韓國經營科學會誌
第26卷 第2號
2001年 6月

Job Shop 일정계획을 위한 혼합 유전 알고리즘*

박병주** · 김현수***

A Hybrid Genetic Algorithm for Job Shop Scheduling*

Byung Joo Park** · Hyun Soo Kim***

Abstract

The job shop scheduling problem is not only NP-hard, but is one of the well known hardest combinatorial optimization problems. The goal of this research is to develop an efficient scheduling method based on hybrid genetic algorithm to address job shop scheduling problem. In this scheduling method, generating method of initial population, new genetic operator, selection method are developed. The scheduling method based on genetic algorithm are tested on standard benchmark job shop scheduling problem. The results were compared with another genetic algorithm-based scheduling method. Compared to traditional genetic algorithm, the proposed approach yields significant improvement at a solution.

Keyword : job shop scheduling, genetic algorithm, benchmark problem

1. 서 론

일정계획 문제는 계산적 측면에서 대부분 NP-hard 문제로 최적해에 보다 가까운 가능해를 찾기 위해 다양한 탐색 방법들이 사용되고 있다. 다윈의 진화과정 방법론을 도입한 유전 알고리즘은 일반적인 반복 탐색 전략으로 조합 최적화 문제를 해결하는데 사용되는 국소 탐색 기법이다. 유전 알고리즘은 시뮬레이티드 어닐링(simulated annealing)과 타부서치(tabu search) 등의 국소 탐색기법들이 하

논문접수일 : 2001년 1월 18일 논문게재확정일 : 2001년 5월 1일
* 이 논문은 2000년도 두뇌한국21사업에 의하여 지원되었음.
** 동아대학교 에이전트 기반 전자상거래팀(BK21) Post-Doc.
*** 동아대학교 경영정보과학부 교수

나의 실행가능한 해를 다루는 것과는 달리 탐색을 위해 해의 집단을 사용함으로써 내재된 병렬성을 지니며, 마지막 세대에서 여러 대안 해를 얻을 수 있다는 특성을 가진다. Davis[8]가 1985년 일정계획 문제를 해결하기 위해 유전 알고리즘을 기반으로 한 기법을 제안한 이후 유전 알고리즘이 일정계획 문제에 사용되어지는 횟수는 증가되고 있다.

전통적인 유전 알고리즘은 문제에서 실행 가능해를 나타내기 위해 이진 스트링을 사용하였다. 그러한 표현은 TSP(Traveling Salesman Problem), JSSP(Job Shop Scheduling Problem)와 같은 문제에 있어서는 적합하지 않다. 왜냐하면 이진 스트링으로 실행 가능해를 나타내는 직접적이고 효과적인 방법을 찾지 못했기 때문이다. 그리고 전통적인 유전 알고리즘의 표현 방법을 사용한 문제에서 스트링에 적용한 단순한 교차연산이나 돌연변이 연산은 항상 실행 불가능한 해를 산출한다. 그래서 이전 연구자들은 이 문제를 표현하기 위해 표준 유전 연산자들을 변형하여 사용했었다.

Davis[8]는 문제 중심의 표현으로 직접적으로 스케쥴을 표현하지는 않지만 대신 디코더나 스케쥴 빌더(builder)로 염색체의 표현을 실행 가능한 스케쥴로 전환시킨다. Syswerda[17]는 TSP 문제처럼 표현된 염색체로 더욱 현실적인 JSSP에 새로운 방법을 실행하였다. JSSP 연구에서 일부 연구자들은 TSP와 유사한 접근법을 포기하고, 더욱 현실적인 표현과 JSSP를 위한 더욱 복잡한 연산자를 개발하였다. Nakano와 Yamada[16]는 미연결 호들을 선택하기 위해 이진 표현을 사용했다. 그리고 새로운 스트링을 산출하기 위해 전통적인 연산자를 사용했다. 여기서 스케쥴 빌더는 실행 가능한 스트링에서 스케쥴을 산출할 뿐만 아니라 실행 불가능한 스트링으로부터 실행 가능한 스케쥴을 찾기 위해 복잡한 보정(repair) 절차를 적용한 뒤 포싱(forcing)한다. 포싱은 보정된 실행 가능한 스트링을 가지고 원래의 실행 불가능한 스트링을 교체하는 과정을 말한다. 그는 이 접근법을 가장 잘 알려진 세 개의 MT 벤치마크 문제에 적용하여 MT6 문제에서 최적해를 발견했다. 하지만 다른 두 문제에서는 그렇지 못했다. 그는 만약 스케쥴 빌더가 잘 설계되어진 전통적인 유전 알고리즘은 JSSP를 효과적으로 해결할 수 있다고 결론지었다.

