Journal of the Korean Institute of Industrial Engineers
Vol. 30, No. 2, pp. 107-119, June 2004.

반복 공정을 가지는 제약적 병렬기계에서의 일정 계획 수립

고효헌[1] · 백종관[2†] · 강용하[1] · 김성식[1]

[1]고려대학교 산업시스템정보공학과/[2]서일대학 산업시스템경영과

A Scheduling Scheme for Restricted Parallel Machines with Cycling Process

Hyo-Heon Ko[1] · Jong-Kwan Baek[2] · Yong-Ha Kang[1] · Sung-Shick Kim[1]

[1]Department of Industrial Systems and Information Engineering, Korea University, Seoul, 136-701
[2]Department of Industrial System Management, Seoil College, Seoul, 131-702

A study on the following parallel machine problem is addressed in this research. An order is completed only when a given number of processes (cycle) are repeated. A new cycle is possible only upon the completion of the previous cycle. Orders are classified into job group according to product feature. For a machine to switch to a different job group from the currently processing one a major setup is required while a minor setup time is inserted in between two jobs of the same job group. The objective of the study is to find a schedule that minimizes total weighted tardiness. An initial solution is obtained by the RATCS(Restricted Apparent Tardiness Cost with Setup) rule, and a Tabu search is applied to improve the solution. Numerical examples are also presented.

Keywords **:** parallel machines, cycling process, restricted tools, sequence dependent major and minor setup

1. 서 론

반도체 제조 공정의 팹(Fab; Fabrication) 공정은 복잡한 생산 과정을 거치기 때문에 효과적인 일정 계획을 수립하기가 힘들다. 특히, 고객의 주문에 따라 반도체를 공급하는 파운더리(Foundry)는, 같은 제품을 대량으로 생산하는 메모리 반도체 팹보다 다양한 종류의 사양과 다양한 납기를 가지기 때문에 효과적인 일정 계획을 수립하기가 더욱 어렵다. 주문형 반도체는 비교적 소량을 고객에게 적시에 공급해야 하기 때문에 납기준수가 가장 중요한 지표가 된다. 특히 기업의 큰 이익을 주는 고객일수록 납기 준수는 더욱 중요하다.

반도체가 완성되기 위해서는 요구되는 수만큼의 층(Layer)을 형성해야 하며 각 층은 유사한 공정을 거치며 이루어진다. 각 층의 가공은 순서를 이루고 있으며 그 전 층의 가공이 완료되어야만 다음 층의 가공을 시작할 수 있다. 또한 팹 제조 공정은 공정의 특성상 포토(Photo; Photolithography) 공정에 부하가 집중되는 경우가 많아 이 공정을 병목 공정으로 가정하여 팹의 능력이 결정되며 일정 관리도 이 공정을 중심으로 이루어진다. 포토 공정은 다수의 동일한 종류의 기계들로 형성되어 있다. 이러한 환경하에서 가동되는 파운더리의 일정 계획은, 팹을 병목 공정인 포토 공정과 포토 외 공정으로 보고, 포토 외 공정에 머무는 시간을 층별 반복 공정 간 지연시간으로 가정하여 수립될 수 있다.

제품은 비슷한 특성을 가지는 제품군으로 나누어지며 기계에서 생산되는 제품군이 바뀌면 메이저 준비시간(Major Setup)이 필요하고 제품군 내에서의 변경은 마이너 준비시간(Minor Setup)이 필요하다. 또한 특정 제품의 특정 층을 가공하기 위해서는 마스크(Mask)라고 불리는 특별한 작업 공구가 필요하다. 같은 제품의 동일한 층을 가공할 수 있는 마스크의 수는 제한되어 있어 특정 제품의 특정 층을 동시에 다른 기계에서 가공할 수 없다. 이러한 제약 때문에 <Figure 1>에서 보듯이 작업이 가지는 실제 투입 가능 시간에 투입하지 못하고 작업이 지연

† 연락저자 : 백종관 교수, 131-702 서울시 중랑구 면목8동 서일대학 산업시스템경영과, Fax : 02-490-7447, E-mail : jkbaek@seoil.ac.kr
2003년 10월 접수; 2004년 3월 수정본 접수; 2004년 3월 게재 확정.

