Application of Hardware Architecture of Genetic Algorithm for Optimal Packet Scheduling

Rong-Hou Wu[1] Yang-Han Lee[2] Shiann-Tsong Sheu[3] Hsien-Wei Tseng[2]
Ming-Hsueh Chuang[2] Yung-Kuang Wang[2]

[1]Dept. of Computer &Communication Engineering, St. John's University, rhwu@mail.sju.edu.tw
[2]Department of Electrical Engineering, TamKang University, 692351157@s92.tku.edu.tw
[3] Department of Communication Engineering, National Central University, stsheu@ce.ncu.edu.tw

Abstract

In Dense Wavelength Division Multiplexing (DWDM) technologies, the optimal packet scheduling is a common encounter issue in multiple channels network. NP-hard problem deals with finding a way to rearrange packets in multiple channels into a finite and rare channel. Genetic algorithm (GA) is one of the most efficient ways to solve this issue. We hope to find a better solution to our task through the GA characteristics of multiprocessor searching and survivor of the fittest. Therefore, a modified and achievable hardware architecture of GA is presented in this paper. This architecture can increase the schedule speed of packet scheduling also can promote the efficiency of DWDM in Optical Communication Networks.

Keywords: Genetic algorithm, Dense Wavelength Division Multiplexing, packet scheduling.

1. Introduction

Due to increasing demands of network bandwidth in our society today, increase bandwidth becomes the critical key to promote our technologies study, with Dense Wavelength Division Multiplexing (DWDM) being one of the conspicuous technology of focus. Through the application of Erbium Doped Fiber Amplifier (EDFA), we segment the communication waveband into 4, 8, or even greater waveband segments for communication processing. Applying this technology can increase the bandwidth and the network transmission speed [1][2]. However, it derives other issues such as channel adjustment and packet scheduling. We hope to arrange packets in the shortest time and use the least amount of wavelength, to accurately and completely achieve packets transmission and receiving. NP-hard issue refers to the difficult process of finding the optimal packet scheduling in a limited timeframe. In recently years, GA has become an important way to solve this problem. It is concluded that crossover and mutation of chromosome and fitness function calculation can converge faster and therefore allow an efficient way to find the approach optimal solution [3].

In this paper, besides using the Matlab simulation software to prove the practicable and superiority of GA, we also presented achievable hardware architecture in DWDM of optimal packet scheduling [4]-[7].

This paper is organized as follows: Section II introduces the ways in which GA solves the optimal packet scheduling and presents the simulation results. Section III describes a hardware architecture designated to GA. Section IV contains our conclusion.

2. Application of GA for Optimal Packet Scheduling

2.1. Optimal of Packet Scheduling

Time stamps refer to the collection of packets in different wavelength channels on multiple channels network through the useable of the multiprocessor collector. All the accepted packets in the Time stamps are rearranged into minority channels for transmission. Generally, sending out all of the packets takes more time than Time stamps, and we refer this time period as the total required switching time (TRST). Since each packet has its unique arrival time and packet length, the TRST of each channel determines the last pack arrival time and the total length of packets using the equation:

$$TRST = Max(trst(w_1), trst(w_2), ..., trst(w_n)) \quad \ldots\ldots (1)$$

In order to achieve the optimal of packet scheduling, we need to decrease the magnitude of TRST. There are two rules to be obeyed during packet scheduling. First, on the same channel, the sequence of packets cannot be altered. Second, packets can only be accepted one at a time and in the order they are transmitted.

An example of packet scheduling is show in Figure 1, which contains 4 different channels to accept coming packets, where P_{ij} represents J's packet of I's channel. Ta represents the arrival time of the packet and L represents the packet length. Figure 2 shows a possible result of packet scheduling, according to Figure 1. Since we have to obey the two restrictions mentioned, there are too many idle spaces, which results in longer scheduling time. Therefore, we hope

to figure out a method to send packets to our finite channels more efficiently through optimal packet scheduling.