비록 많은 접근법들이 논의되었고, 지금까지 많은 지식기반(knowledge-augmented) 연산자가 개발되었다 하더라도 그들 연산자 중 어떤 것도 이론적인 근거를 가지고 있지는 않다. 그리고 접근법들 중에는 유전 알고리즘과 현존하는 알고리즘들의 혼합사용(hybridization)에 기초하고 있는 것들이 있다. Yamada와 Nakano[18]는 표현을 위해 공정완성시간을 사용했고, 새로운 교차 연산자를 제안했다. 그들은 염색체의 재조합 과정에서 실행가능성과 active 스케쥴을 유지하기 위해 Giffler & Thompson(G&T) 알고리즘[13]을 기반으로 한 GA/GT 연산자를 사용하였다. Nakano와 Yamada와 같이 세 개의 MT 벤치마크 문제에 이 방법을 적용하여 좋은 결과를 얻어내고 있다. Dorndorf와 Pesch[9]는 표현방법으로 공정 시작 시간을 코드화하고 표준적인 교차 연산으로부터 생기는 실행 불가능한 자손을 디코더 하기 위해 G&T 알고리즘을 적용하였다. 그리고 유전 알고리즘 과정을 국소 탐색 절차와 통합하였다. G&T 알고리즘 기반 접근법들은 위의 과정과 유사하며, G&T 알고리즘은 active 스케쥴로 어떤 자손을 디코더 하기 위한 해석자로 사용되어졌다.

이들 연구에서는 염색체의 실행 가능성을 유지하기 위해 보정절차, 포싱, 알고리즘과 결합된 디코더 방법 등이 필요했다. 그러나 만약 염색체의 표현에서 유전 연산 후에도 항상 실행 가능성을 유지 할 수 있다면, 보정절차나 포싱, 그리고 G&T와 같은 알고리즘과 결합된 디코더 과정이 필요 없게 되어 알고리즘 적용과정을 단순화 할 수 있다. 또한 대부분의 연구들에서 초기 모집단을 개체의 다양성을 위해 임의대로 생성시켰으나, 국소 탐색에서 최적해에 근접은 나은 초기해에서 탐색을 시작하는 것이 가능성이 더 높다. 그래서 다양성을 유지하면서 좋은 개체로 초기 모집단을 구성할 수 있

는 방법과 이들을 잘 유전시킬 유전 연산자, 선택 방법이 있다면 더 나은 해를 구할 수 있을 것이다. 본 연구는 이들을 바탕으로 일정계획 문제를 보다 효과적으로 해결하기 위한 혼합 유전 알고리즘을 기반으로 한 일정계획 기법을 개발하고자 한다.

2. 유전알고리즘

유전 알고리즘은 자연진화의 법칙인 적자생존과 자연도태의 원리를 바탕으로, 집단을 구성하는 각 개체를 목적함수 값과 제약조건의 위반정도에 따라 적합도를 구하고, 적합성이 큰 개체를 다음 세대로 진화시키기 위한 교차와 돌연변이 과정에 참여할 기회를 높여, 다음 세대에 우수한 형질의 개체를 많이 형성해 좋은 방향으로 탐색을 진행시키는 최적화 기법이다.

