經營科學
第20卷 第1號
2003年 5月

유전알고리즘에 기반한 Job Shop 일정계획 기법*

박병주** · 최형림*** · 김현수***

A Genetic Algorithm-based Scheduling Method for Job Shop Scheduling Problem*

Byung Joo Park** · Hyung Rim Choi*** · Hyun Soo Kim***

■ Abstract ■

The JSSP (Job Shop Scheduling Problem) is one of the most general and difficult of all traditional scheduling problems. The goal of this research is to develop an efficient scheduling method based on genetic algorithm to address JSSP. we design scheduling method based on SGA (Single Genetic Algorithm) and PGA (Parallel Genetic Algorithm). In the scheduling method, the representation, which encodes the job number, is made to be always feasible, initial population is generated through integrating representation and G&T algorithm, the new genetic operators and selection method are designed to better transmit the temporal relationships in the chromosome, and island model PGA are proposed. The scheduling method based on genetic algorithm are tested on five standard benchmark JSSPs. The results were compared with other proposed approaches. Compared to traditional genetic algorithm, the proposed approach yields significant improvement at a solution. The superior results indicate the successful incorporation of generating method of initial population into the genetic operators.

Keyword : Gentic Algorithm, Scheduling, Job Shop

1. 서 론

어떤 분야에서든 의사결정자들은 문제 해결에 있어 최적해를 찾아내려는 노력을 한다. 하지만 불행히도 job shop 일정계획, 조립 라인밸런싱, 프로젝트 일정계획, 설비 위치문제 등과 같은 일반적인

논문접수일 : 2001년 11월 19일 논문게재확정일 : 2002년 11월 28일
* 이 논문은 2001년도 두뇌한국 21 사업에 의하여 지원되었음.
** 동아대학교 경영정보학과
*** 동아대학교 경영정보과학부

사업 운영에 관련된 문제들은 최적해를 얻기 어려운 조합 최적화 문제이다. 이들 대부분에서 실제적인 크기의 문제를 해결하기 위한 시간과 계산 자원들은 매우 부족하나, 여전히 의사 결정자들은 좋은 해들을 요구한다. 이 경우는 휴리스틱 방법들이 유일한 대안이 된다. 1980년대 이후 타부 서치, 시뮬레이티드 어닐링, 유전 알고리즘 등의 반복적 알고리즘에 관한 연구가 활발히 이루어지고 있다. 그들 각자는 좋은 국소 최적해를 찾기 위해 문제의 해 영역들을 탐색해 가는 전략들을 정의하고 있다. 본 연구에서는 적자생존이라는 진화개념을 사용하는 유전 알고리즘을 job shop 일정계획 문제의 접근법으로 사용한다.

초기 유전 알고리즘은 job shop 일정계획 문제에서 염색체 표현을 위해 이진 스트링을 사용하였다. 이러한 표현은 TSP(Traveling Salesman Problem), JSSP(Job Shop Scheduling Problem)와 같은 문제에 있어서는 적합하지 않다. 이진 스트링의 표현을 사용한 경우, 적용한 단순한 교차연산이나 돌연변이 연산은 항상 실행 불가능한 해를 산출한다. 그래서 이 문제를 다루기 위해 Bierwith [5], Syswerda[20]등은 새로운 표현 방법을 사용하였고, Yamada & Nakano[23], Dorndorf & Pesch [12]등은 표준 유전 연산자들을 변형하거나 현존하는 알고리즘과의 혼합사용(hybridization)을 하기도 하였다. Davis[11], Bagchi 등[4], UcKun 등[22]은 염색체 표현이 직접적으로 스케쥴을 표현하지는 않지만, 디코더나 스케쥴 빌더(builder)를 통해 실행 가능한 스케쥴로 전환하는 방법을 사용하였다. Nakano & Yamada[19], Tamaki & Nishikawa[21]은 실행 불가능한 해를 보정(repair)하여 포싱(forcing)하는 방법들을 사용하였다. 포싱은 보정된 실행 가능한 스트링을 가지고 원래의 실행 불가능한 스트링을 교체하는 과정을 말한다.

이들 연구들에서는 염색체의 실행 가능성을 유지하기 위해 적절한 표현 방법이나 보정과 포싱과정 또는 타 알고리즘과 결합된 디코더 방법 등이 필요했다. 그러나 만약 염색체의 표현에서 유전 연산 후에도 항상 실행 가능성을 유지 할 수 있다면, 보정하여 포싱하거나, G&T(Giffler & Thompson) 알고리즘[16]과 결합된 디코더 과정이 필요 없게 되어 알고리즘 적용과정을 단순화 할 수 있을 것이다. 그리고 대부분의 연구들에서 초기 모집단을 개체의 다양성을 위해 임의대로 생성시켰으나, 국소 탐색에서 최적해에 근접은 좋은 초기해에서 탐색을 시작하는 것으로 가능성을 더 높일 수 있다. 그래서 항상 실행가능하고, 다양성을 가진 좋은 개체로 초기 모집단을 구성할 수 있는 방법과 이들을 잘 유전시킬 유전 연산자, 선택 방법이 있다면 더 나은 해를 구할 수 있을 것이다[1].

또한 전통적인 유전 알고리즘에서 일반적인 문제는 조기 수렴(premature convergence)인데, 다른 국소 탐색 절차 보다 조기 수렴을 방지하면서도 이의 영향을 받는다. 유전 알고리즘에서 조기 수렴을 감소시키기 위한 방법은 초기 모집단을 여러 부 집단으로 분리하는 유전 알고리즘의 병렬화이다. 병렬 유전 알고리즘(Parallel Genetic Algorithms : PGA)의 이점은 병렬 프로세서의 사용을 통해 수행 시간을 줄일 수 있다는 것과 모집단의 분리를 통해 개체의 다양성을 유지할 수 있다는 것이다. 본 연구에서는 PGA의 부 집단에서 더욱 다양한 개체를 생성시키기 위해 4가지의 초기 모집단 생성방법과 유전방법을 제시하고, 이들을 바탕으로 job shop 일정계획 문제를 보다 효과적으로 해결하기 위한 일정계획 기법을 제시하고자 한다.

