6

Intelligent Simulation for Computer Aided Design of Optical Networks[1]

I. Chlamtac[1], M. Ciesielski[2], A. Fumagalli[3], C. Ruszczyk[4], and G. Wedzinga[5]

[1] University of Texas at Dallas,
Erik Jonnson School of Engineering and Computer Science, Dallas, USA
E-mail: chlamtac@utdallas.edu

[2] University of Massachusetts at Amherst,
Department of Electrical and Computer Engineering, Amherst, USA
E-mail: ciesiel@esc.umass.edu

[3] Politecnico di Torino,
Dipartimento di Elettronica, Torino, Italy
E-mail: fumagalli@polito.it

[4] Boston University,
Department of Electrical, Computer and Systems Engineering, Boston, USA
E-mail: ruszczyk@bu.edu

[5] National Aerospace Laboratory NLR,
Avionics Department, Amsterdam, The Netherlands
E-mail: wedzing@nlr.nl

Summary

CATO (CAD Tool for Optical networks and interconnects) is a prototype tool for designing LAN/MAN packet-switching optical communication systems. CATO is a knowledge based tool that integrates Artificial Intelligence (AI) techniques and event driven simulation, for optimizing system cost and performance. CATO provides the user with an Optical Device Library (ODL), that contains functional and cost characteristics of the optical devices that can be used to design the optical system. The tool also supplies a user friendly Graphical User Interface (GUI) module, which allows easy specification, evaluation and optimization of the communication system under investigation. Taking into account user defined performance requirements and operational constraints of the optical communication system, CATO searches for the optimal design solution, consisting of the network topology, and the set of devices that minimize the overall system cost. By combining the device library, with AI and event driven simulation, to the best of our knowledge CATO provides the first iterative and automatic optimization procedure for the design of optical communication systems.

1. Introduction

Increasing communication speed requirements by applications such as video-on-demand and image library retrieval have created a great interest in very high speed networks and interconnects. Optical and all-optical solutions are being sought due to the expected Gigabits per second data transfer rates in these systems. Because more and more sophisticated photonic devices, such as Wavelength Division Multiplexing (WDM) add/drop multiplexers, crossconnects, and switches are being brought onto the market, the possibility for implementing and operating such optical high speed communication systems has become a reality. The design of these systems is a highly complex and challenging task, which cannot be carried out without the support of sophisticated Computer Aided Design (CAD) tools.

[1] This research was in part supported by the NSF under Grant Numbers NCR-9596242 and NCR-9628189.

Optical Network Design and Modelling H. van As & A. Jukan (Eds.)

In order to appreciate the usefulness of a CAD tool in the design of an optical system, let us describe the design process of an optical network as carried out today. The design process starts out by defining the overall system requirements. In the case of an optical network, this includes the type of communication medium, number of nodes, distance to cover, etc. Next, a network architecture and protocol are selected, and the topology of the network is laid out. At this point the network subsystems (e.g., nodes) are drafted, and appropriate optical devices are selected for each subsystem. Device experts play an important role in this selection process; the selection of the devices is done based on device characteristics provided by manufacturers in the form of data sheets, diagrams, etc.

The next step is to evaluate the system design using conventional simulation tools. The designer has to create a model of the system and supply it to the simulator. The level of detail of such a model may depend on the current state of the design, or the issues that are being investigated. But nevertheless, the model has to be complete in the sense that it should reflect all the relevant design decisions taken so far. Based on the performance results returned by the simulator, the designer has to evaluate the consequences of the design decisions. If the results are unsatisfactory, the designer has to make new design choices, modify the system's model and resubmit it for evaluation to the simulator. This process of iteratively making design decisions, modeling the system, and running simulations may be extremely time consuming. Even though the final results may be satisfactory, the designer has no guarantee that he has reached an overall optimal design, in terms of system performance and cost.

Currently, the only way to test the system (and to make the final component choices) is to purchase the devices and to create a testbed. This approach can be very expensive considering the high cost of optical components. It is also extremely risky, since the system may not work according to the required specifications with a given selection of components. In this case a new set of devices may have to be purchased and the testbed constructed anew. This costly procedure may require several iterations until a satisfactory prototype system has been built.

The objective of the CATO (CAD Tool for Optical networks and interconnects) project is to provide the designer with an intelligent software tool for the specification, design, simulation and optimization of optical communication systems. CATO allows the user to enter a set of design constraints and, based on these constraints, the tool will automatically complete a design, using a library with specifications of available (or expected to be available) components. The tool then iteratively evaluates the design using simulation techniques and decides on design alterations, using Artificial Intelligence (AI) techniques, until an overall optimal design is obtained. An intelligent tool can thus automatically evaluate a much larger set of possible designs than a designer ever would be able to do manually. It should be noted that due to the complexity of the analyzed systems, which may have many local optima, it cannot be guaranteed that the optimal solution is always found. In addition, the model may not reflect the actual behavior of the system due to modeling approximations. For these reasons only a suboptimal system design will be achieved. Nevertheless, in the remainder of this paper we will continue to use the term optimal.

Once CATO has produced an optimal design, the designer can use CATO's simulation facilities to further investigate the behavior of the design, for example by using different traffic characteristics or by introducing fault conditions. CATO supplies comprehensive system models, enabling the simultaneous evaluation of various system aspects, such as signal transmission, behavior of devices, access protocols, etc., and their interaction. On the basis of the simulation results the designer may decide to further refine the system design. Such simulation activities may reduce the amount of effort needed in the testbed evaluation phase.

