İSTANBUL TECHNICAL UNIVERSITY ★ INSTITUTE OF SCIENCE AND TECHNOLOGY

ANT COLONY OPTIMIZATION FOR SURVIVABLE VIRTUAL TOPOLOGY MAPPING IN OPTICAL WDM NETWORKS

M.Sc. Thesis by
Elif KALDIRIM

Department : Computer Engineering

Programme : Computer Engineering

JUNE 2009

İSTANBUL TECHNICAL UNIVERSITY ★ INSTITUTE OF SCIENCE AND TECHNOLOGY

ANT COLONY OPTIMIZATION FOR SURVIVABLE VIRTUAL TOPOLOGY MAPPING IN OPTICAL WDM NETWORKS

M.Sc. Thesis by
Elif KALDIRIM
504061511

Date of submission : **04 May 2009**
Date of defence examination: **09 June 2009**

JUNE 2009

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

OPTİK AĞLARDA KARINCA KOLONİ ALGORİTMALARI KULLANARAK SANAL TOPOLOJİ ÜZERİNDEKİ IŞIK YOLLARININ HATAYA BAĞIŞIK OLARAK YÖNLENDİRİLMESİ

YÜKSEK LİSANS TEZİ
Elif KALDIRIM
504061511

Tezin Enstitüye Verildiği Tarih :	**04 Mayıs 2009**
Tezin Savunulduğu Tarih :	**09 Haziran 2009**

HAZİRAN 2009

FOREWORD

First and foremost, I would like to thank to my family for their great support during my whole life. I owe a big debt of gratitude to them.

I am deeply indebted to my supervisors Asst. Prof. Dr. Şima Uyar and Asst. Prof. Dr. Ayşegül Yayımlı for their guidance, patience and understanding during the course of my work. I could not have even imagined this much achievement without their encouragement.

I would like to thank to Fatma Corut Ergin, not as a coworker but as a friend, for her help, support and valuable hints.

Grateful thanks are extended to Miraciye Kurtulan for her understanding and encouragement. She gave me the chance to continue M.Sc. while working. She is more than a manager. She is also a teacher, a mother and a friend.

I would like to express my appreciation to the Scientific and Technological Research Council of Turkey (TÜBİTAK) for providing scholarship during my M.Sc. Education.

Especially, I would like to give my special thanks to my fiancee Zekeriya Köse and his family. The preparation of this document would not have been possible without their support and encouragement. I owe an enormous debt of appreciation to Zekeriya. Words alone cannot adequately express my gratitude to him.

May 2009

Elif KALDIRIM
Computer Engineer

TABLE OF CONTENTS

ABBREVIATIONS

ACO	:	Ant Colony Optimization
ACS	:	Ant Colony System
ANTS	:	Approximate Nondeterministic Tree Search
AS	:	Ant System
BWAS	:	Best-Worst Ant System
DAP	:	Disjoint Alternate Path
EA	:	Evolutionary Algorithms
EAS	:	Elitist Ant System
ILP	:	Integer Linear Program
IP	:	Internet Protocol
MMAS	:	MAX-MIN Ant System
RAS	:	Rank-based Ant System
RWA	:	Routing and Wavelength Assignment
TDM	:	Time Division Multiplexing
TSP	:	Travelling Salesman Problem
VT	:	Virtual Topology
WDM	:	Wavelength Division Multiplexing

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLES

$\mathbf{W}$	:	The capacity of each link in physical topology
$\mathbf{N}$	:	Set of nodes in physical topology
$\mathbf{E}$	:	Set of edges in physical topology
$\boldsymbol{l}$	:	The number of lightpaths
$\boldsymbol{k}$	:	The number of shortest paths
$\boldsymbol{m}$	:	The number of ants
$\boldsymbol{N}_L$	:	Set of nodes in virtual topology
$\boldsymbol{E}_L$	:	Set of edges in virtual topology
$\boldsymbol{\tau}_{ij}$	:	Pheromone level between nodes i and j
$\boldsymbol{\tau}_{threshold}$	:	Average pheromone trail on the edges visited by the best-so-far ant
$\boldsymbol{\eta}_{ij}$	:	Heuristic information between nodes i and j
$\boldsymbol{\alpha}$	:	The effect of pheromone level
$\boldsymbol{\beta}$	:	The effect of heuristics information
$\boldsymbol{q}_0$	:	Pseudo-random proportion
$\boldsymbol{P}_m$	:	Pheromone mutation probability
$\boldsymbol{\Delta}\tau_{ij}^{k}$	:	The amount of pheromone ant k deposits on the arc between i and j
$\boldsymbol{\tau}_0$	:	Initial pheromone trails
$\boldsymbol{e}$	:	The weight given to the best-so-far solution
$\boldsymbol{w}$	:	The maximum rank of ants that will deposit pheromone
$\boldsymbol{T}_k$	:	The solution built by the k-th ant
$\boldsymbol{T}_{bs}$	:	The best-so-far solution
$\boldsymbol{T}_{ws}$	:	The worst solution
$\boldsymbol{C}_k$	:	The cost of the solution built by the k-th ant
$\boldsymbol{C}_{bs}$	:	The cost of the best-so-far solution
$\boldsymbol{\rho}$	:	The pheromone evaporation rate
$\boldsymbol{\sigma}$	:	The pheromone mutation power
$\boldsymbol{N}_i^k$	:	The allowed neighborhood of ant k when it is at node i.

ANT COLONY OPTIMIZATION FOR SURVIVABLE VIRTUAL TOPOLOGY MAPPING IN OPTICAL WDM NETWORKS

SUMMARY

As the Internet use increases significantly in everyday life, the need for bandwidth increases accordingly. The most effective technology to meet this high bandwidth need is the optical networking technology. The fiber used in optical networks has the highest bandwidth capacity (50 Tb/s) amongst all other physical layer technologies. This high capacity, using wavelength division multiplexing technology, can be divided into hundreds of different transmission channels, which can transmit simultaneously. Each of these channels work on different wavelengths and each channel can be associated with a different data transmission rate. Wavelength division multiplexing offers an attractive solution to increasing local area network bandwidth without disturbing the existing embedded fiber and continue to be the main choice for the near future.

End-to-end optical connections that the packet layer (IP, Ethernet, etc.) uses are called lightpaths. Since the fibers on the physical topology allow traffic flow on different wavelengths, more than one lightpath, each operating on different wavelengths, can be routed on a single fiber. All the lightpaths set up on the network form the virtual topology. Physical topology is the physical structure of the network that gives information about how the workstations are connected to the network through the actual cables that transmit data whereas the virtual topology is the way that the data passes through the network from one device to the next without regard to the physical connection of the devices. Edges of the virtual topology represent the lightpaths that need to be routed on the physical topology. When a lightpath from one node to another is defined on a physical topology, that means data passes between these nodes and an edge is created on virtual topology to indicate this data transfer.

Any damage to a physical link (fiber) on the network causes all the lightpaths routed through this link to be broken. Since huge data transmission (40 Gb/s) over each of these lightpaths is possible, such a damage results in a serious amount of data loss. Two different approaches can be used in order to avoid this situation:

1. Survivability on the physical layer
2. Survivability on the virtual layer

The first approach is the problem of designing a backup link/path for each link/path of the optical layer. The second approach is the problem of designing the optical layer such that the optical layer remains connected in the event of a single or multiple link failure. While the first approach provides faster protection for time-critical applications (such as, IP phone, telemedicine) by reserving more resources, the second approach, i.e. the survivable virtual topology design, which

has attracted a lot of attention in recent years, aims to protect connections using less resources. The problem that will be studied in this thesis is to develop methods for survivable virtual topology design, that enables effective usage of the resources. Given the physical parameters of the network (physical topology, optical transceivers on the nodes, wavelength numbers on the fibers, etc.) and the mean traffic rates between nodes, the problem of designing the lightpaths to be set up on the physical topology is known as the virtual topology design problem.

The virtual topology design problem can be divided into four different subproblems:

1. Designing a proper virtual topology according to the mean packet traffic rates between nodes,

2. Routing the lightpaths of the virtual topology on the physical topology,

3. Assigning wavelengths to the lightpaths,

4. Routing packet traffic over the virtual topology.

Since any solution to these subproblems affects the solution of other subproblems, the result obtained by solving the subproblems one-by-one and iteratively, may not be the optimum. The pure virtual topology design problem is proved to be NP-complete. This problem, when the survivability constraints are added, gets harder. Because of its complexity, it is not possible to solve the problem optimally in an acceptable amount of time, for real-life sized networks. The main concern of this study is the second subproblem called virtual topology mapping problem. Virtual topology mapping is the problem of routing lightpaths on physical topology in a way that the capacity constraints of fibers in physical topology are not violated. Survivable virtual topology mapping has another constraint stating that in case of a physical link failure, the virtual topology is not disconnected when all the lightpaths routed through this link are deleted from the virtual topology.

Since the problem is NP-complete, it is appropriate to use heuristics to obtain the solutions. There are several studies on this topic. However, in several of these studies, only the second subproblem of virtual topology design problem is considered, without the survivability constraint. According to our literature survey, the only nature inspired heuristics used to solve the routing problem under the survivability constraint are Tabu Search and Simulated Annealing. Nature inspired heuristics are used successfully to solve a great deal of NP-complete problems.

In this thesis, six ant colony optimization algorithms are implemented to solve the virtual topology mapping problem. Ant system has been the basis for many ACO variants which have become the state-of-the-art for many applications. These variants include elitist ant system, rank-based ant system, MAX-MIN ant system, ant colony system, best-worst ant system, the approximate nondeterministic tree search, and the hyper-cube framework. Elitist ant system, rank-based ant system, MAX-MIN ant system, ant colony system and best-worst ant system can be considered as direct variants of ant system since they all use the basic AS framework. The main differences between ant system and these variants are the pheromone update procedures and some additional details in the management of the pheromone trails. In this study, we implemented ant system, elitist ant system, rank-based ant system, MAX-MIN ant system, ant colony system and best-worst ant system for the VT mapping problem since these direct variants of ant system have been successfully applied to many similar problems in literature.

The way ant colony algorithms are applied to the virtual topology mapping problem is described below: The physical topology is used as a graph on which ants travel and construct their solutions. Ants simultaneously try to route lightpaths on the graph one-by-one. Each starts to route a random lightpath. The shortest paths between the end points of the lightpaths are provided to ants at the very beginning of the algorithm. Ants decide their move on the construction graph based on heuristic and pheromone information. Pheromone is modelled as a 2D array which accumulates the information learned by the ants. There are two different pheromones utilized in our problem, one for choosing the next lightpath called "lightpath pheromone", the other is for selecting the shortest path of the chosen lightpath called "shortest path pheromone". The lightpath pheromone has lightpaths in its columns and rows and tells the information which lightpath is more valuable to move from the current lightpath, whereas the shortest path pheromone has lightpaths in its rows and corresponding shortest paths in its columns and tells the information about which shortest path gives the best result when selected for the current lightpath. These pheromones are initialized in the beginning of the algorithm with the same value for each possible choice of the ant. The difference is created by heuristic information that is inversely proportional to the length of the shortest paths. The ants uses lightpath pheromone to decide the next lightpath and shortest path pheromone together with heuristic information to decide the shortest path for the selected lightpath. The pheromones are updated after solutions are constructed. The amount of the accumulated pheromone is proportional to the solution quality. An iteration is the process in which each ant constructs a complete solution. The algorithm stops when both maximum number of iterations are retrieved and the maximum allowed time is completed.

To compare the performance of ant colony optimization algorithms, we perform a series of experiments to calculate resource usage based on both hop-count and link-cost on 100 different instances each, for 3, 4, 5 connected virtual topologies. Link-cost calculation method considers the actual lengths of the physical links whereas hop-count method counts the number of physical links used. For each algorithm and node degree, 3 different numbers of alternative shortest paths for lightpaths are examined. We tried 5, 10, and 15 shortest path cases. Algorithms are run 20 times for each virtual topology and each run is allowed to continue for 15 seconds.

The performance of the ant colony optimization variants are shown quantitatively, both according to the speed, success rates and the effective usage of network resources. As a summary of the experiments, we recommend ant colony optimization algorithms for the survivable VT mapping problem due to their decision policy at each step. They find feasible solutions after each iteration. Based on the results, even though all ant colony optimization algorithms perform well, we can recommend MAX-MIN ant system with hop-count calculation method due to its overall success and better reaction to the increasing search space.

OPTİK AĞLARDA KARINCA KOLONİ ALGORİTMALARI KULLANARAK SANAL TOPOLOJİ ÜZERİNDEKİ IŞIK YOLLARININ HATAYA BAĞIŞIK OLARAK YÖNLENDİRİLMESİ

ÖZET

İnternet kullanımının günlük hayata her geçen gün daha fazla girmesiyle bant genişliği ihtiyacı giderek artmaktadır. Bu yüksek bant genişliği ihtiyacını karşılayabilecek en etkili teknoloji ise optik ağlardır. Optik ağlarda kullanılan fiber kablolar diğer tüm fiziksel katman teknolojilerinden çok daha büyük bant genişliğine sahiptir (50 Tb/s). Bu kapasite, WDM (dalga boyu bölmeli çoğullama - wavelength division multiplexing) tekniği kullanılarak, her biri farklı bir dalgaboyunda çalışan yüzlerce farklı iletim kanalına bölünebilir ve bu kanallar eşzamanlı çalıştırılarak kullanılabilir. Bu kanalların her biri farklı dalga boyunda çalışır ve her bir kanalın veri aktarım hızı istenildiği gibi seçilebilir. Dalga boyu bölmeli çoğullama tekniği, sürekli artan yerel ağ bant genişliğine, fiberin varolan fiziksel yapısını bozmadan dikkat çekici bir çözüm sunmakta ve yakın gelecekte bu konuda ilk akla gelen adres olmaya devam etmektedir.

Paket katmanın (IP, Eternet, vs.) kullanacağı uçtan uca kurulan optik bağlantılara ışıkyolu (lightpath) denir. Fiziksel topolojideki fiber kablolar farklı dalgaboylarında trafik akışına izin verdiğinden, bir fiber üzerinde farklı dalgaboylarında olmak kaydıyla birden fazla ışıkyolu yönlendirilebilir. Ağda kurulan tüm ışıkyolları ağın sanal topolojisini (virtual topology) oluşturur. Fiziksel topoloji, ağın fiziksel yapısıdır ve ağda bulunan bilgisayarların bilgi iletimini sağlayan gerçek kablolarla, ağa nasıl bağlandığı hakkında bilgi verir. Diğer yandan sanal topoloji, aralarında fiziksel bir bağlantının varlığına dikkat etmeden, bir birimden diğerine bilgi geçişinin olup olmadığı hakkında bilgi verir. Sanal topolojinin kenarları, fiziksel topoloji üzerinde yönlendirilmesi gereken ışık yollarını gösterir. Fiziksel topoloji üzerinde bir düğümden diğerine bir ışıkyolu tanımlanması, bu düğümler arasında veri akışının olacağı anlamına gelmektedir ve bu veri akışını anlatmak üzere sanal topolojiye bir kenar eklenir.

Ağ üzerindeki bir fiziksel bağlantının (fiber) herhangi bir şekilde hasara uğraması, bu bağlantı üzerinden geçen tüm ışıkyollarının kopmasına neden olur. Işıkyollarının herbiri üzerinden çok büyük miktarlarda veri akışı sağlanabildiğinden (40 Gb/s) böyle bir hasar durumunda ağda çok ciddi veri kaybı meydana gelir. Bu durumdan korunmak için iki farklı yaklaşım kullanılmaktadır:

1. Fiziksel katmanda hataya bağışıklık

2. Sanal katmanda hataya bağışıklık

Birinci yöntem, optik katmandaki herhangi bir yol/bağlantı için yedek yol/bağlantı tasarlama problemidir. İkinci yöntem ise, fiziksel katmanda bir ya da daha fazla bağlantı koptuğunda sanal topolojinin hala bağlı olabilmesini

sağlayacak şekilde tasarım yapmaktır. Birinci yöntem daha fazla kaynağı rezerve ederek, zamana bağlı kritik uygulamalarda (IP telefon, teletıp gibi) daha hızlı koruma sağlarken; son yıllarda dikkat çeken ikinci yöntem, yani hataya bağışık sanal topoloji tasarımı daha az miktarda kaynak kullanarak bağlantıların korunmasını amaçlamaktadır. Bu projede üzerinde çalışılacak problem de ağ kaynaklarını etkin kullanan bir hataya bağışık sanal topoloji tasarım yönteminin geliştirilmesidir. Ağın fiziksel parametreleri (fiber topolojisi, düğümlerdeki optik alıcı ve verici sayıları (optical transceiver), fiber kablolardaki dalgaboyu sayısı,...) ve düğümler arasındaki ortalama trafik değerleri verildiğinde, bu kaynakları optimum düzeyde kullanarak fiziksel topoloji üzerine kurulacak ışıkyollarını tasarlama problemine "sanal topoloji tasarımı" denmektedir.