2. 유전알고리즘

유전 알고리즘은 하나의 해를 이용하는 다른 최적화 방법과는 달리 후보 해들의 집단을 가지고 수행한다. 그리고 문제의 목적함수에 따라 집단 내의 각 개체들에 값을 할당한 후, 적자생존 단계로 이전 집단에서 수행도가 좋은 개체들을 선택하고, 복제 단계에서는 이전 집단보다 더욱 좋은 집단을 산출하기 위해 선택된 개체들에 교차 또는 돌연변이와 같은 유전 연산자들을 적용해 우수한 형질의 개

체를 생성하는 진화된 방향으로 탐색을 진행시킨다. 유전 알고리즘을 사용하기 위해서는 문제에 대한 특성을 먼저 분석한 다음, 그 문제에 적합한 표현방법, 목적함수, 초기 모집단 구성방법, 유전연산자, 유전 파라미터 등이 결정되어야 한다.

2.1 표현 방법

유전 알고리즘으로 job shop 일정계획 문제를 해결하기 위해서는 먼저 문제의 해를 염색체로 표현해야 한다. 표현은 job 번호를 공정의 수만큼 반복시키는 형태로 이루어진다[1]. 하나의 유전인자는 하나의 공정을 의미하고 표현형태는 처리되는 공정의 job 번호로 표현한다. 이처럼 job 번호의 반복을 통해 표현한 염색체는 job의 공정들이 일정계획 되어지는 순서를 나타낸다. 예를 들어 3개 job과 3대 기계 문제는 순열 형태의 염색체로 <그림 1>과 같이 표현된다. 세 번씩 반복된 숫자는 job의 번호이고, job 번호가 세 번씩 반복된 것은 각 job들이 3개의 공정을 가지고 있기 때문이다. job 번호의 첫 번째 반복은 그 job의 첫 공정을 두 번째 반복은 두 번째 공정을 의미한다. 이 염색체는 job 번호가 공정의 수만큼만 표시된다면 항상 실행가능성을 유지한다. 인덱스(index) 값은 job 번호의 반복 횟수로, 그 job에서 몇 번째 공정인지를 나타낸다. 이는 교차연산시 대응되는 공정을 지정하기 위해 사용된다.

Chromosome	[1 2 2 1 3 1 2 3 3]
Index	1 1 2 2 1 3 3 2 3

〈그림 1〉 염색체의 표현

염색체의 생성은 G&T 알고리즘을 이용한다. G&T 알고리즘은 단계 4의 G집합 내에 속해 있는 공정들을 하나씩 선택하는 과정을 전체 공정의 수만큼 수행하여 일정계획 하게 되는데, 이때 하나씩 선택되어지는 공정을 포함하는 job의 번호를 선택순서대로 전체 공정 수만큼 연결하여 기입함으로써 염색체는 생성되어진다.

2.2 초기 모집단

국소 탐색법에서 초기 값은 해에 많은 영향을 미친다. 기존의 JSSP에 적용된 유전 알고리즘은 대부분 초기 모집단을 임의대로 발생시켜 사용해 왔다. 이로 인해 해를 찾는데 걸리는 시간은 길어지고 좋은 해를 찾을 가능성도 줄어들 수 있다. 본 연구에서는 우수한 개체로 구성된 초기모집단을 생성시키기 위해 G&T 알고리즘을 사용한다. 모집단 구성에서 중요한 것은 다양성이라 할 수 있는데, G&T 알고리즘은 job shop 일정계획 문제에서 다양한 active 스케쥴을 가장 쉽게 산출할 수 있는 알고리즘이다. G&T 알고리즘으로 active, non-delay 스케쥴을 산출하고, 변형된 G&T 알고리즘으로 active' 스케쥴을 산출하여, 이 스케쥴들로 초기 모집단을 구성한다. active' 스케쥴은 단계 3에서 G집합을 구성하는 방법을 수정한 G&T 알고리즘으로 구한다. 그 알고리즘은 다음과 같다.

2.2.1 기 호

j : job의 번호

m : 기계의 번호

P_{jm} : job j의 기계 m에서 가공시간

r_{jm} : job j의 기계 m에서 시작시간

2.2.2 Giffler & Thompson(G&T) 알고리즘 (active 스케쥴)

단계 1 : 각 job들 중에서 가장 먼저 일정계획 해야 할 공정들을 집합 C로 둔다. 집합 C내의 모든 공정 (j, m)에 대한 시작시간 $r_{jm} = 0$ 이라 둔다.

단계 2 : $t(C) = \min_{(j,m)\in C}\{r_{jm} + p_{jm}\}$를 계산한다. 그리고 t(C)가 최소가 되는 기계를 $m^{\star}$로 둔다.

단계 3 : 기계 $m^{\star}$상에서 $r_{jm^{\star}} < t(C)$인 모든 공정 $(j, m^{\star})$들을 집합 G로 둔다.

단계 4 : 집합 G에서 하나의 공정을 임의대로 선택하고 그것을 일정계획 한다.

단계 5 : C에서 그 공정을 삭제한다. 그리고 job에서 그 공정의 후행 공정을 집합 C에 포함시킨다. C에서 r_{jm}을 수정하고 모든 공정이 일정계획 될 때까지 단계 2로 돌아간다.

non-delay 스케쥴을 산출하기 위한 G&T 알고리즘은 위의 알고리즘에서 단계 2와 단계 3이 달라진다.

단계 2 : $t(C) = \min_{(j,m)\in C} \{r_{jm}\}$를 계산한다.

그리고 t(C)가 최소가 되는 기계를 $m^{\star}$로 둔다.

단계 3 : 기계 $m^{\star}$상에서 $t(C) = r_{jm^{\star}}$인 모든 공정 $(j, m^{\star})$들을 집합 G로 둔다.

또한, active 스케쥴을 산출하는 G&T 알고리즘의 단계 3을 아래와 같이 수정하고, 여기서 얻어지는 스케쥴을 active' 스케쥴이라 한다. 이는 기계 $m^{\star}$상에서 처리되어지는 공정 중 가장 빠른 시작시간을 가진 공정들을 먼저 할당하여 기계의 유휴시간을 최소화한 active 스케쥴을 구하기 위함이다.

단계 3 : 기계 $m^{\star}$ 상에서 $r_{jm^{\star}} < t(C)$인 모든 공정 $(j, m^{\star})$들 중에서 $r_{jm^{\star}}$이 가장 작은 공정들을 집합 G로 둔다.

