2002 한국정보처리학회 논문지 제5권 제9호(1989)

유전자 알고리즘을 이용한 광상호연결망에서 파장분할다중방식의 리스트 스케쥴링

정 혜 진† · 위 규 범†† · 예 홍 진†† · 홍 만 표†† · 변 광 준††

요 약

파장분할다중방식의 패킷 전송을 하는 광상호연결망에서 노드들 사이에 송수신하는 패킷의 수가 임의로 주어지는 일반적인 교통량의 스케쥴링은 NP-complete 문제이며, 이 경우에 대하여 리스트 스케쥴링이 좋은 성능을 보임이 알려져 있다. 리스트 스케쥴링은 각 시간 단위에 배치할 송신자를 고려하는 순서에 따라서 스케쥴 결과가 달라질 수 있다. 본 논문에서는 송신자를 고려하는 순서를 다양하게 시도하여 더욱 효율적인 스케쥴을 구하는 방법을 제시한다. 다양한 순서를 탐색하는 방법으로서 유전자 알고리즘을 이용한다.

List Scheduling on WDM Optical Interconnection Networks using Genetic Algorithms

Hea-Jin Jung† · Kyubum Wee†† · Hong-Jin Yeh†† · Manpyo Hong†† · Kwang-June Byeon††

ABSTRACT

The problem of scheduling general packet traffic on WDM optical interconnection networks is NP-complete. It is known that the list scheduling is a good approximation algorithm for this problem. The resulting list schedules vary depending on the order of transmitters considered to be placed on each time slot. We propose an improvement of the list scheduling that tries different orders of transmitters to obtain shorter schedule lengths. Genetic algorithms are used to explore various orders of transmitters.

1. 서 론

대규모 병렬 시스템(massively parallel processing system)에서는 많은 노드들 사이에 대량의 데이터를 교환할 수 있어야 한다. 그러므로 노드들이 이루는 결합구조(interconnection network)의 성능은 대규모 병렬 시스템의 성능에 지대한 영향을 미친다. 최근에 광결합구조(optical interconnection network)의 기술의 발달로 인하여 기존의 전자결합구조(electronic interconnection network)의 한계를 극복할 수 있게 되었다. 광결합구조에서 널리 쓰이는 데이터 전송 방법의 하나가 광섬유의 거대 용량을 이용하는 파장분할다중방식(wavelength division multiplexing; WDM)이다. WDM은 하나의 광섬유에 서로 다른 파장을 사용하는 여러 개의 채널(channel)로 데이터를 전송하는 방법이다. 네

† 준 회 원 : 아주대학교 정보 및 컴퓨터공학부
†† 종신회원 : 아주대학교 정보 및 컴퓨터공학부 교수
논문접수 : 1997년 5월 11일, 심사완료 : 1998년 8월 9일

프로세서의 각 노드는 송신장치(transmitter)와 수신장치(receiver)를 가지고 있다. 서로 다른 파장의 채널에 맞도록 조율(tuning)할 수 있는 송신장치 또는 수신장치를 조율가능(tunable), 조율할 수 없는 것을 고정적(fixed tuned)이라고 부른다. 조율가능한 장치에서 하나의 파장에서 다른 파장으로 조율하는데 걸리는 시간을 조율지연시간(tuning delay)이라 부른다.

많은 노드들 사이에 데이터를 효율적으로 교환할 수 있도록 스케줄링하는 것은 광결합구조의 성능을 향상시키는 핵심적인 문제이다. 일반적으로 전송해야 할 패킷의 수가 사용할 수 있는 채널의 수보다 훨씬 많다. 이 경우에 동시에 두 개 이상의 노드가 같은 파장을 사용하여 전송할 수 없다는 조건과 각 노드가 하나의 파장을 사용하여 전송한 후에 다른 파장을 사용하려면 조율지연 시간이 필요하다는 조건을 만족하면서 최단 시간에 모든 데이터를 다 전송하는 스케줄을 구하는 것은 단순한 문제가 아니다.

이러한 스케줄링 문제의 특별한 경우로서 각 노드가 다른 모든 노드에게 꼭 한 개의 패킷을 보내는 문제를 전방송(all-to-all broadcast)문제라고 한다. 전방송문제에 대하여는 효율적인 스케줄링 알고리즘들이 알려져 있다[1, 2]. 그러나 각 노드가 다른 노드에게 보내는 패킷의 수가 임의로 설정되는 일반적인 경우의 스케줄링은 NP-complete 이다[3, 4]. 이 경우에 대해서는 리스트 스케줄링(list scheduling) 또는 이분 멀티그래프(bipartite multigraph)의 에지 색칠하기(edge coloring)에 기초한 근사 알고리즘이 연구되었다[5]. 리스트 스케줄링 방법에 의한 스케줄은 문제의 복잡도의 하한(lower bound)에 상당히 근접한 좋은 결과를 보였다. 리스트 스케줄은 각 시간 단위(time step)에 배치할 송신장치를 결정할 때에, 송신장치들을 고려하는 순서에 따라 스케줄이 달라질 수 있다. 본 논문에서는 이 경우에 송신장치들을 고려하는 순서를 다양하게 시도하여, 더욱 효율적인 스케줄을 구하고자 한다. 여기서 송신장치들의 순서를 다양하게 시도하는 방법으로서 유전자 알고리즘을 이용한다.