유전 알고리즘을 사용하기 위해서는 문제에 대한 특성을 먼저 분석한 다음 그 문제에 적합한 표현방법, 평가함수, 초기 모집단 구성방법, 유전연산자, 유전 파라미터 등이 결정되어야 한다.

2.1 염색체 표현

job shop 일정계획 문제를 해결하기 위해서는 먼저 문제의 해를 염색체로 표현해야 한다. 염색체 표현은 job 번호를 공정의 수만큼 반복시키는 순열 형태로 표현한다[4]. 하나의 유전인자는 하나의 공정을 의미하고 표현형태는 처리되는 공정의 job 번호로 표현한다. 이처럼 job 번호의 반복을 통해 표현한 염색체는 job의 공정들이 스케쥴 되어지는 순서를 나타낸다. 예를 들어 <표 1>의 3×3 문제가 순열 형태의 염색체 [3 2 2 1 1 2 3 1 3]으로 표현되었다면, 세 번씩 반복된 숫자는 job의 번호를 나타낸다. job 번호가 세 번씩 반복되는 것은 각 job들이 3개의 공정을 가지고 있기 때문인데, job 번호의 첫 번째 반복은 그 job의 첫 공정을, 두 번째 반복은 두 번째 공정을 의미한다. 염색체의 3번째 유전인자가 2인데 이 유전인자는 job 2의 두 번째 공정에 대응된다. 왜냐하면 2의 표현이 염색체에서 두 번 반복되었기에 이는 job 2의 두 번째 공정을 의미한다. 이 공정은 기계 3에서 5의 가공시간이 필요하다. 이 염색체는 job 번호가 공정의 수만큼만 표시된다면 항상 실행가능성을 유지한다.

<표 1> 3개 job, 3대 기계(3×3)를 가진 JSSP

Job	기계번호 (가공시간)
1	3(1) → 1(3) → 2(6)
2	2(8) → 3(5) → 1(4)
3	3(5) → 2(4) → 1(8)

염색체의 생성은 G&T 알고리즘을 이용한다. G&T 알고리즘은 단계 4의 G 집합 내에 속해 있는 공정들을 하나씩 선택하는 과정을 전체 공정의 수만큼 수행하여 스케쥴하게 되는데, 이때 하나씩 선택되어지는 공정을 포함하는 job의 번호를 선택 순서대로 전체 공정 수만큼 연결하여 기입하면 염색체가 생성된다.

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 가능성이 줄어든다. 본 연구에서는 평가함수의 값이 좋은 개체로 구성된 초기모집단을 생성시키기 위해 G&T 알고리즘을 혼합 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' 스케쥴을 산출하고, 이 스케쥴들로 초기 모집단을 구성한다. active' 스케쥴은 단계 3에서 G집합을 구성하는 방법을 변형한 G&T 알고리즘을 사용한다. 그 알고리즘은 다음

과 같다.

2.2.1 기 호

n : job의 수

m : 기계의 수

$P_{j(k)}$: job j의 기계 k에서 가공시간 (j = 1, 2, … n, k = 1, 2,…, m)

$r_{j(k)}$: job j의 기계 k에서 시작시간

2.2.2 수정된 G&T 알고리즘(active′ 스케쥴)

G&T 알고리즘의 단계 3을 아래와 같이 수정하고, 여기서 얻어지는 스케쥴을 active′ 스케쥴이라 한다.

단계1) 각 job들 중에서 가장 먼저 스케쥴 해야 할 공정들을 집합 C로 둔다. 집합 C 내의 모든 공정 j(k)에 대한 시작시간 $r_{j(k)} = 0$ 이라 둔다.

단계2) $t(C) = \min_{(j,k)\in C} \{r_{j(k)} + p_{j(k)}\}$를 계산한다. 그리고 t(C)가 최소가 되는 기계를 m^*로 둔다.