여기서 active 스케쥴은 어떤 다른 공정의 지연이 없이 가공순서를 변경해서도 더욱 일찍 마칠 수 있는 공정이 없는 실행 가능한 스케쥴이고, non-delay 스케쥴은 일부 공정의 가공을 시작할 수 있을 시점에 유휴상태에 있는 기계가 없는 실행가능한 스케쥴이다. 그리고 active' 스케쥴은 수 없이 많은 active 스케쥴 중에서 축소된 active 스케쥴 집합에 속하는 실행가능한 스케쥴이다.

2.3 교차연산자(crossover)

G&T 알고리즘으로 얻은 염색체들이 좋은 스케쥴을 가지고 있기에 가능하면 그들 순서관계를 유지하면서 조금씩 개선시켜 나갈 수 있는 연산자가 필요하다. 본 연구에서는 PGA에서의 적용을 위해 기존 교차연산자들을 개선한 4개의 교차 연산자를 사용한다.

교차 연산자 1은 Bierwith[6]의 PPX 연산자를 변형한 것으로, 먼저 두 부모 1과 두 부모 2를 선택 한 후 두 부모중 한 부모로부터 유전인자를 상속받기 위해 부모를 나타내는 숫자 1, 2를 임의대로 유전인자 수만큼 발생시킨다. 그리고 나서 그 난수에 해당하는 부모로부터 하나의 유전인자를 물려받고 다른 부모에서는 물려받은 유전인자와 같은 속성을 가진 유전인자를 삭제하는 과정을 난수의 수만큼 되풀이한다.

교차 연산자 2는 두 부모 중, 부모 1에서 두 점을 이용한 임의의 구간을 만든다. 그리고 이 임의의 구간내의 유전인자들을 부모 2의 염색체에 삽입하는데, 이때 삽입 위치는 임의의 구간 내 시작 유전인자와 같은 속성을 가지는 유전인자 다음이다. 그리고 부모 2에서 임의 구간내의 유전인자와 같은 속성을 가지고 있는 인자들을 삭제하여 하나의 자식 개체를 생성한다.

교차 연산자 3은 Bierwith[5]의 GPMX 연산자를 변형한 것으로, 먼저 부모 1에 두 점을 이용한 임의의 구간을 만든다. 그리고 부모 2에서 임의 구간내의 유전인자와 같은 속성을 가진 인자들을 먼저 삭제한다. 그리고 임의구간의 시작위치와 같아지도록 부모 2에 삽입한다. 즉 임의구간의 시작위치가 4번째라면 삽입위치는 부모 2에서 삭제된 유전인자를 제외하고 4번째가 될 수 있는 위치가 된다.

교차 연산자 4에서도 먼저 부모 1에서 임의 구간을 정한 뒤 그 구간 내 유전인자들을 부모 2에 삽입한다. 삽입 위치는 임의 구간이 시작된 유전인자 바로 앞이다. 만약 첫 번째 부모에서 임의구간

의 시작 위치가 4번째라면 삽입 위치는 부모 2의 4번째 유전인자 앞이 된다. 그리고 나서 임의 구간 내의 유전인자와 속성이 같은 인자들을 부모 2에서 삭제한다.

이들 4개의 교차연산 과정은 자식 1을 산출한 후, 부모 1과 부모 2를 바꾸어서 같은 난수와 임의의 구간으로 두 번째 자식 개체를 생성한다. 그리고 두 자식 개체 중 평가기준에 적합한 한 개체만을 다음 세대로 보낸다. 교차 연산 결과들은 <그림 2>와 같다.

2.4 돌연변이(Mutation)

돌연변이 연산자는 염색체에 변화를 주어 집단 내의 다양성을 유지하기 위하여 사용한다. 본 연구에서는 이웃 탐색 기법에 근거한 돌연변이 연산자를 사용한다[9]. <그림 3>은 돌연변이 연산의 예를 보여준다. 이 돌연변이 연산자는 case 2에서 case

				임의 구간					
부모 1	3	2	2	2	3	1	1	1	3
index	1	1	2	3	2	1	2	3	3
부모 2	1	1	3	2	2	1	2	3	3
index	1	2	1	1	2	3	3	2	3
난수	1	1	2	2	2	2	1	1	1
교차연산자 1의 자식 1	3	2	1	1	2	1	2	3	3
자식 2	1	1	3	2	2	2	1	3	3
교차연산자 2의 자식 1	3	2	2	1	2	3	1	1	3
자식 2	3	2	2	1	2	3	1	1	3
교차연산자 3의 자식 1	3	2	2	2	3	1	1	1	3
자식 2	3	3	1	2	2	1	2	1	3
교차연산자 4의 자식 1	3	2	3	1	1	2	2	1	3
자식 2	3	2	2	1	2	3	1	1	3

〈그림 2〉 교차연산자들의 결과

부모 염색체	1	2	3	1	2	3	1	2	3
	↓				↓				↓
이웃해 염색체									
case 1	1	2	3	1	2	3	1	2	3
case 2	2	2	3	1	1	3	1	2	3
case 3	2	2	3	1	3	3	1	2	1
case 4	3	2	3	1	2	3	1	2	1
case 5	3	2	3	1	1	3	1	2	2
case 6	1	2	3	1	3	3	1	2	2

〈그림 3〉 돌연변이 연산의 예

6까지의 부모와 다른 5가지만 이웃해로 비교하여 가장 좋은 것을 유전시키는 방법과 현 부모와 5개의 이웃해 모두를 비교해서 가장 좋은 것을 다음 세대로 전달하는 두 개의 형태로 사용한다. 전자의 경우는 일반적인 돌연변이 과정에서 사용하고 후자의 경우는 교차 연산자 수행 후 더 나은 개체 생성 가능성을 확인하기 위해 사용한다.