Initial design and testing of a CATO prototype (CATO-1) has been completed recently. Its objective was to prove the feasibility of combining AI techniques and event driven simulation techniques with an optical device library, and a user friendly graphical user interface into a flexible CAD tool for obtaining optimal WDM communication network designs in terms of system cost and performance. CATO-1 is a prototype tool for designing packet-switching optical communication systems using a

LAN/MAN ring topology as the underlying network topology. It is a knowledge based tool that incorporates artificial intelligence techniques into a Network Constraint Engine (NCE), for optimizing cost and system performance. The two specific AI algorithms incorporated to achieve this goal are Simulated Annealing (SA) and a Genetic Algorithm (GA). In particular, CATO-1 finds an optimal number of transceivers for each network node, using a cost function that includes the cost of the devices and the overall system performance.

The purpose of this paper is to present the results of the CATO-1 development. The paper is structured as follows. In section 2, a survey of existing software tools in support of optical system design is given. In section 3, the concept of CATO-1 is described in detail. Separate subsections are devoted to each of the CATO-1 software modules. Also, the interfaces between the modules and the general flow of operation are described in separate subsections. In section 4 some typical results obtained with CATO-1 are presented. In section 5 conclusions from the CATO-1 development are drawn and directions for further work are given. Some background information on the applied AI algorithms is given in appendix A.

2. Survey of Existing Software Tools for Optical System Design

An optical communication system can be viewed as comprising of three layers:

- The **Network layer**, which refers to protocols, switching and routing mechanisms, control, management, fault monitoring and recovery, etc.
- The **Transmission layer**, which deals with issues related to signal transmission, including device factors that affect transmission, such as noise, crosstalk, dispersion, etc.
- The **Device (or physical) layer**, which refers to modeling and characterization of optical/ optoelectronic devices, and device technology.

Typically, existing design tools for optical systems are suitable for designing a single layer only. Therefore, the design of a communication system has to be carried out at each layer separately, essentially independently of the other layers. In this process for the design of one layer, basic requirements coming from other layers need to be considered, typically as fixed parameters. This methodology is not likely to lead to the optimal design of an all-optical network in terms of system cost and performance [1, 2].

The only commercial tools for optical design available on the market today are those for traditional ray-based imaging, 3-D lens modeling, and generic optomechanical designs. Well-known examples of such tools are LightTools and CODE V of Optical Research Associates (ORA). The popularity of those tools is a testimony to the fact that the field of classical optical and optomechanical modeling is mature, the design methods are well understood and hence CAD support is required for its numerous applications.

As a result of collaboration between the University of Illinois, ARPA and IBM, a prototype computer-aided design tool, iFROST, has been developed for mixed level modeling, simulation and analysis of data links and parallel fiber-optics buses [3]. It allows the user to specify the topology of the link or bus, component parameters, and then it interactively performs waveform simulations and analyses. Its application, however, is limited to single link or bus, and does not consider network configurations with switches and multi-wavelength (WDM) applications. IBM at T.J. Watson Center [4], Bellcore (MAGIC broadband design software), and others, are involved in some aspects of simulation, but only on the network layer. Conversely, a number of companies, such as AT&T, Corning, AMP, and Optikwerks, are involved in device modeling. Optikwerks, for instance, develops optical laser design software.

The Photonics Systems Group at NRC (National Research Council of Canada), is working on the development of simulation of fiber-optic communication links (Optical SPICE), guided-wave photonics (field distribution and propagation), and free-space photonics design tools for VLSI (Very Large Scale Integration) chip interconnects. Another academic research tool, OPALS (Optoelectronic, Photonic, and Advanced Laser Simulator) comes from the University of Melbourne, Australia [5]. In the United States, optical CAD research is also being carried out at George Washington University, Georgia Tech, the University of Colorado at Boulder, UC San Diego, and the University of Pittsburgh [6] using ad-hoc simulation tools as needed. The above research concentrates mainly on fiber-optics simulation and free-space optoelectronic design. Other examples of specialized academic tools include an optical interconnection cube simulator at the University of Arizona [7]; design of holographic memories for optical interconnection [8]; and tools for optoelectronic packaging [9]. The University of Colorado at Boulder is working on a number of tools to aid in the design of their optical systems: in particular a tool for system simulation in multi-hop optical networks; a general purpose simulator Xhatch, to aid in the design at the physical/device system level [10]; CAD tools for thermal modeling of the devices and modules, for the optical coupling between optical devices and optical fibers; and for the attachment of chips using soldering techniques.

Many general purpose network-level, protocol-oriented, simulators have been in existence since the late seventies. The list of such tools is too long to describe here in detail. Currently some of these tools are starting to incorporate protocols for optical networks. For a detailed description of the various university efforts in simulation of networks the interested reader is referred to recent conferences on simulation and networking.

In summary, there is currently no effort in the development of CAD tools for all-optical networks, or WDM communication systems, providing knowledge based design and specification that makes it possible to cross the boundaries between the various layers of network abstraction. Such CAD tools should be able to integrate a working model of the adjacent layers into the model of the layer being designed. Physical characteristics of devices from the device layer need to be considered in modeling, simulation and optimization of the transmission layer. The signal transmission model in turn must be used in the design of the network layer. This can be made practical by creating a single software tool that deals with the issues of the three layers, and an optical device library to provide a complete data base with specifications of optical components.