되고 있다. <Figure 1>에서 보면 실제 작업 투입 가능 시간은 Machine 1의 종료시간이 된다. 하지만 Machine 2에서 필요한 마스크를 사용하여 가공하고 있어, 공구의 제약으로 인해 Machine 1에 투입할 수 없다. 따라서 실제 작업 투입 시간에 Machine 1에 투입되지 못하고 있다. <Figure 1>에서와 같이 같은 마스크가 사용중일 때는, 사용중인 기계의 종료시간 이전에 다른 병렬기계가 유휴하더라도 사용중인 기계의 종료시간까지 대기하여야 한다. <Figure 1>에서 Machine 3은 Machine 2의 작업 종료 시간이 같고 투입하려는 작업의 공구가 사용 가능하므로 대기하던 작업이 투입될 수 있다.

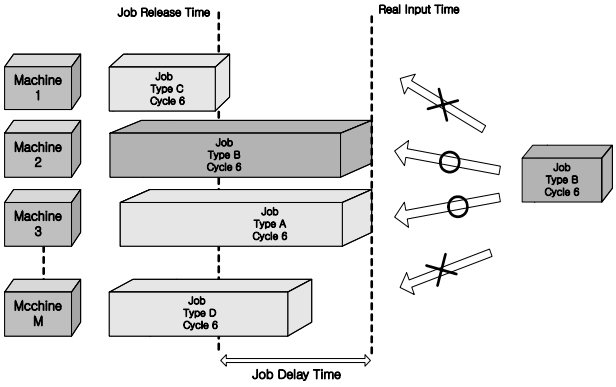

Figure 1. Restricted tools in the parallel machines.

반복 공정을 가지는 문제에서는 앞선 공정이 지연시간을 가지면 앞선 공정에 의해 후속 공정들에서도 계속적인 지연 시간이 발생한다. 각 공정의 지연시간들은 전체 일정 계획에 큰 영향을 준다. 따라서 작업을 진행할 때 병렬기계에서 공구의 제약은 중요한 고려 사항이 된다.본 연구는 파운더리의 팹에서 발생하는 복잡하고 다양한 제약 환경을 해결하기 위해 문제의 범위를 병목 공정인 포토 공정의 기계들로 한정하였다. 따라서 제안하고자 하는 일정 계획 수립 문제는 <Figure 2>와 같이 표현할 수 있다.

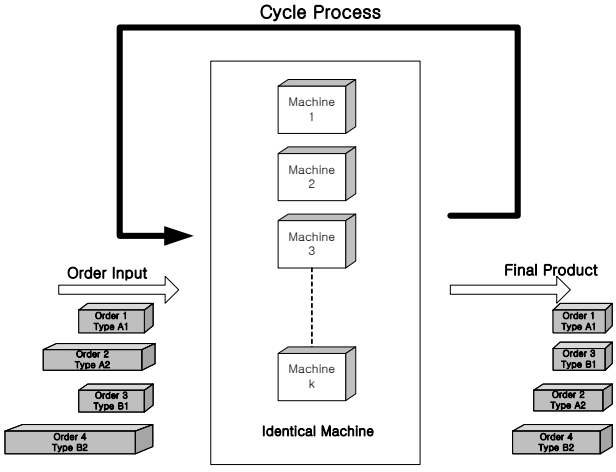

Figure 2. Parallel machines with cycling processes.

<Figure 2>에서 제품은 반복 공정을 수행하고, 각 반복 공정에서의 투입 가능 시간들이 선공정의 완료시간에 따라 변한다. 각 주문은 각기 다른 반복 공정 횟수를 가지고 있고 한 번의 공정을 마치면 다시 돌아와 다음 기계 앞에 대기하게 된다. 필요한 반복 횟수만큼 공정이 완료되면 최종 제품으로 빠져나간다. 주문을 한번 반복하는 것을 사이클(Cycle)이라 하며 모든 주문은 주문된 제품의 종류(Product Type)에 따라서 정해진 사이클 개수를 가진다. 각 사이클의 투입 가능 시간은 이전 사이클의 완료시간에 따라 변한다. 또한 사이클을 가공하는 데 사용되는 공구의 수는 제한적이고 기계에서 동일 제품의 동일 사이클을 가공중일 때 투입하려는 작업은 기계에서 작업이 완료될 때까지 대기한다.

순서 의존적인 작업 준비시간과, 특히 병렬기계에서 작업 공구의 제약을 가지며 반복 공정을 수행하는 복잡한 작업 환경에서 일정 계획을 수립할 때, 단순하고 직관적인 수작업 방식들을 사용한다면 긴 계산시간을 요구할 뿐만 아니라 계산 과정이 복잡하여 생산성 및 비용 측면에서 효과적인 결과를 기대하기 힘들다. 따라서 주어진 제약 조건을 해결할 수 있는 효율적인 알고리듬이 필요하다.