2.5 선택 방법

본 연구에서는 선택 방법으로 씨종자 선택과 토너먼트 선택을 사용한다. 씨종자 선택은 일상에서 사용하는 개체 선택과 좋은 개체 보존 방법을 유전 알고리즘 진화과정에 도입한 것이다[1]. 가축을 사육하는 곳에서는 주로 개체 증식을 위해서 우수한 개체를 주로 씨종자로 사용하여 다음 세대를 구성해 나간다. 이 과정을 선택 방법에 적용하여 두 부모 중 부(父)에 해당하는 개체는 임의로 발생시킨 값이 확률 값(0.9)의 범위 내에 들면 한 집단 내에서 씨종자 범위의 순위 내에 드는 우수한 개체를 선택하고 그렇지 않으면 전체 집단에서 하나를 임의대로 선택한다. 나머지 모(母)는 전체 집단 내에서 임의대로 선택하는데, 두 개체를 임의대로 선택하여 일정한 확률 값에 따라 적합도가 좋은 개체 하나만을 선택한다. 이들을 부모로 사용하고 개체집단에 되돌려 다시 선택할 수 있게 한다. Goldberg & Deb[17]가 제시한 토너먼트 선택은 집단으로부터 두 개의 개체들을 임의대로 선택한 다음, 난수 r을 0과 1사이에서 발생시켜 만약 $r < k$이면 두 개의 개체 중 적합도가 높은 개체를 부모로 선택한다. 그렇지 않으면 덜 적합한 개체를 선택한다. 그런 다음 이 두 개체는 다시 원래의 개체집단에 되돌려지고 다시 선택되어질 수 있다. 여기서 k는 선택확률을 나타내는 파라미터이다.

2.6 평가 함수

JSSP에서 최소의 makespan을 가진 스케쥴은 종종 기계의 높은 효율을 의미한다. 대부분의 JSSP 벤치마크 문제의 목적들이 makespan을 최소로 하는 것이므로, makespan은 JSSP에서 일정계획 기법의 비교나 평가를 위해 선택한다. 순열 형태의 염색체로 표현하였을 때 makespan은 왼쪽에서 오른쪽으로 유전인자를 읽어, job의 선후관계를 지키면서 기계에 할당하여 구한다.

2.7 교 체

다음 세대의 구성은 현 세대에서 선택과 유전 연산자들을 이용하여 새롭게 구성한다. 새로운 개체들을 초기 모집단의 개수만큼 생성하여 다음 세대를 구성하고 난 뒤 엘리티즘을 적용하여 나쁜 개체는 엘리티즘 적용 개수만큼 좋은 개체로 다시 대체한다. 또한 교차율과 돌연변이율에 따라 일부 개체들은 유전 연산자를 거치지 않고 그대로 다음 세대로 이동하도록 한다.

3. 병렬 유전알고리즘

본 논문에서는 다양한 초기 모집단 생성 방법과 교차연산자 그리고 선택 방법을 사용한다. 이들을 각각 단일 집단에 적용하여 해를 산출하는 것보다, 단일 집단을 여러 집단으로 나누어 각 집단에 다른 초기 모집단, 연산자, 선택 방법을 사용하여 독립적으로 진화시키면서 일부 개체를 서로 교류할 수 있게 하면 수렴현상을 줄여 보다 나은 해의 탐색이 가능 할 수 있다. 이것이 PGA의 사용 이유이며, 이처럼 단일 집단을 여러 집단으로 나누어 서로 독립적인 유전 알고리즘으로 전개하고, 일부 개체가 한 집단에서 다른 집단으로 이주(migration)할 수 있는 PGA를 섬모델 PGA라 한다.

섬모델 PGA는 이주 방법, 연결 구조, 부 집단의 동질성에 따라 분류된다. 이주 방법에는 이주가 없는 경우, 동시에 이루어지는 경우, 동시에 이루어지지 않는 경우가 있고, 연결 구조에는 부 집단들 사이의 연결구조가 시작에서 설정되어 수정되지

않는 정적 연결 구조와 변화하는 동적 연결 구조가 있다. 부 집단의 동질성에는 각 부 집단마다 같은 파라미터를 사용하는 경우와 다른 파라미터를 사용하는 경우로 나눌 수 있다. 본 연구에서 섬모델 PGA는 링형의 정적 연결 구조에 집단마다 다른 파라미터를 사용하며, 이주는 동시에 이루어지도록 하였다. 그리고 부 집단의 수를 2개와 4개로 하였는데, 전자를 2-PGA라하고 후자를 4-PGA라 한다. 이 PGA에 사용되는 초기모집단 구성 방법과 교차연산자는 <표 1>과 같다.

또한 이주개체 선별과 교체 방법은 이주시기에 각 집단에서 정한 개수만큼 임의 선별하여 교환하는 방법과 각 부 집단에서 정해진 순위 안에 드는 개체를 복사하여 연결되어 있는 다른 집단의 나쁜 개체를 교체하는 방법 중 나은 결과를 보여준 후자의 방법을 사용하였다. 이주크기와 구간은 실험을 통해 집단크기의 10%와 50세대로 정하였다.

4. 실험 및 분석

제안 알고리즘을 다섯 개의 벤치마크 문제(MT 문제, CAR 문제, ORB 문제, YN 문제, ABZ 문제)에 적용하여 얻은 결과를 다른 연구자들의 결과와 비교해서 수행도를 평가한다. 먼저 알고리즘을 적용하기 위해서는 유전 파라미터들이 결정되어야 하는데, 모집단의 크기와 세대수는 200과 1000으로 하고, 교차율, 토너먼트 선택에서의 선택확률, 엘리티즘의 크기, 씨종자 선택범위는 실험을 통하여 결정하였다. 실험에 사용된 컴퓨터는 팬티엄Ⅱ 350, 메모리 64M의 기종을 사용하였다.

4.1 유전 파라미터의 설정

제시한 알고리즘에 적용할 교차율, 선택확률, 엘리티즘 크기는 MT10 문제에서 하나의 초기 모집단을 구성하고, 파라미터에 따라 100회 실행하여 가장 좋은 값과 평균값을 얻어내는 값으로 하였다. 교차율은 0.6, 0.7, 0.8일 경우, 선택확률은 0.7, 0.75, 0.8일 경우, 엘리티즘의 크기는 5~30일 경우에 대해 실험하였다. 돌연변이율은 0.1로 하였다.

