分散式計算網路中應用程式之資源配置

Resource Allocation for the Applications in Distributed Computing Networks

黃胤傅
Yin-Fu Huang
Department of Electronic Engineering
Yunlin University of Science and Technolog
huangyf@el.yuntech.edu.tw

趙柏偉
Bo-Wei Chao
Department of Computer Science
Tsing Hua University
dr888312@cs.nthu.edu.tw

Abstract

In a distributed computing network, various types of resources attached at nodes, such as disks, main memory, resident p ograms, and printers, could be shared among applications. To promote the resource utilization, an efficient resource allocation algorithm for application requests should be developed. In this paper, our system model considers the factors such as the computation power of nodes, the upper limit of shared programs, th transmission bandwidth of peripherals, the channel bandwidth, and the quality of service (QoS) requirement of applications. Based on it, we propose three resource allocation algorithms used in d ifferent situations, respectively. The simulation results show that our algorithms have a near optimal solution.

Keywords: Distributed computing networks, resource allocation, network bandwidth, QoS, resource utilization

1 Introduction

In recent years, computer networks are quickly expanding and prevalent so that a large amount of applications has been developed to solve different problems. How to make use of the finite resources efficiently in a network is a critical topic. Furthermore network bandwidth could be also viewed as a valuable resource when a remote access is required in an application. That is because when available network bandwidth is not enough to use, data transmission will become very slow. This phenomenon occurs frequently in a multimedia system. To reserve network bandwidth for more applications, QoS concepts or problems have been widely discussed and explored as a means to trade application quality for application availability in case of system resource contention [4]. To support such an application need, there has been a large body of researches on QoS-based system resource management [6], such as RSVP - a resource reservation protocol for communication networks [9], processor reservation [8], and local multi-resource reservation and adaptation [5].

In the past, researches addressing the resource allocation problems in distributed computing networks focus on dispatching resources among the nodes in a network [2, 3], connecting the nodes [1], finding the Minimum Spanning Tree on a connected graph to reduce the overhead of data transmission in a network [10], and allocatin communication channels [7] in order t o fit the requirements of network reliability In our system model, the resources are assumed to have been dispatched among the nod es, and we try to allocate the resources as possible as we can in order to increase the number of concurrent applications and/or reduce the execution time of all applications. In a word, the problems addressed here are totally different from those of the literatures published before.

To reflect the real world, the factors involved in our syste

model are considered as sophisticated as possible, includin the computation power of nodes, the upper limit of sharable programs, the transmission bandwidth of peri pherals, the channel bandwidth, and the QoS requirement of applications. Here we consider three problems that will be solved using proposed resource allocation algorithms. The are 1) with the limitation of network bandwidth, 2) with dynamic environment, and 3) with the specified QoS of applications.

In the remainder of the paper, it is organized as follows. In Section 2, we propose our system model, in which system considerations and problem definitions are described. In Section 3, an objective function to calculate the cost of an application is defined, and three resource allocation algorithms are developed to solve the problems. Next a simulation model is proposed to evaluate these algorithms in Section 4. Finally, we make conclusions in Section 5.

2 System Model

2.1 System Considerations

The premise for our model is that the distributed computin network is a connected graph; that is no partition is permitted in the network. Regarding the other considerations, we outline them as follows

1) Nodes:

Each node in the network is at least equipped with hard disks and main memory, and the other resources such as resident programs and printers are optional. The sizes and/or the numbers of the attached resources could be different among the nodes. Besides, to reflect the real world the computation power of the nodes in our model could be different. Here three levels of computation power are defined.

2) Resources:

The resources attached to the nodes could be versatile. However we only consider four different types of resources in our model; that is hard disks, main memory, residents programs, and printers. This consideration has no effect on the spirit of our method if other different types of resources are considered in the model. As for the characteristic of these four types of resources, we explain as follows

a) Hard Disks:

A hard disk is sharable in essence. An application can request a local/remote disk for reading and writing. To measure the transmission time, an application must specify the required data and/or disk size.

b) Main Memory:

Main memory is another sharable resource. When an application request a local/remote resident program runnin for computation, main memory should be required to download the program from disks. The size of required memory is the same as that of the program.

c) Resident Programs:

Resident programs are also sharable among the applications In other words, different applications can request the same program running concurrently at one node. However each node has an upper limit of sharability for each resident program. Here we do not exclude that the same program can be resident in the different nodes. Besides each resident program has its total execution time.

d) Printers:

Printers are not sharable resources that serve only one application at a time. To measure the transmission time, an application must specify the output data size if it has data to be printed.