3. Description of CATO-1

As stated in the introduction of this paper, CATO-1 (the initial CATO prototype) combines AI techniques and simulation techniques with an optical device library and a graphical user interface to obtain optimal WDM ring network designs. To allow for a parallel development and to improve maintainability, CATO-1 is decomposed into the following four software modules:

- Network Simulation Engine (NSE).
- Network Constraint Engine (NCE).
- Optical Device Library (ODL).
- Graphical User Interface (GUI).

These modules and their (possible) interrelationships are depicted in Figure 1. The Network Simulation Engine (NSE) is a network simulator that implements a multi-channel WDM backbone ring network using a token passing protocol to co-ordinate channel access. Each node can have one or more transceivers, and amplifiers can be inserted in the links between nodes as needed. The Network Constraint Engine (NCE) is an optimization module that operates on the network specification and the simulation results to produce an optimal design. The NCE incorporates two different AI optimization algorithms: Simulated Annealing and the Messy Genetic Algorithm. (The user can select the algorithm to use.) The Optical Device Library (ODL) serves as a data base of optical devices that

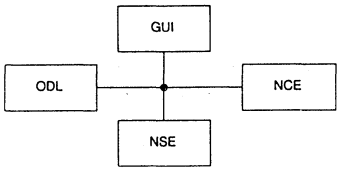

Figure 1: CATO-1 architecture

can be used in the network under design. The device specifications given in the ODL are used by both the NSE and the NCE, thereby avoiding duplicated storage of information. The user can add, modify, and delete devices in the ODL. The Graphical User Interface (GUI) allows the user to input parameters to the NSE and the NCE, and to update the ODL. The GUI displays all the results from the NSE and the NCE to the designer. In addition, the GUI is the control part of CATO-1; it controls the activation of the NCE and the NSE, and controls the communication between the NSE and the NCE by passing data between the two modules.

Sections 3.1 through 3.4 describe each of the CATO-1 modules in more detail. Section 3.5 summarizes the information exchanged between the CATO-1 modules and section 3.6 describes the operation flow of CATO-1.

3.1 Graphical User Interface

The Graphical User Interface (GUI) carries out the interaction between a user and CATO-1. Based on windows and pull-down menu's, this interface provides the user with a simple and intuitive way to enter the network optimization constraints, and the simulation parameters. The GUI provides a graphical representation of changes in the network under investigation. (Such changes may be initiated manually by the user, or automatically by the NCE.) The GUI also allows the user to operate on the device library.

The user gains control of the CATO-1 modules through a control window. This window contains a top level menu and a canvas screen. The top level menu offers the user a set of options for specifying the network architecture and the simulation parameters, for accessing the device library and for controlling the design activities. The canvas is used for monitoring the position of the network components. Network nodes are represented by circular shaped objects marked with the assigned node number. Amplifiers are represented by circular shaped objects marked with an "A". Fiber links are represented by straight interconnecting lines. A GUI window (without network layout) is shown in Figure 2.

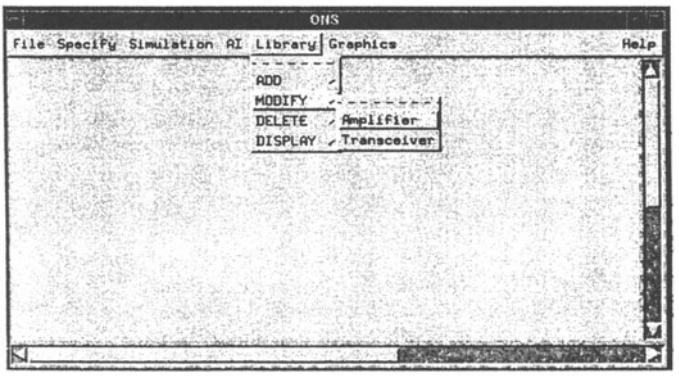

Figure 2: A GUI window

The graphical user interface of CATO-1 was designed using the Tcl/Tk programming language [11]. Tcl (Tool command language), developed by Ousterhout [12] and first released in 1989, is an interpreted language designed to be used as a common extension and customization language for applications. The Tk toolkit, first released in 1991, is a Tcl extension which provides a Tcl interface to the X Window System. It provides an easy way to build a graphical user interface to an application. Tcl/Tk provides a number of functions and widgets which simplify the design of a mouse driven graphical user interface. In addition, there are widget libraries for Tcl/Tk that extend the capabilities of the language. Since Tcl/Tk is designed using the C/C++ programming languages, it is inherently easy to integrate with C/C++ based application programs.

3.2 Optical Device Library

The Optical Device Library (ODL) serves as a data base of optical devices that can be applied in the network under design. Currently, the ODL contains only two types of devices: transceivers and amplifiers, but other types of devices can easily be added. Each transceiver consists of two parts: a transmitter and a receiver, each separately tunable on any of the available wavelength channels. That is, a transceiver can receive and transmit simultaneously on two different wavelengths. Their tuning time is assumed to be zero, i.e., they are instantaneously tunable, and their characteristics such as power, and noise values are assumed to be the same for all wavelengths. Some transceiver parameters and their typical values are listed in Table 1 (transmitter parameters) and Table 2 (receiver parameters).