단계3) 기계 m^* 상에서 $r_{j(m^*)} < t(C)$인 모든 공정 $j(m^*)$들 중에서 $r_{j(m^*)}$이 가장 작은 공정들을 집합 G로 둔다.

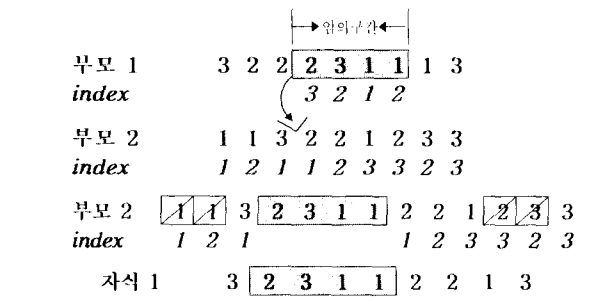
단계4) 집합 G에서 하나의 공정을 임의대로 선택하고 그것을 스케쥴 한다.

단계5) C에서 그 공정을 삭제한다. 그리고 job에서 그 공정의 후행 공정을 집합 C에 포함시킨다. C에서 $r_{j(k)}$를 수정하고 모든 공정이 스케쥴될 때까지 단계 2)로 돌아간다.

2.3 교차연산자(Crossover)

교차연산자는 G&T 알고리즘으로 얻은 염색체들이 좋은 스케쥴을 가지고 있기에 가능하면 그들 순서관계를 유지하면서 조금씩 개선시켜 나갈 수 있는 연산자가 필요하다. 그래서 좋은 순서관계를 유지하면서 진화시킬 수 있는 교차 연산자를 사용한다.

본 연구에서 사용된 교차연산자 1은 먼저 임의 구간을 산출한 뒤 그 구간 내 유전인자들을 부모 2에 삽입한다. 삽입 위치는 임의 구간이 시작된 유전인자 바로 앞이다. 만약 첫 번째 부모에서 임의 구간의 시작 위치가 4번째라면 삽입 위치는 부모 2의 4번째 유전인자 앞이 된다. 그리고 나서 임의 구간내의 유전인자와 같은 인덱스(index)를 가진 유전인자들을 부모 2에서 삭제한다. 이들 과정을 부모 1과 2를 바꾸어서 수행하여 두 개의 자식 개체를 생성한다. 그리고 두 자식 개체 중 평가기준에 적합한 한 개체만을 다음 세대로 보낸다. 그 과정은 [그림 1]과 같다.

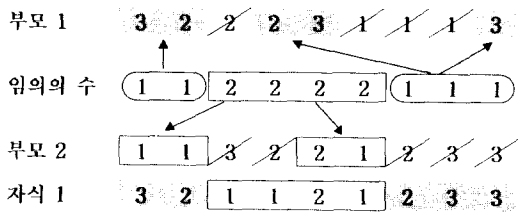


부모 1과 2를 바꾸어 같은 구간으로 두 번째 자식을 생성한다.

자식 2 3 2 2 1 2 3 1 1 3

[그림 1] 교차연산자 1 과정

교차 연산자 2는 Bierwith[3]의 PPX 연산자를 변형한 것으로, 먼저 두 부모 1과 2를 선택 한 후 두 부모 중 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 유전인자를 삭제하는 과정을 난수의 수만큼 되풀이한다. 이 과정을 부모 1과 2를 바꾸어서 다시 수행한다. 이렇게 산출된 두 자식 중에서 적합도가 높은 한 개체를 다음 세대로 보낸다. 그 과정은 [그림 2]와 같다.



부모 1과 2의 순서를 바꾸어 같은 난수로 두 번째 자식을 생성한다.

자식 2 1 1 3 2 2 2 1 3 3

[그림 2] 교차연산자 2 과정

2.4 돌연변이(Mutation)

돌연변이 연산자는 염색체에 변화를 주어 집단 내의 다양성을 유지하기 위해 사용한다. 본 연구에서는 이웃 탐색 기법에 근거한 돌연변이 연산자를 사용한다. [그림 3]은 돌연변이 연산의 예를 보여준다.

