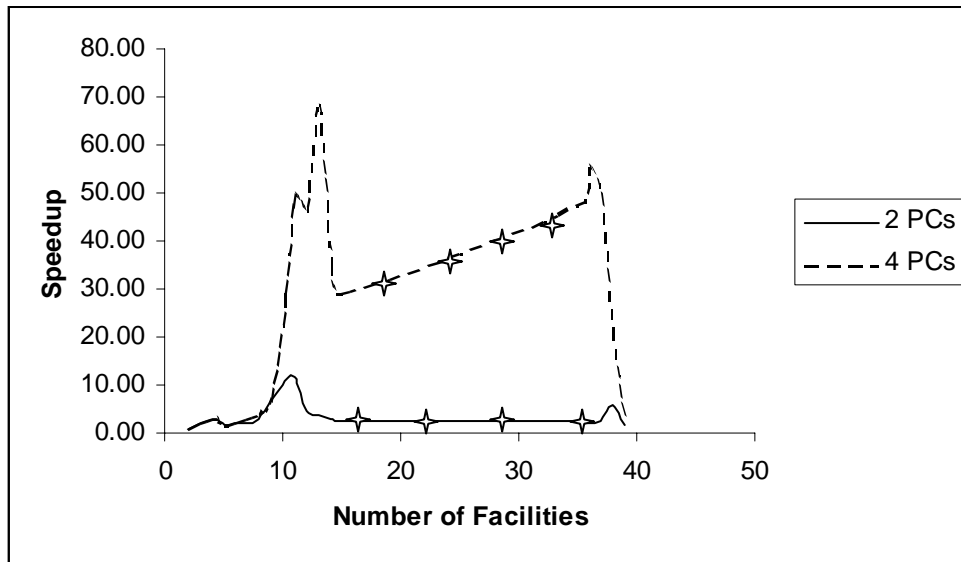


Figure 48: The speedup versus the number of facilities and  $N=40$ .

Figure 49 shows how the efficiency changes with the number of facilities, while changing the number of computers, and size of data is 40.

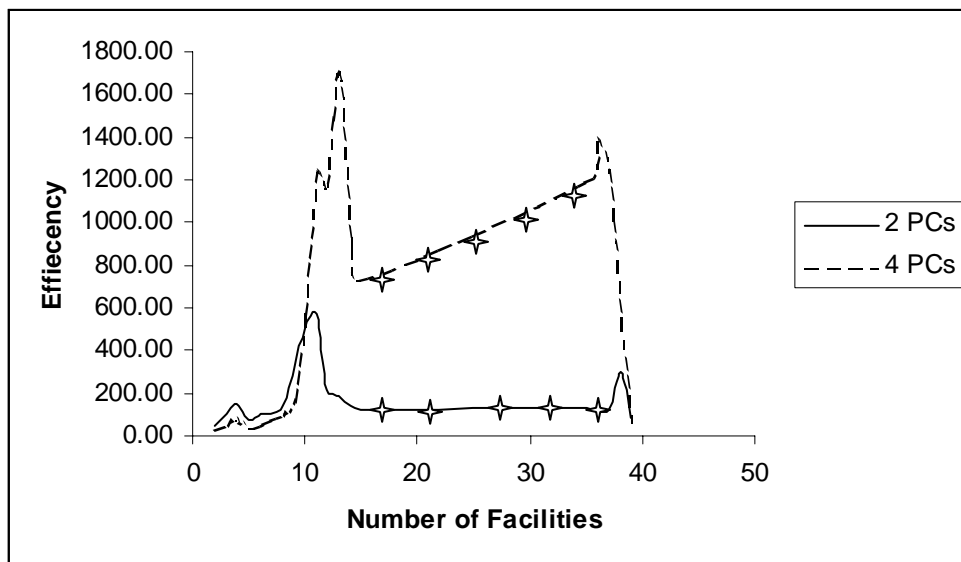


Figure 49: The efficiency versus the number of facilities for  $N=40$ .