따라서 본 연구에서는 병렬기계에서의 작업 공구 제한 등의 제약 사항이 존재하며, 반복 공정을 고려한 병렬기계에서 납기 만족 위주의 효율적인 일정 계획 수립 알고리듬을 제안한다. 또한 순서 의존적인 메이저와 마이너 작업 준비시간과 반복 공정에 따른 작업 선·후행 관계에서 투입 가능 시간이 변하는 것도 고려하였다. 연구의 목적은 중요도가 높은 고객의 납기를 우선적으로 만족시키기 위한 납기 지연 가중치 합(Total Weighted Tardiness 이하 TWT)의 최소화이다.

순서 의존적인 작업 준비시간과 작업 투입시간을 고려하는 일정 계획 수립에 대한 연구로는 Kim and Bobrowski (1994)의 연구가 있다. Kim and Bobrowski(1994)은 작업 준비시간이 주문 생산과 같은 생산 시스템의 성능에 미치는 영향을 연구하기 위하여 시뮬레이션을 수행하였다. 특히 작업 준비시간이 순서 의존적일 경우에는 일정 계획 문제를 풀 때 반드시 고려되어야 하는 요소임을 보였다.

작업 준비시간이 존재하지 않는 단일 기계 일정 계획 문제에서 L_{Max}를 최소화하는 방법들은 지금까지 많이 연구되어 왔다. 단일기계 일정 계획 문제는 작업 투입시간의 제약이 없을 경우 EDD(Earliest Due Date) 규칙을 이용하면 쉽게 최적 해를 구할 수 있다(Baker and Su, 1974). 그러나 작업 투입시간의 제약이 존재하는 문제($1|r_j|L_{Max}$)는 Np-Hard가 된다(Garey and Johnson, 1979).

Monma and Potts(1994), Tang(1990) 그리고 Bitran and Gilbert(1990) 등은 메이저 준비시간이 존재하는 동일 병렬기계 문제를 다루었다. Monma and Potts(1994)는 Preemption이 허용되는 상황에서 작업 완료시간의 최대값을 최소화하는 휴리스틱을 제안하였다. Tang(1990)은 동일한 메이

저 사이에서의 마이너 준비시간까지 고려한 경우의 작업 완료 시간의 최대값을 최소화하여 근사 최적해를 구하는 휴리스틱과 최적해에 대한 두 개의 하한(Lower Bound)을 제안하였다. Bitran and Gilbert(1990)는 분지한계(Branch & Bound)법을 이용하여 메이저 준비시간의 횟수를 최소화하는 방법을 제안하였다. 특히 So(1990), Tang and Wittrock (1985)은 동종 병렬기계에 대한 작업의 메이저/마이너 준비시간과 제품군(Family)에 대한 문제를 연구하였다. So(1990)는 기계의 처리 능력을 고정한 상태에서 Total Reward 함수를 최대화하는 문제를 해결하는 세 가지의 휴리스틱을 개발하였다.

TWT(Total Weighted Tardiness)를 목적으로 하는 많은 연구가 있었으며 Vepsalainen and Morton(1987)은 동종 기계에서 납기 지연 가중치의 합을 최소화하는 문제를 해결하고자 Apparent Tardiness Cost(ATC) 알고리듬을 개발하였다. 기본적으로 ATC 규칙은 Weighted Shortest Processing Time 규칙과 Least Slack Remaining 규칙을 결합한 것이다. Raman *et al.*(1989)은 (1|s_{ij}|TWT) 문제에서 ATC에 셋업을 추가하는 시도를 하였다. 그들은 셋업 s_{ij}을 작업 i을 수행하고 작업 j를 시작할 때 발생한다고 정의하였고, 남은 작업들은 우선순위를 가지고 우선순위가 높은 작업이 다른 작업에 우선하여 계획된다고 규정하였다. Raman *et al.* (1989)은 ATC 규칙의 작업 처리시간을 작업 준비시간으로 대체하여 풀었다.