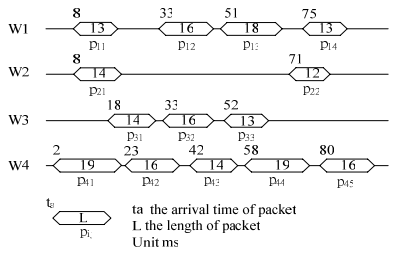

Fig. 1: First example of packet scheduling

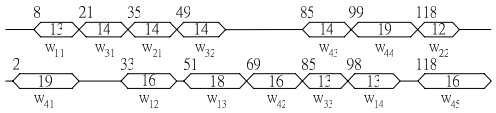

Fig. 2: Second example of packet scheduling

2.2. Application of GA

GA is an algorithm method that imitates creature evolution and uses basic genetic adaptation idea of survivor of the fittest. It has advantage on multiprocessor searching. We hope to use this multiprocessor searching characteristics to find the ideal solution on packet scheduling problem.

In the issue of packet scheduling, one of the possibilities is to search all of the packet scheduling sequences. However, this is time consuming; certain kinds of packet scheduling may be illegal, and it is difficult to construct a trial and error method. Therefore, we try to present a new structure, which selects the smallest and the best packet in advance. Because of different amount of packets in different channels and under rules confine, this solution may not be the most effective. However, this can approach the optimal solution within the require time. This is critical because it offers users a steady system. So we apply multiprocessor scheduling characteristics of GA and merit extensibility of survivor of the fittest to help us search for the optimal solution.

We select the smallest arrival time (Ta) of the available packets in different input channels as the first transmit packet. Take Figure 1 for example, we will take P_{41} as the first schedule packet, because it has the smallest arrival time (Ta = 2 ms) of all four input channels. Then we select the next smallest Ta of the remaining packets as the second schedule packet. The searching procedures will be repeated, in hopes of finding a better packet scheduling solution. However, finding the smallest Ta in the huge packets of a lot of input channels situation may depend on GA's searching capability. Initially, we randomly select the number of M and N as input and output channels, respectively. Then we read the Ta value of all these channels, which is determined by the follow equation

$$Ta_n = \max(ta(p_{nj}), pt(n) + td(p_{n(j-1)})) \ldots\ldots \quad (2)$$

P_t represents the start transmission time of selected packet of each input channel; n represents the n^{th} channel and j represents the j^{th} packet. After comparison, we will select the smallest Ta of each N Channel. These selection numbers will proceed crossover and mutation process in order to produce the next generation. In a similar way, we use one variable to recode the present allocation situation of each output channels and select the fastest placement channel as the packet output channel. Since the selecting the packet with the smallest Ta value may not guarantee the optimal solution, we also recode the result of every generation solution at the same time. Then we will generate the output of the optimal solution. Thus, this process allows the finding of a better solution and at the same time match the rules of solution for every generation.

2.3. Matlab Simulation

In order to verify that the present architecture can find a better solution, we use Matlab simulation software to verify the excellence of this structure. We use Poisson distribution to determine the packet outcome probability. We use exponential distribution to determine packet length, using 1ms as the smallest unit and setting collection window as 20ms. At the same time, simulation of GA for the conventional architecture, as shown in Figure 3, is processed. We conduct crossover and mutation of chromosome (packet data), similar to the optimal packet scheduling architecture. However, the largest difference is that the entire packets in one generation must all be processed before the system can proceed on to the next generation. According to these parameters, the result of convergence, between these two architectures, is shown in Figure 4.

The solid line (HG-GAPS) and the dash line (G-GAPS) represents the hyper-generation Gas and general Gas, respectively. It is obvious that the new architecture not only has outstanding result in the first attempt, but also has faster convergence.

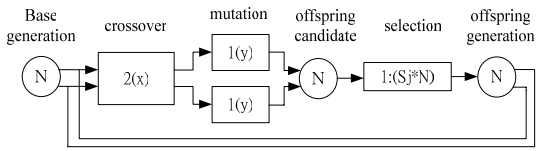

Fig. 3: Scheme of Genetic algorithm of conventional architecture

3. Hardware Architecture

We will take a system with four input channels and two outputs channels to design a feasible hardware architecture in order to achieve the proceeding algorithm and to promote its practical value.