## 7. Applying the sequential p-median algorithm and parallel p-median algorithm on the Baboon image data set:

Digital Image Processing (DIP) is concerned with the manipulation and analysis of images discretised from continuous signals. The main goals of classical image processing are (1) image enhancement, which improves the appearance and/or highlights certain details of an image, (2) image segmentation, which is the categorization and/or classification of elements/ structures within an image and (3) image manipulation, which is a general term that refers to the geometric alteration of an image, including translation, scaling and sheering (The Importance of Phase in Image Processing).

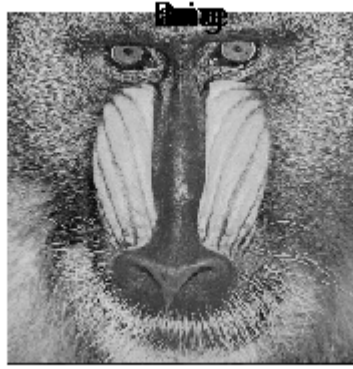


Figure 50: The Baboon image.

This study will focus on image enhancement where the type of decomposition in the parallel algorithm is horizontal, the size of data is fixed  $N=30$ , and number of facilities  $P$  is change from 2 to 29. The results are shown in the following tables.

### 7.1 Apply the sequential p-median algorithm:

Table 28: Time to compute F for p-median problem on one computer for the Baboon image.

$P$	Tsec	F
2	1	301,73
3	1	217,17
4	2	205,05
5	2	198,56
6	2	175,66
7	2	161,61
8	3	150,09
9	3	138,82
10	4	125,53
11	4	122,55
12	5	120,79
13	6	117,30
14	7	109,76
15	12	109,16
16	10	103,64
17	10	103,52
18	9	100,83
19	8	99,30
20	7	97,83
21	6	95,42
22	5	94,98
23	5	93,37
24	5	93,37
25	4	91,61
26	4	89,73
27	3	87,26
28	3	86,44
29	2	83,83



Figure 51 shows how the time changes during the change of the number of facilities for fixed size of data  $N=30$ .

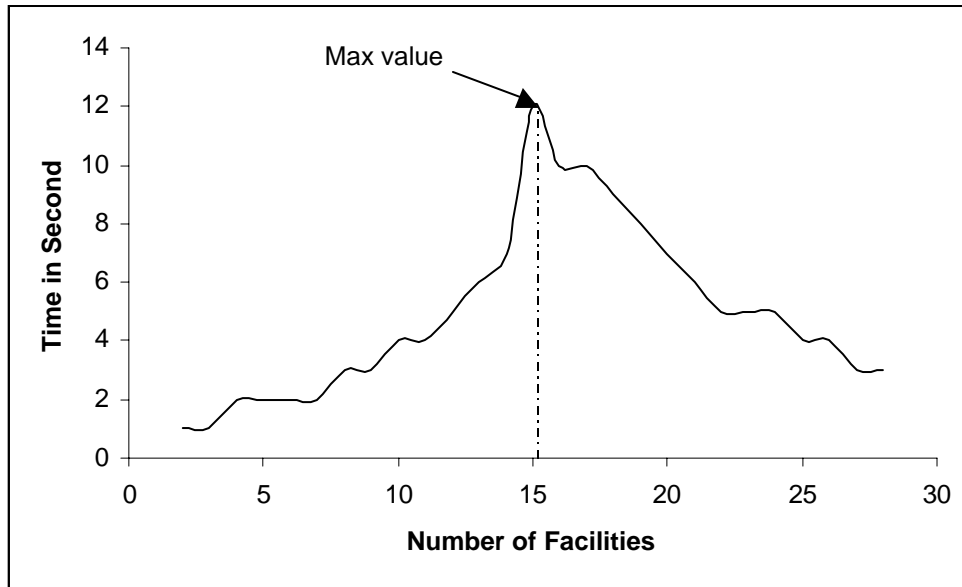


Figure 51: Time versus the number of facilities for one PC.