이 돌연변이 연산자는 case 2에서 case 6까지의 부모와 다른 5가지만 이웃으로 비교하여 가장 좋은 것을 유전시키는 방법과 현 부모와 5개의 이웃 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 사용하는 돌연변이 연산자로 사용한다.

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

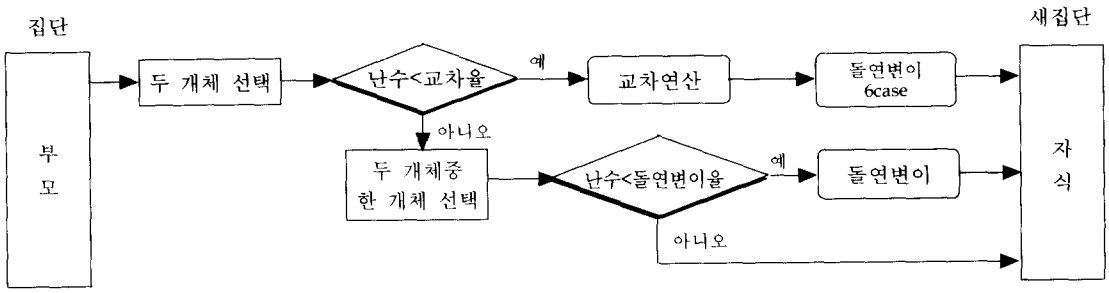
[그림 3] 돌연변이 연산의 예

2.5 선택방법

본 연구에서 새로운 선택방법으로 씨종자 선택을 제시한다. 씨종자 선택은 일상에서 사용하는 개체 선택과 좋은 개체 보존 방법을 유전 알고리즘 진화과정에 도입한 것이다. 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 정해진 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 토너먼트 선택에서처럼 일정한 확률 값에 따라 적합도가 좋은 개체 하나만을 선택한다. 이들을 부모로 사용하고 개체 집단에 되돌려 다시 선택할 수 있게 한다. 본 연구에서는 씨종자 선택을 각 연산자와 결합시켜 적용하고, 수행도는 토너먼트 선택을 사용한 결과와 비교한다. 토너먼트 선택은 각 개체마다 선택확률을 부여하지 않는다는 점에서 여러 선택방법들 중 씨종자 선택과 가장 유사한 방법이다. Goldberg와 Deb[14]가 제시한 토너먼트 선택은 집단에서 두 개의 개체를 임의대로 선택한 다음, 난수 r을 0과 1사이에서 발생시켜 만약 $r < k$이면 두 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 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 개



[그림 4] 세대의 생성 과정

체들은 유전 연산자를 거치지 않고 그대로 다음 세대로 이동하도록 한다. 한 세대의 생성과정은 [그림 4]와 같다.

2.7 평가함수

JSSP에서 최소의 makespan을 가진 스케줄은 종종 기계의 높은 효율을 의미한다. 대부분의 정적 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. 그 과정은 [그림 5]와 같다.