본 논문의 구성은 다음과 같다. 제 2장에서는 문제를 정의하고, 제 3장에서는 리스트 스케줄에서 송신자를 고려하는 순서에 따라 스케줄이 달라지는 예를 본다. 제 4장에서는 유전자 알고리즘을 간략히 설명한다. 제 5장에서는 이 문제에 유전자 알고리즘을 어떻게 적용하는지 설명한다. 제 6장에서는 실험 과정을 설명하고, 실험을 분석한다.

2. 문 제

네트워크는 n개의 노드로 이루어지고, k개의 파장 $w_1, w_2, \ldots, w_k$을 사용할 수 있으며 n은 k로 나누어 떨어진다고 가정한다. 여기서 n이 k의 배수라는 가정은 단지 문제의 모델을 간단히 설정하기 위함이며, 본 논문에서 제안하는 방법은 n이 k의 배수라는 조건이 없는 일반적인 경우에도 사용할 수 있다. 각 노드는 1 개의 조율 가능한(tunable) 송신장치와 1 개의 고정된(fixed-tuned) 수신장치를 가지고 있다. 각 송신장치 t_i ($1 \le i \le n$)가 각 수신장치 r_j ($1 \le j \le n$)으로 보내는 패킷의 개수는 음이 아닌 임의의 정수이며, 그림 1과 같은 교통량 행렬(traffic matrix)로 나타낸다. 각 노드는 자기 자신에게는 패킷을 보내지 않는다. 다시 말해서, 각 노드의 송신장치는 같은 노드에 있는 수신장치에게는 아무 것도 전송하지 않는다. 따라서 교통량 행렬의 대각선은 모두 0으로 채워진다. 일반적으로 사용할 수 있는 파장의 수 k는 노드의 수 n보다 훨씬 작으므로, 여러 수신장치가 같은 파장을 사용해야 한다. 파장을 공유하는 수신장치들의 집합을 수신장치 그룹이라고 부르며, 각 수신장치 그룹 $R_p = \{r_{p-1)n/k+1}, \cdots, r_{p(n/k)}\}$는 파장 w_p를 공유하는 $\frac{n}{k}$ 개의 수신장치로 이루어진다 ($1 \le p \le k$).

패킷들의 크기는 모두 일정하다고 가정하며, 각 1 개의 패킷의 전송을 완료하는 데 걸리는 시간을 단위시간(unit time)으로 정한다. 송신장치가 하나의 파장으로부터 다른 파장으로 조율하는데 걸리는 조율지연시간(tuning delay)을 δ로 표시한다. 이 경우에 모든 패킷들을 최단시간에 송수신하도록 스케줄링하고자 한다. 그림 2는 그림 1의 교통량 행렬에 대한 하나의 스케줄을 보여준다. 여기서 노드의 수 N은 6, 사용할 수 있는 파장의 수 k는 3, 조율지연시간 δ은 2이며, 같은 파장을 사용하는 수신장치들의 그룹은 $R_1 = \{r_1, r_2\}$, $R_2 = \{r_3, r_4\}$, $R_3 = \{r_5, r_6\}$이다.

그림 2의 처음 두 열(column)이 비어 있는 이유는 처음에 t_1이 파장 w_1으로, t_6가 w_2로, t_5가 w_3로 각각 조율하는 데 시간이 2 가 걸리기 때문이다. 하나

		t_1	t_2	t_3	t_4	t_5	t_6
R_1	r_1	0	1	0	1	3	0
	r_2	2	0	1	2	1	0
R_2	r_3	3	1	0	1	0	1
	r_4	0	3	0	0	0	2
R_3	r_5	0	1	2	2	0	0
	r_6	1	1	2	0	1	0

(그림 1) 교통량 행렬
(Fig. 1) traffic matrix

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
R_1					t_4	t_4	t_4	t_3	t_5	t_5	t_5	t_5	t_2								
R_2			t_6	t_6	t_6	t_2	t_2	t_2	t_2				t_4								
R_3			t_5								t_3	t_3	t_3	t_3		t_4	t_4	t_2	t_2		

(그림 2) 패킷 전송 스케줄
(Fig. 2) packet transmission schedule

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
R_1			t_1	t_1	t_4	t_4	t_4		t_2	t_3	t_5	t_5	t_5	t_5							
R_2			t_2	t_2	t_2	t_2	t_6	t_6	t_6	t_4	t_1	t_1	t_1								
R_3			t_3	t_3	t_3	t_3	t_5	t_1				t_2	t_2	t_4	t_4						

(그림 3) 리스트 스케줄링한 결과
(Fig. 3) A result of list scheduling

의 송신장치가 동시에 여러 개의 파장 을 사용할 수 없으므로, 그림 2의 각 열(column)에는 같은 송신장치가 여러 번 나올 수 없다. 각 송신장치는 조율지연시간이 필요하므로, 같은 송신장치의 여러 블락(block)들은 거리가 δ 이상 떨어져 있어야 한다. 예를 들어서, 그림 2에서 t1의 블락 3 개는 서로 거리가 2 이상 떨어져 있다. 그림 2의 스케줄 길이는 19 이다.