Sanal topoloji tasarımı problemi dört farklı alt problem şeklinde ele alınabilir:

1. Düğümler arasındaki paket trafiği yoğunlukları gözönüne alınarak uygun sanal topoloji belirlenmesi,

2. Sanal topolojideki ışıkyollarının fiziksel topolojideki bağlantılar üzerinde yönlendirilmesi,

3. Işıkyollarına dalgaboyu atanması,

4. Paket trafiğinin sanal topoloji üzerinde yönlendirilmesi.

Bu alt problemlerden herbirinin çözümü diğerlerini etkilediğinden, ayrı ayrı ve sırayla çözüldüklerinde ortaya çıkan sonuç en iyi çözüm olmayabilmektedir. Salt sanal topoloji tasarımı probleminin NP-karmaşık olduğu kanıtlanmıştır. Bu problem, hataya bağışıklık koşulu da eklendiğinde, daha da zorlaşmaktadır. Problemin karmaşıklığı nedeniyle, gerçek uygulamalardaki boyutlarda hızlı bir şekilde ve optimum olarak çözülmesi mümkün olmamaktadır. Bu çalışmanın esas konusu, yukarıdaki alt problemlerden ikincisi olan ışıkyollarının sanal ağ üzerinde yönlendirilmesi problemidir. Sanal topolojinin yönlendirilmesi problemi, ışıkyollarının sanal topoloji üzerinde fiziksel topolojide bulunan fiberlerin kapasite kısıtlarını aşmadan yönlendirilmesidir. Hataya bağışık olarak sanal topolojinin yönlendirilmesi problemi bir kısıta daha sahiptir, bu kısıta göre, ağ üzerindeki bir fiziksel bağlantının (fiber) herhangi bir şekilde hasara uğraması sonucu, bu bağlantı üzerinden geçen tüm ışıkyollarının sanal topolojiden silinmesi durumunda sanal topolojinin hala bağlı bir graf olması gerekir.

Yukarıda bahsettiğimiz problem NP-karmaşık olduğundan çözümü için sezgisel yaklaşımlar kullanılması uygundur. Bu konuda yapılmış birçok çalışma vardır. Ancak bu çalışmaların çoğunda sanal topoloji tasarımı probleminin sadece ikinci alt problemi, hataya bağışıklık kısıtı göz önünde bulundurulmadan ele alınmıştır. Yaptığımız araştırmalar sonucunda hataya bağışıklık kısıtı altında yönlendirme problemi için sadece Tabu Arama (Tabu Search) ve Tavlama Benzetimi (Simulated Annealing) gibi doğa esinli algoritmaların kullanılmış olduğunu gördük. Doğa esinli algoritmalar birçok NP-karmaşık problemin çözümünde başarılı olarak kullanılmaktadır.

Bu tez kapsamında, sanal topoloji üzerindeki ışık yollarının yönlendirilmesi problemine altı farklı karınca koloni optimizasyon algoritması gerçeklenmiştir. Ant system, pek çok problemin literatürde en gelişmiş yöntemi olan çeşitli karınca koloni optimizasyon algoritmalarının temelini oluşturmaktadır. Karınca koloni optimizasyon algoritmalarının türevleri elitist ant system, rank-based ant system, MAX-MIN ant system, ant colony system, best-worst ant system, approximate nondeterministic tree search ve hyper-cube framework olarak listelenebilir. Bu algoritmalar içinde ant system algoritmasının direk türevi olan algoritmalar elitist ant system, rank-based ant system, MAX-MIN ant system, ant colony system ve best-worst ant system algoritmalarıdır, çünkü bu algoritmalar ant system algoritmasının ana çatısını kullanmaktadır. Ant system ve türevleri arasındaki en belirgin farklılıklar hormon güncelleme yöntemleri ve hormonun peşinden gitme stratejileri arasındaki değişikliklerdir. Bu çalışmada, literatürde benzer problemler üzerindeki başarıları nedeniyle, ant system, elitist ant system, rank-based ant system, MAX-MIN ant system, ant colony system ve best-worst ant system algoritmaları sanal ağ üzerinde ışıkyollarının yönlendirilmesi problemini çözmek amacıyla gerçeklenmiştir.

Karınca koloni optimizasyon algoritmalarının, sanal ağ üzerinde ışıkyollarının yönlendirilmesi problemine uygulanması şöyle olmuştur: Fiziksel topoloji, karıncaların üzerinde dolaştığı ve çözüm ürettiği bir graf olarak kullanılmıştır. Karıncalar eş zamanlı olarak, ışıkyollarını graf üzerinde birer birer yönlendirmektedir. Herbiri rasgele bir ışıkyolunu yönlendirmekle işe başlar. Algoritmanın en başında, karıncalara bütün ışıkyollarının uçları arasındaki en kısa yolların bilgisi verilmektedir. Karıncalar çözüm ürettikleri graf üzerindeki hareketlerine hormon ve sezgisel bilgilerini kullanarak karar verirler. Hormon karıncaların öğrenilmiş bilgilerini tutan iki boyutlu bir dizi olarak modellenmiştir. Problemimizde iki tane hormon bilgisi kullanılmaktadır, birisi sonraki ışık yolunu bulmak amacıyla kullanılan "ışıkyolu hormonu", diğeri ise seçilen ışıkyolu için en kısa yolun bulunması için kullanılan "en kısa yol hormonu" 'dur. Işıkyolu hormonunun satır ve sütunlarında ışıkyolları bulunmaktadır ve yönlendirilmesi bitmiş olan şimdiki ışıkyolundan sonra hangi ışıkyolunun seçilmesinin daha faydalı olacağına dair bilgi verir. En kısa yol hormonunun ise satırlarında ışıkyolları, sütunlarında bu ışıkyollarına karşılık gelen en kısa yollar bulunmaktadır ve seçilen ışıkyolu hangi en kısa yoldan yönlendirilse daha iyi sonuç elde edileceği hakkında bilgi verir. Bu hormon matrisleri, algoritmanın en başında karıncanın yapabileceği her seçim için aynı değer ile ilklendirilir. Farklılık sezgisel bilgi ile yaratılır, öyle ki, bu bilgi en kısa yolların uzunlukları ile ters orantılıdır. Karıncalar sonraki ışıkyoluna karar verirken ışıkyolu hormonunu, seçilen ışıkyolunun hangi yolla yönlendirileceğine karar verirken ışık yolu hormonunu sezgisel bilgi ile beraber kullanırlar. Karıncaların her biri çözümürettiğinde hormon bilgileri güncellenir. Hormon bilgisine eklenen yeni hormon değerleri çözüm kalitesi ile doğru orantılıdır. Bütün karıncaların bir çözüm üretmesi için geçen süreye iterasyon denir. Algoritma, daha önceden tanımlanan maksimum iterasyon sayısı tamamlandığında ve kendisine verilen maksimum süre dolduğunda sonlanır.

Karınca koloni algoritmalarının performanslarını karşılaştırmak amacıyla bir dizi testler yapılmıştır. 3, 4 ve 5 bağlı 100'er farklı sanal topoloji kullanılmış ve bulunan çözümlerin ağ kullanımı değerleri fiberler üzerinden geçiş sayısı ve geçilen fiberlerin maliyetleri gözönüne alınarak hesaplanmıştır. Maliyet dikkate alındığında kullanılan fiberlerin kilometre olarak uzunlukları hesaba alınırken, geçiş sayısı dikkate alındığında sadece kullanılan fiberlerin sayısı hesaplanmıştır. Her algoritmaya kaç bağlı sanal topoloji olduğuna dikkat edilmeksizin 3 farklı

sayıda en kısa yolların bilgisi sağlanmıştır. Testlerimizde sırasıyla 5, 10 ve 15 tane en kısa yol kullanılmıştır. Algoritmalar, her sanal topoloji için 20 kez çalıştırılmış ve her koşumun 15 saniye sürmesine izin verilmiştir.

Geliştirilen yöntemlerin performansı hem hız, hem başarım hem de ağ kaynaklarının etkin kullanımı açısından nitel olarak ortaya koyulmuştur. Yapılan testlerin sonucu olarak, karınca koloni algoritmaları her adımda kısıtları gözönüne alarak karar verme stratejileri sayesinde, hataya bağışık olarak ışıkyollarının sanal topoloji üzerinde yönlendirilmesi problemine uygulanabilir. Karınca koloni algoritmaları her iterasyon sonunda kısıtları aşmayan uygun çözüm üretebilmişlerdir. Sonuçları incelediğimizde, her karınca koloni algoritması iyi sonuç üretmiş olsa da, MAX-MIN ant system algoritmasının gerek başarısı gerekse artan arama uzayına toleransından dolayı geçiş sayısı hesaplama yöntemi ile birlikte kullanılmasını öneririz.

1. INTRODUCTION

Today, optical networking [1] is the most effective technology to meet the high bandwidth network demand. The high capacity of fiber used in optical networks, can be divided into hundreds of different transmission channels, using the wavelength division multiplexing (WDM) technology. Each of these channels work on different wavelengths and each channel can be associated with a different optical connection.

Any damage to a physical link (fiber) on the network causes all the channels on this link to be broken. Huge amount of data (40 Gb/s) can be transmitted over each of these channels, so a fiber damage may result in a serious amount of data loss. To avoid data loss, these channels can be designed in a way that in the event of a single or multiple link failures, all workstations on the network can still accomplish data transfer. In this study, our aim is to route data traffic through these channels while considering single link failures and the capacity of the fibers.

The communication between nodes in the physical topology is mapped on a graph called virtual topology (VT). Lightpaths are the edges of the VT representing communication channels to be routed on the physical topology. VT mapping is the problem of routing lightpaths on the physical topology in such a way that the capacity constraints of fibers in the physical topology are not violated. Survivable VT mapping has another constraint stating that in case of a physical link failure, the VT is not disconnected when all the lightpaths routed through this link are deleted from the VT.

The VT mapping problem is known to be NP-complete [2]. Because of its complexity, for real-life sized networks, it is not possible to solve the problem optimally in an acceptable amount of time using classical optimization techniques. Therefore, heuristic approaches should be used. In this study, we chose ant colony algorithms (ACO) because of their successful applications on NP-complete problems. We used ACO to find a survivable mapping of a given VT while minimizing the resource usage. We implemented six different ACO algorithms and compared their performance to determine which algorithm is more suitable for this problem and we investigated possible reasons.

The rest of the thesis is organized as follows. In Section 2, a brief introduction to optical networks and WDM is given. The definition of the problem together with its mathematical formulation is given in Section 3, followed by the related literature. In Section 4, the details of the implemented six ACO algorithms are given. Section 5 interprets the way ACO algorithms are applied to the survivable VT mapping problem. In Section 6, the experimental results are given and these results are discussed thoroughly. Finally, in Section 7, conclusion and future work are given.

2. OPTICAL NETWORKS

A revolution in telecommunications networks evolved in the early 1980s and became widespread by the use of a relatively unassuming technology: fiber optic cable. Since then, optical networks have been commonly used due to increased network quality and the tremendous cost savings. The benefits of optical networks have been increased by the advances in the technologies required for optical networks.

There are many factors driving the need for optical networks. A few of the most important reasons for migrating to the optical layer can be listed as fiber capacity, restoration capability, reduced cost and wavelength services [3].

Fiber Capacity

Optical networks were first implemented on fiber-limited routes. However, a few years later, the capacity of fibers became inadequate to meet the increased demand. More capacity is needed between two sites. As higher bit rates were not available in a fiber, there remains no other options except installing more fiber or placing more time division multiplexed (TDM) signals on the same fiber. The first choice is expensive and labor-intensive. Using WDM technology, many "virtual" fibers are provided on a single physical fiber. Network providers managed to send many signals on one fiber by transmitting each signal at a different frequency.

Restoration Capability

A failure in a fiber can result in enormous consequences because of the increased capacity. Each network element performs its own restoration in current electrical architectures. Whereas, in a WDM system with many channels on a single fiber, a fiber cut would cause multiple failures to happen, causing many independent systems to fail. Optical networks can perform protection switching faster and more economically when restoration is performed in the optical layer instead of the electrical layer. Moreover, networks that currently do not have a protection scheme can also be restored using the optical layer. As a result of this technology, providers are able to add restoration capabilities to embedded asynchronous systems without first upgrading to an electrical protection scheme.

Reduced Cost

In optical networks, the high cost of electronic cross-connects is avoided by providing space and wavelength for routing of traffic and network management is simplified.

In WDM technology, each optical switch that demultiplexes signals will utilize an electrical network element for each channel, without regard to the existence of traffic routed through that node. By implementing an optical network, only those wavelengths that add or drop traffic at a site need corresponding electrical nodes. Other channels can simply pass through optically. That provides enormous cost savings in network and equipment management.

Wavelength Services

One of the great advantage of optical networks is the ability to resell bandwidth instead of fiber. Service providers can improve revenue by selling wavelengths by maximizing capacity available on a fiber, without regard to the data rate required. Customers think this service provides the same bandwidth as a dedicated fiber.

2.1 Fiber Optic Communication

An optical fiber (or fibre) is a plastic or glass fiber that is used for carrying light along its length. Fiber optics is arised from the common studies of applied science and engineering on the design and application of optical fibers.

Optical fibers are widely used in fiber-optic communications, that permits data transmission over longer distances and at higher bandwidths than other communications systems. Metal wires are replaced by fibers because signals flow through them with less loss, and they are not affected by electromagnetic interference. Fibers can be used to carry and brighten images. They can also be designed specially to be used for a variety of other applications, such as sensors and fiber lasers.

Fiber-optic communication systems were first developed in the 1970s and revolutionized the telecommunications industry [4]. They have a significant contribution to the advent of the Information Age. In the developed world, optical fibers have been largely used instead of the copper wire communications in core networks due to their benefits of electrical transmission.

Fiber-optic communication is a method that sends pulses of light through an optical fiber to transmit data from one place to another. The light behaves as an electromagnetic carrier wave that is responsible for carrying data.

The procedure of fiber-optic communication involves the following basic steps: Creating the optical signal via a transmitter, flowing the signal through the fiber, verifying that the signal is not too deformed or weak, receiving the optical signal, and converting it into an electrical signal.

Until the late 1980s, optical fiber communications was mainly restricted to transmitting data using a single optical channel that is required periodic maintenance because signals in a fiber get weaker after a time period. This maintenance includes detection, electronic processing, and optical retransmission that causes a high-speed optoelectronic traffic delay and can handle only a single wavelength [5]. The development of the new generation amplifiers enabled us to accomplish high-speed repeaterless single-channel transmission.