3) Transmission Bandwidth of Peripherals:

For the peripherals attached to the nodes, their transmission bandwidth could be different. For hard disks, we consider several transmission bandwidths in our model, such as IDE (1MB/s), EIDE (5, 10MB/s), U-DMA33 (33MB/s) for internal disks, and SCSI (2MB/s), SCSI-2 (12MB/s), U-SCSI-3 (48MB/s) for external disks. For printers, only one transmission bandwidth is considered; that is the parallel port (1, 2Mb/s).

4) Channel bandwidth:

The c hannels between nodes are assumed to be point-to-point and bi-directional. Their individual

bandwidths could be T1 (1.544Mb/s), T3 (45Mb/s), and ATM (4 , 100, 155, and 622Mb/s). To transmit data smoothly, all channels of a path from the issuing node to the target node with the required resource are assumed t have the same allocated transmission bandwidth; that is the overall network bandwidth of a path is the minimum available bandwidth of all allocated channel bandwidths. Then no buffers for the flow control are required in our model.

5) Network bandwidth requirement of applications:

To match the transmission bandwidth of remote peripherals that applications request, the network bandwidth requirement of the applications is assumed to be the range from 50% to 100% of the transmission bandwidth of the peripherals. It means that when we allocate network bandwidth for applications, the required network bandwidth is exactly the transmission bandwidth of the peripherals. However if the available network bandwidth is not sufficient for the requirement, at least half of the required network bandwidth should be met.

6) QoS requirement of applications:

Sometimes when the network bandwidth is not quite satisfactory to an application, the application is still able t tolerate the worse QoS if it has not strong requirement for the network bandwidth. This phenomenon occurs frequently in a multimedia system. In our model, we will consider the applications with different QoS.

2.2 Problem Definitions

Based on the system model proposed above, a distributed computing network where the channel bandwidth between nodes has been assigned is given. For the nodes in the network, there exists a resource table describing the resources information. When a node called **resource allocation node** receives the applications from users, we will explore how to allocate local and/or remote resources to these applications efficiently according to the requirements of the applications. Herein we have several problems to be addressed as follows

1) With the limitation of network bandwidth:

With the limitation of network bandwidth, we expect to find an efficient resource allocation that has the most concurrent applications running at the same time.

2) With dynamic environment:

This problem is associated with the execution time of applications. When the resources have been allocated t some applications in the first round as many as we can, the rest applications will wait for the completion of these applications. As long as an application finishes and releases its resources, the allocation mechanism will be triggered again in the next round till all applications acquire the resources. Here we expect to find an efficient resource allocation that has the shortest completion time of all applications.

3) With the specified QoS of applications:

With the specified QoS of applications, we also expect t find an efficient resource allocation as problem 1 and problem 2 if the requirement of the network bandwidth is not insisted in an application. In a multimedia system, sacrificing the QoS of individual applications and then increasing the concurrent degree of applications are usuall the pursued topic.

3 Resource Allocation Algorithms

3.1 The Objective Function

The concepts behind the objective function are that the application with the least ratio of required resources over available resources in the system should be scheduled first to allocate resources. Only done in this way, the goal t achieve the most concurrent applications running at the same time will probably be reached. In essence, we try to utilize the system resources as possible as we can. Before the objective function is formalized, the notations for the terms used in the function will be given. These terms have been mentioned in Section 2.1.

The notations used in the objective function are explained as follows

N set of the nodes in the distributed computing network

P set of the resident programs in the distributed computing network

A set of the applications in the distributed computing network

d_a the required data and disk size of application a for reading and writing

D_n the available disk space at node n

$r_{a,p}$ the memory size of program p requested b application a

$R_{n,p}$ the available memory space at node n storing program p

$prg_{a,p}$ the number of program p requested by application a

PRG_p the available number of sharability for program p in the system

prn_a the number of printers requested by application a

PRN the available number of printers in the system

The objective function is composed of four parts; that is 1) the cost for hard disks, 2) the cost for main memory, 3) the cost for resident programs, and 4) the cost for printers. Here we define these costs as follows