Lee *et al.*(1997)는 역시(1|s_{ij}|TWT) 문제의 해법으로 ATCS(Apparent Tardiness Cost With Setup) 규칙을 사용하는 3단계 발견적 기법 절차를 제안하였다. Lee and Pinedo (1997)는 TWT를 최소화하는 (P|s_{ij}|TWT)문제를 풀기 위해 할당 규칙과 시뮬레이티드 어닐링(Simulated Annealing)기법을 결합한 3단계의 발견적 기법을 제시하였다. 이 연구에서 그들은 할당 규칙 방안으로 Lee *et al.*(1997)가 단일 기계 문제에서 제안하였던 ATCS 규칙을 수정하여 사용하였다.

Kim *et al.*(1995)은 TWT를 최소화하는(1|s_{ij}|TWT) 문제를 풀기 위한 방안으로 할당 규칙과 신경망(Neural Networks)을 결합한 혼합적인 방법(Hybrid Approach)을 사용하였다.

Schutten and Leussink(1996)은 동종 병렬기계의 납기 지연의 합을 최소화하는 문제(P|s_{ij}|TWT)를 분지한계법을 이용하여 해결하였다. 분지한계법을 이용하면 수많은 이웃해를 생성해서 수행 속도상의 문제를 가지게 된다.

Eom *et al.*(2002)은 제품군 작업시간(Family Setup Time)이 존재하는 병렬기계에서 TWT를 최소화하는 3단계 발견적 기법을 제시하였다. 이 연구는 TWT를 최소화하기 위한 방법으로 제품군 작업 준비 횟수를 줄이기 위하여 제품군 작업 위주의 할당 규칙을 사용하였다. 그러나 제품군 작업과 개별 부품 작업을 동시에 고려하지는 않았다.

대부분의 연구들에서 제안한 할당 규칙은 근시안적 한계를 벗어나지 못한다. 이러한 근시안적 한계를 극복하고 최적해의 근접해(Near Optimal Solution)에 도달하기 위해서는 타부탐색(Tabu Search)과 같은 메타 휴리스틱이 필요하다. 타부탐색(Tabu Search), 시뮬레이티드 어닐링(Simulated Annealing) 그리고 유전자 알고리듬(Genetic Algorithm)과 같은 메타휴리스틱(Meta Heuristic)은 조합 최적화 문제에 많이 적용되어 왔다 (Shin *et al.*, 2001).

타부탐색 알고리듬은 많은 연구에서 이용되었으며 Laguna *et al.*(1991)는 단일 기계 문제에서 모든 작업의 납기 지연과 작업준비 비용의 합을 최소화하기 위해 타부탐색 기법을 제시하였다. 그들이 사용한 타부탐색 기법은 해 영역을 탐색하는 데 있어서 교환이동(Swap Moves)과 삽입이동(Insert Moves)방법을 사용하였다.

Islam and Eksioglu(1997)은 납기 지연시간의 평균을 최소화하는 단일 기계 문제에 타부탐색 기법을 적용하였다. James (1997)는 조기 완료/납기 지연을 최소화하기 위한 단일 기계 일정 계획 수립 문제를 풀기 위해 타부탐색 기법을 제안하였다.

그들은 우수한 해를 찾기 위한 방안으로 압축해 영역(Compressed Solution Space)이라는 개념을 도입하였다. Shin *et al.* (2001)은 순서 의존적인 작업 준비시간이 존재하고 작업 가능시간이 존재하는 단일 기계에서 L_{Max}를 갖는 작업을 최소화하기 위해 제한적 타부탐색(Restricted Tabu Search) 해법을 제시하였다.

본 연구에서는 앞에서 언급한 작업 공구 제한 등의 제약 사항이 존재하며, 반복 공정을 고려한 병렬기계 상황에서 TWT를 최소화하는 효율적인 일정 계획 수립 알고리듬을 제안한다.

2. 본 론

2.1 병렬기계 문제의 정의

본 연구는 순서 의존적인 작업 준비시간과, 작업의 시작 가능 시간과 납기가 존재하고 반복적인 작업에서 작업들이 선·후행 관계에 따른 작업 시작 가능 시간의 변화, 병렬기계에서 작업을 가공하는 공구에 대한 제약 조건을 가진다. 순서 의존적인 작업 준비시간은 메이저 종류와 마이너 종류에 각각 존재하며 여러 번의 반복 공정을 통해야 최종 제품이 생산된다. 제품마다 각 사이클을 가공하기 위한 작업 공구가 하나만 존재하기 때문에 여러 대의 기계에서 같은 제품의 동일한 사이클 작업을 동시에 작업할 수 없고 병렬기계는 서로 다른 사이클 작업만을 한다. 주어진 문제는 <Figure 3>와 같이 표현될 수 있다.