Table 1: Transmitter parameters

Parameter	Typical value
Transmitting wavelength	1530 - 1560 nm
Maximum transmit power	+10 dBm
Long term wavelength stability	±1 nm
Price	4 kUS$

Table 2: Receiver parameters

Parameter	Typical value
Reception wavelength	1530 - 1560 nm
Sensitivity	-30 dBm
Maximum power	0 dBm
Equivalent noise input current	8 pA/√Hz
Price	3 kUS$

Currently, amplifiers are modeled as a simple signal boosting device. An amplifier has a flat gain over the wavelengths being amplified. The gain model for the amplifiers is:

$$\frac{P_{in}}{P_{sat}} = \frac{1}{G-1} ln\left(\frac{G_{max}}{G}\right), \tag{1}$$

where P_{in} is the total input power across all wavelengths to the amplifier, P_{sat} is the saturation power, G is the actual gain achieved, and G_{max} is the maximum small signal gain. Table 3 lists amplifier parameters and their typical values.

Table 3: Amplifier parameters

Parameter	Typical value
Frequency range	1530 - 1560 nm
Sensitivity	-30 dBm
Maximum small signal gain	+20 dB
Maximum total output power	0 dBm
Saturation power	-10 dBm
Noise figure	6 dB
Price	20 kUS$

3.3 Network Simulation Engine

The Network Simulation Engine (NSE) models an all-optical WDM multi-channel backbone ring LAN/MAN, based on SCM-NET [14]. The ring network consists of M nodes interconnected by means of unidirectional fiber optic links. Optical amplifiers may be incorporated in the links to compensate for power losses due to the fiber links and the nodes themselves. An example of a six node ring with one amplifier is shown in Figure 3. Using wavelength division multiplexing, N data

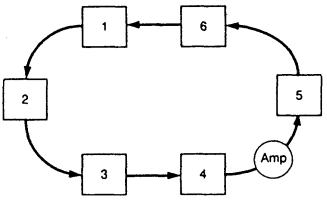

Figure 3: Example of a 6 node ring

channels of 1.2 Gbps plus one control channel of 100 Mbps are created on the fiber links. The maximum number of nodes is currently set to 15 and the maximum number of channels to 8, reflecting present day technology constraints. As technology evolves these maxima can be easily increased in the model. Each of the M nodes on the ring is equipped with K transceivers ($1 <= K <= N$), depending on for instance the node's traffic requirements. In addition to transmitting and receiving, nodes have the capability to relay data channels and to strip data channels. The channel relay function is needed to provide all-optical connections between arbitrary source/destination combinations and the channel stripping function is needed to remove data from the channel (after it has made a complete trip around the ring). The control channel is used to provide access to the data channels. The access protocol is based on a multi-token extension of the FDDI token passing protocol [15], with the extension that a node is allowed to hold multiple tokens at the same time. The control channel itself operates as a slotted ring [16], where slots of fixed length are either empty, or contain a token. The architecture of a node is shown in Figure 4. In this architecture, the Node Manager performs the token passing protocol function and controls the operation of the channel stripping function and the tuning of the transceivers to the data channels.

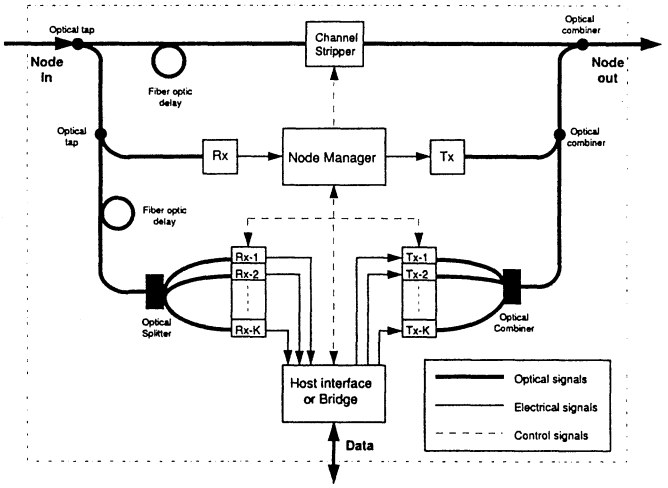

Figure 4: Node architecture

The control channel carries a number of tokens equal to the number of data channels, i.e., N. Each token has a length of 4 bytes and contains the data channel identification, the status bit ("free" or "busy"), the destination address, the source address, and the acknowledgment (ACK) bit. The reception of a free token indicates that the associated data channel is available for data transmission and the reception of a busy token indicates that the associated data channel is being used for data transmission. The operation of the token passing protocol is shown (slightly simplified) in Figure 5.

Every node with a packet to transmit waits for a free token on the control channel. Whenever a free token is received, the node checks whether it has a free transmitter, and if so, tunes the transmitter to that channel and puts the intended destination address in the token to indicate to the remote node that a packet is coming on that particular channel. The node starts transmitting after the captured token has been transmitted. The source node releases a free token after completion of transmission of the packet.