<표 2>의 평균해의 비교를 통해, 교차연산자 1은 교차율 0.7, 선택확률 0.7, 엘리티즘 크기 10으로, 교차연산자 2는 교차율 0.7, 선택확률 0.75, 엘리티즘 크기 10으로, 교차연산자 3은 교차율 0.7, 선택확률 0.75, 엘리티즘 크기 20으로, 교차연산자 4는 교차율 0.8, 선택확률 0.75, 엘리티즘 크기를 10으로 결정한다. 그리고 씨종자 선택에서 씨종자(父)의 선택범위는 앞에서 정해진 파라미터 값에 선택범위를 상위 20위 내에서 60위까지로 하여 가장 좋은 범위를 찾기 위한 실험을 하였다. <표 3>의 결과를 통해 선택 범위를 교차연산자 1과 2는 40, 3은 60, 4는 30으로 한다.

〈표 1〉 PGA 모델

모 델		교차 연산자(선택 방법)	부집단 구성방법	부집단 크기	이주 구간	이주 크기
2-PGA	M 1	Crossover 1 (seed) Crossover 4 (tournament)	active' non-delay	100 100	50	10
	M 2	Crossover 2 (tournament) Crossover 3 (seed)	active' active'	100 100	50	10
4-PGA	M 3	Crossover 1 (seed) Crossover 2 (tournament) Crossover 3 (seed) Crossover 4 tournament)	active' non-delay active random	50 50 50 50	50	5
	M 4	Crossover 1 (seed) Crossover 2 (tournament) Crossover 3 (seed) Crossover 4 tournament)	active' active' active' active'	50 50 50 50	50	5

〈표 2〉 파라미터 실험 결과

파라미터		교차연산자 1			교차연산자 2			교차연산자 3			교차연산자 4		
교차율	선택 확률	엘리티즘	평균해	최고해	엘리티즘	평균해	최고해	엘리티즘	평균해	최고해	엘리티즘	평균해	최고해
0.6	0.75		966.06	937		961.66	937		966.34	937		960.27	951
	0.75		965.17	937		961.02	951		966.84	937		958.76	945
0.7	0.7	5	966.18	939	5	963.79	951	10	967.95	937	10	959.69	945
	0.8		965.92	937		964.54	951		967.54	937		959.26	951
0.8	0.75		967.61	939		961.32	951		966.52	937		957.49	951
0.6	0.75		972.16	937		962.87	937		966.52	937		961.27	935
	0.75		966.85	937		960.84	951		965.8	937		958.83	937
0.7	0.7	10	964.52	937	10	966.07	937	15	967.58	937	20	960.42	951
	0.8		964.67	937		963.27	951		968.68	951		960.54	951
0.8	0.75		964.59	937		961.49	951		968.91	951		960.67	951
0.6	0.75		971.5	937		962.44	937		966.98	951		962.57	940
	0.75		969.27	937		963.17	951		965.5	937		962.53	951
0.7	0.7	15	969.67	937	15	963.59	951	20	968.58	937	30	963.62	937
	0.8		972.26	942		964.35	951		967.86	951		959.52	951
0.8	0.75		969.71	936		962.39	937		966.04	937		960.05	951

〈표 3〉 씨종자 선택 범위에 대한 실험 결과

교차연산자	파라미터				평균해	최고해
	엘리티즘	교차율	선택 확률	씨종자 선택범위		
교차연산자 1	10	0.7	0.7	20	969.99	937
				30	968.54	937
				40	962.15	937
				50	964.35	937
				60	962.51	936
교차연산자 2	10	0.7	0.75	20	960.84	951
				30	960.57	937
				40	960.45	937
				50	962.75	951
				60	962.48	951
교차연산자 3	20	0.7	0.75	20	967.11	937
				30	967.8	937
				40	968.19	951
				50	969.13	951
				60	967.79	937
교차연산자 4	10	0.8	0.75	20	960.25	951
				30	959.13	951
				40	959.28	951
				50	959.25	951
				60	959.49	945

4.2 초기모집단 구성

단일집단 유전 알고리즘에서 초기모집단 구성 방법을 선정하기 위해 MT10 문제에서 active', active, non-delay, random, active 스케줄과 non-delay 스케줄의 혼합으로 각각 초기 모집단을 구성하여 결과 해들을 비교하였다. 실험에는 교차연산자 1과 4가 사용되었고, 토너먼트 선택을 사용한 경우로 하였다. 파라미터 값은 엘리티즘 크기 10, 교차율 0.7, 돌연변이율 0.1, 선택확률 0.75로 고정하고, 한 번 실행마다 스케줄 별로 새로운 초기 모집단을 구성하여 사용하였다. 이를 각 모집단 별로 100회 실행하여 최고해와 평균해를 구해 비교하였다. <표 4>에서 수정된 G&T 알고리즘으로 구한 active' 스케줄로 초기 모집단으로 구성한 경우의 결과가 가장 좋음을 알 수 있다. 단일 집단 유전 알고리즘을 벤치마크 문제에 적용할 때의 초기 모집단 구성은 active' 스케줄로 한다.

4.3 유전 알고리즘의 수행도 평가

각 벤치마크 문제마다 수정된 G&T 알고리즘으로 새로운 초기 모집단을 구성하고, 유전 연산자를 통해 최대 세대수만큼 진화시킨 집단에서 가장 좋은 해를 산출하는 과정을 50회 수행하여 그 중 가장 좋은 해를 구한다. 그리고 타 알고리즘으로 구한 해와의 비교를 통해 제시한 기법의 수행도를 평가한다.

4.3.1 MT 문제

MT 문제는 벤치마크 문제 중 가장 많이 사용되는 문제로 Muth & Thomson[18]에 의해 제시되었다. 이 문제는 3가지 크기의 문제를 가지고 있는데, 이 중 MT10, MT20은 제안되는 거의 모든 job shop 일정계획 알고리즘들이 벤치마크 문제로 사용하고 있다. 본 연구에서는 MT6, MT10 문제에서는 55, 930으로 최적해를 구할 수 있었으며, MT20 문제에서는 1173으로 근접해를 구하였다.

<표 5>는 세 개의 MT 문제에서 이전의 연구에 의해 얻어진 최고의 결과들이고, <표 6>은 교차연산자와 선택방법을 달리 적용해서 얻어진 결과들이다.