We found that in the p-median algorithm the time reached maximum value when the number of facilities is  $N/2$ . In conclusion, we must avoid solving the p-median algorithm in the number of facilities equal to the middle of data set.

Figure 52 shows how the objective function changes during the change of the number of facilities for fixed size of data  $N=30$ .

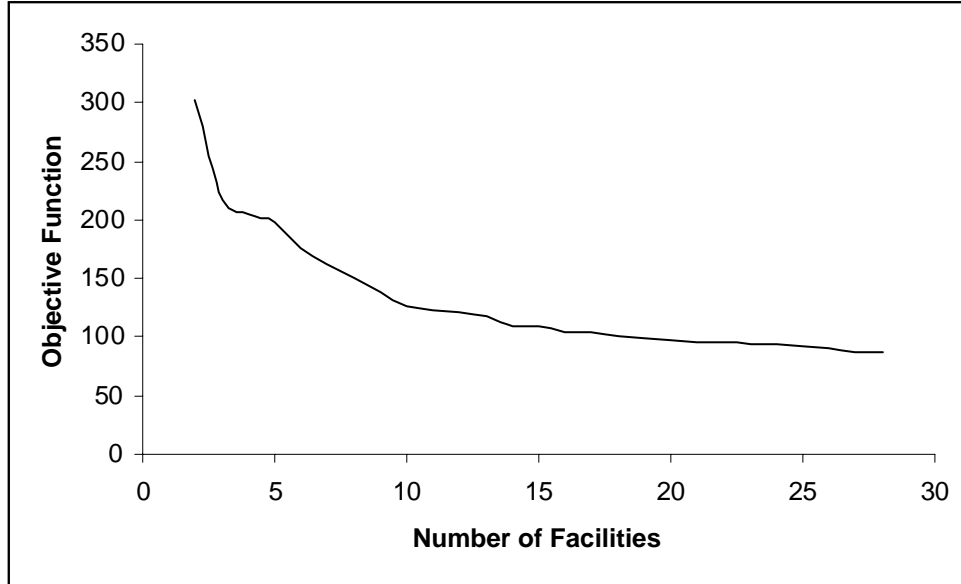


Figure 52: The objective function  $F$  versus the number of facilities  $P$  for one PC and  $N=30$ .

## 7.2 Apply the parallel p-median algorithm:

The parallel p-median algorithm was applied on two computers for fixed number of nodes and different number of facilities. The type of decomposition was horizontal. The results are shown in Table4.26:

Table 29: Time to compute F for p-median problem on two computers.

$P$	$T(sec)$
2	1
3	1
4	1
5	1
6	2
7	2
8	2
9	2
10	3
11	3
12	3
13	3
14	3
15	4
16	4
17	4
18	4
19	4
20	4
21	4
22	4
23	4
24	3
25	3
26	3
27	2
28	2
29	1

Figure 53 shows how the time changes during the change of the number of facilities for fixed size of data  $N=30$ .