Whenever a node receives a busy token, it checks whether it is intended for the node itself. If so, it checks whether it has a free receiver, and it tunes the receiver to the wavelength of the channel

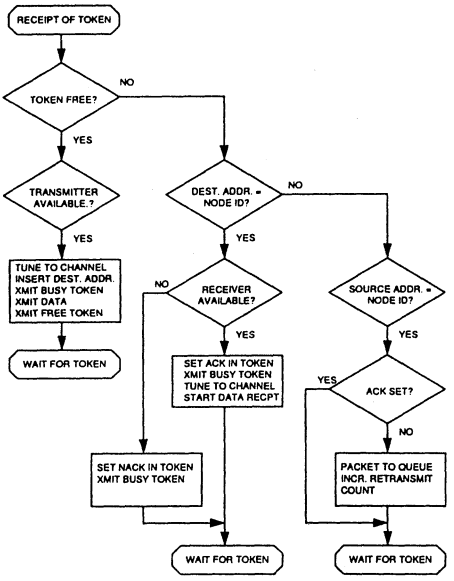

Figure 5: Token passing protocol

specified in the token, turns the ACK bit on in the token and releases the token. If the node does not have a free receiver, it turns off the ACK and releases the token. When the token has traveled back to the source node, it checks whether ACK has been set or not. If there is no ACK, the source moves the packet back onto the queue of packets to be transmitted. Otherwise, it removes it from its buffers. Finally, the returned busy token is purged.

A packet might fail to reach its destination due to two reasons:

- The destination has no free receiver to tune.
- The optical power in the communication link is inadequate or the Signal to Noise Ratio (SNR) at the receiver end is too low.

In the first case, the packet will be retransmitted as the node will not give an ACK. In the second case, the node will give an ACK (which has been given as soon as the receiver is tuned before starting to receive). The NSE records the number of packets that failed to reach their destination, along with the reason for failure of reception.

The NSE is designed as an event-driven simulator. Every event which results in a change of the overall network state is modeled, and the simulation runs by serially executing these events in temporal order. The NSE can also be used as a stand-alone simulation tool for evaluating the performance of the optimized system under different operating conditions, e.g., various arrival rates.

3.4 Network Constraint Engine

The Network Constraint Engine (NCE) is responsible for optimizing the network specification that has been entered by the user. Basically, it determines an optimal number of transceivers for each node on the ring network, based on its input from the simulator (NSE) which provides the failure rate of packet reception at each node of the network. As described in the previous subsection, there are two possible scenarios for failure of packet reception: 1) unavailability of a receiver and 2) inadequate optical power in the communication link or low SNR at the receiver. Based on the number of packets lost due to unavailability of a receiver, the NCE can assign additional transceivers at the individual nodes in order to reduce the retransmit count (and hence the average packet delay time). The NCE could use the number of packets lost due to low power, or low SNR to place amplifiers on the ring. Although all the necessary provisions within the GUI, ODL, and NSE modules are available, this last functionality is however not yet implemented in the NCE.

The NCE can use two different AI algorithms for transceiver placement, i.e., Simulated Annealing (SA) [17] and a Genetic Algorithm (GA) [18], selectable by the user. Appendix A provides a brief introduction to these two algorithms. SA and GA were chosen in favor of other optimization methods such as neural networks and fuzzy logic, because no training with existing data sets is required. The SA algorithm implemented in the NCE is the Adaptive Simulated Annealing (ASA) algorithm developed by Ingber [19], which is made publicly available under a GNU copying license. The ASA code was first developed in 1987 as Very Fast Simulated Reannealing (VFSR) [20]. In 1993 many adaptive features were developed, leading to the present ASA code. ASA is now used world-wide across many disciplines. Other examples of the application of SA in the design of communication networks are presented in [21, 22]. The GA implemented in the NCE is the Messy GA (MGA)

developed by Goldberg, Korb, and Deb [23]. The most important difference between MGA and traditional GA is that MGA uses variable-length chromosomes that may be over- or under specified with respect to the problem being solved. In addition, MGA uses simple cut and splice operators instead of fixed-length crossover operations (see appendix A). NCE uses the MGA coded in the C language [24].

To optimize the number of transceivers, the NCE calls the selected AI algorithm to optimize a cost function. This cost function is based on the information that was passed from the simulator (i.e., the number of packets lost due to unavailability of receivers). The cost function associated with each AI algorithm is:

$$A * (\%\ lost\ packets) + B * (total\ \#\ of\ transceivers / X), \tag{2}$$

where A and B are (currently) fixed weights and

$$X = M*N \quad (1 <= N <= 8,\ 2 <= M <= 15), \tag{3}$$

equals the maximum total number of transceivers allowed in the system under study. M is the number of nodes in the system and N is the number of data channels. Variable X biases the optimal solution towards a higher number of transceivers, when more data channels are available. Equation (2) is optimized under the constraints:

$$\#\ of\ transceivers\ for\ node\ i <= N \quad (1 <= i <= M) \tag{4}$$

In the current implementation, the NCE optimizes the number of transceivers using only the device types provided by the designer, e.g., it does not intermix different types of transceivers. Optimization over different types of devices has to be done manually by the designer, by selecting in successive iterations different device types and finally choosing the device types for which the lowest cost function is achieved.

3.5 Information exchange

Table 4 gives summary descriptions of the information exchanged between the CATO-1 modules.

Table 4: Information exchanged between the CATO-1 modules

From	To	Information transferred
GUI	NSE	Network specification, including number of nodes, distances between nodes, number of data channels, number of transceivers at each node, fiber specifications, and device specifications. Simulation parameters, including traffic characteristics, total number of packets to be transmitted and initial token locations.
NSE	GUI	Simulation results, including the number of lost packets for each node, positions in the ring where power or SNR dropped below an acceptable level, average queue lengths, average message delays, and the network throughput.
GUI	NCE	Identification and prices of devices that are used in the network. Simulation results.
NCE	GUI	Final value of the cost function for the optimal network. Number of transceivers at each node. Dollar cost of the network.