2.2 Wavelength Division Multiplexing

WDM is the method of dividing the wavelength capacity of an optical fiber into multiple channels to send more than one signal using the same fiber [6]. This requires a wavelength division multiplexer in the transmitting equipment and a wavelength division demultiplexer in the receiving equipment. Using WDM technology now commercially available, the bandwidth of a fiber can be divided into as many as 80 channels to support a bit rate combination into the range of terabits per second. That is why WDM in optical fiber networks has been rapidly gaining acceptance as a means to meet the increasing bandwidth demands of network users [1]. To illustrate the WDM technology, we can assume the highway as a optical fiber. The single high-speed lane in this highway is thought

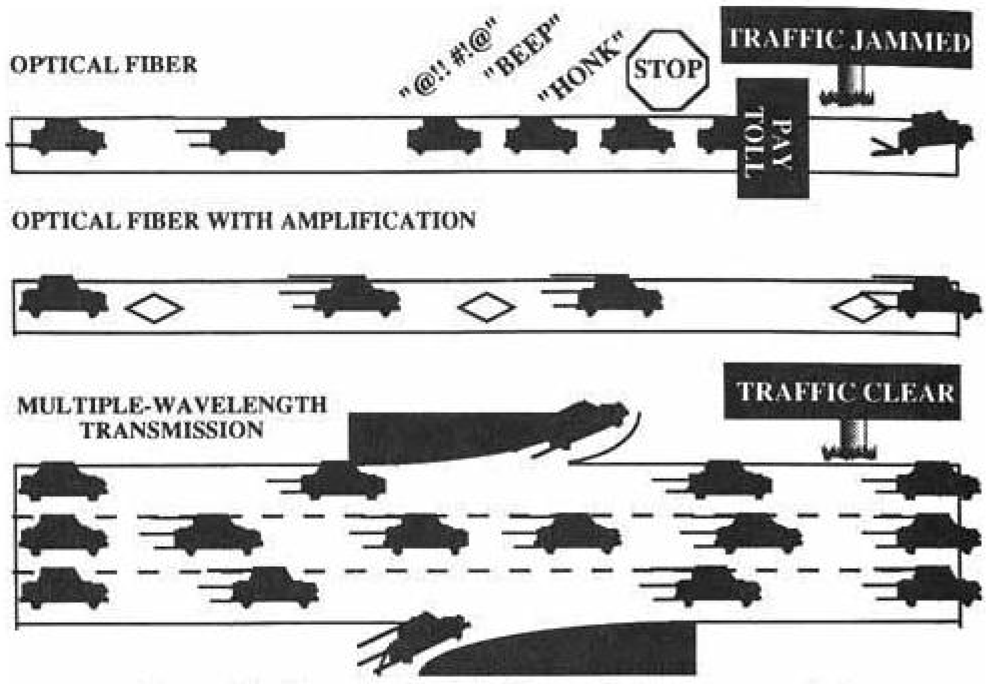

Figure 2.1: Multiwavelength optical transmission as represented by a multiple-lane highway.

of a single channel that has a capacity around Gbps. The cars are packets of optical data. However, the 25 THz optical fiber can accommodate much more bandwidth than the traffic from a single lane. To increase the system capacity

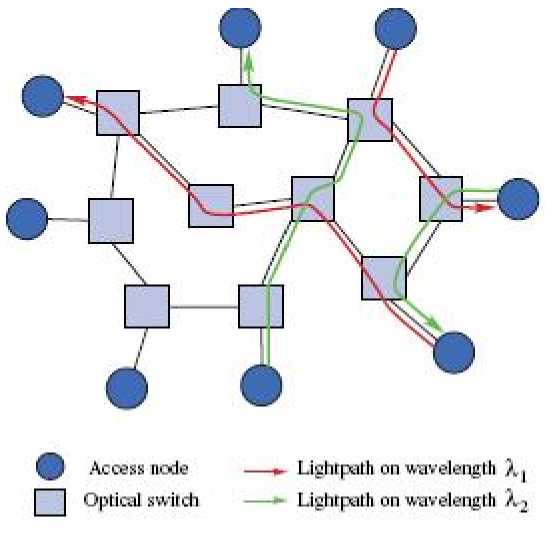
Access node
Optical switch
Lightpath on wavelength λ_1
Lightpath on wavelength λ_2

Figure 2.2: WDM network with lightpath connections.

we can fully utilize this huge fiber bandwidth by transmitting several different independent wavelengths simultaneously through this fiber. Therefore, the intent was to develop a multiple-lane highway, with each lane representing data traveling on a different wavelength. Thus, a WDM system enables the fiber to carry more amount of data. By using wavelength-selective devices, independent signal routing can also be accomplished. The highway principle illustrated in Figure 2.1 is taken from [7].

In a wavelength-routed WDM network, the communication between end users is provided via all-optical WDM channels, which are referred to as lightpaths [8]. A lightpath is used to provide a connection in a wavelength-routed WDM network, and it may spread over multiple fiber links.

When there are not any wavelength converters, a lightpath must hold the same wavelength on all the fiber links through which it crosses. Figure 2.2 illustrates a wavelength-routed network in which lightpaths have been set up between pairs of access nodes on different wavelengths.

3. SURVIVABLE VIRTUAL TOPOLOGY MAPPING PROBLEM

Optical WDM networks use a technology which multiplexes multiple optical signals on a single optical fiber by using different wavelengths (colours) of laser light to carry different signals. Any damage to a physical link (fiber) on the network causes all the signals carried by this link to be broken. Huge amount of data (40 Gb/s) can be transmitted over each of these channels, so a fiber damage may result in a serious amount of data loss. Two different approaches can be used to avoid data loss [9]:

1. Survivable design of the physical layer

2. Survivable design of the virtual layer

The first approach is the problem of designing a backup link/path for each link/path of the virtual layer. The main concern of this topology design is to protect or restore the link at the logical layer. A backup lightpath can always be found in the physical layer in any of the considered failure scenarios if the logical topology is designed as described [10]. This consideration assumes that either adequately high capacities are available or enough traffic can be dropped in case of a failure.

The second approach is the problem of designing the virtual layer such that it remains connected in the event of a single or multiple link failures. While the first approach provides faster recovery for time-critical applications (such as, IP phone, telemedicine) by reserving more resources; the second approach, i.e. the survivable VT design, which has attracted a lot of attention in recent years, aims to protect data communication using less resources. In this study, our main aim is to compare the performance of six different ACO algorithms to find a survivable mapping of a given VT while minimizing the resource usage.

VT design problem is defined as modelling the lightpaths to be set up on the physical topology when the physical parameters of the network (physical topology, optical transceivers on the nodes, wavelength numbers on the fibers, etc.) and the mean traffic rates between nodes are provided as an input. VT mapping problem, which is a subproblem of VT design, is to find a proper route for each lightpath of the given VT and to assign wavelengths to these lightpaths.

The VT design problem can be divided into four different subproblems:

1. Designing a proper VT according to the mean packet traffic rates between nodes,

2. Routing the lightpaths of the VT on the physical topology,

3. Assigning wavelengths to the lightpaths,

4. Routing packet traffic over the VT.

The main concern of this study is the second one. Given the physical and the virtual network topologies, our aim is to find a survivable mapping of the VT. Physical topology is the physical structure of the network that gives information about how the workstations are connected to the network through the actual cables that transmit data. VT is the way that the data passes through the network from one device to the next without regard to the physical connection of the devices. Edges of the VT represent the lightpaths that need to be routed on the physical topology.

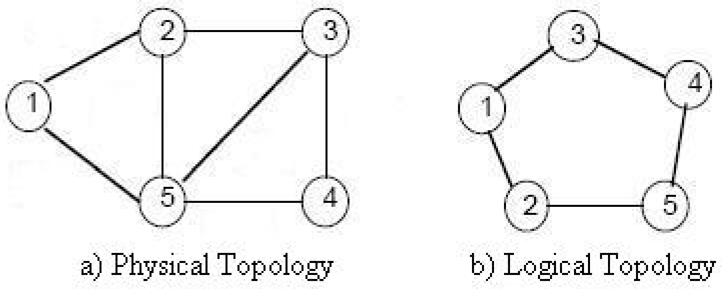

Figure 3.1: The difference between the physical and the logical topologies.

Figure 3.1 interprets the difference between the physical and the logical topologies. The nodes 3 and 5 are connected with actual cables in the physical topology but according to the VT, there is no lightpath between 3 and 5, so data transfer is not needed between these two nodes.

VT mapping is the problem of routing lightpaths on the physical topology in such a way that the capacity constraints of fibers in the physical topology are not violated. Survivable VT mapping has another constraint stating that in case of a physical link failure, the VT is not disconnected when all the lightpaths routed through this link are deleted from the VT.

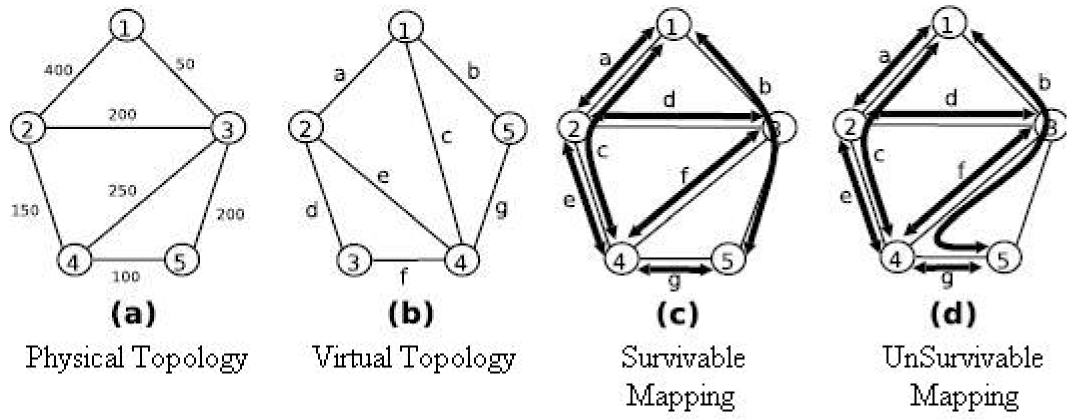

Figure 3.2: Illustration of the survivable VT mapping problem.

To illustrate the survivable VT mapping problem, assume that we have a physical network topology as in Figure 3.2.a and a virtual network topology representing lightpaths to be routed on this physical topology as in Figure 3.2.b. Figures 3.2.c and 3.2.d show the way the lightpaths are routed, e.g., the lightpath *c* in figure 3.2.b is routed through the nodes 1, 2 and 4 in both figures 3.2.c and 3.2.d while the lightpath *b* is routed through the nodes 1, 3 and 5 in figure 3.2.c, 1, 3, 4 and 5 in figure 3.2.d. If we route these lightpaths as in Figure 3.2.c we obtain a survivable mapping, that is, a failure on any physical link does not disconnect the VT. However, if the routing of only one lightpath is changed, e.g., as in Figure 3.2.d, we end up with an unsurvivable mapping. In this case, if a failure occurs on the physical link between nodes 4 and 5, the nodes connected with lightpaths *b* and *g* will not be able to find an alternative path to communicate. If we remove the lightpaths *b* and *g* from the VT, node 5 will be disconnected to the other nodes, so the VT will be unsurvivable.

3.1 Formal Problem Definition

The physical topology is composed of a set of nodes $N = \{1..N\}$ and a set of edges E where (i, j) is in E if $i \& j$ exist in N and there is a link between nodes i and j. Each link has a capacity of W wavelengths. The VT, on the other hand, has a set of virtual nodes N_L, which is a subset of N, and virtual edges (lightpaths) E_L, where an edge (s,t) exists in E_L if both node s and node t are in N_L and there is a lightpath between them.

An Integer Linear Program (ILP) formulation of survivable lightpath routing of a VT on top of a given physical topology is given in [2]. Based on this formulation, a number of different objective functions can be considered for the problem of survivable mapping. The simplest objective is to minimize the number of physical links used. Another objective is minimizing the total number of wavelength-links used in the whole physical topology. A wavelength-link is defined as a wavelength used on a physical link. To illustrate the difference between link and wavelength-link, assume that we have a VT routing as in figure 3.2.c. Here the number of physical links used is 7, whereas the total number of wavelength-links is 9. Our choice as the objective is the latter one, since it gives a better idea of the actual resource usage.

The optimal survivable routing problem that minimizes total number of wavelengths used can be expressed using the following ILP [2].

$$\text{Minimize} \quad \sum_{\substack{(i,j) \in E \\ (s,t) \in E_L}} f_{ij}^{st} \tag{3.1}$$

Let $f_{ij}^{st} = 1$ if lightpath (s,t) is routed on physical link (i, j) and 0, otherwise. Clearly $f_{ij}^{st} > 0$ means that there exists a physical link between nodes i and j.

The ILP formulation of the constraints are given as the following equations:

a. Capacity Constraint

$$\forall (i, j) \in E, \sum_{(s,t) \in E_L} f_{ij}^{st} \leq W \tag{3.2}$$

If the number of wavelengths on a fiber is limited to W, a capacity constraint can be imposed as in Eq. **(3.2)**.

b. Survivability Constraint

$$\begin{matrix}\forall(i,j)\in E\\ \forall S\subset N_L\end{matrix}\ ,\quad \sum_{(s,t)\in CS(S,N_L-S)} f_{ij}^{st}+f_{ji}^{st} <\mid CS(S,N_L-S)\mid \tag{3.3}$$

where $CS(S,N_L-S)$ is the set of cuts of the VT that divides the VT into two node sets S and $N-S$. Each cut defines a set of edges consisting of edges in E with one endpoint in S and the other endpoint in $N-S$. Removal of these edges from the VT seperates the VT into two parts. $\mid CS(S,N_L-S)\mid$ in Eq. **(3.3)** means the number of edges in the cut-set. This equation means that to route all the edges (s,t) in a cut of the VT, the fiber between nodes i and j should not be used more than the number of edges in the cut. As another explanation, the survivability constraint states that for all proper cuts of the VT, all edges(lightpaths) in this proper cut should not be routed through the same physical link.

c. Connectivity Constraint

For each pair (s,t) in E_L:

$$\sum_{(i,j)\in E} f_{ij}^{st}-\sum_{(j,i)\in E} f_{ji}^{st}=\begin{cases}1 & \text{if } s=i\\ -1 & \text{if } t=i\\ 0 & \text{otherwise}\end{cases} \tag{3.4}$$

Eq. **(3.4)** means that while routing the lightpath (s,t), the same amount of flow enters and leaves each node that is not the source or destination of (s,t). Moreover, node s has an outer input of one more unit of traffic that has to find its way to node t. There are many possible combinations that can satisfy the constraint of Eq. **(3.4)**.

d. Integer Flow Constraint

$$f_{ij}^{st}\in\{0,1\}$$

The integer flow constraint ensures that the information whether the lightpath (s,t) is routed on physical link (i,j) can take values only either true or false.

The aim of lightpath routing is to find a set of physical links that connect the nodes of the lightpaths. Since our objective is to minimize the total number of wavelength-links used in the whole physical topology, we can formulate the objective as in Eq. **(3.5)**:

$$\text{Minimize} \sum_{\substack{(i,j)\in E \\ (s,t)\in E_L}} f_{ij}^{st} * cost(i,j) \qquad \textbf{(3.5)}$$

where $cost(i,j)$ is considered to be equal to 1 when hop-count method is used as an objective. On the other hand, when the link-cost method is used as an objective, $cost(i,j)$ is considered to be equal to the actual length of the physical path in kilometers.

The survivable VT mapping problem implemented in this study has two constraints: survivability constraint and capacity constraint. The mathematical formulations are the same as Eq. **(3.3)** and Eq. **(3.2)** respectively.

a) Survivability constraint:

The survivability constraint means that all the lightpaths of a cut-set cannot be routed using the same physical link. The cut-set of a graph G is the subgraph G_x of G consisting of the set of edges satisfying the following properties:

- The removal of G_x from G reduces the rank of G exactly by one.

- No proper subgraph of G_x has this propery.

- If G is connected then the first property in the above definition can be replaced by the following phrase: The removal of G_x from G separates the given connected graph G into exactly two connected subgraphs.

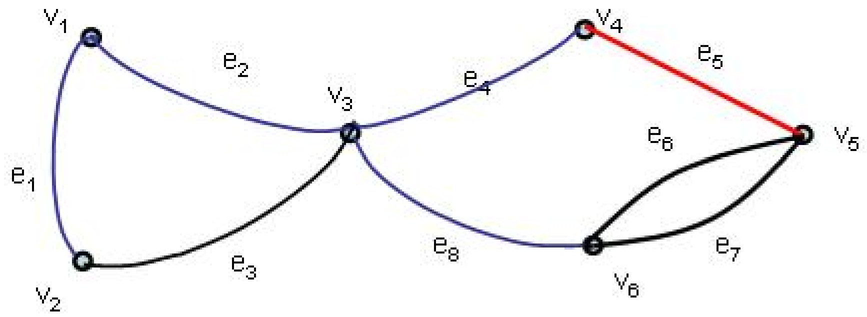

Figure 3.3: The cut-set of a graph.

Consider the graph in Figure 3.3. The edges $e4, e6, e7$ are the cut-set of the graph because these edges divide graph G into exactly two connected subgraphs. The edges $e1, e2$ are also a cut-set. But $e2, e3, e4, e8$ is not a cut-set, because the removal of these edges from G results in three connected subgraphs.

b) Capacity constraints:

The capacity constraint ensures that the number of wavelengths on a physical link is no more than its capacity W.