3. 리스트 스케줄링

위와 같이 정의된 문제의 스케줄을 구하는 방법으로서, 잘 알려진 리스트 스케줄링을 사용할 수 있다[6]. Choi 등[5]은 리스트 스케줄이 최적 스케줄 길이의 두 배 이하가 됨을 증명했으며, 시뮬레이션 결과 최적 스케줄 길이에 매우 근접함을 보였다.

그림 3은 그림 1의 교통량 행렬을 리스트 스케줄한 결과이다.

리스트 스케줄은 다음과 같이 구한다: 그림 3과 같은 스케줄 표의 각 칸에 송신장치 t_i를 배치하는 것이 가능한가를 알기 위하여 아래의 조건들을 검사한다:

(1) 그 칸이 속한 행(row)에 이미 t_i가 배치되어 있지 않아야 한다.
(2) 그 칸이 속한 열(column)에 이미 t_i가 배치되어 있지 않아야 한다.
(3) 이미 배치된 t_i의 블락들과의 거리가 δ 이상 떨어져 있어야 한다.

위의 조건들을 만족시키는 송신장치를 찾기 위하여,

송신장치들을 어떤 정해진 순서대로 고려한다. 쉽게는 $t_1, t_2, \ldots, t_n$ 의 순서로 고려할 수 있다. 다시 말해서, 위의 조건들을 만족시키는 송신장치가 여러 개인 경우에, 첨자가 가장 작은 송신장치를 배치시킨다.

예를 들어서, 그림 3에서 좌표 $(R_1, 5)$에 배치할 송신장치를 선택하기 위하여, $t_1, t_2, \ldots, t_6$ 의 순서로 위의 조건을 만족시키나 검사한다: t_1은 이미 그 행에 배치되어 있다. t_2와 t_3는 이미 그 열에 배치되어 있다. t_4는 위의 조건들을 만족시키고 t_4가 R_1에게 보내는 패킷의 수가 3 개이므로, $(R_1,5),(R_1,6),(R_1,7)$에 t_4를 배치시킨다. 여기서 t_5도 조건을 만족시키지만, t_4를 t_5보다 먼저 고려하므로, t_4가 배치된다. 이번에는 $(R_3, 9)$에 아무 것도 배치되어 있지 않은 이유를 알아보자: t_1 은 이미 배치되어 있다. t_2는 같은 열에 $(R_1, 9)$에 배치되어 있다. t_3는 이미 배치되어 있다. t_4는 이미 $(R_1,5),(R_1,6),(R_1,7)$에 배치되어 있는 t_4의 블락과의 시간적인 거리가 δ (= 2) 이상이 되지 않는다. t_5는 이미 배치되어 있다. t_6는 R_3에 보낼 패킷이 없다. 따라서 $(R_3, 9)$ 칸은 비어 있게 된다.

리스트 스케쥴에서 각 칸에 배치할 송신장치를 선택하기 위하여 고려하는 순서를 $t_1, t_2, \ldots, t_6$ 이 아닌 아래와 같이 달리하는 경우에 그림 4와 같은 더 좋은 스케쥴을 얻을 수 있다. 그림 4에서 송신장치들을 고려한 순서는 $t_1, t_2, t_5, t_3, t_4, t_6, t_5, t_1, t_2, t_4, t_1, t_3, t_2, t_4$이다.

여기서 송신장치들을 고려하는 가능한 순서들의 경우의 수는 매우 많다. 스케쥴 표에 배치해야 하는 송신장치 블락의 개수가 최악의 경우에는 nk 개이므로, 가능한 순서들의 경우의 수는 $(nk)!$ 개이다. 따라서, 모든 경우를 다 시도해 볼 수는 없다. 여기서, 좋은 스케쥴을 내는 순서를 찾는 방법으로서 유전자 알고리즘을 제안한다.

4. 유전자 알고리즘

유전자 알고리즘은 생명체의 적자생존(survival of the fittest)과 진화(evolution)현상에 기초한 확률적 검색 알고리즘으로 크로모좀(chromosome)이라 부르는 스트링(string)으로 인코딩(encoding)한 후보해(candidate solution)를 나타내는 개체(individual)들을 군집(population)으로 구성하고, 이 군집에 교배(crossover), 돌연변이(mutation) 등의 유전자 연산을 반복적으로 적용함으로서 다음 세대에 더 좋은 해를 나타내는 개체군으로 진화시킨다. 유전자 알고리즘은 여러 가지 최적화 문제(optimization problems)에 좋은 성능을 보일 수 있음이 알려져 있다[7, 8]. 유전자 알고리즘은 다음과 같은 과정들로 구성된다.

(1) 크로모좀들을 무작위로 생성하여 초기 집단을 형성한다.

(2) 집단의 각 개체들이 얼마나 좋은 후보해인지를 나타내는 적합도(fitness)를 계산한다. 여기서 적합도를 계산하는 규칙을 적합도 함수(fitness function)이라고 부른다.