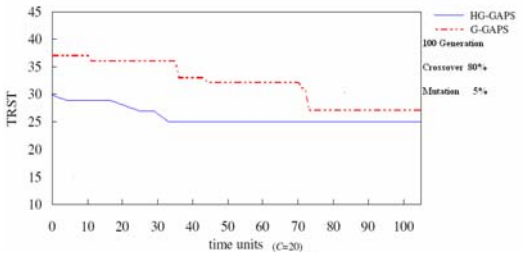

Fig. 4: Simulation result of packet scheduling using Matlab software

3.1. Main Architecture

In reality, the numbers of packet vary according to different situations. To solve this problem, when we choose the hardware, we should set the numbers of offspring generation in advance. This way, can be suitable for different amount of packets, and also can achieve the efficiency of pipeline. Figure 5 is a designation of the main hardware architecture for one offspring generation. The number of identical architectures in a series represents the number of offspring generations, which are present in the system. In the beginning of the system, we need to put all the accepted packets data into a memory or a register of collection window. These data includes arrival time of each packet, length of each packet, and the total packets. The total packets will determine the execution times from generate to generation, i.e., how many process will completely achieve scheduling for all the packets. The Pack-counter unit, composed of simple counter and comparator, is used to calculate the transmission number. The counter tallies the number of packets that enters the system. If one packet enters, the counter will add one to the system.

Simultaneously, the comparator checks whether this number is equal to the total packets. If the desired number of packets has entered the system, the system will stop. Other units are described below.

3.2. Genetic Crossover and Mutation Unit

In this hardware architecture, crossover unit and mutation unit represents the crossover and mutation of GA, respectively. Figure 6 shows the inner design of crossover unit. We use two switches to compare input random number and crossover-rate, which is set up by system. When random number is greater than crossover-rate, the output is crossover and the number 1 represents this situation. On the contrary, when random number is less than crossover-rate, output is its original value and the number 0 represents this situation, there is no crossover therefore the offspring is identical to the mother.

Figure 7 shows a mutation unit, which reverses the point out bit by means of inverter to achieve mutation effect. The output control is similar to the crossover unit.

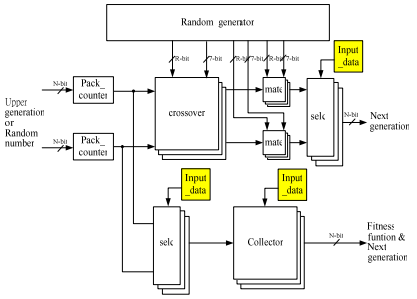

Fig. 5: Architecture of main hardware

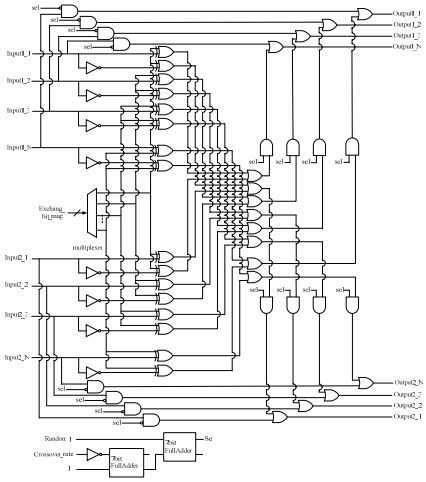

Fig. 6: Architecture of crossover hardware

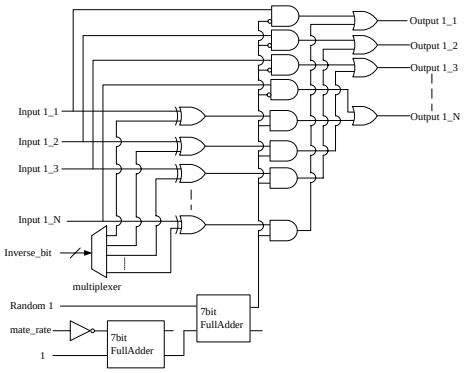

Fig. 7: Architecture of mutation hardware

3.3. Selection Unit

The function of selection unit is to choose the better packet. It will read through each packet data from memory allocation unit, compare the fastest available transmission packet values of selected channels, and transmit the better channel value. Therefore, this architecture needs a comparator, multiplexer, and demultiplexer. Finally, it needs register spaces to recode some variables, which are the fastest available

transmission packet value of each input channels. It also needs a simple adder for calculation.

3.4. Collector Unit

The purpose of the collector unit is to maintain the most outstanding output for the next generation. It uses a register to record the best solution and compares solutions from each generation, making sure that the data they use is the most compatible. This unit needs a similar structure of fitness function architecture of GA.