The exchange of information between the NSE, the NCE, and the GUI is accomplished through predefined ASCII formatted data files. Since the GUI controls the interactions, files are always passed between modules via the GUI. The formats of the files are know to the GUI and can be read

by the GUI as well. On user demand, the GUI displays the information from these files in predefined windows.

3.6 Flow of operation

The typical flow of operation in the design process is depicted in Figure 6; it shows the usual cycling through the CATO-1 states. Three operational states are distinguished within CATO-1, i.e., the GUI state, the NSE state and the NCE state. Each of these states is now briefly described.

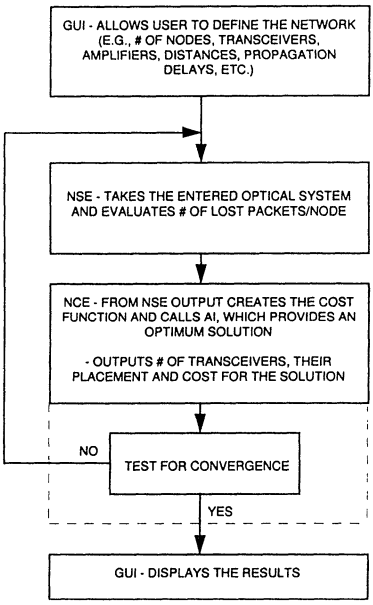

Figure 6: CATO-1 flow of operation

GUI state. In this state, CATO-1 allows the designer to define a network specification. The inputs to be specified include: number of nodes, distances between nodes, number of transceivers at every node, transceiver type, number of amplifiers and their positions on the ring, amplifier type, the number of packets generated during simulation, and the method of optimization to be used. An output file is generated, including all the information required by the NSE to run a simulation.

NSE State. In this state, the NSE receives the input data from the GUI and the simulation is run. The simulation will stop once the total number of packets specified in the GUI state have been transmitted. An output file is generated containing the results of the simulation. The number of packets lost at every node is presented in tabular format, together with an indication why the packets were lost.

NCE State. The file generated during the NSE state is used as input by the NCE. Based on this file and on the method of optimization specified in the GUI state, an AI algorithm is executed. After a number of iterations an output file containing the number of transceivers to be used at every node is generated. The results are presented to the designer in tabular format.

4. Results

CATO-1 has been programmed in the C programming language. It consists of over 7400 lines of code (excluding the AI algorithms), divided over the modules as follows: GUI: 3400 lines, NCE: 2000 lines, NSE: 1600 lines, and ODL: 400 lines.

A number of network design experiments were carried out with CATO-1 on a Silicon Graphics Indy, with R5000 processor and 64 Mbyte of RAM. In terms of the network cost and design the ASA and MGA produced similar results. The main difference was the execution time. For a particular test case MGA required about 2 hours to attain the results while ASA took only about half a minute. It should however be noted that MGA achieved 90% of its final result in about the same time that ASA executed. MGA and ASA performed consistently the same for different network configurations and traffic characteristics. Ingber also made a comparison study between SA (VFSR) and GA [25]. For a suite of six standard test functions he found that VFSR performed orders of magnitude more efficient than GA. These results are however not conclusive enough to discard MGA for future CATO developments. In CATO-1 only one type of transceiver is used throughout the system, resulting in fairly simple constraints. This may have caused the cost function evaluation to be weighted in favor

of ASA over MGA in terms of performance. If an intermix of devices was allowed, the complexity of the constraints would increase and this could effect the comparative performance of the two algorithms. An exhaustive re-examination of the two AI methods under a variety of design problems is therefore required for future CATO developments.

5. Conclusion

The development of an initial CATO (CAD Tool for Optical networks and interconnects) prototype proved the *feasibility* of intelligent CAD software for designing optical networks and interconnects considering performance aspects of both network and transmission layers. This first prototype (CATO-1) is a tool for designing packet-switching optical communication systems using a LAN/MAN ring topology as the underlying network. It incorporates Artificial Intelligence techniques for optimizing system performance and cost. The two specific algorithms incorporated to achieve this goal are Adaptive Simulated Annealing (ASA) and Messy Genetic Algorithm (MGA).

The tool supplies a user friendly Graphical User Interface (GUI) module which allows easy specification, evaluation and optimization of the communication system under consideration. It also provides the user the capability to maintain a simple Optical Device Library (ODL), such that devices can easily be added, modified, or deleted. CATO-1 also contains a Network Simulation Engine (NSE), which allows the Network Constraint Engine (NCE) to provide an iterative and automatic optimization procedure of the communication system design. The NCE is responsible for determining the optimal number of transceivers at each node in the network. It takes its input from the NSE which provides the number of packets lost at each network node due to unavailability of transceivers and places additional transceivers in order to minimize a cost function, which includes the cost of the transceivers and the number of lost packets.

The major conclusion regarding further needed research is that photonics is a rapidly growing field which affects many important applications, such as communications, networking and interconnections, data storage, image processing, etc. It is still a relatively new field, and thus lacking many of the design tools that are available to electrical engineers. Specifically there is a lack of tools on the market to support optical system designers with the specification, design, simulation, performance analysis and optimization of optical/photonics systems.