3.2 Related Literature

The survivable VT mapping problem was first addressed as Design Protection [11] in the literature. In this first study, tabu search was used to find the minimum number of source-destination pairs that become disconnected in the event of a physical link failure. Their aim is to find a systematic plan to protect a WDM optical network against component or link failures that may cause the simultaneous failure of several optical channels. To address this, they introduce the concept of Design Protection, which aims at making such failure propagations impossible. They present the Disjoint Alternate Path (DAP) algorithm which places optical channels in order to maximise design protection. The capacity constraint is the same as our problem but unlike ours, survivability is treated as objective in their study. The number of source-destination pairs that become disconnected in the event of a physical link failure must be zero in our study for a feasible solution while their aim is to minimise that number.

Nucci et. al. also used tabu search to solve the survivable VT design problem [12]. Their design methodology relies on the dynamic capabilities of IP routing to re-route IP datagrams. They first consider the resilience properties of the topology during the logical topology optimization process, so the optimization of the network resilience performance can be extended also on the logical topology space. The constraints in this study include transmitter and receiver constraints as well as wavelength capacity constraints.

Modiano and Narula-Tam used ILP to solve the VT mapping problem [2]. They added the survivability constraint in the problem formulation, such that, no physical link is shared by all virtual links belonging to a cut-set of the VT graph. However, they did not consider the capacity constraint. Their objective was to minimize the number of wavelengths used. For the cases when ILP cannot find an optimum solution in a reasonable amount of time due to the problem size, Modiano et. al. proposed two relaxations to ILP, which consider only small-sized cut-sets. These relaxations reduce the problem size; however, they may lead to suboptimal solutions. In order to overcome the long execution time problem in ILP formulation, Todimala and Ramamurthy proposed a new ILP formulation for computing the survivable routing of a virtual topology. As ILP is not scalable when the network size extends to a few tens of nodes, in their work, they present sub-graphs which more accurately model an actual network and for which a survivable routing can be easily computed using an ILP. They solved the problem for networks of up to 24 nodes [13]. In [13], besides the physical network and the virtual network topologies, the shared risk link groups should be known in advance. In their study, Todimala and Ramamurthy considered both capacity and survivability constraints.

A heuristic approach to VT mapping is developed by Ducatelle et. al. [14]. They consider the survivability constraint in this study. They consider a routing as survivable, if the connectivity of the logical network is guaranteed in case of a failure in the physical network. They introduce a local search algorithm which can provide survivable routing in case of not only physical link failures but also node failures and multiple simultaneous link failures. Unlike our problem, they considered survivability as an objective. Their aim is to minimise the total number of node pairs that make VT unsurvivable in case of a physical link failure.

Kurant and Thiran [15] used an algorithm that divides the survivable mapping problem into subproblems. Heuristic algorithms usually start with some initial mapping and then try to improve it. This involves the evaluation of the entire topology at each iteration, which is costly for large topologies. To overcome that cost, they propose a different approach that breaks down the current problem into subproblems. The combination of solutions of these subproblems is a survivable mapping.

There are a few studies on VT mapping [16] and design [17] using evolutionary algorithms (EA), however, the survivability is not considered in any of them except [18]. Ergin et. al. proposed the only EA based approach for survivable

VT mapping problem. Their objective is to minimize the resource usage without violating the capacity constraint. They experiment with different EA components to develop an efficient EA for this problem.

Swarm intelligence algorithms are used in a few studies for Routing and Wavelength Assignment (RWA) problem. Ant colony optimization(ACO) is applied to the static [19] and dynamic [20, 21] RWA problem without the survivability constraint. The only study using ACO considering back-up paths on physical layer is [22]. Particle swarm optimization is applied to RWA problem in only a single study [23], in which no survivability constraint is considered.

In [19], the objective is to minimize the wavelength used in the given network. They use a simple greedy heuristic for wavelength assignment. According to this approach, ants select their routes according to the weight of attraction of each physical link. Ants use a tabu list of previously visited nodes in order to avoid loops and backtracking. They use different methods for pheromone updating.

Garlick et. al. [20] is the first group that used ACO on dynamic RWA problem. In this approach, whenever a new connection request arrives, some of ants are launched from source to destination. While deciding which path to use, ants use the length of the path and the number of available wavelengths along the path.

Ngo et. al. also proposed an approach for the dynamic RWA problem. They designed a new routing table structure to solve this problem [21]. They use ants to observe the state changes in the network and to update these tables regularly. The results show that this algorithm outperforms the other alternate methods in terms of blocking probability. In a further study [22], Ngo et. al. handled the RWA problem considering the back-up paths on the physical layer, and used ACO to solve this problem.

The work that use particle swarm optimization for RWA problem in WDM networks use a hybrid algorithm inspired from ant systems [23]. For the routing part of the problem, particles are used to determine the path together with the ant system (AS). For the wavelength assignment part, a first-fit algorithm is used.

4. ANT COLONY OPTIMIZATION ALGORITHMS

ACO is one of the most commonly used swarm intelligence techniques and is based on the behavior of real ants. ACO has been applied successfully to many combinatorial optimization problems such as routing problems [24], assignment problems [25], scheduling and sequencing problems [26] and subset problems [27]. One of the first successful implementations of ACO is the Ant System (AS) developed by Dorigo [28], in 1992. AS has been the basis for many ACO variants which have become the state-of-the-art for many applications. These variants include elitist AS (EAS), rank-based AS (RAS), MAX-MIN AS (MMAS), ant colony system (ACS), best-worst AS (BWAS), the approximate nondeterministic tree search (ANTS), and the hyper-cube framework. AS, ACS, EAS, RAS, MMAS and BWAS can be considered as direct variants of AS since they all use the basic AS framework. The main differences between AS and these variants are the pheromone update procedures and some additional details in the management of the pheromone trails. In this study, we implemented AS, ACS, EAS, RAS, MMAS and BWAS for the VT mapping problem since it can be seen in [28] that these direct variants of AS have been successfully applied to many similar problems in literature [27].

ACO algorithms are inspired from the social behavior of ants that provide food to the colony. Ants deposit a substance called pheromone on the way they search, find food and return to the nest. Pheromone trails guide the colony during the food search process. Ants are able to smell the pheromone and remember the way they had used to reach food. When an ant is positioned at a location, it makes a decision about the next path to take based on a probability defined by the amount of pheromone existing in each trail. When a path betweeen the nest and the food is constructed, the ants stops depositing pheromone. The length of the path is reduced step by step because of the progressive action of the ants in the colony. The pheromone concentration becomes higher on the shortest paths because they are visited more frequently. On the contrary, the longest paths are less visited and the associated pheromone trails are evaporated.

The algorithmic flow of the basic ACO algorithm is given in Algorithm 1. An iteration consists of the solution construction and pheromone update stages. Iterations are finished when stopping criteria are met, which may be the time when both max solutions are generated and the allowed time is completed.

Algorithm 1 Basic ACO outline

```
 1: set ACO parameters
 2: initialize pheromone levels
 3: while stopping criteria not met do
 4:     for each ant $k$ do
 5:         select random initial node
 6:         repeat
 7:             select next node based on decision policy
 8:         until complete solution achieved
 9:     end for
10:     update pheromone levels
11: end while
```

In each iteration, each ant in the colony constructs a complete solution. Ants start from random nodes and move on the construction graph by visiting neighboring nodes at each step. For each node, the next node to visit is determined through a stochastic local decision policy based on the current pheromone levels and heuristic information between the current node and its neighbors. Heuristic information is proportional to the knowledge that makes the solution optimum. Better solutions have higher heuristic levels, i.e., for travelling salesman problem (TSP), heuristic information is usually set as $1/d_{ij}$ where d_{ij} is the distance between cities i and j.

An ant k determines its next move from i to j with a probability p^k_{ij} as calculated in Eq. **(4.1)**,

$$p^k_{ij} = \begin{cases} \dfrac{\tau^{\alpha}_{ij} \cdot \eta^{\beta}_{ij}}{\Sigma_{l \in N^k_i} \tau^{\alpha}_{il} \cdot \eta^{\beta}_{il}} & \text{if } j \in N^k_i \\ 0 & \text{otherwise} \end{cases} \qquad \textbf{(4.1)}$$

where τ_{ij} and η_{ij} are the pheromone level and heuristic information between nodes i and j respectively, α and β are the parameters used to determine the effect of the pheromone level and heuristics information respectively, N_i^k is the allowed neighborhood of ant k when it is at node i. The probabilistic action choice in Eq. (4.1) is called random proportional rule. The effect of α and β on heuristic and pheromone information is the following: if $\alpha = 0$, the neighbor node that has the biggest heuristic information is selected, if $\beta = 0$, heuristic information is not used while deciding the next move, only pheromone is used.

Pheromone trails are modified when all ants have constructed a solution. First the pheromone values are lowered (evaporated) by a constant factor on all edges. Then pheromone values are increased on the edges the ants have visited during their solution construction. Pheromone evaporation and pheromone update by the ants are implemented as given in Eq. (4.2) and Eq. (4.3) respectively,

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij} \tag{4.2}$$

$$\tau_{ij} \leftarrow \tau_{ij} + \sum_{k=1}^{m} \Delta\tau_{ij}^k \tag{4.3}$$

where $0 < \rho \leq 1$ is the pheromone evaporation rate, m is the number of ants and $\Delta\tau_{ij}^k$ is the amount of pheromone ant k deposits on the arcs it has visited. Evaporation prevents adding unlimited pheromone trails so that ants can forget bad decisions they had taken previously. $\Delta\tau_{ij}^k$ is defined as given in Eq. (4.4), where C_k is the cost of the solution T_k built by the k-th ant.

Based on the equation Eq. (4.4), ants that construct better solutions, deposit more pheromone on the edges they have traversed. So the edges that lead to minimum costs and used by many ants receive more pheromone, so they are more likely to be selected in future iterations.

$$\Delta\tau_{ij}^k = \begin{cases} 1/C_k & \text{if } edge(i,j) \in T_k \\ 0 & \text{otherwise} \end{cases} \tag{4.4}$$

4.1 Ant System

The AS algorithm [29] implements the basic ACO procedure detailed above in this section. The following sections explain the differences between the other ACO variants used in the experiments in this thesis and the AS algorithm.

4.2 Elitist Ant System

The main idea of EAS [28] is to provide additional reinforcement to the edge pairs which belong to T_{bs}, the best solution found since the start of the algorithm. This additional reinforcement of solution T_{bs} is achieved by adding a quantity e/C_{bs} to its edges, where e defines the weight given to the best-so-far solution T_{bs}, and C_{bs} is its cost. The new equation for the pheromone deposit is given in Eq. (4.5).

$$\begin{aligned} &\tau_{ij} \leftarrow \tau_{ij} + \textstyle\sum_{k=1}^{m} \Delta\tau_{ij}^{k} + e\Delta\tau_{ij}^{bs} \\ &\text{where } \Delta\tau_{ij}^{bs} = \begin{cases} 1/C_{bs} & \text{if } edge(i,j) \in T_{bs} \\ 0 & \text{otherwise} \end{cases} \end{aligned} \tag{4.5}$$

where $\Delta\tau_{ij}^{k}$ is calculated as in Eq. (4.4). The pheromone evaporation of EAS is the same as it is in AS.

4.3 Rank-Based Ant System

The main idea of RAS [28] is to allow each ant to deposit an amount of pheromone which decreases with its solution rank. The ants are sorted in decreasing order according to the quality of the solutions they constructed. The amount of pheromone an ant deposits is weighted according to its rank r. In each iteration only the $(w-1)$ best-ranked ants and the ant which has constructed the best-so-far solution are allowed to deposit pheromones. The best-so-far solution has the largest weight w, while the r-th best ant of the current iteration contributes pheromones with a weight given by $max\ \{0, w-r\}$. The new pheromone deposit rule is given in Eq. (4.6) where C_r denotes the solution cost of r-th best ant.

$$\begin{aligned} &\tau_{ij} \leftarrow \tau_{ij} + \textstyle\sum_{r=1}^{w-1} (w-r).\Delta\tau_{ij}^{r} + w.\Delta\tau_{ij}^{bs} \\ &\text{where } \Delta\tau_{ij}^{r} = \begin{cases} 1/C_{r} & \text{if } edge(i,j) \in T_{r} \\ 0 & \text{otherwise} \end{cases} \end{aligned} \tag{4.6}$$

4.4 MAX-MIN Ant System

The MMAS has four major differences from AS [30].

- Only either the ant which found the best solution in the current iteration, or the best-so-far ant is allowed to deposit pheromones.

- Allowed range of pheromone trail values is limited to the interval $[\tau_{min}, \tau_{max}]$. This modification is implemented in MMAS because allowing only best-so-far or iteration-best ant to deposit pheromone may lead to a stagnation situation that is all ants construct the same solution so keeping pheromone trails between boundaries will prevent the excessive accumulation of pheromone trails of suboptimal solutions.

- Pheromone trail values are initialized to the upper limit to increase exploration in the beginning.

- Pheromone trails are initialized when diversity is lost or when no improvement occurs for a given number of consecutive iterations.

Pheromones are deposited on the edges according to Eq. **(4.3)** and Eq. **(4.4)** as in the AS, but the ant which is allowed to add pheromone may be either the best-so-far or the iteration-best. Commonly in MMAS implementations, both the iteration-best and the best-so-far update rules are used alternatively.

Pheromone update is managed as follows: in the beginning pheromone trails are initialized with the upper bound of pheromone limits (τ_{max}) so that initial search space is very explorative, when an ant constructs the complete solution, pheromone trails are evaporated by a small evaporation rate so the unvisited edges have bigger pheromone levels. This procedure makes the search space explorative. To increase the probability of selecting unsearched edges, pheromone trails are initialized when algorithm approaches to a stagnation situation or solution is not improved for a number of iterations [30].

4.5 Ant Colony System

ACS differs from AS in three main points [31].

- It has a modified action selection rule.

- Pheromone evaporation and pheromone deposit take place only on the edges belonging to the best-so-far solution.
- Each time an ant uses an edge (i, j) it removes some pheromone from the edge.

In ACS, with a probability q_0, an ant makes the best possible move based on the pheromone trails and the heuristic information, and with probability $(1 - q_0)$ it performs a biased exploration of the edges. This method is called pseudo-random proportional action choice rule (see Algorithm 2). The parameter q_0 modulates the degree of exploration performed by the ants.

Algorithm 2 Choosing next solution component

```
1: if random(0-1) < $q_0$ then
2:     choose best next
3: else
4:     choose next according to pseudo-random proportional action choice rule
5: end if
```

At the end of each iteration in ACS, the pheromone trails are updated according to Eq. (4.7). The pheromone trail update, both evaporation and new pheromone deposit, are implemented only for the edges belonging to the best-so-far solution.

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij} + \rho\Delta\tau_{ij}^{bs}, \quad \forall(i,j) \in T_{bs} \tag{4.7}$$

Here $\Delta\tau_{ij}^{bs} = 1/C_{bs}$ and ρ represents pheromone evaporation. In addition to the global pheromone update performed at the end of each iteration, in ACS, the ants also use the local pheromone update rule given in Eq. (4.8). They apply local pheromone update immediately after having used an edge (i, j) during solution construction.

$$\tau_{ij} \leftarrow (1-\xi)\tau_{ij} + \xi\tau_0 \tag{4.8}$$

Here ξ $(0 < \xi < 1)$, and τ_0 are two parameters. The value for τ_0 is set to be the same as the initial value for the pheromone trails. Experimentally, a good value for τ_0 was found to be $1/nC_{min}$, where n is the number of nodes and C_{min} is the cost of the trivial solution [31]. C_{min} is determined using the shortest path of each lightpath.

4.6 Best-Worst Ant System

BWAS differs from AS in three main points [32].

- While only the best-so-far ant is allowed to deposit pheromones, the worst ant of the current iteration subtracts pheromones on the arcs it does not have in common with the best-so-far solution
- Search diversification is achieved through frequently reinitializing the pheromone trails
- To further increase diversity, pheromone mutation is used [28].

The pheromone trail update rule of BWAS is based on the consideration that the best-so-far solution can perform a positive contribution of trails. Whereas the worst ant of the current solution is penalized to decrease the desirability of selecting the same nodes in the construction graph.

$$\tau_{ij} \leftarrow \tau_{ij} + \Delta\tau_{ij}^{bs}, \quad \forall (i,j) \in T_{bs} \tag{4.9}$$

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij}^{ws}, \quad \forall (i,j) \in T_{ws} \quad and \quad (i,j) \notin T_{ws} \tag{4.10}$$

The deposition of pheromone rule for the best-so-far ant is given in Eq. (4.9). The evaporation of pheromones on the edges visited by the worst-ant that are not common with the best-so-far is given in Eq. (4.10) where T_{ws} is the worst solution found since the start of the algorithm.