주문 레벨(Order Level)에 있는 각 주문은 작업 레벨(Job Level)에서의 반복 작업에 대한 투입 가능 시간과 납기로 나뉘며 나누어진 작업 순서대로 기계에 투입되어야 한다. 주문을 완료하기 위해 첫 번째 사이클 작업부터 차례로 기계에 투입

된다. 가공을 마치고 완료된 사이클 작업은 다음 사이클 작업의 투입 가능 시간을 결정한다. 모든 사이클을 완료하면 주문은 완료되고 최종 완제품이 된다. 만일 기계에서 투입하고 싶은 작업과 같은 제품의 동일한 사이클이 다른 기계에서 가공중일 때 진행중인 사이클 작업이 완료될 때까지 투입하려는 작업은 기계 앞에 대기하게 된다. 만일 다른 기계가 유휴 하더라도 투입하려는 작업은 투입 가능 시간 전까지 작업에 대한 준비를 할 수 없다.

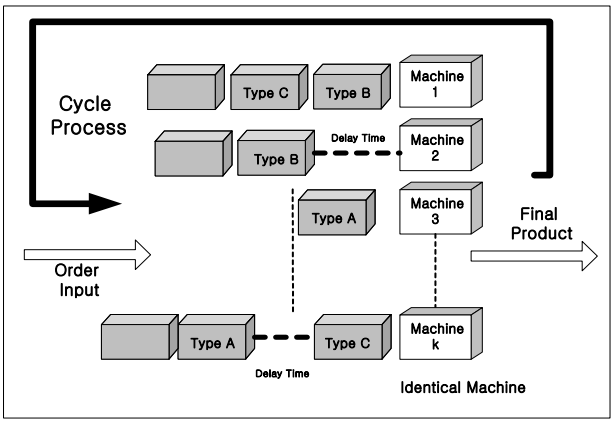

Figure 3. Definition of the parallel machines.

이러한 순서 의존적인 작업 준비시간, 작업의 선·후행 관계에서의 작업 투입시간 변화, 동일한 작업에 사용되는 공구에 대한 제한 등의 제약 조건을 가지는 환경에서의 효율적인 생산 관리 알고리듬을 제안한다. 알고리듬의 목적은 병렬기계에서의 TWT를 최소화시키는 것이다.

2.2 전체 알고리듬 구성

일정 계획 수립의 목적인 TWT를 최소화하기 위해 전체 알고리듬은 선행 계획단계와 초기해 생성단계 그리고 해의 개선단계로 구성된다.

선행 계획단계는 일정 계획을 수립하기 위해 주문에 대한 선행 작업을 수행하는 단계로, 주문을 계획 수립을 위한 작업으로 나눈다. 먼저, 납기별로 주문을 정렬하고 정렬된 주문을 세분화하여 작업으로 분할한다. 또한, 주문의 투입 가능 시간과 납기를 이용하여 분할된 작업의 납기와 투입 가능 시간을 계산한다.

초기해 생성단계에서는 선행 계획단계에서 나누어진 작업에 대한 초기해를 생성하는 단계이다. 선행 계획단계에서 나누어진 작업들의 투입 순서를 계산하고 계산된 순서대로 병렬기계에 할당하여 초기해를 생성한다. 초기해를 생성하기 위하여 TWT를 최소화할 수 있는 ATCS를 변형한 새로운 규칙을 제안한다.

해의 개선단계에서는 목적함수인 TWT를 최소화하기 위해 생성된 초기해를 향상시키기는 단계이다. 초기해의 근시안적 한계를 극복하고 최적해의 근접해에 도달하기 위해서는 메타 휴리스틱이 필요하다. 따라서 본 연구에서는 메타 휴리스틱의 방법인 타부탐색 기법을 사용하여 목적함수인 TWT를 최소화한다.

타부탐색 알고리듬은 계산시간을 많이 필요로 하기 때문에 속도를 높이기 위해 타부탐색 알고리듬에 Rolling Horizon 방법을 적용하였다.

2.3 선행 계획단계

선행 계획단계는 일정 계획을 수립하기 위해 주문을 작업으로 나누는 단계이다. 주문을 각 반복 공정에 따라서 작업으로 나누는 것은 각 사이클에 사용되는 작업 공구의 제약을 고려해야 하기 때문이다. 또한 본 문제의 정의에서 병목 공정인 포토 공정만을 고려하기 때문에 포토 공정의 반복 작업으로 주문을 나눌 수 있다.