(3) 두 개의 크로모좀을 선택(selection)하여 이들을 교배해서 얻은 두 개의 새로운 자손(offspring)을 다음 세대(generation)의 군집에 개체로 더해준다. 이때 새로운 개체를 나타내는 크로모좀을 이루는 각 유전자(gene)는 정해진 확률로 돌연변이(mutation)을 일으켜 그 값이 변할 수 있다. 각 개체가 선택될 확률은 적합도에 비례한다. 이 과정을 반복하여 다음 세대의 군집을 형성한다.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
R_1			t_1	t_1	t_4	t_4	t_4	t_5	t_5	t_5	t_5	t_3	t_2								
R_2			t_2	t_2	t_2	t_2	t_6	t_6	t_6	t_4	t_1	t_1	t_1								
R_3			t_5	t_3	t_3	t_3	t_3	t_1	t_2	t_2			t_4	t_4							

(그림 4) 송신장치의 순서를 달리하는 경우의 리스트 스케쥴링한 결과
(Fig. 4) A result of list scheduling with a different order of transmitters

(4) (2)번과 (3)번 과정을 정해진 횟수만큼 또는 원하는 해가 생성될 때 까지 반복한다.

유전자 알고리즘의 기본 개념은 적합도가 높은 개체가 가진 좋은 형질의 유전자들이 선택에 의하여 그 수가 불어나면, 교배에 의하여 서로 다른 개체에 있던 좋은 형질의 유전자들이 같은 개체로 모이게 되며, 이에 따라 점점 적합도가 높은 후보해로 진화해 나아간다는 것이다. 여기서 돌연변이 연산자의 역할은 크로모좀의 어떤 특정한 위치에 있는 유전자 값이 모든 개체에 걸쳐서 동일하게 되어 유전자의 다양성을 잃어버리는 경우가 발생하지 않도록 보장하는 장치이다.

주어진 문제를 유전자 알고리즘으로 해결하고자 하는 경우에 다음과 같은 점들을 설정해야 한다:

(1) 문제의 후보해를 크로모좀 스트링으로 인코딩하는 방법
(2) 적합도 함수의 디자인
(3) 각 유전 연산자를 구현하는 방법
(4) 각 유전 연산자를 적용하는 비율의 설정
(5) 집단의 크기와 최대 세대수의 설정

위에서 나열한 점들을 어떻게 설정하느냐에 따라서 유전자 알고리즘의 성능은 많은 영향을 받는다. 병렬처리의 스케쥴링 문제에 유전자 알고리즘을 이용한 연구들은 [9], [10]등이 있으며, 유전자 알고리즘 일반에 대한 참고문헌들은 [7], [8], [11]등이 있다.

5. 유전자 알고리즘을 이용한 리스트 스케쥴링

5.1 후보해의 인코딩(encoding)

각 송신장치가 어느 수신 장치 그룹에 몇 개의 패킷을 보내야 하는지를 하나의 순서쌍으로 구성하여 나타낸다. 이 경우에 패킷의 개수가 0인 것은 순서쌍으로 만들지 않는다. 이렇게 생성된 순서쌍들에 일련번호를 매긴다. 이와 같은 순서쌍들의 번호를 무작위(random)로 순서를 정하여 하나의 크로모좀을 형성한다. 이러한 크로모좀이 나타내는 스케쥴은 순서쌍들이 나타내는 전송해야 할 패킷들을 리스트 스케쥴한 결과이다. 여기서 리스트 스케쥴할 때에, 송신장치들을 고려하는 순서는 크로모좀에 나타난 순서쌍들의 순서에 의한다. 위의 그림 1의 교통량 행렬을 순서쌍들로 아래와 같이 나타낼 수 있다.

1:$(t_1, R_1, 2)$ 2:$(t_2, R_1, 1)$ 3:$(t_3, R_1, 1)$ 4:$(t_4, R_1, 3)$ 5:$(t_5, R_1, 1)$
6:$(t_1, R_2, 3)$ 7:$(t_2, R_2, 4)$ 8:$(t_4, R_2, 1)$ 9:$(t_6, R_2, 3)$ 10:$(t_1, R_3, 1)$
11:$(t_2, R_3, 2)$ 12:$(t_3, R_3, 4)$ 13:$(t_4, R_3, 2)$ 14:$(t_5, R_3, 1)$

(그림 5) 순서쌍 번호 :
(송신장치의 번호, 수신장치 그룹의 번호, 패킷수)
(Fig. 5) index of ordered pair :
(transmitter no., receiver no., number of packets)

이러한 순서쌍으로부터 아래 그림 6과 같은 크로모좀을 구성할 수 있으며, 이 크로모좀이 나타내는 스케쥴은 그림 7과 같다.

1	4	9	14	7	3	5	6	2	8	10	12	13	11

(그림 6) 크로모좀의 예
(Fig. 6) an example of a chromosome

그림 7에서 처음 두 열은 초기 조율지연시간이다. $(R_1, 3)$에 t_1이 배치된 것은 크로모좀의 맨 처음 유전자가 순서쌍 1번 $(t_1, R_1, 2)$를 나타내기 때문이다. $(R_2, 3)$에 배치할 송신자를 찾기 위하여 크로모좀의 유전자들을 순서대로 검사하면, 4번 순서쌍은 수신자 그룹이 R_2가 아니고, 9번 순서쌍이 R_2에 보내므로 t_6가 배치된다. $(R_3, 3)$에는 14번 순서쌍에 의해서

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
R_1			t_1	t_1	t_4	t_4	t_4	t_5	t_5	t_5	t_5	t_3				t_2					
R_2			t_6	t_6	t_6	t_2	t_2	t_2	t_2	t_4	t_1	t_1	t_1								
R_3			t_5	t_3	t_3	t_3	t_3	t_1				t_2	t_2	t_4	t_4						