〈표 4〉 초기모집단 구성 방법에 따른 결과 비교

교차 연산자 \ 초기 모집단		Active' 스케줄 (200)	Non-delay 스케줄 (200)	Active 스케줄 (200)	Random 스케줄 (200)	Active' 스케줄(100) + Non-delay 스케줄(100)
교차 연산자 1	평균해	978.14	985.13	988.59	999.1	983.25
	최고해	936	938	939	942	937
교차 연산자 4	평균해	979.1	980.67	981.12	984.67	979.73
	최고해	930	930	937	945	938

〈표 5〉 MT 문제에서 이전 연구들의 결과

문제 \ 비교 논문	최적해	Nakano & Yamada[19]	Yamada & Nakano[23]	Gen[15]	Fang[14]	Dorndorf1 & Pesch[13]	Dorndorf2 & Pesch[13]	Croce [10]	Cheng [15]	Bierwirth [5]
MT 6 (6×6)	55	55	55	55	-	55	55	55	55	55
MT 10 (10×10)	930	965	930	962	949	960	938	946	948	936
MT 20 (20×5)	1165	1215	1184	1175	1189	1249	1178	1178	1196	1181

〈표 6〉 MT 벤치마크 문제의 결과

문제 \ 기법	수행시간 (1회 Run)	교차연산자 1		교차연산자 2		교차연산자 3		교차연산자 4	
		토너먼트	씨종자	토너먼트	씨종자	토너먼트	씨종자	토너먼트	씨종자
MT6	38초	55	55	55	55	55	55	55	55
MT10	77초	936	936	930	937	936	936	930	930
MT20	76초	1178	1173	1178	1178	1173	1178	1173	1173

여기서 이전 연구의 결과들에 비해 제시한 기법으로 더 나은 해를 구할 수 있음을 볼 수 있다. 표에서의 음영은 최적해나 결과들 중의 최고해를 의미한다.

4.3.2 CAR 문제

CAR 문제는 Carlier[7]에 의해 제시된 다양한 크기의 벤치마크 문제이다. 이 문제는 제시한 기법의 작고 다양한 크기의 문제에서의 수행도를 평가하기 위해 사용하였다. <표 7>에서 괄호안의 숫자는 50회 수행에서 얻은 최적해의 산출 횟수이다. 이 결과들을 통해 작고 다양한 크기의 문제에서는 교차연산자 1과 씨종자 선택이 가장 적합함을 확인할 수 있다. 교차연산자 1, 3, 4와 씨종자 선택을 사용한 경우는 모든 CAR 문제에서 최적해를 구하였다. PGA의 모델들도 모두 최적해를 구하였다.

4.3.3 ORB 벤치마크 문제

ORB 벤치마크 문제는 Applegate & Cook[3]에 의해 특별히 어렵게 만들어진 10개의 10×10 문제이다. 이 문제에서 얻은 결과들은 <표 8>과 같다. 이 표에서 2열은 Adams 등[2]의 Bottle-5로 얻은 결과이고, 3열은 Applegate & Cook의 shuffle 알고리즘으로 구한 결과이다. 4열은 Chamber[8]의 tabu search에 의한 결과이다. 여기서는 CAR 문제와 달리 교차연산자 1의 수행도가 낮음을 확인 할 수 있고, 나머지 경우는 대부분 타 알고리즘의 결과 보다 우수함을 볼 수 있다.

4.3.4 ABZ 벤치마크 문제

ABZ 벤치마크 문제는 Adams 등[2]에 의해 만들어진 5개의 문제이다. Adams는 5개의 문제에 shifting bottleneck(SB) 알고리즘을 적용하였다. <표 9>는 SB I, SB II와 제안 유전알고리즘으로 구한 결과들을 보여준다.

4.3.5 YN 벤치마크 문제

YN 문제는 Yamada & Nakano[23]에 의해 만들

〈표 7〉 CAR 벤치마크 문제의 결과

문 제	크 기	최적해	수행시간 (1회 Run)	교차연산자 1		교차연산자 2		교차연산자 3		교차연산자 4	
				토너먼트	씨종자	토너먼트	씨종자	토너먼트	씨종자	토너먼트	씨종자
CAR1	11×5	7038	48초	7038(50)	7038(50)	7038(38)	7038(40)	7038(34)	7038(35)	7038(40)	7038(43)
CAR2	13×4	7166	43초	7166(33)	7166(35)	7166(15)	7166(13)	7166(11)	7166(9)	7166(8)	7166(14)
CAR3	12×5	7312	47초	7400(0)	7312(1)	7399(0)	7312(1)	7312(2)	7312(1)	7312(1)	7312(1)
CAR4	14×4	8003	44초	8003(32)	8003(29)	8003(24)	8003(21)	8003(20)	8003(20)	8003(25)	8003(29)
CAR5	10×6	7702	47초	7732(0)	7702(1)	7714(0)	7720(0)	7720(0)	7702(1)	7720(0)	7702(1)
CAR6	8×9	8313	56초	8313(10)	8313(13)	8313(13)	8313(11)	8313(15)	8313(7)	8313(11)	8313(15)
CAR7	7×7	6558	44초	6558(23)	6558(19)	6558(15)	6558(16)	6558(15)	6558(21)	6558(20)	6558(19)
CAR8	8×8	8264	52초	8264(3)	8264(5)	8264(4)	8264(2)	8264(3)	8264(2)	8264(3)	8264(1)

〈표 8〉 ORB 벤치마크 문제의 결과

문 제	Bot 5	Shuffle	Tabu search	최적해	수행시간 (1회 Run)	교차연산자 1		교차연산자 2		교차연산자 3		교차연산자 4	
						토너먼트	씨종자	토너먼트	씨종자	토너먼트	씨종자	토너먼트	씨종자
ORB1	1092	1070	1073	1059	78초	1089	1089	1077	1072	1064	1060	1077	1060
ORB2	894	890	890	888	〃	892	889	892	890	890	889	889	890
ORB3	1031	1021	1024	1005	〃	1023	1025	1027	1027	1023	1022	1024	1020
ORB4	1031	1019	1013	1005	〃	1014	1011	1011	1011	1019	1011	1005	1012
ORB5	896	896	899	887	〃	894	894	889	889	890	894	890	889
ORB6	-	-	1026	1010	〃	1023	1013	1023	1023	1023	1013	1021	1013
ORB7	-	-	397	397	〃	397	397	397	397	397	397	397	397
ORB8	-	-	899	899	〃	914	912	899	899	899	899	899	899
ORB9	-	-	934	934	〃	947	943	934	934	934	939	943	934
ORB10	-	-	944	944	〃	952	952	946	946	944	944	944	944