1) For hard disks

$$Z_1 = \sum_n \frac{d_a}{D_n} \quad \text{where } a \in A \ \& \ n \in N$$

2) For main memory:

$$Z_2 = \sum_p \sum_n \frac{r_{a,p}}{R_{n,p}}$$

where $a \in A \ \& \ n \in N \ \& \ p \in P$

3) For resident programs

$$Z_3 = \sum_p \frac{prg_{a,p}}{PRG_p} \quad \text{where } a \in A \ \& \ p \in P$$

4) For printers:

$$Z_4 = \frac{prn_a}{PRN} \quad \text{where } a \in A$$

After summing up these costs, the objective function can be expressed as $Z = \sum_{i=1}^{4} Zi$ where $a \in A$.

All applications to which resources have not been allocated will be evaluated using the objective function. The application with the least cost will be selected to allocate resources. The evaluation and allocation procedures will kee n running in turn till no sufficient resources can be allocated to an application. Here we take an example showing how to evaluate applications using the objective function.

Example 1: Assumed that there are five nodes in a distributed computing network. he resources installed at the five nodes are shown in Table I. For an application which requires 5MB disk size for writing, a printer for output, and resident program 1 and 3 for running, its cost can be computed using the objective function as follows

$$Z_1 = \frac{5}{40} + \frac{5}{20} + \frac{5}{5} + \frac{5}{10} + \frac{5}{30} \approx 2.042$$

$$Z_2 = \frac{1}{64} + \frac{1}{8} + \frac{0.5}{4} \approx 0.266$$

$$Z_3 = \frac{1}{3+1} + \frac{1}{2} \approx 0.75 \qquad Z_4 = \frac{1}{1+1} = 0.5$$

$$Z = Z_1 + Z_2 + Z_3 + Z_4 \approx 3.558$$

Table I Resource Table

Node id	1	2	3	4	5
Program id & size (MB)	-, -	3, 0.5	1, 1	2, 1.5	1, 1
Sharable no. of programs	-	2	3	4	1
Memory size (MB)	16	4	64	32	8
Printer no.	0	1	0	1	0
Disk space (MB)	40	20	5	10	30

3.2 Resource Allocation Algorithms

For the problems mentioned in Section 2.2, we propose the following algorithms to allocate network resources according to the requests of applications. First these algorithms determine one application with the minimu value each time, based on the objective function discussed in Section 3.1. Next they will try to allocate resources for the picked application. If there are more than one node capable of providing resources to the application, the algorithms select one node with the most benefit. In principle, the most benefit means that residual resources

have the most possibility to provide other unallocated applications, after the resource allocation. The method used here is to find the node whose available resources have the same size as the requested size of the application. If not found, it selects the node with the most available resources, like the worst-fit strategy used in the memory allocation of operating systems. Finally the algorithms determine the routing using the variant Dijkstra's algorithm.

The method to find an appropriate node providin resources is outlined as follows

Node_selection(type, request)

{ switch (type) {

case 1:

Max_size ← 0;

For i = 0 to n

{ If D_i = request

then D_i ← 0; return {i};

else If (D_i > request and D_i > Max_size)

then Max_size ← D_i; Max_node ← i; }

If (Max_size ≠ 0)

then D_{Max_node} ← D_{Max_node} – request; return {Max_node};

else return ∅;

case 2:

For each program p in request, proceed as follows

{ Max_size ← 0;

For i = 0 to n

{ If (program p exists at node i)

then If ($R_{i,p}$ = $r_{a,p}$)

then Max_size ← $R_{i,p}$; Max_node ← i;

break;

else If ($R_{i,p}$ > $r_{a,p}$ and $R_{i,p}$ > Max_size)

then Max_size ← $R_{i,p}$;

Max_node ← i; }

If (Max_size ≠ 0)

then $R_{max\_node,p}$ ← $R_{Max\_node,p}$ – $r_{a,p}$;

insert Max_node into Node_set;

PRG_p ← PRG_p – 1;

else return ∅; }

return Node_set;

case 3:

For i = 0 to n

{ If (a printer is available at node i)

then PRN ← PRN – 1; return {i}; }

return ∅; }}

3.2.1 With the Limitation of Network Bandwidth

For this problem, we expect to find an efficient resource allocation that has the most concurrent applications runnin at the same time.