(그림 7) 그림 6의 크로모좀의 순서에 따른 스케쥴
(Fig. 7) the schedule according to the chromosome of Fig. 6

t_5가 배치된다. $(R_1, 5)$에는 4번 순서쌍에 의해서 t_4가 배치된다. $(R_2, 6)$에는 7번 순서쌍에 의해서 t_2가 배치된다. $(R_3, 4)$에는 3, 5, 6, 2, 8, 10번 순서쌍들을 지나쳐서, 12번 순서쌍에 의하여 t_3가 배치된다. $(R_1, 8)$에는 5번에 의해서 t_5가 배치된다. $(R_3, 8)$에는 3, 6, 2, 8번을 지나쳐서, 10번에 의하여 t_1이 배치된다. $(R_3, 9)$에는 배치할 것을 찾아보면 11번 순서쌍이 있지만, 같은 시간대에(column 9) t_2가 이미 배치되어 있으므로 t_2를 배치할 수 없다. 다음 12번 순서쌍은 이미 배치되었으며, 13번은 t_4가 시각 7에 R_1에게 전송을 끝낸 후 R_3가 수신하는 과정 w_3로 아직 조율이 끝나지 않았다. 14번은 같은 시간대에(column 9) 이미 t_5가 배치되어 있다. 따라서 $(R_3, 9)$에는 아무 것도 배치할 수 없다. 이 이후의 각 칸에도 같은 방법으로 송신장치들을 배치해 나아간다.

여기서 순서쌍들의 순서를 달리하여 아래와 그림 8과 같은 크로모좀을 구성하면, 이 크로모좀이 나타내는 스케쥴도 그림 9와 같이 달라짐을 알 수 있다.

2	6	13	5	12	7	4	10	9	14	3	1	11	8

(그림 8) 크로모좀의 다른 예
(Fig. 8) another example of a chromosome

5.2 적합도 함수(fitness function)

각 크로모좀의 적합도는 처음 세대부터 그 크로모좀이 속한 세대까지 생성된 모든 크로모좀들의 스케쥴 길이의 최대값과 그 크로모좀이 나타내는 리스트 스케쥴의 길이와의 차이로 설정한다. 따라서 적합도의 값은 음이 아닌 정수이며, 스케쥴의 길이가 짧을수록 적합도가 높게 된다.

5.3 선택(selection)

교배 또는 돌연변이를 적용할 크로모좀을 선택하는 방법은 널리 쓰이는 룰렛휠 선택(roulette wheel selection)을 사용하였다. 이때 각 세대에서 적합도 값이 가장 높은 개체는 교배 또는 돌연변이 연산을 적용하지 않고 다음 세대에 그대로 복사해 주는 정예주의(elitism)를 가미하였다.

5.4 유전 연산자(genetic operators)

5.4.1 교배(crossover)

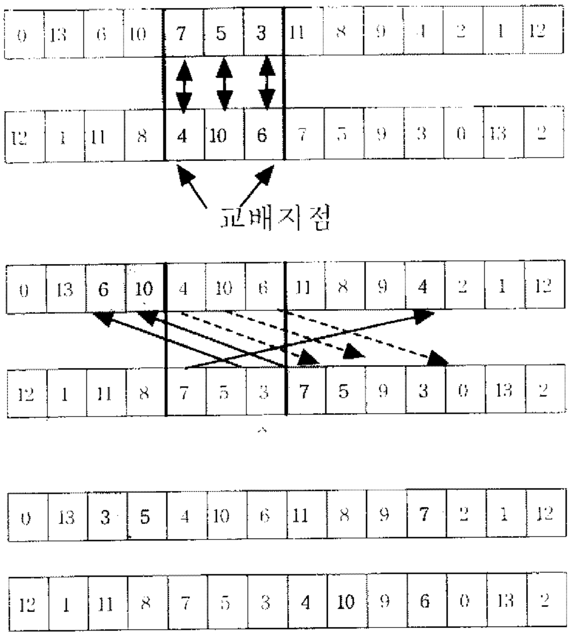
크로모좀이 순서쌍들의 번호의 순열(permutation)을 나타내므로, 두 개의 크로모좀에 일반적인 교배연산을 적용하면 그 결과로 얻어지는 자손들은 순서쌍들의 번호의 순열을 나타내지 못한다. 그러므로 외판원 문제(travelling salesman problem)에 유전자 알고리즘을 적용하는 경우의 교배 연산으로 잘 알려진 PMX(partially matched crossover)방법을 사용하였다[7]. PMX는 두 개의 크로모좀을 선택하여 무작위로 선정한 두 점 사이의 부분스트링(substring)을 교환하고, 나머지 부분의 유전자(gene)들은 두 부분스트링 사이에서 대응되는 값으로 바꾸어 준다. 그림 10은 PMX 연산을 예를 들어서 보여준다.