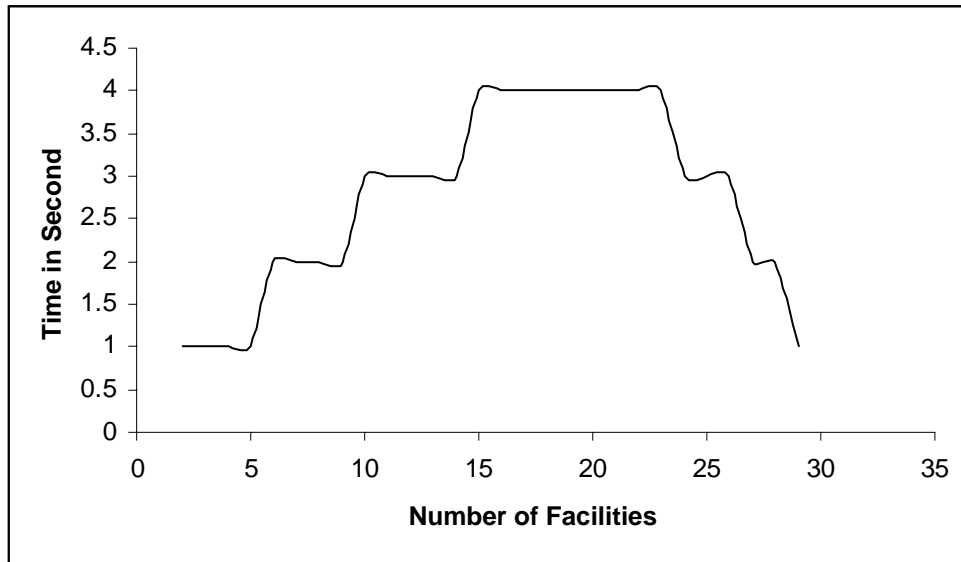


Figure 53: Time versus the number of facilities for two PCs and  $N=30$ .

When using the  $p$ -median algorithm to highlight 400 nodes from 1400 nodes from the Baboon image the concluded image is shown in Figure 54. This is to explain the usability of the  $p$ -median problem on compression images.



Figure 54: The highlighted Baboon image.

### 7.3 Comparison results:

Figure 55 shows how the time decreases while the number of computers increases:

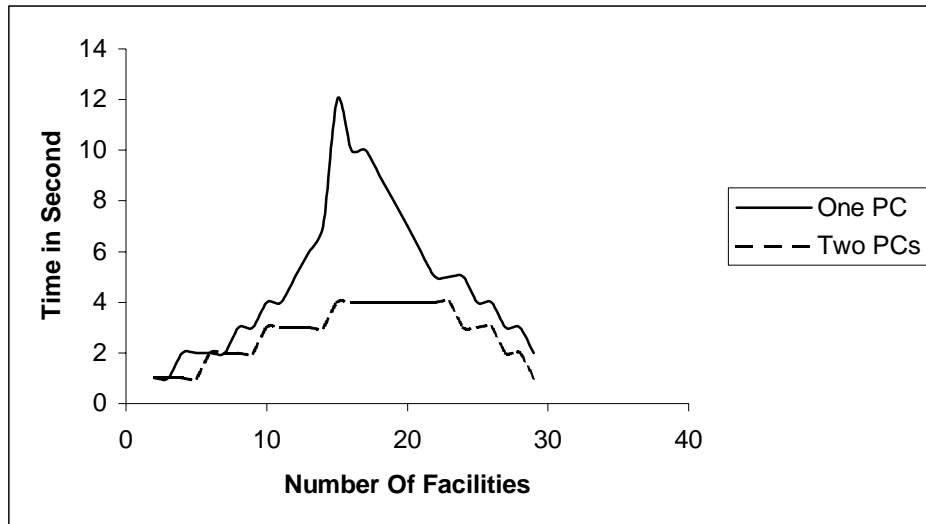


Figure 55: Time versus the number of facilities for one PC and two PCs.

The features of the parallel p-median algorithm can be generated from the above results by collecting all these results in one table for each state, drawing the curves of the speedup and efficiency for each table. All results indicate that the p-median problem can be solved on multiple computers for applications when it is impossible to solve it on one computer.

## 8. Comparison with other works:

Lopez et al. (2004) provide a design and implementation of the Scatter Search metaheuristic parallelized to solve selection problems. They consider the application of Parallel Scatter Search to the p-Median Problem and Feature Subset Selection Problem. The Scatter Search is a population-based metaheuristic that constructs solutions by combining others in a reference set of good solutions. They show several ways of parallelizing the Scatter Search that are applied to the mentioned problems (Lopez F.G *et al.*, 2002, b).