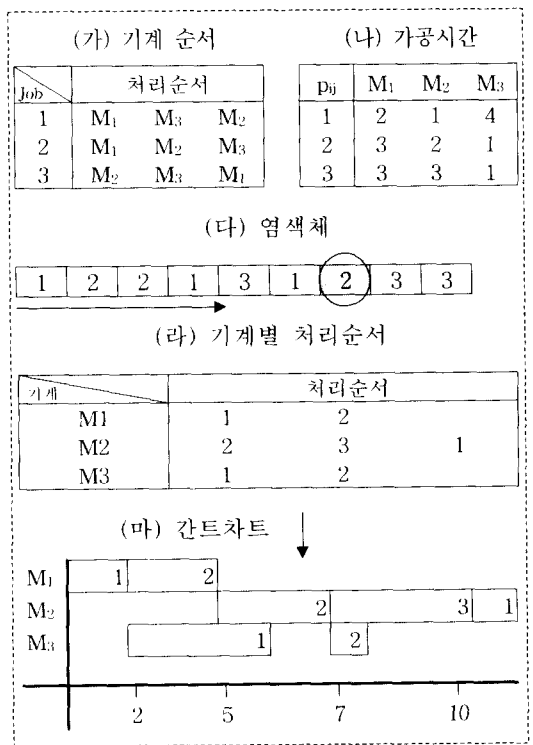
3. 실험 및 분석

제안알고리즘의 수행도를 평가하기 위해 세 개의 벤치마크 문제(MT문제, CAR문제, ORB문제)에 적용하여 다른 알고리즘의 결과와 비교한다. 알고리즘을 적용하기 위해서는 유전 파라미터들이 결정되어야 하는데, 먼저 모집단의 크기는 200으로 세대수는 1000으로 하고, 교차율, 토너먼트 선택에서의 선택확률, 엘리티즘의 크기, 씨종자 선택범위는 실험을 통하여 결정한다.

3.1 유전 파라미터의 설정

제시한 알고리즘에 적용할 교차율, 선택확률, 엘리티즘 크기는 MT10 문제에 하나의 초기 모집단을 구성하고, 파라미터에 따라 100번 실행하여 가장 좋은 해와 평균해를 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>에서 가장 좋은 최고해와 평균해는 교차연산자 1에서 교차율 0.8, 선택확률 0.75, 엘리티즘



[그림 5] 염색체에서 makespan을 구하는 과정

〈표 2〉 파라미터 실험 결과

파라미터		교차연산자 1			교차연산자 2		
교차율	선택확률	엘리티즘	평균해	최고해	엘리티즘	평균해	최고해
0.6	0.75	10	960.27	951	5	966.07	937
0.7	0.75		958.76	945		965.17	937
	0.7		959.69	945		966.18	939
	0.8		959.26	951		965.92	937
0.8	0.75		957.49	937		967.61	939
0.6	0.75	20	961.27	935	10	972.16	937
0.7	0.75		958.83	937		966.85	937
	0.7		960.42	951		964.52	937
	0.8		960.54	951		964.67	937
0.8	0.75		960.67	951		964.59	937
0.6	0.75	30	962.57	940	15	971.5	937
0.7	0.75		962.53	951		969.27	937
	0.7		963.62	937		969.67	937
	0.8		959.52	951		972.26	942
0.8	0.75		960.05	951		969.71	936

크기를 10으로 했을 경우가, 교차연산자 2는 교차율 0.7, 선택확률 0.7, 엘리티즘 크기를 10으로 했을 경우가 가장 좋다. 그래서 이들을 파라미터 값으로 결정한다.

〈표 3〉 씨종자 선택 범위에 대한 실험 결과

교차연산자	파라미터				평균해	최고해
	엘리티즘	교차율	선택확률	씨종자 선택범위		
교차연산자 1	10	0.8	0.75	20	960.25	951
				30	961.33	951
				40	959.28	951
				50	959.25	951
				60	959.49	945
교차연산자 2	10	0.7	0.7	20	969.99	937
				30	968.54	937
				40	967.15	937
				50	964.35	937
				60	962.51	936

씨종자 선택에서 씨 종자(父)의 선택범위를 결정하기 위해 앞에서 정해진 파라미터에 선택범위를 상위 20위 내에서 60위까지로 하여 가장 좋은 범위를 찾기 위한 실험을 하였다. <표 3>의 결과를 통해 범위를 교차연산자 1, 2에서 50, 60으로 결정한다.

3.2 유전알고리즘의 수행도 평가

각 벤치마크 문제마다 변형된 G&T 알고리즘으로 새로운 초기 모집단을 구성하고, 유전연산자를 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. 상대오차는 매회 실험에서 구해진 최선해와 최적해 사이의 편차이며, (100×(최선해 - 최적해) /최적해)로 구한다.