The pheromone trail mutation is used in BWAS to introduce diversity in the search space. Each row of the pheromone matrix is mutated with a probability of P_m by depositing or evaporating the same amount of pheromone based on the current iteration [32]. The pheromone mutation is given in Eq. (4.11).

$$\tau_{ij}^{'} = \begin{cases} \tau_{ij} + mut(it, \tau_{threshold}) & \text{if } a = 0 \\ \tau_{ij} - mut(it, \tau_{threshold}) & \text{if } a = 1 \end{cases} \tag{4.11}$$

where a is a random variable in $0,1$ and it is the current iteration, $\tau_{threshold}$ is the average pheromone trail on the edges visited by the best-so-far ant and $mut(.)$ is given in Eq. (4.12).

$$mut(it, \tau_{threshold}) = \frac{it - \tau_{threshold}}{Nit - it_r} \;.\; \sigma \;.\; \tau_{threshold} \tag{4.12}$$

with Nit being the maximum number of iterations and it_r being the last iteration when a restart was performed. The parameters $\tau_{threshold}$ and σ specify the maximum power of the mutation.

5. APPLICATION OF ACO TO THE VT MAPPING PROBLEM

ACO can be applied to the survivable VT mapping problem in a straightforward way. The VT mapping problem can be seen as a search for the best routing of lightpaths through physical links. Therefore, we use a solution encoding inspired from [16]. For this encoding, first, the shortest k paths between the end points of each lightpath are determined. Then, a solution candidate is represented as an integer string of length l, where l is the number of lightpaths in the VT. Each integer gives the index of the shortest path for the corresponding lightpath. These integers can take values between $[1..k]$ where k is the predefined number of shortest paths for the lightpaths.

The Figure 5.1 interprets the encoding used during solution construction. According to this encoding, the fourth shortest path is selected for the sixth lightpath. Similarly the third shortest path is chosen for the first lightpath. As the algorithm may lead to different solutions, ants route the lightpaths in a random order. For example, an ant may route the fifth lightpath and then the second lightpath and so on. The flexibility of selecting lightpaths in a random order may increase the number of feasible solutions because for a survivable solution if a lightpath can only be routed using a few shortest paths, routing this lightpath earlier may result in better solutions.

The following sections introduce ACO steps implemented in this thesis.

5.1 Construction Graph

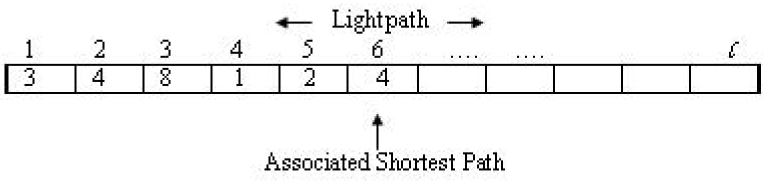

Figure 5.1: Solution encoding to survivable VT mapping.

The construction graph is identical to the physical topology. The physical topology is used as a graph on which ants travel and construct their solutions. Ants simultaneously try to route lightpaths on the graph one-by-one. Each starts to route a random lightpath. The shortest paths between the end points of the lightpaths are provided to ants at the very beginning of the algorithm. Ants determine one of the shortest paths of the selected lightpath while visiting the nodes of the shortest paths on the construction graph and checking if the chosen shortest path violates the constraints or not. If the solution becomes infeasible for a shortest path selected for the lightpath, another shortest path is examined. If none of the shortest paths leads to a feasible solution, this ant is removed from the colony.

5.2 Constraints

Survivable VT mapping problem has two constraints:

1) The number of wavelengths on a physical link should not exceed its capacity

2) All the lightpaths of a cut-set cannot be routed using the same physical link

The first limitation explains the capacity constraint while the second one introduces survivability constraint. To illustrate the second constraint assume that we have a logical topology as in Figure 5.2. According to this figure, the lightpaths 10, 8 and 3 cannot be routed using the same link in the physical topology because they are the cut-set of the virtual topology graph. Similarly, 6, 7, 2 and 10 cannot be routed through the same physical link.

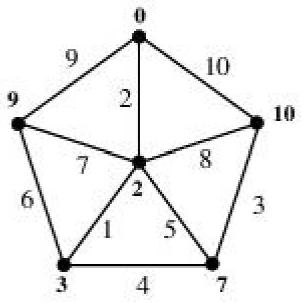

Figure 5.2: Example virtual topology.

5.3 Pheromone Trails and Heuristic Information

Ants decide their move on the construction graph based on heuristic and pheromone information. Two pheromone trails are implemented in the survivable VT mapping problem: the lightpath pheromone trails τ^l_{ij} refer to the desirability of choosing lightpath j directly after i for mapping, the shortest path pheromone trails τ^s_{ij} show the desirability of selecting j^{th} shortest path of lightpath i.

Pheromone is modelled as a 2D array which accumulates the information learned by the ants. The lightpath pheromone has lightpaths in its columns and rows and gives the information which lightpath is more valuable to choose after the current lightpath, whereas the shortest path pheromone has lightpaths in its rows and corresponding shortest paths in its columns and gives the information about which shortest path leads to the better result when selected for the current lightpath. These pheromones are initialized in the beginning of the algorithm with the same value for each possible choice of the ant. The difference is created by heuristic information η_{ij} that is inversely proportional to the length of the j^{th} shortest path of lightpath i i.e. $\eta_{ij} = 1/d_{ij}$. The ants use the lightpath pheromone to decide the next lightpath. The heuristic information is used together with the shortest path pheromone trails while deciding the proper shortest path of the chosen lightpath. A combined pheromone called total pheromone is used which is computed as τ^{α}_{ij} . η^{β}_{ij} for this responsibility. The pheromones are updated after solutions are constructed. The amount of the accumulated pheromone is proportional to the solution quality.

Algorithm 3 Global Pheromone Update

```
1: for each ant $k$ do
2:     for each lightpath $i$ chosen $j^{th}$ shortest path do
3:         $\tau^s_{ij} + \frac{1}{resource\ usage} -> \tau^s_{ij}$
4:     end for
5:     for each lightpath $i$ chosen before lightpath $t$ do
6:         $\tau^l_{it} + \frac{1}{resource\ usage} -> \tau^l_{it}$
7:     end for
8: end for
```

There are three different pheromone update procedures: local pheromone update, global pheromone update and global pheromone update weighted. In global pheromone update (see Algorithm 3), after each ant constructed its solution, both shortest path and lightpath pheromones are updated on the edges the ants visited. The pheromones are updated according to the solution quality. In this thesis, $\frac{1}{resource\ usage}$ is accumulated on pheromones where ***resource usage*** is the number of wavelength-links used by the corresponding ant.

The weighted global pheromone update (see Algorithm 4) differs from global update in the amount that is added pheromones. The pheromones are increased with the amount $\frac{weight}{resource\ usage}$ where weight is used to deposit more or less pheromone for the selected ants.

Algorithm 4 Global Pheromone Update Weighted

```
1: for each ant $k$ do
2:     for each lightpath $i$ chosen $j^{th}$ shortest path do
3:         $\tau^s_{ij} + \frac{weight}{resource\ usage} -> \tau^s_{ij}$
4:     end for
5:     for each lightpath $i$ chosen before lightpath $t$ do
6:         $\tau^l_{it} + \frac{weight}{resource\ usage} -> \tau^l_{it}$
7:     end for
8: end for
```

The local pheromone update algorithm that is used by ACS can be found in Algorithm 5. Every ant without regard to the solution quality updates pheromone using the parameters ξ and τ_0 where $0 \leq \xi \leq 1$ and τ_0 is the initial pheromone value. As ants perform pheromone update after each move when the whole solution is not created, unlike other five ACO algorithms, both feasible and infeasible solutions are allowed to update pheromone trails.

Algorithm 5 Local Pheromone Update

```
1: for each ant $k$ do
2:     for each lightpath $i$ chosen $j^{th}$ shortest path do
3:         $\tau^s_{ij} * (1-\xi) + \xi * \tau_0 -> \tau^s_{ij}$
4:     end for
5:     for each lightpath $i$ chosen before lightpath $t$ do
6:         $\tau^l_{it} * (1-\xi) + \xi * \tau_0 -> \tau^l_{it}$
7:     end for
8: end for
```

There are differences in pheromone update stages of ACO algorithms.

5.3.1 The pheromone update of AS

The pheromone update of AS is shown in Algorithm 6. In AS, every ant uses global update pheromone procedure [29].

Algorithm 6 AS Pheromone Update

```
1: for each ant do
2:     Global Pheromone Update
3: end for
```

5.3.2 The pheromone update of EAS

In pheromone update procedure of EAS (see Algorithm 7) in addition to AS, global pheromone update weighted procedure is used for the best-so-far ant. The weight is determined with the parameter e.

Algorithm 7 EAS Pheromone Update

```
1: for each ant do
2:     Global Pheromone Update
3: end for
4: for best-so-far ant do
5:     Global Pheromone Update Weighted
6: end for
```

5.3.3 The pheromone update of RAS

The pheromone update of RAS is shown in Algorithm 8. In RAS, weighted global update pheromone procedure is used with increasing weight for better solutions. The ant that constructs the best solution uses the weight w, the second best solution uses the weight $w-1$ and this update continues for the first w ranked ants.

Algorithm 8 RAS Pheromone Update

```
1: for each ant $k$ that has rank $\leq w$ do
2:     Global Pheromone Update Weighted
3: end for
```

5.3.4 The pheromone update of EAS

In pheromone update procedure of ACS (see Algorithm 9) only best-so-far ant is allowed to deposit pheromone. Evaporation is implemented at the same time of accumulation. Apart from this, every ant uses local pheromone update after each move as in Algorithm 5.

Algorithm 9 ACS Pheromone Update
```
1: for best-so-far ant do
2:     for each lightpath $i$ chosen $j^{th}$ shortest path do
3:         $\tau^s_{ij}*(1-\rho)+\frac{\rho}{resource\ usage} -> \tau^s_{ij}$
4:     end for
5:     for each lightpath $i$ chosen before lightpath $t$ do
6:         $\tau^l_{it}*(1-\rho)+\frac{\rho}{resource\ usage} -> \tau^l_{it}$
7:     end for
8: end for
```

5.3.5 The pheromone update of MMAS

The pheromone update of MMAS is shown in Algorithm 10. MMAS alternatively allows iteration-best ant, best-so-far ant or restart-best ant to deposit pheromone. Iteration-best ant is the ant that constructs the best solution in the current iteration. Best-so-far ant constructs the best solution since the start of the algorithm, that is the best of all iterations. MMAS initializes the pheromone trails when diversity is lost or when no improvement occurs for a given number of consecutive iterations. Restart-best-ant is the best solution constructed after this initialization. *u_gb* in Algorithm 10 is set as 2 to give the same chance to these three ants.

Algorithm 10 MMAS Pheromone Update
```
1: if iteration % u_gb then
2:     Global Pheromone Update for iteration-best ant
3: else
4:     Global Pheromone Update for best-so-far ant or restart-best ant
5: end if
```

5.3.6 The pheromone update of BWAS

The pheromone update of BWAS is shown in Algorithm 11. Global pheromone update is used by only best-so-far ant and the worst ant of the current iteration subtracts pheromones on the arcs it does not have in common with the best-so-far solution. When there are a few differences between the solutions of the best-so-far and iteration-worst-ant ants, the pheromone trails are reinitialized to increase search diversification. To further increase diversity, pheromone mutation is used as explained in [28].

Algorithm 11 BWAS Pheromone Update

```
1: for best-so-far ant do
2:     Global Pheromone Update
3: end for
4: for iteration-worst-ant do
5:     for each lightpath $i$ chosen $j^{th}$ shortest path do
6:         if this pair is not used by best-so-far ant then
7:             $\tau^s_{ij}*(1-\rho)$ -> $\tau^s_{ij}$
8:         end if
9:     end for
10:    for each lightpath $i$ chosen before lightpath $t$ do
11:        if this pair is not used by best-so-far ant then
12:            $\tau^l_{it}*(1-\rho)$ -> $\tau^l_{it}$
13:        end if
14:    end for
15: end for
16: Find distance between best-so-far ant and iteration-worst-ant
17: if distance $< 5$ % then
18:     restart the search by initializing pheromones and restart-best-ant
19: else
20:     mutate pheromones
21: end if
```

The algorithm flow of pheromone trail update is shown in Algorithm 12. Each ant, after constructing a solution, first evaporates pheromone trails on the visited edges and then calls the pheromone update procedure associated with the selected ACO algorithm. If MMAS is used, pheromone trail limits are checked and put in the bounds. Last, without regard to the selected algorithm, total pheromone trails are updates as $\tau^{\alpha}_{ij} \ . \ \eta^{\beta}_{ij}$.

Algorithm 12 Pheromone Trail Update

```
1: for each algorithm do
2:     evaporate pheromones except ACS
3:     call associated pheromone update procedure
4: end for
5: if MMAS then
6:     check pheromone trail limits
7: end if
8: for each algorithm do
9:     compute total pheromone as $\tau^{\alpha}_{ij} \ . \ \eta^{\beta}_{ij}$
10: end for
```

5.4 Solution Construction

Each ant is initially placed on a randomly chosen start lightpath and one of its shortest paths is selected. At each step, the ant iteratively adds an unvisited lightpath to its partial solution and decides the shortest path of the selected lightpath. The solution construction terminates once all lightpaths have been visited.

Algorithm 13 Solution Construction

```
1:  for each ant do
2:      place ant on randomly selected lightpath
3:      choose random shortest path for the selected lightpath
4:  end for
5:  while step $< n - 1$ do
6:      step ++
7:      for each ant do
8:          move to next step
9:          if ACS then
10:             local acs pheromone update
11:         end if
12:     end for
13: end while
14: for each ant do
15:     Pheromone Trail Update
16: end for
```

Solutions are constructed by applying the following simple constructive procedure to each ant:

(1) choose a start lightpath and one of its shortest paths,

(2) use lightpath pheromone information to select the next lightpath to route,

(3) use shortest path pheromone information together with the heuristic values to probabilistically determine the path between the nodes of the corresponding lightpath, until all lightpaths have been visited. If the ant cannot select a shortest path that makes the solution feasible, this ant is removed from the current iteration.

The algorithm flow of solution construction is shown in Algorithm 13.

Moving next step is implemented using pseudo-random proportional action choice rule. According to this rule, each lightpath is assigned a probability proportional to the lightpath pheromone. Cumulative probability is calculated and a random point is selected in this probability array. The corresponding lightpath is selected. The shortest path for the selected lightpath is chosen using the same way but total pheromone is used instead of lightpath pheromone. Algorithm 14 shows the pseudo-random proportional action choice rule.

Algorithm 14 Pseudo-random proportional action choice rule

```
sum_prob = 0
for each lightpath $i$ do
    if visited then
        prop_ptr[i]=0
    else
        prop_ptr[i]= lightpath_pheromone[current lightpath][i]
        sum_prob += prop_ptr[i]
    end if
    select randomly a point in sum_prob
    calculate the associated lightpath $l$
end for
sum_prob = 0
for each shortest paths $i$ of lightpath $l$ do
    if not feasible then
        prop_ptr[i]=0
    else
        prop_ptr[i]= total_pheromone[$l$][i]
        sum_prob += prop_ptr[i]
    end if
    if sum_prob = 0 then
        remove ant from colony
    else
        select randomly a point in sum_prob
        calculate the associated shortest path
    end if
end for
```

The following example can be used to summarize and illustrate the problem that is the main concern of this thesis.

Our objective is to minimize the total cost of resources used throughout the network. This cost is evaluated in two different ways: (1) by considering the actual lengths of the physical links (link-cost), and (2) by counting the number of physical links used (hop-count).