The goal for further research will be to fill this gap by providing a CATO-1 based software tool for *virtual prototyping* of photonics systems for these emerging applications. In particular, all-optical communication systems based on Wavelength Division Multiplexing (WDM) and multi-wavelength routing technology are among the most promising and fastest growing areas. The next versions of CATO will significantly reduce the time needed to design such photonics systems. Based on the design constraints provided by the user, CATO will automatically complete an optimal design, using mathematical programming techniques (linear, integer, and dynamic), intelligent search methods and AI. CATO will relieve the user of manually evaluating each design alternative, thereby allowing him/her to concentrate on the critical aspects of the system design. The simulation facility of CATO will enable the analysis of each system layer separately, but also the combined analysis of multiple system layers, and their interaction. We believe that this will reduce the amount of effort needed for constructing and evaluating a working testbed. Overall, the application of CATO is expected to significantly reduce the cost of designing photonics systems.

6. References

1. *Special Issue on Multiwavelength Optical Technology and Networks*, IEEE/OSA Journal of Lightwave Technology, Vol. 14, No. 6, June 1996.

2. *Optical Networks*, IEEE Journal on Selected Areas in Communications, Vol. 14, No. 5, June 1996.

3. Whitlock, B. K., *Computer Modeling and Simulation of Digital Lightwave Links Using iFROST, illinois FibeR-optic and Optoelectronic Systems Toolkit*, M.S. Thesis, University of Illinois at Urbana-Champaign, May 1993. (Web site: http://www.ccsm.uiuc.edu/people/whitlock/ifrost.html)

4. Kahn, C., Kershenbaum, A., *Design of All-optical Networks by Using a CAD Tool*, OFC, W03, 1995.

5. Gurney, P.C.R., and Lowery, A.J., *Opals - A New Computer Aided Learning Package for Photonics*, IEEE Conf. on Multi-Media Eng. Educ., Melbourne, Australia, July 1994, p115-123. (Web site: www.ee.mu.oz.au/papers/prl/VP/opalhtdg.htm)

6. Levitan, S., Marchand, P., et al., *Computer-Aided Design and Simulation of Free Space Optoelectronic Information Processing Systems*, University of Pittsburgh, Pittsburgh. (Web site: kona.ee.pitt.edu:80/steve)

7. Kostuk, R., *Optical Interconnects and Micro-optic System Design*, COEDIP Industrial Advisory Board meeting, University of Arizona, Toucson, May 9-10, 1996.

8. Gmitro, A., *Design, Fabrication, and Testing of Holographic Optical Interconnects*, COEDIP Industrial Advisory Board meeting, University of Arizona, Toucson, May 9-10, 1996.

9. Mansuripur, M., *Simulation Tools for Optoelectronic Packaging*, COEDIP Industrial Advisory Board meeting, University of Arizona, Toucson, May 9-10, 1996.

10. Sivesind, J., *XHatch Users Manual, Version 2.0*, Optoelectronics Computer System Technical Report 95-04, University of Colorado at Boulder, Boulder, CO. (Web site: http://www-ocs.colorado.edu)

11. Welch, B., *Practical Programming in Tcl and Tk*, Prentice Hall, 1995. (Web site: ftp://ftp.neosoft.com/pub/tcl/)

12. Ousterhout, J.K., *Tcl and the Tk Toolkit*, Addison-Wesley, 3rd printing, April 1994.

13. Green, P.E., *Fiber-Optic Networks*, Prentice Hall, Englewood Cliffs, NJ, 1992.

14. Olshansky, R., Bugos, A.R., and Hofmeister, R.T., *Multigigabit, Multichannel Lightwave Networks Using Subcarrier Multiplexing*, Journal of High Speed Networks, Vol. 2, 1993, p63-79.

15. International Organization for Standardization (ISO), *Information Processing Systems - Fiber Distributed Data Interface (FDDI) - Part 2: Token Ring Media Access Control (MAC)*, ISO 9314-2, 1989.

16. Tanenbaum, A.S., *Computer Networks*, 2nd edition, Prentice Hall PTR, New Jersey, 1995, p160-161.

17. Kirkpatrick, S., Gelatt Jr, C.D., Vecchi, M.P., *Optimization by Simulated Annealing*, Science, Vol. 220, No. 4598, 1983, p671.

18. Goldberg, D.E., *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison Wesley, Reading, MA, 1989.

19. Ingber, A.L., *Adaptive Simulated Annealing (ASA)*, Ingber Research, McLean, VA, 1993. (Web site: http://www.ingber.com)

20. Ingber, A.L., *Very Fast Simulated Reannealing*, Mathematical Computer Modeling, Vol. 12, No. 8, 1989, p967-973.

21. Chardaire, P., Lutton, J.L., *Using Simulated Annealing to Solve Concentrator Location Problems in Telecommunication Networks*, CNET Paris, France.

22. Andersen, K., Iversen, V.B., *Design of Teleprocessing Communication Network Using Simulated Annealing*, Institute of Telecommunications, The Technical University of Denmark.

23. Goldberg, D.E., Korb, B., and Deb, K., *Messy Genetic Algorithms: Motivation, Analysis and First Results*, Computer Systems, Vol. 3, p493-530, 1989.

24. Deb, K., Goldberg, D.E., *mGA in C: A Messy Genetic Algorithm in C*, University of Illinois at Urbana-Champaign, IlliGAL Report No. 91008, September 1991. (Web site: http://gal4.ge.uiuc.edu/illigal.home.html)

25. Ingber, A.L, Rosen, B., *Genetic Algorithms and Very Fast Simulated Reannealing: A Comparison*, Mathematical Computer Modeling, Vol. 16, No. 11, 1992, p87-100.