Scatter Search consists of five components processes: Diversification Generation method, that generates a set of divers solutions, Improvement Method, that improves a solution to reach a better solution, Reference Set Update Method, which builds and updates the reference set consisting of RefSetSize good solutions, Subset Generation Method, that combines the solutions in the produced subsets.

There are three parallelizations of Scatter Search to solve the p-Median Problem (Lopez F.G *et al.*, 2002, a). The aim of these strategies was to reduce the running time of the algorithm and increasing the exploration in the solution space.

1. Synchronous Parallel Scatter Search (SPSS): That enables solving, in parallel, the local search.
  1. Replication Combination Scatter Search (RCSS): The procedure is parallelized by selecting several subsets from the reference set that are combined and improved by the computers.
  2. Replication Parallel Scatter Search (RPSS): Consists of multistart search where the local searches are replaced by Scatter Search methods using different populations that run on the parallel computers.

### 8.1 Computational experiments:

The algorithms of the Parallel Scatter Search were coded in C, and tested with large instances of the p-Median Problem. The distance matrix was taken from the instance TPSLIB RL1400 that includes 1400 points (Symmetric traveling salesman problem ).

Table 30: Synchronous Parallel Scatter Search and Grid Decomposition Search

P	1 Computer				2 Computers				4 Computers				8 Computers			
	SPSS		Grid		SPSS		Grid		SPSS		Grid		SPSS		Grid	
	F	T	F	T	F	T	F	T	F	T	F	T	F	T	F	T
40	35002.02	1858	-	-	35002.02	1047	34835.24	410	35002.02	461	34835.24	207	35002.02	283	34835.24	110
50	29089.71	1443	-	-	29089.71	847	28911.20	1185	29089.71	612	28911.20	352	29089.71	311	28911.20	210
60	25185.79	1930	-	-	25185.79	1119	24735.11	3210	25186.24	634	24735.11	377	25167.84	628	24735.11	231
70	22125.46	1864	-	-	22125.46	1088	21816.01	3721	22125.46	649	21816.01	389	22125.46	416	21816.01	238
80	19884.51	1794	-	-	19884.51	1049	19112.12	4211	19870.51	1085	19112.12	418	19870.51	471	19112.12	257
90	17987.91	4168	-	-	17987.91	2322	17245.47	4766	18006.23	780	17245.47	614	18002.35	488	17245.47	316
100	16563.93	3341	-	-	16563.93	1912	16254.78	4928	16551.68	1266	16254.78	1112	16554.08	520	16254.78	334

Table 31: Replication Combination Scatter Search and Grid Decomposition Search

P	1 Computer				2 Computers				4 Computers				8 Computers			
	RCSS		Grid		RCSS		Grid		RCSS		Grid		RCSS		Grid	
	F	T	F	T	F	T	F	T	F	T	F	T	F	T	F	T
40	35002.02	1858	-	-	35002.02	1173	34835.24	410	35002.02	578	34835.24	207	35002.02	557	34835.24	110
50	29089.71	1443	-	-	29089.71	965	28911.20	1185	29089.71	953	28911.20	352	29089.71	410	28911.20	210
60	25185.79	1930	-	-	25185.79	1240	24735.11	3210	25186.24	760	24735.11	377	25167.84	433	24735.11	231
70	22125.46	1864	-	-	22125.46	1263	21816.01	3721	22125.46	828	21816.01	389	22125.46	486	21816.01	238
80	19884.51	1794	-	-	19884.51	1214	19112.12	4211	19870.51	1015	19112.12	418	19870.51	519	19112.12	257
90	17987.91	4168	-	-	17987.91	2871	17245.47	4766	18006.23	1389	17245.47	614	18002.35	537	17245.47	316
100	16563.93	3341	-	-	16563.93	12285	16254.78	4928	16551.68	1113	16254.78	1112	16554.08	827	16254.78	334