Resource_Allocation_Algorithm_1( )

{ /* A: a set of applications,

N_d, N_{prg}, and N_{prn} denote the sets of selected nodes for hard disks, programs, and printers requested b application a, respectively.

Alloc_Band(P_i): allocated bandwidth of path P_i

Band(P_i): bandwidth of path P_i obtained using ***variant Dijkstra's algorith***

Band(peri): transmission bandwidth of allocated peripherals */

From A, find an application a with the minimum value, based on **the objective function**;

If d_a ≠ 0

then If (N_d ← **Node_selection**(1, d_a)) = ∅ then stop;

If prg_a ≠ ∅

then If (N_{prg} ← **Node_selection**(2, prg_a)) = ∅

then release all allocated resources of application a

stop;

If prn_a ≠ 0

then If (N_{prn} ← **Node_selection**(3, prn_a)) = ∅

then release all allocated resources of application a;

stop;

Call ***variant Dijkstra's algorith*** ;

If (there exists a required path whose bandwidth is completely consumed)

then release all allocated resources of application a; stop;

else for each required path, proceed as follows

Alloc_Band(P_i) ← Min(Band(P_i), Band(peri));

If (Alloc_Band(P_i) < Band(peri)*50%)

then release all allocated resources of application a;

stop;

$A \leftarrow A - \{a\}$;

Repeat; }

When the total number of nodes is much larger than that of requested programs in an application (i.e. n >> | prg_a|), the time complexity of this algorithm is dominated b *variant Dijkstra's algorithm*, and is $O(|A| \cdot n^2)$; otherwise dominated by procedure **Node_selection**, and is $O(|A| \cdot |prg_a| \cdot n)$.

3.2.2 With Dynamic Environment

For this problem, we expect to find an efficient resource allocation that has the shortest completion time of all applications. When the resources have been allocated t some applications in the first round as ***Algorithm_1***, other unallocated applications will wait for the completion of these applications. Whenever an application finishes and releases its resources, the allocation mechanism will be triggered again in the next round till all applications acquire the resources.

Resource_Allocation_Algorithm_2( )

{ Call ***Resource_Allocation_Algorithm_1( );***

If (there exists an unallocated application)

then find an application a with the minimum execution time, from the allocated applications;

Release all allocated resources of application a;

Repeat;

else stop; }

3.2.3 With the Specified QoS of Applications

In a multimedia system, sacrificing the QoS of individual applications and then increasing the concurrent degree of applications are usually the topic to be pursued. When the available network bandwidth is not sufficient to offer an application, the rest resources attached at nodes will not be able to be used and the overall resource utilization will degrade. For this problem, we expect to find an efficient resource allocation as problem 1 and problem 2 if the Qo requirement of network bandwidth is not insisted in an application. The QoS specification of applications has a close correlation with the maximum concurrent degree and the completion time of all applications. On the whole, the lower the QoS specification is, the more possibility the resources are utilized. Here only some statements in ***Resource_Allocation_Algorithm_1*** are needed to modify to correspond the QoS specification.

Resource_Allocation_Algorithm_3( )

{ /*QoS: the percentage of QoS specified by applications */

: : :

Call ***variant Dijkstra's algorith*** ;

If (there exists a required path whose bandwidth is completely consumed)

then release all allocated resources of application a; stop;

else for each required path, proceed as follows

If (Band(P_i) ≥ Band(peri)*QoS)

then Alloc_Band(P_i) ← Band(peri)*QoS;

else release all allocated resources of application a;

stop;

$A \leftarrow A - \{a\}$;

Repeat; }

4 Performance Evaluation

4.1 Simulation Model

In the simulation model, the distributed computing network is a connected graph. Each node in the network is at least equipped with hard disks and main memory, and the other resources such as resident programs and printers are optional. The sizes and/or the numbers of the attached resources could be different among the nodes. We collect these information in a resource table shown in Table II. For hard disks, the transmission bandwidth could be IDE (1MB/s), EIDE (5, 10MB/s), U-DMA33 (33MB/s), SCSI (2MB/s), SCSI-2 (12MB/s), and U-SCSI-3 (48MB/s). For printers, the transmission bandwidth could be 1 or 2Mb/s. In addition, we also have the channel bandwidth between nodes, shown in Fig. 1. The channel bandwidth could be T1

(1.544Mb/s), T3 (45Mb/s), and ATM (45, 100, 155, and 622Mb/s). Here all data above are randomly generated.