Table 5.1: Four different shortest paths for the lightpaths of the example virtual topology given in Figure 3.2.

	hop-count				**link-cost**			
lightpath	sp_1	sp_2	sp_3	sp_4	sp_1	sp_2	sp_3	sp_4
1-2 (a)	1-2	1-3-2	1-3-4-2	1-3-5-4-2	1-3-2	1-2	1-3-4-2	1-3-5-4-2
1-5 (b)	1-3-5	1-2-4-5	1-2-3-5	1-3-4-5	1-3-5	1-3-4-5	1-3-2-4-5	1-2-4-5
1-4 (c)	1-2-4	1-3-4	1-3-2-4	1-3-5-4	1-3-4	1-3-5-4	1-3-2-4	1-2-4
2-3 (d)	2-3	2-1-3	2-4-3	2-4-5-3	2-3	2-4-3	2-1-3	2-4-5-3
2-4 (e)	2-4	2-3-4	2-1-3-4	2-3-5-4	2-4	2-3-4	2-3-5-4	2-1-3-4
3-4 (f)	3-4	3-2-4	3-5-4	3-1-2-4	3-4	3-5-4	3-2-4	3-1-2-4
4-5 (g)	4-5	4-3-5	4-2-3-5	4-2-1-3-5	4-5	4-3-5	4-2-3-5	4-2-1-3-5

The constraints for the problem, i.e. the survivability and the capacity constraints, are explained in section 3.1. In order to determine if the solution is survivable or not, each physical link is deleted from the physical network one by one. If the VT graph becomes disconnected in the event of a broken physical link, the solution is considered as unsurvivable.

The following example illustrates the fitness evaluation techniques.

Consider the physical and virtual topologies given in Figure 3.2. The first 4 shortest paths calculated based on hop-counts and based on link-costs can be seen in Table 5.1. Here, the first column shows the lightpaths as source-destination node pairs. Four shortest paths found using hop-counts are given in the next four columns, and 4 shortest paths found using link-costs are given in the last four columns.

Assume we have an individual encoded as [1 2 1 3 1 1 2]. This encoding means that the first lightpath uses the 1^{st} shortest path (1-2), the second one uses the 2^{nd} shortest path (1-2-4-5), and the third one uses the 1^{st} shortest path (1-2-4), etc. If we sum up the number of wavelength-links used in this solution, we have a total of 12 wavelength-links for hop-count evaluation, and 2250 kilometers for link-cost evaluation.

6. EXPERIMENTAL STUDY

To compare the performance of ACO algorithms, we perform a series of experiments to calculate resource usage based on both hop-count and link-cost on 100 different instances each, for 3, 4, 5 connected virtual topologies. In these experiments, we used four metrics for performance evaluation, namely *success rate*, *first hit iteration*, *first hit time* and the *resource usage*. Success rate is defined as the percentage of program runs in which a survivable mapping that does not violate the capacity constraint is found. First hit iteration is the first iteration during which the best-so-far solution is encountered. First hit time is similarly, the first time when the best-so-far solution is encountered. When calculating shortest paths based on hop-count, each wavelength-link is considered to have a length of 1 for each physical link. On the other hand, when calculating the shortest paths based on the link-cost, each wavelength-link is considered to have a length equal to the actual length of the physical path in kilometers.

6.1 Experimental Setup

ACO algorithm specific parameters are determined after a series of tests. A sample of 20 VTs is selected between 100 VTs and MMAS algorithm is selected due to its better results among ACO algorithms for TSP in [28].

First, 15 shortest paths are provided to the algorithm and 5 connected VTs are used as data set for the problem to investigate the effect of maximum allowed time on resource usage. The largest search space is selected because maximum allowed time should be determined according to the problem that require the longest time. The results in Figure 6.1 shows the effect of maximum allowed time on resource usage when the shortest path calculation method is based on hop-count. We used hop-count method in our parameter tuning tests because unlike link-cost method, it gives the exact number of used physical links so when

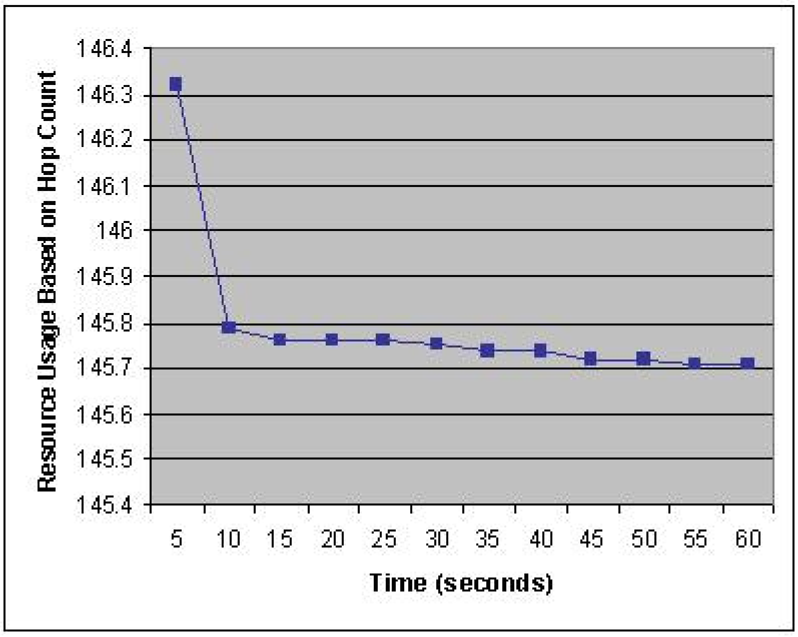

Figure 6.1: Effect of maximum allowed time on resource usage.

we compare two solutions based on hop-count we can easily calculate how much better one solution is than the other but in link-cost one solution may give a higher resource usage value by using just one more physical link. According to the Figure 6.1, algorithm does not need to be run more than 15 seconds because no valuable contribution is provided after 15 seconds.

Other parameters are determined using a data set of 4 connected 20 VTs and 10 shortest paths are provided. MMAS algorithm is used except the algorithm specific parameters. The effect of total number of ants used for solution construction is investigated using this sample data set. According to the Figure 6.2 increasing number of ants negatively affect resouce usage after 5 ants. Until maximum number of solutions are generated, ants use pheromone matrices to construct solutions. Pheromones are updated according to the solution quality after each ant completed their solution. When the maximum number of solutions is selected as 100 and 100 ants are used to construct the solutions, each ant will generate only one solution. They cannot use pheromone matrices as they will have updated the pheromones but will not use it because termination condition will be met. So when the number of ants decrease, the use of pheromone increases. We set the number of ants to 10 because it is a mid value and there is no distinct difference between 5 and 10 ants.

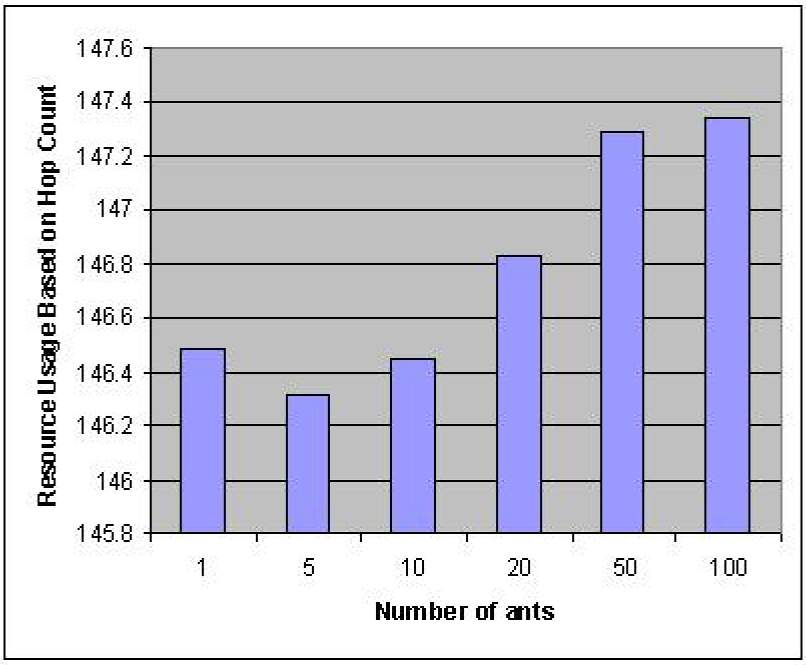

Figure 6.2: Effect of number of ants on resource usage.

The parameters α and β represent the weight of heuristic and pheromone values on total pheromone matrix. As it is explained in section 4, if $\alpha = 0$, the neighbor node that has the biggest heuristic information is selected, if $\beta = 0$, heuristic information is not used while deciding the next move, only pheromone is used. The effect of these two parameters when they are greater than 0 is also investigated. Table 6.1 shows the effect of α and β on resource usage. The parameters do not contribute much on resource usage after the values of 2. α does not seem to have much effect on resource usage. As pheromones are initialized with small values between 0 and 1, the increasing power of pheromones may not change the value significantly. Tests are done until the β values of 4 and α values of 3. The parameters are set when the best solutions are retrieved as $\alpha = 3$ and $\beta = 4$.

Table 6.1: Effect of β and α on resource usage

		β				
		0	1	2	3	4
α	0	221.67 ∓ 0.36	149.84 ∓ 0.35	148.79 ∓ 0.35	148.05 ∓ 0.34	147.44 ∓ 0.35
	1	203.52 ∓ 0.41	149.10 ∓ 0.34	147.86 ∓ 0.34	147.18 ∓ 0.34	147.01 ∓ 0.34
	2	203.56 ∓ 0.44	148.61 ∓ 0.35	147.40 ∓ 0.34	146.96 ∓ 0.34	146.63 ∓ 0.34
	3	203.76 ∓ 0.42	149.15 ∓ 0.36	147.05 ∓ 0.34	146.60 ∓ 0.34	146.48 ∓ 0.34

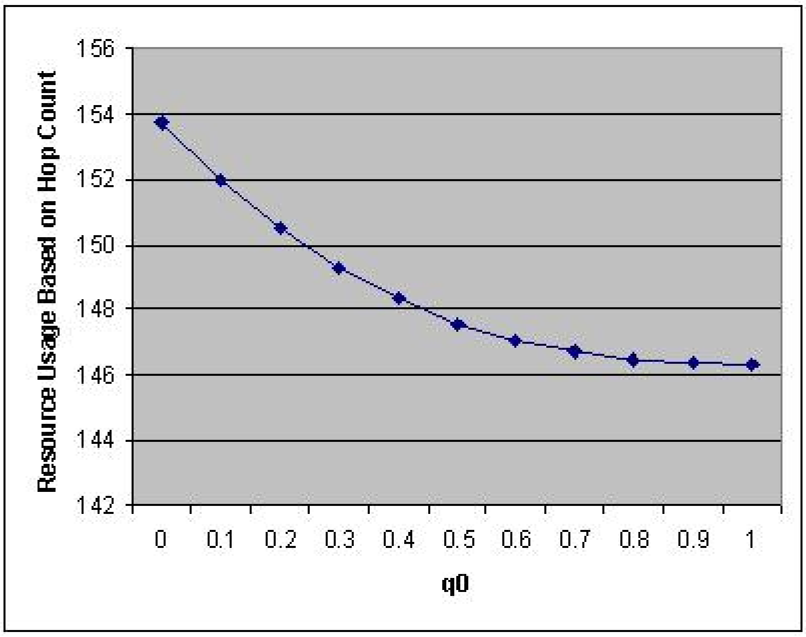

Figure 6.3: Effect of q_0 on resource usage.

The effect of q_0 is also investigated for the sample data set. q_0 is the parameter that is used to determine the next step of the ant. With q_0 probability ants select the best pheromone as a next step, while with $1 - q_0$ probability ants use the pseudo random proportional choice rule. Details can be found in section 4.5. According to the Figure 6.3, higher q_0 values lead better solutions. This parameter is set as 0.8 because random selection should not be ignored as it may lead to different solutions and increase diversity.

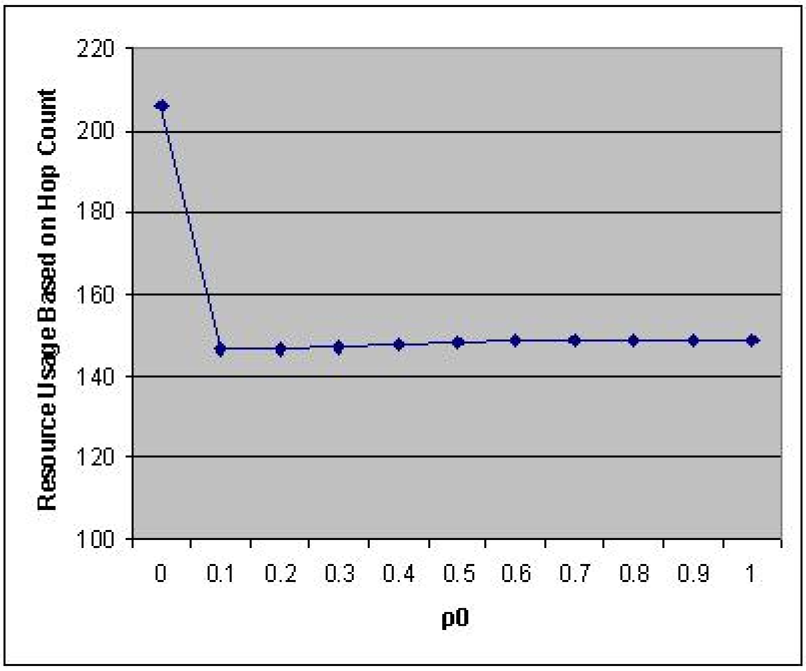

Figure 6.4: Effect of ρ_0 on resource usage.

ρ_0 is determined according to the Figure 6.4. ρ_0 is the parameter used while updating pheromone values. ρ_0 is selected as 0.1 where the best solutions are retrieved.

The effect of w is also investigated for the sample data set using RAS algorithm. w is determined according to the results in Figure 6.5. w is the maximum rank of ants that will deposit pheromone in RAS algorithm. w is selected as 5 where the best solutions are retrieved.

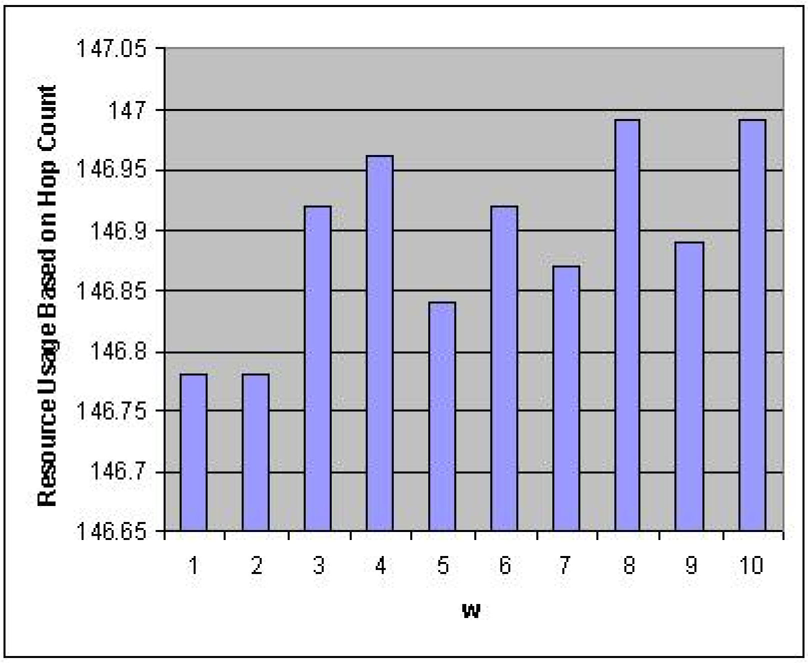

Figure 6.5: Effect of w on resource usage in RAS.

In MMAS, the branching factor for a lightpath i is defined as follows: if τ^i_{max} is the maximum and τ^i_{max} is the minimum pheromone trail value on edges incident to lightpath i; the branching factor is given by the number of edges incident to lightpath i that have a pheromone trail value $\tau_{ij} \geq \tau^i_{min} + \alpha(\tau^i_{max} - \tau^i_{min})$. The value of α ranges over the interval [0,1], while the values of the branching factors range over the interval [2, n-1], where n is the number of lighpath in the VT. The average branching factor is the average of the branching factors of all lightpaths and gives an indication of the size of the search space efectively being explored by the ants. If, for example the average branching factor is vey close to 5%, on average 5% of the lightpaths have a high probability of being chosen. In MMAS, when average branching factor reaches the lower limit, pheromone trails are initialized to increase diversity. Table 6.2 shows the effect of lower limit of branching factor on resource usage. The columns except the first one, show the time passed since the start of the algorithm. The rows represent the resource usage after each 5 seconds when the lower limit is selected as the first column. The lower limit of average branching factor is selected as 10% because the algortihm converges to better solutions earlier when the lower limit selected as 10%.