3.2.1 MT 문제

MT 문제는 벤치마크 문제 중 가장 많이 사용되는 문제로 Muth와 Thomson[15]에 의해 제시되었다. 이 문제는 3가지 크기의 문제(MT6, MT10, MT20)를 가지고 있는데 이 중 MT10, MT20 문제는 제안되는 거의 모든 job shop 일정계획 알고리즘들이 벤치마크 문제로 사용하고 있다.

본 연구에서는 MT6, MT10 문제에서 55, 930으로 최적해를 얻을 수 있었으며, MT20 문제에서는 1173으로 근접해를 구해낼 수 있었다. <표 4>는 세 개의 MT 문제에서 이전의 연구에 의해 얻어진

최고의 결과들이다. 이 표에서 이전 연구의 결과들에 비해 제시한 기법이 현저한 해의 개선을 이루어 냄을 볼 수 있다.

〈표 4〉 MT 벤치마크 문제의 결과 비교

비교논문	FT6 (6×6)	FT10 (10×10)	FT20 (20×5)
최적해	55	930	1165
Nakano & Yamada[16]	55	965	1215
Yamada & Nakano[18]	55	930	1184
Gen[12]	55	962	1175
Fang[11]	-	949	1189
Dorndorf1 & Pesch[10]	55	960	1249
Dorndorf2 & Pesch[10]	55	938	1178
Croce[7]	55	946	1178
Cheng[12]	55	948	1196
Bierwirth[3]	55	936	1181
제안 GA	55	930	1173

3.2.2 CAR 문제

CAR 문제는 Carlier[5]에 의해 제시된 다양한 크기의 벤치마크 문제이다. 이 문제는 제시한 기법의 작고 다양한 크기의 문제에서 수행도를 평가하기 위해 사용되었다. 결과는 <표 5>에서처럼 8개 문제 모두에서 최적해를 찾을 수 있었다.

3.2.3 ORB 문제

ORB 문제는 Applegate와 Cook[2]에 의해 특별히 어렵게 만들어진 10개의 10×10 문제이다. 이 문제에서 얻은 결과들은 <표 6>과 같다. 이 표에서 3열은 Adams 등[1]의 Bottle-5 version으로 얻은 결과이고, 4열은 Applegate와 Cook의 shuffle 알고리즘으로 구한 결과이다. 5열은 Chamber[6]의 tabu search에 의한 결과이다.

〈표 5〉 CAR 벤치마크 문제의 결과

문제	크기	제안 GA	평균해	평균 상대오차	최적해
CAR1	11×5	7038	7038	0	7038
CAR2	13×4	7166	7233.8	0.94	7166
CAR3	12×5	7312	7521.58	2.86	7312
CAR4	14×4	8003	8059.34	0.7	8003
CAR5	10×6	7702	7818.38	1.51	7702
CAR6	8×9	8313	8476.84	1.97	8313
CAR7	7×7	6558	6609.04	0.77	6558
CAR8	8×8	8264	8412.42	1.79	8264

3.3 씨종자 선택의 수행도 평가

씨종자 선택의 수행도를 확인하기 위해 각 벤치마크 문제에서 선택방법만을 달리 한 경우의 결과들을 비교하였다. MT10, MT20 문제에서의 결과들은 <표 7>과 같다. 여기서 교차연산자 2를 사용한 MT10 문제를 제외하고는 씨종자 선택을 사용한 경우가 최고해와 평균해가 더 우수함을 볼 수 있다. 특히 MT20 문제에서는 평균해의 많은 개선이 이루어졌는데, 이는 적은 수행에서는 더 나은 해를 구해낼 수 있음을 보여준다. 그리고 <표 8>에서도 씨종자 선택으로 더 나은 해(음영으로 표시)를 얻어낸 횟수가 많음을 확인 할 수 있다. 이