Table II Resource Table

Node id	1	2	3	4	5	6	7	8	9	10
Program id	-	-	2	2	3	4	1	-	4	-
Sharable no. of programs	-	-	1	2	1	5	3	-	3	-
Memory size (MB)	16	13	24	21	28	18	19	14	30	16
Printer no.	0	1	0	1	0	1	0	1	1	0
Disk space (GB)	1	4	10	1	6	2	6	10	2	4
Band. of Printer (Mb/s)	-	1	-	2	-	1	-	2	2	-
Band. Of Disks (MB/s)	12	10	1	33	12	2	2	5	1	48
Computation power level	3	2	3	1	2	2	1	2	2	1

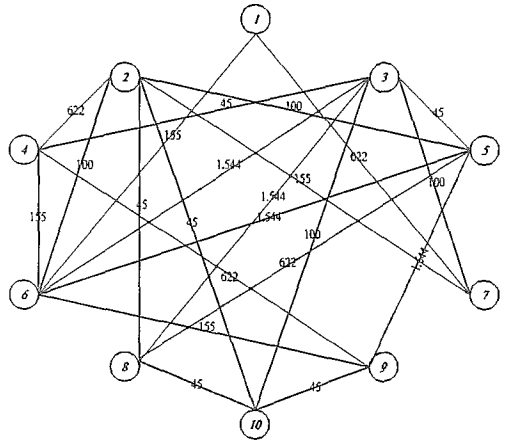

Fig. 1 Channel Bandwidth between Nodes

4.2 Experiments

Three experiments are conducted in the simulation. The request patterns of applications in each experiment are randomly generated according to the resource table shown in Table II.

Experiment 1: Limitation of Network Bandwidth

In this experiment, we consider limited network bandwidth as in real environment. From Fig. 2, we find that our solution has the same number of concurrent applications as the optimal one in all cases.

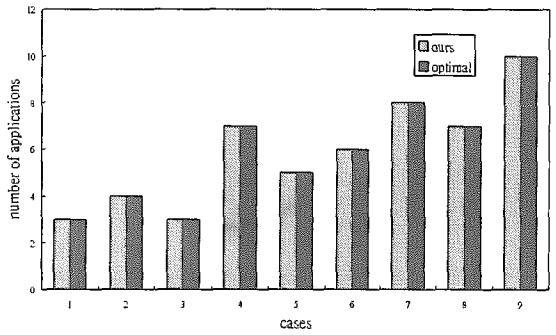

Fig. 2 Number of applications with the limitation of network bandwidth

Experiment 2: Dynamic Environment

In this experiment, we explore a dynamic environment; that is resources can be reallocated when a completed application has released all its resources. Here the evaluated parameter is the total execution time of all applications. From Fig. 3, we find that the total execution time of our solution is a lit tle more than that of the optimum solution only in some cases. These cases occur when most applications request the same resident program with low sharability. Besides we also measure the execution time of the random resource allocation that shows the worst performance in all cases.

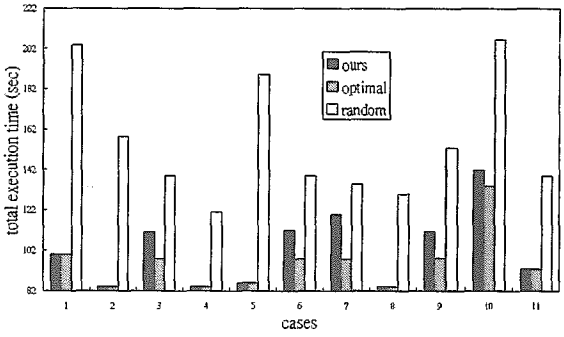

Fig. 3 Total execution time for dynamic environment

Experiment 3: QoS Specification

In this experiment, we explore the QoS effect in a dynamic environment. The QoS range is from 10% to 100%. Here the evaluated parameters are the number of concurrent applications and the total execution time of all applications. For each case shown in Fig. 4(a), we find that the more QoS the applications claim, the less the concurrent degree is. However the total execution time of all applications shown in Fig. 4(b) shows a parabola. The reason why the total execution time under low QoS is long is that the execution of applications is dominated by the low transmission rate when the applications request remote accesses. On the other hand, since less concurrent applications are executed under high QoS, the total execution time of all applications is lengthened. Thus the

best case occurs frequently when QoS is specified between 40% and 60%.