Table 6.2: Effect of lower limit of average branching factor on resource usage in MMAS

	5	10	15	20	25	30	35	40	45	50
0 %	146.81	146.27	146.24	146.22	146.21	146.2	146.19	146.18	146.17	146.16
10 %	146.78	146.25	146.20	146.20	146.19	146.19	146.18	146.15	146.14	146.13
20 %	146.80	146.29	146.25	146.23	146.22	146.21	146.20	146.17	146.15	146.15
30 %	146.82	146.27	146.25	146.24	146.24	146.22	146.19	146.16	146.14	146.14
40 %	146.85	146.3	146.27	146.26	146.25	146.24	146.22	146.18	146.15	146.15
50 %	146.86	146.29	146.25	146.24	146.23	146.22	146.21	146.16	146.15	146.14
60 %	146.83	146.27	146.23	146.22	146.22	146.21	146.19	146.16	146.15	146.14
70 %	146.82	146.28	146.24	146.23	146.23	146.22	146.21	146.18	146.16	146.16
80 %	146.81	146.27	146.23	146.21	146.20	146.2	146.18	146.16	146.15	146.14
90 %	146.77	146.30	146.25	146.22	146.21	146.21	146.20	146.16	146.15	146.14
100 %	146.79	146.28	146.25	146.24	146.23	146.22	146.21	146.18	146.17	146.16

The effect of e is determined according to the Figure 6.6. e is the weight of pheromone deposited for the best-so-far solution in EAS algorithm. e is selected as 3 where the best solutions are retrieved. In the local pheromone trail update

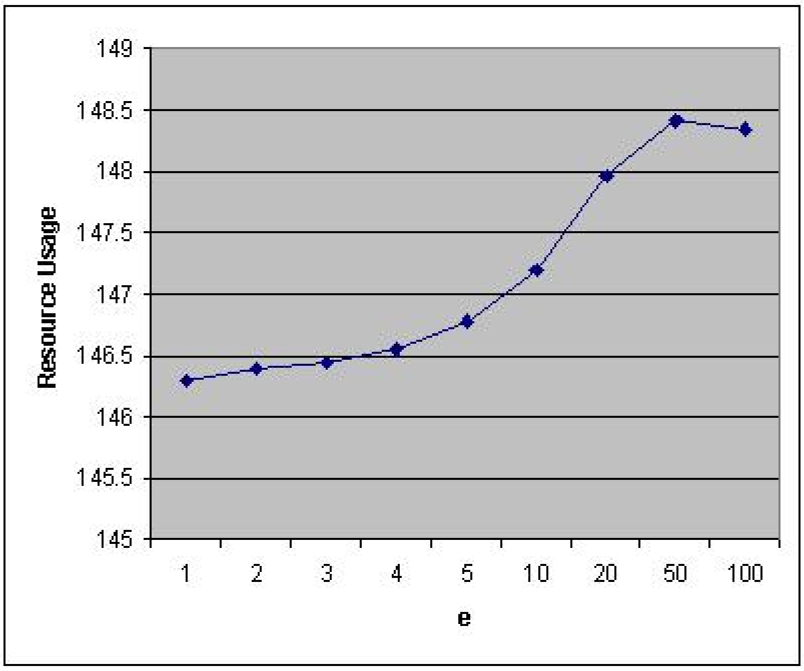

Figure 6.6: Effect of *e* on resource usage in EAS.

in ACS $\xi = 0.1$ and τ_0 is set as the default settings in [33] shown in Table 6.3.

Table 6.3: The settings of τ_0 per ACO algorithm

	AS-RAS-EAS	ACS-BWAS	MMAS
τ_0	$1/\rho C_{min}$	$1/nC_{min}$	$1/2n\rho C_{min}$

For the experiments, we use a physical topology with 24 nodes and 43 links (see Figure 6.7). We created 100 random VTs with average connectivity degrees of 3, 4, and 5 to map onto this physical topology. We assumed that each physical link has a capacity of 10 wavelengths.

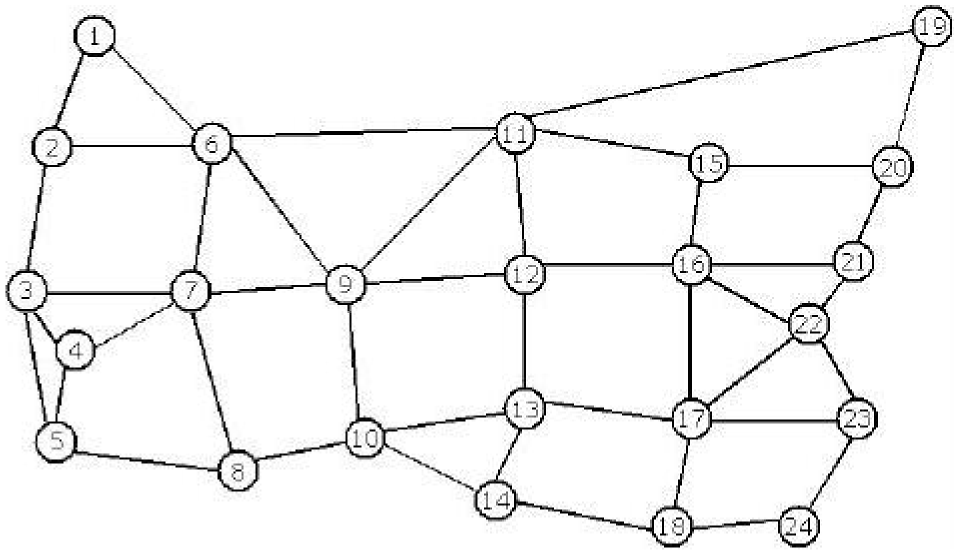

Figure 6.7: US wide 24-node 43-link physical topology.

In ACO performance tests, for all elements in the problem set, we run each algorithm 20 times. Each run is allowed to continue for 15 seconds.

6.2 Experimental Results

We present the results of the experiments in Tables 6.4, 6.5, 6.6, and 6.7. Table 6.4 shows the success rates of all ACO algorithms for randomly generated 100 VTs. We have 3 different sets of topologies where the average node degrees are 3, 4, and 5. For each algorithm and node degree, 3 different numbers of alternative shortest paths for lightpaths are examined. We tried 5, 10, and 15 shortest path cases. Both hop-count and link-cost results for each case are given in the table.

The success probability numbers are averaged over 2000 runs (20 runs per VT instance).

Table 6.4: Success rates for 24-node network

		5 shortest paths		10 shortest paths		15 shortest paths	
		hop count	link cost	hop count	link cost	hop count	link cost
as	3	0.946	0.857	0.988	1	1	1
	4	0.963	0.915	0.98	1	1	1
	5	0.992	0.795	0.998	0.979	1	0.984
ras	3	0.988	0.881	0.998	1	0.999	1
	4	0.975	0.955	0.999	1	1	1
	5	0.996	0.851	1	0.992	1	0.997
eas	3	0.979	0.882	0.997	1	0.999	1
	4	0.982	0.942	0.999	1	1	1
	5	0.998	0.824	1	0.995	1	0.996
mmas	3	0.974	0.915	0.999	1	1	1
	4	0.987	0.977	0.996	1	1	1
	5	0.999	0.862	1	0.971	1	0.986
bwas	3	0.976	0.889	0.998	1	1	1
	4	0.987	0.941	0.998	1	1	1
	5	0.999	0.852	1	0.994	1	0.996
acs	3	0.854	0.704	0.979	0.997	0.999	0.999
	4	0.939	0.871	0.979	1	1	1
	5	0.986	0.709	0.998	0.977	1	0.984

Table 6.4 shows that ACO algorithms can successfully be used for the survivable VT mapping problem. Success rates are mostly greater than 99% when more than 5 shortest paths are provided to the algorithms. Although algorithms have relatively the same performance, MMAS is the best of all when 5 shortest paths are utilized. It is followed by BWAS, RAS, and EAS. They have relatively the same performance. ACS is the worst of all, and AS has slightly better performance than ACS.

From Table 6.4, we can see that success rates are increasing with the number of shortest paths because the probability of finding feasible potential mappings increases with the number of alternative shortest paths. When ants can use 15 shortest paths, each algorithm finds feasible solutions with success rates of almost 100%. Hop-count based calculation results in better success rates when 5 shortest paths are provided to each ACO algorithm. For 10 shortest path cases, when the virtual topology has an average node degree of 3 and 4, link-cost calculation is more successful than hop-count. For average node degrees of 5, hop-count success rates are greater than link-cost.

Table 6.5: Lower and upper bounds of resource usage in terms of link-cost with 95% confidence interval

		5 shortest paths	10 shortest paths	15 shortest paths
		lower - upper	lower - upper	lower - upper
as	3	110314 - 110940	110403 - 110973	110504 - 111072
	4	142246 - 142908	142393 - 143007	142426 - 143038
	5	180846 - 181750	182193 - 183007	182409 - 183231
ras	3	110008 - 110614	110831 - 111407	111071 - 111651
	4	142641 - 143273	143162 - 143794	143226 - 143860
	5	181913 - 182793	183835 - 184693	184105 - 184973
eas	3	110263 - 110869	110731 - 111305	110792 - 111370
	4	142644 - 143290	142744 - 143368	142718 - 143344
	5	181131 - 181983	183151 - 183997	183153 - 183995
mmas	3	109860 - 110442	110215 - 110781	110255 - 110821
	4	142071 - 142681	142248 - 142858	142485 - 143099
	5	180654 - 181496	182101 - 182915	183051 - 183887
bwas	3	109743 - 110337	110750 - 111318	111471 - 112043
	4	142289 - 142919	143801 - 144439	144445 - 145093
	5	181897 - 182773	184970 - 185844	185988 - 186882
acs	3	109738 - 110436	111136 - 111728	111532 - 112122
	4	143778 - 144518	144012 - 144654	144578 - 145222
	5	182790 - 183816	185675 - 186587	186473 - 187381

Table 6.5 and Table 6.6 show resource usage of all ACO algorithms for link-cost and hop-count respectively. Here 3, 4, and 5 on the second column, represent the average node degrees for randomly generated VTs, whereas 5, 10, and 15 are the number of shortest paths provided to ACO algorithms. Both tables show lower and upper bounds of resource usage for the proposed solutions to the VTs when the confidence interval is 95%.

When we analyse Table 6.5 and Table 6.6, we conclude that for each algorithm, resource usage does not increase in the same proportion as the number of shortest paths. For perturbative search algorithms, such as EAs, the search space increases exponentially when the number of alternative shortest paths increases. The problem becomes harder for these algorithms and their success rates drop.

However for ACO algorithms, the increase in search space size is linear. Therefore ACO algorithms are not affected significantly by the increased search space. Both Tables 6.5 and 6.6 show that when 5 or 10 shortest paths are provided to ACO algorithms, MMAS has the best performance whereas when 15 shortest paths are provided to ACO algorithms AS is the best of all. In 5 shortest path cases, BWAS is comparable to MMAS. As RAS, MMAS, AS and EAS do not decrease in performance, we can conclude that they can better react to the increase in the search space size.

Table 6.6: Lower and upper bounds of resource usage in terms of hop-count with 95% confidence interval

		5 shortest paths	10 shortest paths	15 shortest paths
		lower - upper	lower - upper	lower - upper
as	3	110.03 - 110.57	110.30 - 110.83	110.37 - 110.88
	4	143.53 - 144.12	143.56 - 144.14	143.58 - 144.16
	5	181.39 - 182.10	181.29 - 181.99	181.32 - 182.01
ras	3	109.90 - 110.47	110.35 - 110.87	110.67 - 111.19
	4	143.49 - 144.08	143.90 - 144.49	144.10 - 144.69
	5	181.53 - 182.23	182.13 - 182.84	182.09 - 182.81
eas	3	110.07 - 110.59	110.32 - 110.84	110.49 - 111.01
	4	143.46 - 144.04	143.68 - 144.27	143.77 - 144.36
	5	181.37 - 182.07	181.59 - 182.29	181.57 - 182.27
mmas	3	109.73 - 110.25	109.98 - 110.49	109.97 - 110.48
	4	143.33 - 143.92	143.35 - 143.93	143.48 - 144.06
	5	181.06 - 181.75	181.21 - 181.91	181.65 - 182.35
bwas	3	109.92 - 110.44	110.38 - 110.90	111.22 - 111.73
	4	143.46 - 144.04	144.26 - 144.85	145.01 - 145.61
	5	181.55 - 182.25	182.74 - 183.46	183.62 - 184.36
acs	3	110.28 - 110.86	111.21 - 111.78	111.42 - 111.94
	4	144.86 - 145.52	145.48 - 146.13	145.70 - 146.32
	5	183.20 - 183.99	184.37 - 185.16	184.62 - 185.37

As can be seen in Tables 6.5 and 6.6 the performance of ACS is the worst of all as resource usage is relatively higher than the other algorithms. The main difference between ACS and the other algorithms is the local pheromone update where each ant decreases the pheromone trail in each step when a lightpath and the shortest path is chosen. Local pheromone update helps ants explore new solutions, but it does not work well in VT mapping problem. For the cases in which the ξ parameter used in local pheromone update is chosen as $\xi \geq 0.1$, the evaporation

Table 6.7: Average first hit iterations

		5 shortest paths		10 shortest paths		15 shortest paths	
		hop-count	link-cost	hop-count	link-cost	hop-count	link-cost
as	3	24.15	42.65	12.83	15.76	7.75	13.14
	4	15.22	24.57	8.09	10.86	5.52	8.94
	5	9.36	22.09	6.07	9.98	5.06	8.22
ras	3	19.20	51.75	16.22	29.81	11.92	20.54
	4	10.78	30.24	8.37	16.10	6.56	11.17
	5	7.97	22.24	6.14	11.98	5.04	8.91
eas	3	8.50	20.48	6.96	13.13	6.36	10.28
	4	5.26	13.44	4.54	8.31	4.28	6.88
	5	4.05	10.97	4.19	7.12	4.10	6.48
mmas	3	14.56	30.91	10.60	17.07	9.52	14.40
	4	9.41	19.89	9.40	12.94	8.76	10.87
	5	7.84	18.04	8.92	10.68	7.52	8.61
bwas	3	22.97	46.16	25.33	39.96	21.55	29.65
	4	19.19	36.28	16.56	18.12	11.24	12.78
	5	16.42	30.63	12.55	13.84	7.94	9.17
acs	3	28.28	46.65	19.05	24.01	12.31	16.47
	4	18.69	29.68	11.50	13.73	7.23	9.43
	5	11.71	20.15	7.47	9.60	5.24	7.41

of pheromones after each step, may make ants forget the heuristic information. The effect of ξ parameter should also be investigated to increase the performance of ACS algorithm.

From Table 6.6, we can conclude that although MMAS has the smallest resource usage values, all algorithms have relatively the same performance. There is no significant difference between the resource usage values of the implemented algorihtms for 4 connected VTs. Table 6.7 shows the first hit iterations in which the best solutions are retrieved. Average of the first hit iterations are given based on both hop-count and link-cost. When we compare the results for link-cost and hop-count we can see that best solutions are found in earlier iterations when the hop-count calculation method is used. Hop-count calculation method also results in better success rates not only for 5 connected virtual topologies, but also in 5 shortest path cases (see Table 6.4). We can see from Table 6.7 that EAS finds its best solution in earlier iterations than the other algorithms. From Table 6.7, we can conclude that BWAS finds its best solution in the latest iteration. The reason may be the pheromone mutation procedure. When BWAS cannot find a better solution after a predefined number of iterations, the pheromones are updated and different solutions are explored. The best solution may be found after this stage.