〈표 6〉 ORB 벤치마크 문제의 결과

문제	크기	Bot 5	Shuffle	Tabu search	제안 GA	최적해
ORB 1	10×10	1092	1070	1073	1060	1059
ORB 2	10×10	894	890	890	889	888
ORB 3	10×10	1031	1021	1024	1020	1005
ORB 4	10×10	1031	1019	1013	1005	1005
ORB 5	10×10	896	896	899	889	887
ORB 6	10×10	-	-	1026	1013	1010
ORB 7	10×10	-	-	397	397	397
ORB 8	10×10	-	-	899	899	899
ORB 9	10×10	-	-	934	934	934
ORB 10	10×10	-	-	944	944	944

〈표 7〉 MT 문제에서 선택방법에 따른 결과 비교

일정계획기법 \ 문제		MT10			MT20		
		최고해	평균해	평균 상대오차	최고해	평균해	평균 상대오차
교차연산자 1	씨종자 선택	930	976.92	5.04	1173	1211.85	4.02
	토너먼트 선택	930	977.48	5.1	1173	1217.24	4.48
교차연산자 2	씨종자 선택	936	982.96	5.6	1173	1198.98	2.91
	토너먼트 선택	936	977.80	5.14	1180	1245.28	6.89

〈표 8〉 CAR, ORB 문제에서 선택방법에 따른 결과 비교

벤치마크 문제	교차연산자 1		교차연산자 2		최적해
	씨종자 선택	토너먼트 선택	씨종자 선택	토너먼트 선택	
CAR 1	7038	7038	7038	7038	7038
CAR 2	7166	7166	7166	7166	7166
CAR 3	7312	7312	7312	7400	7312
CAR 4	8003	8003	8003	8003	8003
CAR 5	7702	7720	7702	7732	7702
CAR 6	8313	8313	8313	8313	8313
CAR 7	6558	6558	6558	6558	6558
CAR 8	8264	8264	8264	8264	8264
ORB 1	1060	1077	1089	1089	1059
ORB 2	890	889	889	896	888
ORB 3	1020	1024	1025	1023	1005
ORB 4	1012	1005	1011	1014	1005
ORB 5	889	890	894	894	887
ORB 6	1013	1021	1013	1023	1010
ORB 7	397	397	397	397	397
ORB 8	889	899	915	916	899
ORB 9	934	943	943	947	934
ORB 10	944	944	944	944	944

결과들을 통해 교차연산자 1과 씨종자 선택이 더 나은 수행도를 보임을 확인 할 수 있다.

4. 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 선택방법과 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 구성방법과 유전 연산자의 성공적인 결합을 의미하는 것으로 제시된 일정계획 기법의 효율성을 보여주는 것이다.

본 연구에서 제시한 일정계획 기법은 기존 알고리즘의 혼합과 창조적인 진화과정으로 얻은 좋은 수행도와 알고리즘의 단순화로 보다 현실적인 적용이 용이 할 것으로 기대된다.

참 고 문 헌

[1] Adams, J., Balas, E., and D. Zawack, "The Shifting Bottleneck Procedure in Job Shop Scheduling," Management Science, Vol.

34(1988), pp.391-401.

[2] 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.

[3] 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.

[4] Bierwirth, C., Mattfeld, D., 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.

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

[6] Chambers, J.B., "Classical and flexible Job Shop Scheduling by Tabu Search," Ph.D. thesis, Department of Computer Science, University of Texas, 1996.

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

[8] 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.

[9] 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.

[10] Dorndorf, U. and E. Pesch, "Evolution based Learning in a Job Shop Scheduling Environment," Computers & Operations Research, Vol.22, No.1(1995), pp.25-40.

[11] Fang, H., Ross, P. 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.

[12] Gen, M. and R. Cheng, Genetic Algorithms and Engineering Design, New York, John Wiley & Sons, 1997.

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

[14] 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.

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

[16] 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.

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

[18] 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.