〈표 9〉 ABZ 벤치마크 문제의 결과

문 제	크 기	SB I	SB II	최적해	수행시간 (1회 Run)	교차연산자 1		교차연산자 2		교차연산자 3		교차연산자 4	
						토너먼트	씨종자	토너먼트	씨종자	토너먼트	씨종자	토너먼트	씨종자
ABZ5	10×10	1306	1239	1234	89초	1238	1238	1236	1238	1236	1238	1236	1236
ABZ6	10×10	962	943	943	78초	943	943	943	943	943	943	943	943
ABZ7	20×15	730	710	?	368초	696	704	685	698	695	698	689	694
ABZ8	20×15	774	716	?	364초	711	714	704	707	709	714	710	704
ABZ9	20×15	751	735	?	366초	728	730	723	728	726	730	730	723

〈표 10〉 YN 벤치마크 문제의 결과

문 제	active 스케쥴	GA/GT	수행시간 (1회 Run)	교차연산자 1		교차연산자 2		교차연산자 3		교차연산자 4	
	최고해	최고해		토너먼트	씨종차	토너먼트	씨종자	토너먼트	씨종자	토너먼트	씨종자
YN1	1126	967	380초	951	951	934	936	941	936	925	933
YN2	1104	945	〃	956	944	956	966	966	962	966	957
YN3	1107	951	〃	962	962	928	940	950	936	941	931
YN4	1202	1052	〃	1069	1061	1018	1033	1039	1028	1027	1027

어진 4개의 20×20 문제이다. Yamada & Nakano는 G&T 알고리즘 기반 연산자인 GA/GT 교차연산자를 개발하였다. 이들은 4개의 YN 문제에 이 연산자를 적용하고, 각 문제에서 임의대로 산출된 400,000개의 active 스케쥴과 GA/GT의 결과들을 비교하였다. <표 10>은 20회 수행에서 얻은 결과와 GA/GT, 임의대로 산출된 active 스케쥴의 결과들을 보여준다.

4.3.6 PGA의 수행도 평가

PGA 모델을 5개의 벤치마크 문제에 적용해 보았다. <표 12>와 <표 13>의 결과에서처럼 PGA는

SGA에서 얻은 최고해의 개선은 이루지 못했다. 하지만 각 SGA와 비교해서는 대체로 좋은 해를 산출하며, <표 12>에서는 평균해를 개선시키고 있음을 확인할 수 있었다. 이는 적은 실행(run) 회수에서는 나은 해를 구해낼 수 있음을 의미한다. PGA에 사용한 파라미터 값은 <표 11>과 같다.

〈표 11〉 PGA에서의 파라미터 값

교차연산자 (선택방법)	2-PGA					4-PGA				
	엘리티즘	교차율	돌연변이율	선택확률	씨종자 크기	엘리티즘	교차율	돌연변이율	선택확률	씨종자 크기
교차연산자 1 (씨종자)	5	0.7	0.1	0.7	20	2	0.7	0.1	0.7	10
교차연산자 2 (토너먼트)	5	0.7	0.1	0.75	·	2	0.7	0.1	0.75	·
교차연산자 3 (씨종자)	10	0.7	0.1	0.75	30	3	0.7	0.1	0.75	20
교차연산자 4 (토너먼트)	5	0.8	0.1	0.75	·	2	0.8	0.1	0.75	·

〈표 12〉 MT문제에서 PGA의 결과 비교

문 제		2-PGA		4-PGA		교차연산자 1		교차연산자 2		교차연산자 3		교차연산자 4	
		M 1	M 2	M 3	M 4	토너먼트	씨종자	토너먼트	씨종자	토너먼트	씨종자	토너먼트	씨종자
MT 10	최고해	930	936	930	936	936	936	930	937	936	936	930	930
	평균해	969.69	974.74	964.23	970.12	977.80	982.96	977.08	975.96	980.91	981.91	977.48	976.92
MT 20	최고해	1173	1178	1173	1173	1178	1173	1178	1178	1173	1178	1173	1173
	평균해	1210.92	1210.62	1207.29	1204.44	1245.28	1224.22	1217.67	1217.05	1215.85	1214.09	1217.24	1211.85

〈표 13〉 ORB, ABZ 문제에서 PGA의 결과 비교

문제	2-PGA	4-PGA	교차연산자 1		교차연산자 2		교차연산자 3		교차연산자 4	
	M 1	M 3	토너먼트	씨종자	토너먼트	씨종자	토너먼트	씨종자	토너먼트	씨종자
ORB1	1061	1060	1089	1089	1077	1072	1064	1060	1077	1060
ORB2	889	889	892	889	892	890	890	889	889	890
ORB3	1025	1020	1023	1025	1027	1027	1023	1022	1024	1020
ORB4	1011	1011	1014	1011	1011	1011	1019	1011	1005	1012
ORB5	889	889	894	894	889	889	890	894	890	889
ORB6	1013	1013	1023	1013	1023	1023	1023	1013	1021	1013
ORB7	397	397	397	397	397	397	397	397	397	397
ORB8	908	899	914	912	899	899	899	899	899	899
ORB9	939	934	947	943	934	934	934	939	943	934
ORB10	944	944	952	952	946	946	944	944	944	944
ABZ5	1236	1236	1238	1238	1236	1238	1236	1238	1236	1236
ABZ6	943	943	943	943	943	943	943	943	943	943
ABZ7	694	694	696	704	685	698	695	698	689	694
ABZ8	704	704	711	714	704	707	709	714	710	704
ABZ9	726	723	728	730	723	728	726	730	730	723

5. 결 론

본 연구에서는 JSSP를 풀기 위해 유전 알고리즘을 기반으로 한 일정계획 기법을 제안하였다. 이 기법에서 표현은 공정의 순서를 job의 번호로 코드화 하여 항상 해가 실행 가능하도록 하였고, 이를 G&T 알고리즘과 연결하여 초기 모집단을 구성하였다. 그리고 새로운 선택 방법과 유전 연산자들은 집단 내 개체들의 일시적인 관계를 보다 좋게 유전시키도록 하여 초기의 좋은 스케쥴이 계속 진화될 수 있도록 하였다. 이 기법을 표준 벤치마크 JSSP에 적용하여 좋은 결과들을 산출할 수 있었는데, 이는 새로운 모집단 구성 방법과 유전 연산자의 성공적인 결합을 의미하는 것으로 제시된 일정계획 기법의 효율성을 보여주는 것이다.