Table 32: Replication Parallel Scatter Search and Grid Decomposition Search

P	2 Computers				4 Computers				8 Computers			
	RPSS		Grid		RPSS		Grid		RPSS		Grid	
	F	T	F	T	F	T	F	T	F	T	F	T
40	35002.02	2048	34835.24	410	35002.02	2058	34835.24	207	35002.02	1750	34835.24	110
50	29089.71	1667	28911.20	1185	29089.71	1672	28911.20	352	29089.71	1605	28911.20	210
60	25185.79	2067	24735.11	3210	25186.24	2124	24735.11	377	25167.84	2382	24735.11	231
70	22125.46	2035	21816.01	3721	22125.46	2044	21816.01	389	22125.46	3040	21816.01	238
80	19884.51	3557	19112.12	4211	19870.51	3591	19112.12	418	19870.51	2796	19112.12	257
90	17987.91	22359	17245.47	4766	18006.23	4730	17245.47	614	18002.35	12742	17245.47	316
100	16563.93	3477	16254.78	4928	16551.68	4076	16254.78	1112	16554.08	3838	16254.78	334

The three methods of the Parallel Scatter Search are metaheuristic search. While, the presented methods are exact search. Consequently, the presented method gives more precise solution than the scatter search. See the results in tables 4.27, 4.28, 4.29.

For the sequential search, the Scatter Search needs running time smaller than the presented method. For the parallel search, the presented methods give in general a better running time this is expected because we have a better hardware.

The three methods of the Scatter Search are parallel task decomposition methods, while the presented methods are parallel data decomposition methods. The values of F are very close, which assumes that our parallel algorithm is working correctly.

## CONCLUSIONS AND RECOMENDATIONS

### 1. Summary:

In this work, the p-median problem is investigated. A literature review for previous work is presented in Chapter2. Two algorithms to solve the problem are presented in Chapter3. The first algorithm is a sequential one. The second one is parallel one.

The parallel p-median algorithm is the major achievement of this research. The algorithm was studied on horizontal, vertical, and grid data decomposition. The values of the objective functions generated by the sequential algorithm matched with those produced by the parallel one. Detailed results are presented in Chapter4. These show the values of the objective functions for different values for (1) the number of facilities, (2) the number of computers, and (3) the sizes of applications.

The data set were: some generated randomly and three others were: Baboon image, Lina image, and Maximum Covering Location Problem. The speedup and efficiency were computed for the tested data samples.

### 2. Conclusions:

In the conclusion of this work, we find that:

1. By applying the parallel p-median algorithm we can solve the problem of huge size of data with acceptable execution time.
2. Applying the parallel p-median algorithm, we have a speedup greater than the real time of number of computers.
3. We can apply the parallel p-median algorithm on one computer more than one time. Each time we apply the algorithm on part of the data set. The total time for solving p-median problem in parallel is smaller than on sequential p-median algorithm.

4. The better results can be reached at number of facilities more than 4, and size of data more than 100. Then apply the parallel p-median algorithm at greater these numbers.
5. All the results can be enhanced by increasing the characteristics of computers.

### 3. Future research:

Thus, we recommend the following for future research:

1. Avoiding using the parallel p-median algorithm when  $p = N / 2$ , where  $p$  is the number of facilities, and  $N$  is the size of data set.
2. Apply the parallel p-median algorithm on a network of other operating systems besides windows 2000 professional operating system, TCP/IP network protocol, and UTP network cables.
3. Apply the parallel p-median algorithm on other network topologies.