5.4.2 돌연변이(mutation)

돌연변이 연산은 일반적으로 크로모좀의 각 유전자가 작은 확률로 다른 값으로 변하는 것이다. 그러나 위의 교배연산에서와 마찬가지로 우리의 문제에서는

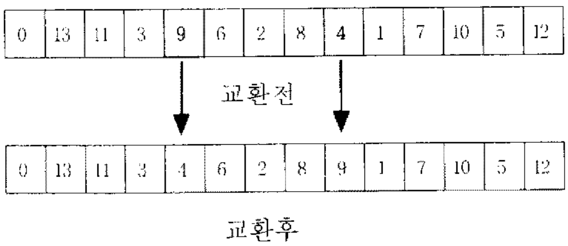
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
R_1			t_2	t_5	t_5	t_5	t_5	t_4	t_4	t_4	t_3	t_1	t_1								
R_2			t_1	t_1	t_1	t_2	t_2	t_2	t_2	t_6	t_6	t_6	t_4	t_4							
R_3			t_4	t_4	t_3	t_3	t_3	t_3	t_1	t_5		t_2	t_2								

(그림 9) 그림 8의 크로모좀에 의한 리스트 스케쥴
(Fig. 9) the schedule according to the chromosome of Fig. 8



(그림 10) 교배연산
(Fig. 10) crossover operator

크로모좀이 번호들의 순열이므로 어떤 유전자가 임의의 다른 값으로 변하면 더 이상 순열이 되지 못한다. 그러므로 여기서는 돌연변이 연산으로서 역시 외판원 문제에서 쓰이는 상호 교환(reciprocal exchange)라는 방법을 사용하였다. 상호 교환은 하나의 크로모좀의 두 점을 무작위로 선정하여 그 위치에 있는 유전자를 서로 교환하는 방법이다. 그림 11은 상호교환을 예를 들어서 보여 준다.



(그림 11) 돌연변이 연산
(Fig. 11) mutation operator

5.5 제어변수(control parameters)

두 개의 개체를 선택했을 때 반드시 교배를 하는 것은 아니다. 교배를 하지 않는 경우에는 두 개의 개체를 각각 다음 세대에 그대로 복사해 준다. 이때 선택한 두 개의 개체를 교배할 확률을 교배 비율(crossover rate)이라고 한다. 또한 크로모좀의 각 위치에 있는 유전자가 돌연변이를 일으킬 확률을 돌연변이 비율(mutation rate)이라고 부른다. 교배 비율 Pc 는 0.8, 돌연변이 비율 Pm은 0.001로 설정하였으며, 군집의 크기(population size)는 30, 세대수(number of generations)는 30로 설정하였다.

6. 실험 및 결과

본 논문에서 제안한 스케쥴 방법을 C 언어로 구현하여 삼보 ULTRA-1s 워크스테이션에서 수행하였다. 노드의 수 N 는 16개로 하였으며, 교통량 행렬은 포와송 분포(Poisson distribution)를 사용하여 생성하였다. 하나의 노드가 다른 한 노드에 보내는 패킷의 개수가 매개변수 λ 인 포와송 분포를 이룬다. 포와송 분포의 매개변수 λ 의 값은 0.1, 1, 3, 10 으로 하였으며, 각 λ 에 대하여 100개의 교통량 행렬을 생성하였다. 각 교통량 행렬에 대하여, 파장의 수 k 가 4인 경우와 8인 경우, 조율지연시간 δ 가 1, 2, 11, 20인 경우를 각각 수행하였다. 또한 각 경우에 대하여 송신자의 순서를 $t_1, t_2, \ldots, t_n$ 로 고정한 경우의 리스트 스케쥴을 구하여, 제안한 방법의 스케쥴과 비교하였다.

λ 가 각각 0.1, 1, 3, 10 인 경우에 두 노드 사이에 전송할 패킷의 개수의 범위는 각각 0 ~ 2, 0 ~ 6, 0 ~ 10, 0 ~ 22 이다. 각 노드가 다른 모든 노드에게 꼭 한 개의 패킷을 보내는 전방송(all-to-all broadcast)의 경우에 최적 스케쥴의 길이는 $\max\{\frac{N(N-1)}{k}, k\delta+N-1\}$ 임이 알려져 있다[1]. 여기서 $k=4$ 인 경우에 조율지연시간 δ 가 최적 스케쥴의 길이에 영향을 미치기 시작하는 $\delta=2$, $k=8$ 인 경우에 δ 가 영향을 미치기 시작하는 $\delta=11$ 을 선정하였으며, δ 가 상당히 큰 20도 선정하였다.

각 실험의 결과는 아래의 표 1, 2, 3, 4, 5, 6이 보여준다. 표 1, 3, 5는 $k=4$ 인 경우를, 표 2, 4, 6은 $k=8$ 인 경우를 나타낸다. 표에서 f 는 송신장치의 순서를 고정한 리스트 스케쥴을 나타내고, g 는 유전자 알고리즘을 이용한 리스트 스케쥴을 나타낸다. 표 1, 2는 스케쥴 길이의 평균을 나타낸다. 표 3, 4는 송신장치의 순서를 고정한 리스트 스케쥴에 의한 길이와 제

〈표 1〉 k=4 인 경우의 스케줄 길이의 평균값

〈Table 1〉 average schedule lengths when $k=4$

λ \ δ	1			2			11			20		
	f	g	$\frac{f-g}{f}$ %	f	g	$\frac{f-g}{f}$ %	f	g	$\frac{f-g}{f}$ %	f	g	$\frac{f-g}{f}$ %
0.1	9.97	9.75	2.21	11.91	11.29	5.21	37.19	35.95	3.33	64.38	63.15	1.91
1	69.57	69.38	0.27	70.51	70.38	0.19	83.14	79.62	4.23	116.15	109.18	6.00
3	195.52	195.41	0.06	198.3	198.22	0.04	208.31	207.22	0.52	218.21	216.22	0.91
10	626.47	626.43	0.006	628.32	626.81	0.24	632.5	632.08	0.06	648.95	645.69	0.50