We can calculate the TRST of each achieved packet scheduling using simple multiplexer and adder.

3.5. Random Generator Unit

Random selection is an important idea in GA. In order to achieve pseudo random number in hardware, we use PN code method and apply a lot of shift registers to generate a large random number range, as shown in Figure 8. The value of N is quite big and each output of registers is one bit of the random number. Thus if the system requires 8 bits random number, then 8 outputs from this unit must be selected. The outputs do not need to be in a series. Using the non-synchronous clock to shift these outputs, we can achieve a random number in the hardware.

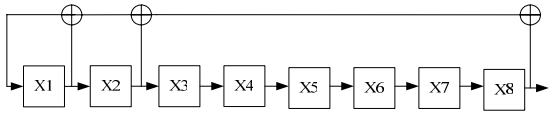

Fig. 8: Architecture of random number generator

3.6. Register Allocation Unit

Packet scheduling needs spaces to storage channel packet data and output results. The register allocation space presented in this paper is shown in Table 1.

4. Conclusion

Aiming at the meditation of using GA for proceeding packet scheduling, this paper not only verify its excellence by means of software simulation, but also in the architectural design of the hardware. We can use this architecture to solve the packet scheduling issue and to promote the system function and transmission efficiency in the future.

5. Acknowledgements

The authors would like to express their thanks to Ph.D. Yue-Ru Chuang for his helpful discussions. This work was supported by the National Science Council, Taipei, Taiwan, R.O.C. under Contract NSC 94 - 2213 - E - 129 - 005, NSC 94 - 2213 - E - 032 - 005, NSC 94 - 2745 - E - 032 - 001 - URD, NSC 94 - 2745 - E - 032 - 004 - URD, and the funding from St. John's University and Tamkang Unversity for the University-Department joint research project.

Table 1: Register allocation

Name	Purpose	Number
Pack_all (PA)	Total numbers of packet	1
Pack_used	The number of Scheduled packets	1
Pack_last	Remaining packets of each channel	N
Ta_in	Arrival time of pack of each input	PA
Td_in	Length of each channel	PA
Ta_out	Available time of each channel	N
Put_time(Pt)	At present transmission Time of each channel	N
Ta_used	Available transmission time of each channel	O
Out_data	Sequence of packet scheduling	O
PA = Total number of packet N = Numbers of input channel O = Numbers of output channel		

6. Reference

[1] Shiann-Tsong Sheu, Yue-Ru Chuang, Yu-Jie Cheng, Hsuen-Wen Tseng, “A Novel Optical IP Router Architecture for WDM Networks.” *In Proceedings of IEEE ICOIN-15*, pp. 335-340, 2001

[2] Paul Green, ” Progress in Optical Networking.” *IEEE Communications magazine*, Vol. 39, No. 1, pp. 54-61, January 2001

[3] Shiann-Tsong Sheu**,**and U.-J. Chuang, “An Optimization Solution for Packet Scheduling: A Pipeline-Based Genetic Algorithm Accelerator”, *Proc. ofAAAI GECCO’2003*, Chicago, July 2003.

[4] Wallace Tang, Leslie Yip, ”Hardware Implementation of Genetic Algorithms Using FPGA” *Circuits and Systems, 2004. MWSCAS '04. The 2004 47th Midwest Symposium on* Vol. 1, pp. I-549 - I-552, 25-28 July 2004.

[5] Kallel I., Jmaiel M. et Alimi A. M. ”A Multi-agent Approach for Genetic Algorithm Implementation” *Systems, 2002 IEEE International Conference on Man and Cybernetics*, Vol. 7, 6-9 Oct. 2002.

[6] M.S. Sharawi, J. Quinlan, Abdel-Aty-Zohdy, H.S. “ A Hardware Implementation of Genetic Algorithms for Measurement Characterization” *9th International Conference on Electronics, Circuits and Systems*, Vol. 3, pp. 1267–1270, 15-18 Sept. 2002

[7] C. Aporntewan, P. Chongstitvatana, ”A Hardware Implementation of the Compact Genetic Algorithm;” *Evolutionary Computation, 2001. Proceedings of the 2001 Congress on Evolutim Computatim 1,* pp. 624–629, 27-30 May 2001