## REFERENCES

- Al-Zoubi, M.B, Sharieh, A, El-Hanbali, N, and Al-Dahoud, A. (submitted Feb 2005). **A Heuristic Algorithm for Solving the P-Median Problem**. Department of computer science, University of Jordan, Amman-Jordan.
- Berman, O, Chiu, S.S, Larson, R.C, Odoni, A.R, and Batta, R. (1990). **Location of Mobile Units in a Stochastic Environment**. In *Discrete Location Theory*, pages 503-549. R.L. Francis and P.B. Mirchandani, Editors, Wiley-Interscience, New York, NY.
- Charikar, M, Guha, S, Tardos, E, and shmoys, D. B. (March 2005). **A Constant-Factor Approximation Algorithm for the p-Median Problem**. *Journal of Heuristics* 6:332-341.
- Chajed, D, and Lowe, T.J. (1992). **M-median and M-center Problems with Mutual Communication: Solvable Special Cases**. *Operations Research*, Vol. 40 Supp. 1:S57-S66.
- Correa, E.S, Steiner, M.T.A, Freitas, A.A, and Carnieri, C. (2001). **A Genetic Algorithm for the P-Median Problem**. *SIAM Journal of Applied Mathematics*, 37(3):539{560, 1979.
- Hansen, P, and Mladenovie', N. (1992). **Heuristic Solution of the Multisource Weber Problem as a p-Median Problem**. GERARD Research Report G-92-35, University of Montreal, Canada. *Operation Research*, 11,301-343.
- Hansen, P, and Mladenovie', N. (1997). **Variable Neighborhood Search for the p-median**, *Location Sci.* 5: 207-226.

Hansen, P, and Mladenovie, N. (1999). **An introduction to Variable Neighborhood Search**. In S. Voss *et al.* (eds.). *Metaheuristics, advances and trends in local search paradigms for optimization*, pp. 433-458, Kluwer Academic Publishers Dordrecht.

Lopez, F.G, Torres, M.G, Batista, B.M, Perez, J.A.M, and Vega, J.M.M. (2004).

**Parallelizations of Scatter Search for Selection Problems**. Partially supported by the project TIC2002-04242-C03-01 (70% of which are FEDER funds) and TIC2002-10886E.

Lopez, F.G, Torres, M.G, Batista, B.M, Perez, J.A.M, and Vega, J.M.M. (2002). **The Parallel Variable Neighborhood Search for the  $p$ -Median Problem**. *Journal of Heuristics* 8: 375-388.

Markatos, E. P. (1996). Using **Remote Memory to avoid Disk Thrashing**.

A Simulation Study. In Proceedings of the MASCOTS'96 , San Jose, CA, USA.

Mladenovie, N, Labbe, M, and Hansen, P. (1995). **Solving the  $p$ -center Problem with Tabu Search and Variable Neighborhood Search**. Les Cahiers du GERARD, G-95-38, Montreal.

Resende, M.G.C, and Werneck, R.F. (2002). **On the Implementation of a Swap-Based Local Search Procedure for the  $p$ -Median Problem**. Department of Computer Science, Princeton University, Technical Report TD-5E4QKA.

Scaparra, M.P, and Scutella, M.G. (2001). **Facilities, Location, Customers: Building Blocks of Location Models**. Technical Report: TR-01-18, University of Pisa, Department of Information.

Shmoys D.B, Tardos E, and Aardal K. (1997). **Approximation Algorithms for Facility Location Problems**. School of Operations Research & Industrial

Engineering and Department of Computer Science, Cornell University, Ithaca, NY 14853.

Silberschatz, A, Galvin, P.B, and Gagne, G. (2002). **Operating System Concepts**. pp. 317-344, John Wiley & Sons Publishers.

Suzuki, A, and Okabe, A. (1995). **Using Voronoi Diagrams**. In Z. Drezner, editor, *Facility Location. A Survey of Applications and Methods*, Chapter 6. Springer-Verlag, New York, Inc.

### **Computer Science 150/Core 142 Contemporary Issues in Computer Science**

Retrieved August 3, 2005, from

<http://cs.colgate.edu/faculty/brackett/C150Fa98/Notes150/algorithms2.htm>.

**The Importance of Phase in Image Processing** Retrieved August 3, 2005, from

[http://www.personal.psu.edu/users/m/d/mde11/importance\\_of\\_phase\\_in\\_image\\_processing.htm](http://www.personal.psu.edu/users/m/d/mde11/importance_of_phase_in_image_processing.htm).