26. Anon., *Mathematical Optimization*, Hong Kong Baptist University, Department of Mathematics, Computational Science Education Project, October 1991. (Web site: http://www.math.hkbu.edu.hk/CSEP/mo/mo.html)

27. Metropolis, N., Rosenbluth, A.W., Rosenbluth, M.N., Teller, A.H., and Teller, E., *Equation of State Calculations by Fast Computing Machines*, Journal of Chemistry and Physics, Vol. 21, No. 6, p1087-1092, 1953.

28. Holland, J., *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor, MI, 1975.

APPENDIX A AI METHODS

This appendix gives a brief overview of Simulated Annealing (section A.1) and Genetic Algorithms (section A.2). The text and figures in this appendix are largely based on reference [26].

A.1 Simulated Annealing

Simulated Annealing (SA) [26] exploits an analogy between the way in which a metal cools and freezes into a minimum energy crystalline structure (the annealing process) and the search for a minimum of a cost function. SA's major advantage over other methods is the ability to avoid becoming trapped at local minima. SA is based on an algorithm developed by Metropolis et al. [27].

Figure 7 shows the basic structure of the SA algorithm. SA generally begins by reading in an existing solution to the problem or generating a random solution. Once the starting temperature and termination criteria have been established the actual annealing begins. At each iteration, the current solution is randomly perturbed to create a new solution (this step is referred to as a "move"). If the constraints are not satisfied by this configuration, the move is immediately rejected. Otherwise, the change in cost δC from the previous to the new solution is calculated, and the move is accepted if a randomly drawn number in [0, 1) is less than (or equal to) $exp(-\delta C/T)$. All moves which lead to a decrease in cost will be accepted; moves leading to an increase will be accepted with a probability which depends on the increase and the current temperature. If the move is not accepted, the previous solution is restored. After a certain number of moves the

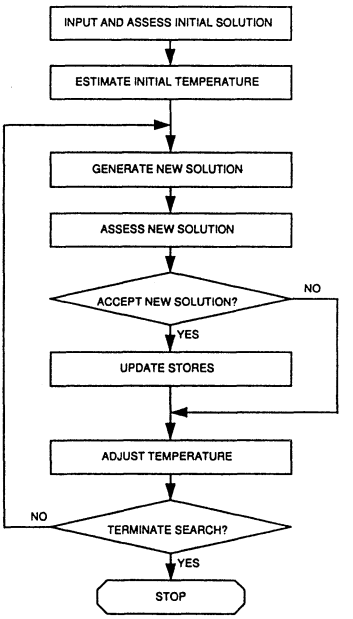

Figure 7: Basic structure of the Simulated Annealing algorithm

temperature will be decreased, thus decreasing the probability of accepting uphill moves. The algorithm terminates when the specified stopping criterion is met, e.g., when no improvement has been found for some number of moves.

The manner in which the temperature T is decreased over the iterations is referred to as the cooling schedule. Many schedules have been proposed, but the simple geometric schedule works fairly well on a variety of problems. In this schedule the temperature is reduced by multiplying it with a constant factor smaller than 1 (e.g., 0.99) at each chain of moves. The chains are of equal length.

A.2 Genetic Algorithms

Genetic Algorithms (GAs) [26] are optimization techniques based on the phenomenon of natural evolution. In natural evolution each species searches for beneficial adaptations in a changing environment. As species evolve new attributes are encoded in the chromosomes of individual members. This information does change by random mutation, but the driving force behind evolutionary development is the combination and exchange of chromosomal material during breeding. The original concepts of GAs were developed by Holland [28].

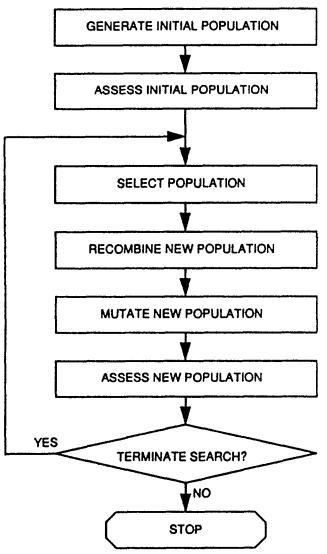

Figure 8: Basic structure of a Genetic Algorithm

The basic structure of a GA is shown in Figure 8. When applied to a cost function that needs to be minimized, the variables are represented as genes on a chromosome. GA searches from a group of candidate solutions (population) to another, rather than from individual to individual (as e.g., Simulated Annealing does). Through natural selection and the genetic operators - recombination and mutation - chromosomes with better fitness (i.e., a lower value of the cost function) are found. In the natural selection step, the chromosomes with the largest fitness scores are placed one or more times into a mating subset in a semi-random fashion. Using the recombination operator, the GA combines genes from two parent chromosomes to form two new chromosomes (children) that have a high probability of having better fitness than their parents. The two most common recombination operators are the one-point and two-point crossover methods. In the one-point method, a crossover point is selected along the chromosome and the genes up to that point are swapped between the two parents. In the two-point method, two crossover points are selected and the genes between the two points are swapped. The children then replace the parents in the next generation. The purpose of mutation is to provide insurance against the irrevocable loss of genetic information and hence to maintain diversity within the population. A mutation simply changes the value for a particular gene. GAs offer a generational improvement in the fitness of the chromosomes and after many generations create chromosomes containing the optimized variable settings.