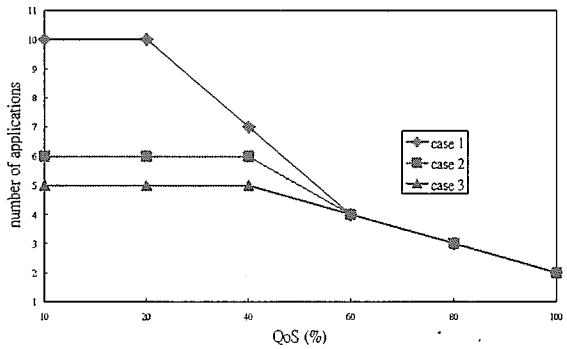

Fig. 4(a) Number of applications for QoS specification

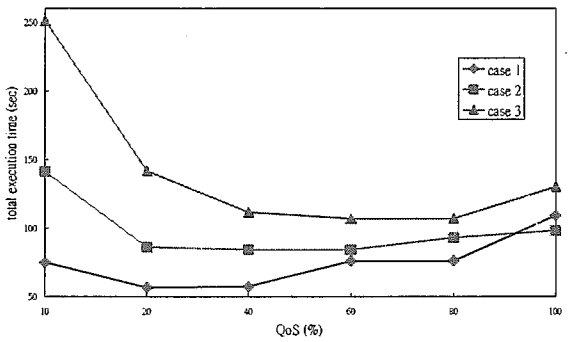

Fig. 4(b) Total execution time for QoS specification

5 Conclusions

Till now, we have not yet found any researches focusing the resource allocation problem for applications in a distributed computing network. From the viewpoints of applications, we propose three resource algorithms based on the objective function to solve different problems such as 1) with the limitation of network bandwidth, 2) wit dynamic environment, 3) with the specified QoS of applications. Since these resource allocation problems are NP-hard ones, the objective function is used to simplif the searching method. Although we cannot guarantee to obtain an optimal solution, the solution is definitely a near optimal one through the validation of simulation experiments.

References

[1] G. Chartrand and O. R. Oellerann, *Applied and Algorithmic Graph Theory*, McGraw-Hill International Editions, 1993.

[2] G. M. Chiu and C. S. Raghavendra, "A Model for Optimal Resource Allocation in Distributed Computing Systems," Proc. the Seventh Annual Joint Conference of the IEEE Computer and Communications Societies, pp. 1032 -1039, March 1988.

[3] A. E. El-Abd, "Modeling Resources Allocation and Performance Measures in Distributed Computer Networks," Proc. IEEE International Conference on Networks and Information Engineering, pp. 581 -586, July 1995.

[4] J. Huang, Y. Wang, N. R. Vaidyanathan, and F. Cao, "GRMS: A Global Resource Management System for Distributed QoS and Criticality Support," P roc. IEEE International Conference on Multimedia Computin and Systems, pp. 424-432, June 1997.

[5] J. Huang, D. Kenchamanna-Hosekote, M. Agrawal, and J. Richardson, "Presto- A System Environment for Mission-Critical Multimedia Applications," Journal of Real-Time Systems, July 1997.

[6] D. Hutchison, G. Coulson, A. Campbell, and G. S. Blair, *Quality of Service Management in Distributed Systems Management*, Ed. Morris Sloman, Imperial College London, 1995.

[7] K. B. Irani and N. G. Khabbaz, "A Methodology for the Design of Communication Networks and the Distribution of Data in Distributed Super Computer Systems," IEEE Trans. on Computer, Vol. C -31, No. 5, 1982.

[8] C. Lee, R. Rajkumar, and C. Merser, "Experiences with Processor Reservation and Dynamic QoS in Real-Time Mach," Proc. Multimedia'96, 1996.

[9] L. Zhang, S. Deering, D. Estrin, S. Shenker, and D. Zappala, "RSVP: A New Resource Reservation Protocol," IEEE Network, Vol. 7, No. 5, pp. 8 -18, 1993.

[10] G. Zhou, M. Gen, and T. Wu, "A New Approach to the Degree-Constrained Minimum Spanning Tree Problem Using Genetic Algorithms," Proc. IEEE International

Conference on Systems, Man, and Cybernetics, pp. 2683-2688, Oct. 1996.