〈표 2〉 k=8 인 경우의 스케줄 길이의 평균값

〈Table 2〉 average schedule lengths when $k=8$

λ \ δ	1			2			11			20		
	f	g	$\frac{f-g}{f}$ %	f	g	$\frac{f-g}{f}$ %	f	g	$\frac{f-g}{f}$ %	f	g	$\frac{f-g}{f}$ %
0.1	9.27	9.18	0.97	11.83	11.59	2.03	41.74	41.26	1.15	73.42	72.98	0.59
1	39.75	39.32	1.08	42.75	40.89	4.35	111.3	109.07	2.00	183.06	180.98	1.14
3	105.68	105.27	0.39	159.96	147.37	0.53	159.96	147.37	7.87	231.84	218.86	5.59
10	327.75	325.83	0.59	329.95	328.11	0.56	340.44	334.01	1.89	375.74	359.11	4.43

〈표 3〉 k=4 인 경우의 스케줄 길이의 차이의 최대값

〈Table 3〉 maximum among the 100 schedule length differences when $k=4$

λ \ δ	1			2			11			20		
	f	g	$\frac{f-g}{f}$ %	f	g	$\frac{f-g}{f}$ %	f	g	$\frac{f-g}{f}$ %	f	g	$\frac{f-g}{f}$ %
0.1	12	10	16.67	14	11	21.42	39	36	7.69	67	63	5.97
1	76	71	6.58	77	72	6.49	89	77	13.48	130	113	13.08
3	189	185	2.12	185	179	3.24	245	228	6.94	247	226	8.50
10	641	637	0.62	654	617	5.66	650	616	5.23	689	636	7.69

〈표 4〉 k=8 인 경우의 스케줄 길이의 차이의 최대값

〈Table 4〉 maximum among the 100 schedule length differences when $k=8$

λ \ δ	1			2			11			20		
	f	g	$\frac{f-g}{f}$ %	f	g	$\frac{f-g}{f}$ %	f	g	$\frac{f-g}{f}$ %	f	g	$\frac{f-g}{f}$ %
0.1	10	9	10	12	10	16.67	38	36	5.26	66	63	4.55
1	43	38	11.63	45	39	13.33	114	107	6.14	187	181	3.23
3	109	102	6.42	112	106	5.36	180	147	18.33	262	224	14.50
10	368	335	8.97	330	308	6.67	347	320	7.78	395	360	8.86

안된 방법에 의한 스케쥴 길이의 차이가 가장 큰 경우를 보여준다. 표 5, 6은 송신장치의 순서를 고정한 리스트 스케쥴의 결과가 최적 스케쥴임이 확인된 경우의 수와 제안된 방법에 의한 스케쥴의 결과가 최적 스케쥴임이 확인된 경우의 수를 보여 준다. 최적 스케쥴임이 확인된 경우는, 결과 스케쥴에서 모든 패킷의 수신

이 다 끝날 때 까지 걸리는 시간이 가장 긴 수신자 그룹의 수신 스케줄의 초기 조율시간은 제외하고는 idle time이 전혀 없이 송신장치들이 배치되어 있는 경우이다. 이러한 경우는 결과 스케줄이 최적 스케줄임을 알 수 있다. 예를 들어서, 그림 9에서 수신자 그룹 R2가 위에서 설명한 조건을 만족한다. 그러므로 그림 9의 스케줄은 최적 스케줄임을 알 수 있다.

실험 결과를 보면, 제안한 방법의 평균 스케줄 길이가 0.006%부터 7.87%까지 짧아짐을 알 수 있다. 제안한 방법의 스케줄 길이와 송신자의 순서를 고정한 리스트 스케줄 길이의 차이가 가장 큰 경우는 0.62% 와 21.42% 사이의 비율로 단축됨을 볼 수 있다. 또한 유전자 알고리즘을 이용한 방법이 많은 경우에 최적 스케줄을 구했음을 볼 수 있다. 표5, 6이 나타내는 것은 최적 스케줄임을 확인한 경우의 수들이므로, 확인하지는 못했지만 실제로 최적 스케줄인 경우의 수는 더 많을 것이다. 표 5, 6에서 δ 값이 커지면, idle time이 전혀 없는 수신자 그룹이 존재할 가능성이 희박해지므로, 최적 스케줄임이 확인된 경우의 수가 줄어듦이 당연하다. 이러한 경우들에는 최적 스케줄을 구하지 못한 것이 아니라, 최적 스케줄임을 확인하지 못한 것이다.