**Image Characteristics Retrieved** August 3, 2005, from

<http://cnx.rice.edu/content/m11085/latest/>.

**Symmetric traveling salesman problem (TSP)** Retrieved August 3, from

<http://elib.zib.de/pub/Packages/mp-testdata/tsp/tsplib/tsplib.html>.

## حل مشكلة ب-الوسيط بالتجزئة المتوازية للبيانات

إعداد

صبحي إبراهيم الشبيخة

المشرف الرئيسي

الدكتور أحمد الشرايعة

المشرف المشارك

الدكتور محمد بلال الزعبي

### الملخص

حل مشكلة ب-الوسيط باستخدام حاسوب واحد (تسلسليا) مكلف بالزمن والمساحة التخزينية وغير قابل للحل للتطبيقات ذات الحجم الكبير. إن الخوارزمية التسلسلية تبدو غير فعالة لحل تطبيقات ذات حجم كبير. فهذه الأطروحة تقدم بديل لحل مشكلة ب-الوسيط باستخدام الحواسيب المتعددة.

عرضت هذه الدراسة طريقة جديدة لحل مشكلة ب-الوسيط تعتمد على تقسيم مجموعة البيانات للمسألة إلى أجزاء صغيرة: عموديا أو أفقيا أو شبكيا. وهذه الأجزاء توزع على عدد من الحواسيب التي تملك نفس المواصفات وتعمل بتزامن بحيث أن كل حاسوب يحل مشكلة ب-الوسيط ولنفس العدد الكلي لاماكن التوضع ولكن على جزء البيانات الخاص به.

و الخوارزمية المتوازية قد طبقت وجربت على مجموعة البيانات وأجهزة الحاسوب المتوفرة. وبيانات الدراسة تقسم إلى: القسم الأول منتج بشكل عشوائي وبأحجام مختلفة، و القسم الثاني أخذ من تطبيقات حقيقية وبعض هذه التطبيقات هي: مسألة توضع التغطية الشاملة و صورة بابون وصورة لينا. والمتغير التابع والمطلوب حسابه هو (ف) لإيجاد ب وسيلة وتحديد موقعها. وحسب كذلك زمن التشغيل و السرعة والفعالية. و العوامل الداخلية (المتغيرات المستقلة) هي عدد الحواسيب وحجم مجموعة البيانات وعدد الوسائل ب. والمتغيرات الوسيطة هي حجم الذاكرة لأجهزة الحاسوب وسرعة وحدة المعالجة المركزية للأجهزة المستخدمة.

ولقد وجد أن حاسوبا واحدا غير قادر على إكمال العمليات الحسابية عندما يكون حجم البيانات أكثر من ٣٠٠ نقطة من أجل توضع وسيلة واحدة و ٨٠ نقطة من أجل توضع ٥ وسائل و ٤٠ نقطة من أجل توضع ٩ وسائل. وعند تطبيق خوارزمية ب-الوسيط على أكثر من حاسوب تمكنا من زيادة حجم مجموعة البيانات وتم الحصول على زمن تشغيل أفضل من حاسوب واحد.



فعلى سبيل المثال: عندما كان عدد الحواسيب يساوي ٢ تمكنا من تطبيق الخوارزمية على بيانات بحجم يساوي ٥٢٠ نقطة حيث عدد الوسائل المطلوب توضعها يساوي ١، و ١٠٠ نقطة من أجل ٥ وسائل و ٦٠ نقطة من أجل ٩ و ١٠ وسائل. و استخدام ٩ حواسيب تمكنا من تطبيق الخوارزمية على ٩٣٥ نقطة حيث عدد الوسائل يساوي ١، و ١٢٠ نقطة من أجل ٧ و ٨ و ٩ وسائل، و ١٠٠ نقطة من أجل ١٠ وسائل.