5개의 벤치마크 문제의 결과를 통해서 제시한 교차연산자와 선택방법들 중 CAR 문제처럼 작은 크기의 문제에서는 교차연산자 1과 씨종자 선택을 사용한 경우가 좋은 결과를 산출하였고, ABZ7-9, YN 문제처럼 큰 크기의 문제에서는 교차 연산자 2와 토너먼트 선택을 사용한 경우가 좋은 결과를 산출함을 확인할 수 있었다. 그리고 전체적으로는 교차연산자 4와 씨종자 선택을 사용한 경우가 최고해와 최적해를 구한 횟수가 가장 많음을 확인할 수 있었다. 또한 PGA의 결과를 통해서는 대체적으로 PGA가 SGA 보다는 나은 해를 산출해 주며, 평균해를 개선시킨다는 것을 확인할 수 있었다. 충분한 시간을 가지지 못해 실행 횟수에 제한이 있는 경우는 PGA를 사용하는 것이 더욱 좋은 해를 기대할 수 있을 것이다. 본 연구에서 제시한 일정계획 기법은 기존 알고리즘과의 혼합과 창조적 진화과정으로 얻은 좋은 수행도와 알고리즘의 단순화로 보다 현실적인 적용이 용이 할 것으로 기대된다.

참 고 문 헌

[1] 박병주, 김현수, "Job Shop 일정계획을 위한 혼합 유전 알고리즘", 「한국경영과학회지」, 제26권 제2호(2001), pp.59-68.

[2] Adams, J., E. Balas, and D. Zawack, "The Shifting Bottleneck Procedure in Job Shop Scheduling," Management Science, Vol.34 (1988), pp.391-401.

[3] Applegate, D. and W. Cook, "A Computational Study of the Job Shop Scheduling Problem," ORSA Journal on Computing, Vol.3, No.2(1991), pp.149-156.

[4] Bagchi, S., S. Uckun, Y. Miyabe and K. Kawamura, "Exploring Problem-Specific Recombination Operators for Job Shop Scheduling," Proc. Fourth Int'l Conf. on Genetic Algorithms, Morgan Kaufmann, San Mateo, 1991, pp.10-17.

[5] Bierwirth, C., "A Generalized Permutation Approach to Job Shop Scheduling with Genetic Algorithms," OR-Spektrum, Special Issue : Applied Local Search, Pesch, E. and Vo, S.(eds), Vol.17, No.213(1995), pp.87-92.

[6] Bierwirth, C., D. Mattfeld and H. Kopfer, "On permutation representations for scheduling problems," In Voigt, H M., et al. editors, Proceedings of Parallel Problem Solving from Nature Ⅳ, Springer Verlag, Berlin, Germany, 1996, pp.310-318.

[7] Carlier, J. and P. Chretienne, "Problèmes d'ordonnancement," col. ERI, Masson, Paris, 1988.

[8] Chambers, J.B., Classical and flexible job shop scheduling by tabu search, Ph.D. thesis, Department of Computer Science, University of Texas, 1996.

[9] Cheng, R., A study on Genetic Algorithms-based Optimal Scheduling Techniques, Ph. D. thesis, Tokyo Institute of Technology, 1997.

[10] Croce, F.D., R. Tadei and G. Volta, "A Genetic Algorithm for the Job Shop Problem," Computer & Operations Research, Vol.22, No.1(1995), pp.15-24.

[11] Davis, L., "Job Shop Scheduling with Genetic Algorithms," Proc. Int'l Conf. on Genetic Algorithms and their Applications, Lawrence Erlbaum, Hillsdale, 1985, pp.136-149.

[12] Dorndorf, U. and E. Pesch, "Combining Genetic and Local Search for Solving the Job Shop Scheduling Problem," APMOD93 Proc. Preprints, Budapest, Hungary, 1993, pp.142-149.

[13] Dorndorf, U. and E. Pesch, "Evolution based Learning in a Job Shop Scheduling Environment," Computers & Operations Research, Vol.22(1995), pp.25-40.

[14] Fang, H., P. Ross and D. Corne, "A Promising Genetic Algorithm Approach to Job-Shop Scheduling, Rescheduling, and Open-Shop Scheduling Problems," Proc. Fifth Int'l Conf. on Genetic Algorithms, Morgan Kaufmann, San Mateo, 1993, pp.375-382.

[15] Gen, M. and R. Cheng, Genetic algorithms and engineering design, New York, John Wiley & Sons, 1997.

[16] Giffler, J. and G.L. Thompson, "Algorithms for Solving Production Scheduling Problems," Operations Research, Vol.8(1960), pp. 487-503.

[17] Goldberg, D.E. and K. Deb, "A Comparative Analysis of Selection Schemes used in Genetic Algorithms," In G. Rawlins, ed., Foundations of Genetic Algorithms, Morgan Kaufmann, 1991.

[18] Muth, J.F. and G.L. Thompson, Industrial Scheduling, Prentice-Hall, Englewood Cliffs, N.J., 1963.

[19] Nakano, R. and T. Yamada, "Conventional Genetic Algorithms for Job Shop Problems," Proc. Fourth Int'l Conf. on Genetic Algorithms, Morgan Kaufmann, San Mateo, 1991, pp.474-479.

[20] Syswerda, G., Schedule Optimization Using Genetic Algorithms, Handbook of Genetic Algorithms, Davis, L. (ed), Van Nostrand Reinhold, New York, 1991, pp.332-349.

[21] Tamaki, H. and Y. Nishikawa, "A Parallel Genetic Algorithm based on a Neighborhood Model and It's Application to the Job shop Scheduling," Parallel Problem Solving from Nature, 2, North-Holland, Amsterdam, 1992, pp.573-582.

[22] Uckun, S., S. Bagchi and K. Kawamura, "Managing Genetic Search in Job Shop Scheduling," IEEE Expert, Vol.8, No.5(1993), pp.15-24.

[23] Yamada, T. and R. Nakano, "A Genetic Algorithm Applicable to Large-Scale Job Shop Problems," Parallel Problem Solving from Nature, 2, North-Holland, Amsterdam, 1992, pp.281-290.