〈표 5〉 k=4 인 경우에 최적 스케줄임이 확인된 경우의 수
〈Table 5〉 The number of cases where the produced schedules are verified to be optimal when $k=4$

δ / λ	1		2		11		20	
	f	g	f	g	f	g	f	g
0.1	79	98	41	66	0	0	0	0
1	94	100	95	100	13	86	0	0
3	96	100	98	100	83	100	72	100
10	100	100	93	100	97	100	83	100

〈표 6〉 k=8 인 경우에 최적 스케줄임이 확인된 경우의 수
〈Table 6〉 The number of cases where the produced schedules are verified to be optimal when $k=8$

δ / λ	1		2		11		20	
	f	g	f	g	f	g	f	g
0.1	23	28	2	4	0	0	0	0
1	76	99	22	75	0	0	0	0
3	84	100	73	100	0	0	0	0
10	75	100	73	100	29	99	0	4

7. 결 론

파장분할다중방식의 패킷 전송 방법으로 통신하는 광상호연결망에서 노드들 사이에 전송할 패킷의 수가 임의로 주어지는 일반적인 경우의 스케줄링 문제에 대한 효율적인 방법을 제안하였다. 이 문제에 대하여 좋은 성능을 보이는 리스트 스케줄이 알려져 있다. 리스트 스케줄링은 각 시각에 배치하는 송신자를 고려하는 순서에 따라 스케줄 결과가 달라질 수 있다. 이 순서를 유전자 알고리즘을 이용하여 다양하게 시도하여 더욱 효율적인 스케줄을 얻는 방법을 제안하였다.

다양한 부하를 가지는 교통량 행렬들을 생성하여, 조율지연시간과 사용할 수 있는 파장의 수를 변화시키며 제안한 방법을 실험하여, 송신자의 순서를 고정한 경우의 리스트 스케줄보다 좋은 성능을 보임을 확인하였다. 또한 제안한 방법은 많은 경우에 최적 스케줄을 구해 내었다. 제안한 방법은 파장분할다중방식의 광상호연결망에서만이 아니라 리스트 스케줄링을 이용하는 여러 문제들에 적용하여 더욱 효율적인 스케줄을 구하는 방법으로 사용할 수 있다.

참 고 문 헌

[1] S. K. Lee, A. D. Oh, H. Choi, and H. A. Choi, "Optimal Transmission Schedules in TWDM Optical Passive Star Networks", Discrete Applied Mathematics 75(1), pp.81-91, Jan. 1997.

[2] H. Choi, H. A. Choi, M. Azizoglu, "On the All-to-All Broadcast Problem in Optical Networks", Proceedings of Infocom '97.

[3] G. N. Rouskas and V. Sivaraman, "On the Design of Optimal TDM Schedules for Broadcast WDM Networks with Arbitrary Transceiver Tuning Latencies", Proceedings of Infocom '96, pp.1217-1224, March 1996.

[4] H. Choi and H. A. Choi, "Complexity Results on Packet Transmissions Scheduling Problem in Broadcast-and-Select WDM Networks", Technical Report, Department of Electrical Engineering and Computer Science, George Washington University.

[5] H. Choi, H. A. Choi, M. Azizoglu, "Efficient

Scheduling of Transmissions in Optical Broadcast Networks", IEEE/ACM Trans. on Networking 4(6), pp.913-920, 1996.

[6] E. G. Coffman and P. J. Denning, "Operation Systems Theory", Englewood Cliffs, NJ: Prentice-Hall, 1973.

[7] D. E. Goldberg, "Genetic Algorithms in Search, Optimization, and Machine Learning", Addison Wesley, 1989.

[8] M. Mitchell, "An Introduction to Genetic Algorithms", MIT Press, 1996.

[9] E. S. H. Hou, N. Ansari, and H. Ren, "A Genetic Algorithm for Multiprocessor Scheduling", IEEE Trans. on Parallel and Distributed Systems 5(2), pp.113-120, 1994.

[10] Y. K. Kwok and I. Ahmad, "A Parallel Genetic Search-Based Algorithm for Scheduling Arbitrary Task Graphs to Multiprocessors", Proc. of PDCS '97, pp.255-344, 1997.

[11] M. Srinivas and L. M. Patnaik, "Genetic Algorithms: A Survey", Computer, pp.17-26, June 1994.

정 혜 진

1993년 경기대학교 전자계산학과(학사)

1997년~현재 아주대학교 컴퓨터공학과 석사과정

관심분야 : 컴퓨터 이론, 유전자 알고리즘

위 규 범

1978년 서울대학교 수학과(학사)

1984년 University of Wisconsin 전산학과(석사)

1992년 Indiana University 전산학과(박사)

1993년~현재 아주대학교 정보 및 컴퓨터공학부 조교수

관심분야 : 컴퓨터 이론

예 홍 진

1986년 서울대학교 수학교육과(학사)

1988년 아주대학교 전자계산학과(석사)

1990년 Univ. Joseph Fourier-INP Grenoble(佛), 응용수학과(D.E.A.)

1993년 Univ. Claude Bernard - ENS Lyon(佛), 전자계산학과(박사)

1993년~현재 아주대학교 정보 및 컴퓨터공학부 조교수

관심분야 : 컴퓨터 산술, 병렬 알고리즘과 구조, VLSI 알고리즘과 구조

홍 만 표

1981년 서울대학교 계산통계학과(학사)

1983년 서울대학교 계산통계학과(석사)

1991년 서울대학교 전산과학과(박사)

1985년~현재 아주대학교 정보 및 컴퓨터공학부 교수

관심분야 : 병렬처리, 광상호연결망, 시스템 성능분석

변 광 준

1985년 서울대학교 전자계산기공학과(학사)

1987년 Pennsylvania State Univ., Computer Science(석사)

1993년 Univ. of Southern California, Computer Science(박사)

1994년~현재 아주대학교 정보 및 컴퓨터공학부 조교수

관심분야 : 데이터베이스, 분산객체 컴퓨팅