Table 6.8: Average and standard error of first hit times

			5 shortest paths		10 shortest paths		15 shortest paths	
			hop count	link cost	hop count	link cost	hop count	link cost
acs	3	a	7.00	11.10	8.01	11.54	8.08	11.12
		e	0.12	0.11	0.10	0.08	0.10	0.08
	4	a	7.32	12.35	8.43	11.53	8.09	10.97
		e	0.12	0.08	0.11	0.08	0.11	0.09
	5	a	8.07	12.81	8.62	11.98	8.94	12.86
		e	0.11	0.09	0.11	0.09	0.12	0.11
as	3	a	2.22	4.33	2.86	4.92	3.18	5.89
		e	0.07	0.09	0.07	0.08	0.06	0.09
	4	a	2.38	5.46	3.02	6.15	4.03	7.09
		e	0.07	0.10	0.06	0.09	0.06	0.08
	5	a	2.57	7.00	3.97	8.35	5.72	9.74
		e	0.06	0.11	0.05	0.09	0.06	0.08
bwas	3	a	3.78	7.55	6.83	10.98	8.07	11.21
		e	0.08	0.09	0.10	0.08	0.10	0.08
	4	a	4.36	8.93	6.46	8.29	6.66	8.23
		e	0.09	0.08	0.09	0.08	0.09	0.08
	5	a	5.67	10.50	7.30	9.51	7.18	8.77
		e	0.10	0.08	0.10	0.08	0.09	0.08
eas	3	a	2.24	5.73	3.15	6.44	4.21	6.97
		e	0.06	0.10	0.07	0.09	0.08	0.09
	4	a	2.32	6.60	3.38	7.04	4.79	8.06
		e	0.06	0.10	0.06	0.09	0.06	0.08
	5	a	2.81	7.81	4.87	9.10	7.05	11.47
		e	0.06	0.10	0.06	0.09	0.07	0.09
mmas	3	a	2.79	6.86	4.00	8.09	6.08	9.65
		e	0.07	0.10	0.06	0.08	0.07	0.07
	4	a	2.60	7.65	5.95	10.77	9.67	12.70
		e	0.05	0.09	0.06	0.07	0.08	0.06
	5	a	4.07	9.80	9.70	13.19	12.69	14.94
		e	0.05	0.09	0.07	0.06	0.09	0.07
ras	3	a	3.11	8.54	4.30	8.21	4.44	7.79
		e	0.09	0.11	0.10	0.10	0.09	0.10
	4	a	2.96	8.60	3.72	7.62	4.12	7.25
		e	0.09	0.10	0.09	0.10	0.08	0.09
	5	a	3.34	9.10	4.09	8.53	4.81	8.68
		e	0.09	0.10	0.08	0.09	0.08	0.09

Table 6.8 shows the first hit times in which the best solutions are retrieved. The time unit is shown as seconds. Average (a) and the standart error(e) of the first hit

times are given based on both hop-count and link-cost. Similar to the Table 6.7, the best solutions are found earlier when the hop-count calculation method is used. We can see from Table 6.8 that AS finds its best solution earlier than the other algorithms. It is not suprising because AS has the basic procedures. As the quality of the solutions found by AS is as good as the solutions found by the other algorithms (see Tables 6.5 and 6.6), in time critical systems, AS can be used to gather good solutions in a short time. MMAS, ACS and BWAS finds their best solutions later than the other algorithms. EAS and RAS do not need much time to retrieve their best solutions. As their qualities are comparable to the other algorithms, they can be used in time critical systems confidently.

6.3 Discussion

As a summary of the experiments, we recommend ACO algorithms for the survivable VT mapping problem due to their decision policy at each step. They find feasible solutions after each iteration. To increase the performance and the success rates, as many shortest paths as necessary should be used. Based on the results, even though all ACO algorithms perform well, we can recommend MMAS with hop-count calculation method due to its overall success and better reaction to the increasing search space. MMAS finds its best solutions later than the other algorithms when 15 shortest paths are provided. In time critical systems, AS can be used because it converges to its best solutions earlier than the other algorithms but from Table 6.4, we can conclude that success rates of AS is not as good as the other algorithms so in 15 shortest path cases, to decrease time utilized by MMAS, EAS and RAS can be used reliably to retrieve best solutions in a short time with high success rates.

We also performed a series of experiments to compare the performance of the EA and ACO for the survivable VT mapping problem. In these experiments, we used two metrics for performance comparisons: success rate, and resource usage. For the experiments, we used the same physical topology as we did to compare the ACO variants(see Figure 6.7). We created 50 random VTs with average connectivity degrees of 3, 4, and 5. We assumed capacity of physical link as 10 wavelengths.

Table 6.9: Success rates retrieved by ACO and EA

		5 shortest paths		10 shortest paths		15 shortest paths	
		EA	ACO	EA	ACO	EA	ACO
hop count	3	0.26	0.72	0.55	0.92	0.59	0.97
	4	0.87	0.94	0.95	1	0.97	1
	5	0.97	0.98	1	1	1	1
link cost	3	0.19	0.60	0.53	0.93	0.60	0.98
	4	0.84	0.88	0.94	0.96	0.97	1
	5	0.98	0.83	1	1	1	1

Here, we used the EA approach and the parameter set with the best performance from [18]. In the EA performance tests, we considered a mutation probability of $1/l$, where l is the number of lightpaths, a crossover probability of 1.0 and a population size of 100. A penalty factor of 200 is used in the tests using hop count for shortest path calculation, and 300 in the tests using link cost.

The termination criterion for both algortihms is to create a predefined number of points in the solution space. We applied separate tests to each algorithm and determined this number as the iteration count after which there is no improvement in solution quality. As a result, we decided this number to be 5000 for the EA, and 100 for the ACO. We should note that each run of the programs take less than a minute on the average. We performed 20 runs for each experiment.

The results of the experiments are given in Tables 6.9, and 6.10. Table 6.9 shows the success rates of both heuristics averaged over 1000 runs (20 runs per VT instance). For each algorithm and VT connectivity degree, we examined three different numbers of alternative shortest paths for each lightpath. We used 5, 10, and 15 shortest paths calculated according to hop count and link cost. Table 6.10 shows the lower (l) and upper (u) bounds for the resource usage of both algorithms with 95% confidence interval calculated using only the results of the successful runs.

A quick observation of Table 6.9 shows that success rates for both algorithms are very high. Generally ACO achieves higher success rates. The experiments using hop count calculation method performs better in terms of success rate for both heuristics. In Table 6.9, we can see that success rates increase with the increase

Table 6.10: Lower and upper bounds of resource usage with 95% confidence interval for ACO and EA

			5 shortest paths		10 shortest paths		15 shortest paths	
			EA	ACO	EA	ACO	EA	ACO
hop count	3	l	113.24	111.53	115.15	115.88	117.48	116.99
		u	114.86	112.48	116.39	116.80	118.84	117.91
	4	l	146.8	146.54	150.84	154.51	153.9	156.82
		u	147.64	147.31	151.7	155.41	154.78	157.78
	5	l	185.64	187.01	192.32	197.18	197.86	201.55
		u	186.8	188.19	193.53	198.42	199.17	202.85
link cost	3	l	115775	111175	115733	116329	118524	117838
		u	117751	112507	117104	117391	119954	118934
	4	l	147272	147319	151183	154017	154928	156137
		u	148379	148317	152166	155061	156528	157223
	5	l	200687	189479	199330	199969	203287	204412
		u	209674	191047	204847	201535	207230	205966

in the number of alternative shortest paths. In EA tests; for 3 connected VTs, success rates for 10 shortest paths are more than twice the success rates for 5 shortest paths, however, the increase of success rate from 10 shortest paths to 15 shortest paths is not that high. This is because of the exponential growth of the search space.

The probability of random candidate solutions being survivable increases with the connectivity degree of the VT. Therefore, for higher connectivity degrees of VTs, both algorithms have higher success rates. However, the EA has a much smaller success rate than ACO for 3 connected VTs.

Table 6.10 shows that the resource usage increases slightly with the increase in the number of alternative shortest paths. This is an expected result, since, the probability of getting stuck at local optima is higher in larger search spaces and both algorithms are allowed to run up to a predefined maximum time. If the run times are increased, the resource usage results for different number of shortest paths will converge. Even though success rates in Table 6.9 are higher for ACO on 4 and 5 connected VTs, Table 6.10 shows that the solution quality of EA is better.

High success rates show that both heuristics are promising for the survivable VT mapping problem. ACO performs better for sparse VTs both according to success rate and resource usage. However, for dense VTs, ACO gives better success rates whereas EA gives better quality results. Since the time needed to find a feasible solution is less than a minute, these heuristics can easily be applied to real world applications.

7. CONCLUSION

The aim of this thesis is to route lightpaths on a physical topology in such a way that the capacity constraints of the links are not violated and in case of a physical link failure, the VT is not disconnected when all the lightpaths routed through this link are deleted from the VT. Survivable VT mapping is an optimization problem with the objective of minimizing the resource usage that is the number of wavelength used in the whole physical topology.

VT mapping problem has been the area of interest to many studies. However most of them did not consider the survivability constraint. Their objective is to minimize the number of wavelength used without violating capacity constraints. The studies that take survivability constraint into account mostly did not consider capacity constraints and retrieved solutions in a long time. In this study, we have implemented six ACO algorithms and considered not only survivability constraint but also capacity constraints of the links in the physical topology. Another important contribution of this study is that remarkable solutions are retrieved in quarter of a minute. Unlike other algorithms, the user does not need to wait for a long time to obtain good solutions. Moreover, the implemented algorithms does not decrease in performance when the search space is getting bigger.

Our results show that ACO algorithms can be successfully used for the survivable virtual topology mapping problem. When we compare our results with EAs implemented in [18], we see that if we run the algorithms for the same amount of time, we obtain lower resource usage with higher success rates using ACO. Overall, the results are promising and promote further study to improve the ACO performance.

As future work, we will examine the effect of other ACO specific parameters such as σ, τ_0, P_m and the number of shortest paths on the performance of the algorithms. We will also use classical optimization algorithms to compute lower bounds of the solutions to better asses the quality of our results.

The future work includes also testing the algorithms on different data. Moreover, the graph structures of the data sets can be examined and a relation between the graph structure and best variation to route the lightpaths can be set up.

REFERENCES

[1] **Mukherjee, B.**, 2006. Optical WDM Networks, Springer, New York, USA.

[2] **Modiano, E. and Narula-Tam, A.**, 2002. Survivable Lightpath Routing: A New Approach to the Design of WDM-Based Networks, *IEEE Journal on Selected Areas in Communications*, **20-4**, 800–809.

[3] **Agrawal, G.P.**, 2002. Fiber-optic communication systems, John Wiley, New York, USA.

[4] **Bates, R.**, 2001. Optical Switching and Networking Handbook, McGraw-Hill, New York, USA.

[5] **Kazovsky, L.G.**, **Benedetto, S. and Willner, A.E.**, 1996. Optical Fiber Communication Systems, Artech House.

[6] **Tomlinson, W.J. and Lin, C.**, 1978. Optical wavelength-division multiplexer for the 1-1.4-micron spectral region, *Electronics Letters*, **14**, 345–347.

[7] **Shtainhart, A., R., S. and Tsherniak, A.**, 1999. Wavelength Division Multiplexing, www2.rad.com/networks/1999/wdm/wdm.htm.

[8] **Chlamtac, I.**, **Ganz, A. and Karmi, G.**, 1992. Lightpath Communications: An Approach to High-Bandwidth Optical WAN's,, *IEEE Transactions on Communications*, **40**, 1171–1182.

[9] **Ou, C.S. and Mukherjee, B.**, 2005. Survivable Optical WDM Networks, Springer.

[10] **Todimala, A. and Ramamurthy, B.**, 2004. Survivable virtual topology routing under shared risk link groups in WDM networks, First International Conference on Broadband Networks (BROADNETS'04), San Jose, CA, USA.

[11] **Armitage, J.**, **Crochat, O. and Le Boudec, J.Y.**, 1997. Design of a Survivable WDM Photonic Network, Sixteenth Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '97), Kobe, Japan.

[12] **Nucci, A.**, **Sanso, B.**, **Crainic, T.**, **Leonardi, E. and Marsan, M.A.**, 2001. Design of Fault-Tolerant Logical Topologies in Wavelength-Routed Optical IP Networks, Proceedings of IEEE Globecom.

[13] **Todimala, A. and Ramamurthy, B.**, August 2007. A Scalable Approach for Survivable Virtual Topology Routing in Optical WDM Networks, *IEEE Journal on Selected Areas in Communications*, **25-6**, 63–69.

[14] **Ducatelle, F. and Gambardella, L.M.**, 2004. A Scalable Algorithm for Survivable Routing in IP-Over-WDM Networks, *First International Conference on Broadband Networks (BROADNETS'04)*, **25**, 54–63.

[15] **Kurant, M. and Thiran, P.**, 2004. Survivable Mapping Algorithm by Ring Trimming (SMART) for Large IP-over-WDM Networks, First International Conference on Broadband Networks (BROADNETS'04).

[16] **Banerjee, N. and Sharan, S.**, 2004. An EA for Solving the Single Objective Static RWA Problem in WDM Networks, International Conference on Intelligent Sensing and Information Processing.

[17] **Saha, M. and Sengupta, I.**, 2005. A GA Based Approach for Static Virtual Topology Design in Optical Networks, IEEE Conference and Exhibition on Control, Communication and Automation (Indicon).

[18] **Ergin, F.**, **Yayımlı, A. and Uyar, Ş.A.**, 2009. An Evolutionary Algorithm for Survivable Virtual Topology Mapping in Optical WDM Networks, EvoWorkshops09, Tubingen, Germany.

[19] **Varela, G.N. and Sinclair, M.C.**, 1999. Ant Colony Optimisation for Virtual-Wavelength-Path Routing and Wavelength Allocation, Proceedings of the 1999 Congress on Evolutionary Computation.

[20] **Garlick, R.M. and Barr, R.S.**, 2002. Dynamic Wavelength Routing in WDM Networks via Ant Colony Optimization, *Springer-Verlag*, **3**, 250–255.

[21] **Ngo, S.**, **Jiang, X. and Horiguchi, S.**, 2006. An Ant-Based Approach for Dynamic RWA in Optical WDM Networks, *Photonic Network Communications*, **11**, 39–48.

[22] **Ngo, S.**, **Jiang, X.**, **Le, V. and Horiguchi, S.**, 2006. Ant-based survivable routing in dynamic WDM networks with shared backup paths, *Journal of Supercomputing*, **36**, 297–307.

[23] **Hassan, A.**, **Phillips, C. and Pitts, J.**, 2007. Dynamic Routing and Wavelength Assignment using Hybrid Particle Swarm Optimization, The Engineering and Physical Sciences Research Council (EPSRC), PGNet2007.

[24] **Bullnheimer, B.**, **Hartl, R. and Strauss, C.**, 1999. An improved ant system algorithm for the vehicle routing problem, *Annals of Operations Research*, **89**, 319–328.

[25] **Gambardella, L., Taillard, I. and Dorigo, M.**, 1999. Ant colonies for the quadratic assignment problem, *Journal of Operational Research Society*, **50**, 167–176.

[26] **Gagne, C., Price, W. and Gravel, M.**, 2002. Comparing an ACO algorithm with other heuristics for the single machine scheduling problem with sequence-dependent setup times, *Journal of the Operational Research Society*, **53**, 895–906.

[27] **M, D. and T., S.**, 2002. The ant colony optimization metaheuristics: algorithms, applications, and advances, *International Series in Operations Research and Management Science*, **57**, 251–85.

[28] **Dorigo, M. and Stützle, T.**, 2004. Ant Colony Optimization, Cambridge Massachusetts.

[29] **Dorigo, M., Maniezzo, V. and Colorni, A.**, 1996. Ant System: Optimization by a colony of cooperating agents, *IEEE Transactions on Systems, Man, and Cybernetics-Part B*, **26**, 29–41.

[30] **Stützle, T. and Hoos, H.H.**, 2000. MAX-MIN Ant System, *Future Generation Computer Systems*, **16**, 889–914.

[31] **Dorigo, M. and Gambardella, L.M.**, 1997. Ant Colony System: A cooperative learning approach to the traveling salesman problem, *IEEE Transaction on Evolutionary Computation*, **6**, 317–365.

[32] **Cordon, O., de Viana, F., Herrera, F. and Moreno, L.**, 2000. A New ACO Model Integrating Evolutionary Computation Concepts: The Best-Worst Ant System, ANTS'2000 - From Ant Colonies to Artificial Ants: Second International Workshop on Ant Colony Optimization, Brussels, Belgium.

[33] **Stützle, T.**, 2004. Ant Colony Optimization Source Code, http://www.aco-metaheuristic.org/aco-code.

CURRICULUM VITAE

Candidate's full name: Elif KALDIRIM

Place and date of birth: Bayburt, 1985

Universities and
Colleges attended:
B.Sc.: Computer Engineering Department, ITU, 2006
B.Sc.: Industrial Engineering Department, ITU, 2007

Publications:

- **Kaldırım, E., Ergin-Corut, F., Uyar, A. Ş., Yayımlı, A.**, 2009: Ant Colony Optimization for Survivable Virtual Topology Mapping in Optical WDM Networks. *24th International Symposium on Computer and Information Sciences, ISCIS 2009*, Guzelyurt, Northern Cyprus.