선행 계획단계에서는 먼저 O개의 주문을 각 제품의 종류가 갖는 N개의 반복 공정으로 나누고 하나의 반복 공정을 작업이라고 정의한다. 그리고 주문이 가지는 투입 가능 시간과 납기를 이용하여 나누어진 작업들의 투입 가능 시간과 납기를 계산한다. 각 주문의 인덱스를 $o(o=1,2,\cdots,O)$라 하고 제품의 종류에 따라 반복되어 처리되는 사이클 인덱스를 i $(i=1,2,\cdots,N)$이라고 하면 작업 인덱스는 주문 인덱스와 사이클 인덱스를 합친 oi로 나타낸다.

작업의 투입시간(Release Time) r_{oi}와 납기(Due Date) d_{oi}는 주문이 가지는 투입시간(Release Time) r_o와 납기(Due Date) d_o를 사이클 가공시간(Processing Time) p_{oi}를 사용하여 계산한다.

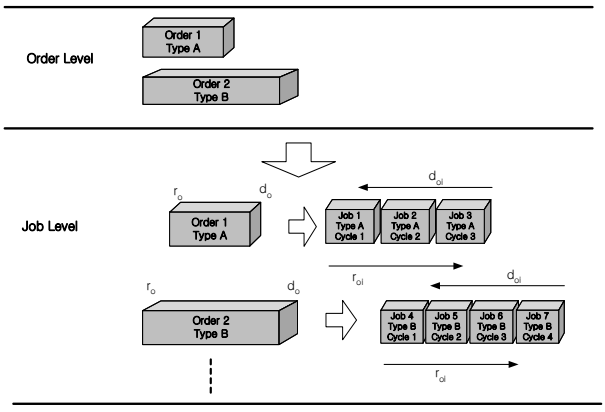

Figure 4. Subdivision of job in the cycling process.

<Figure 4>에서 보듯이 납기는 주문의 납기에서부터 후진전개(Backward)방법으로 계산하고, 투입 가능 시간은 주문의 투입 가능 시간에서 전진전개(Forward)방법으로 계산한다. 이러한 방법은 각 작업의 가장 빠른 작업 가능 투입시간과 최대 납기 준수시간을 구할 수 있게 된다. 단, 하나의 주문은 처리되는

공정 수에 따라 각 작업으로 분할되지만 각 분할된 작업은 더 이상의 분할을 허용하지 않는 최소 단위가 된다. 또한 기계는 아직 도착하지 않은 작업을 위해 미리 해당 작업의 작업 준비를 할 수 없다. 즉, 작업 투입 가능 시간 이후에서야 작업 준비와 가공을 할 수 있게 된다. 나누어진 작업들은 투입 가능한 선행 작업에 대해 빠른 납기를 가지는 순서로 정렬한다.

2.4 초기해 생성 단계

2.4.1 초기해 생성 알고리듬

초기해 생성단계는 선행 계획단계에서 나누어진 작업들에 대해 초기해를 생성하는 단계이다. 병렬기계 문제에서 초기해를 생성하는 방법으로, Lee and Pinedo(1997)가 제안한 ATCS (Apparent Tardiness Cost With Setup) 규칙이 있다. ATCS 규칙은 순서 의존적인 작업 준비시간과 작업의 납기를 고려하여 TWT를 최소화하는 (P| s_{ij} |TWT) 문제를 풀기 위해 적용된다. ATCS는 작업의 우선순위가 높을수록, 가공시간이 짧을수록, 그리고 순서 의존적인 작업 준비시간이 작을수록 그 해당 작업은 높은 인덱스를 갖게 된다. 하지만 ATCS는 반복적인 공정에서의 문제를 고려하지 않았고, 병렬기계에서 사용하는 공구의 제약을 해결할 수 없다. 또한 작업 투입 가능 시간이 변할 때의 문제를 해결할 수 없다.

따라서 본 연구에서는 순서 의존적인 작업 준비시간과 작업의 투입시간이 계속적으로 변하고, 작업 공구 제한 등의 제약사항을 고려한 RATCS (Restricted Apparent Tardiness Cost With Setup) 규칙을 제안한다. RATCS 규칙은 반복 공정에서 작업을 분할할 수 있도록 허용하며, 메이저 작업과 마이너 작업의 순서 의존적인 작업 준비시간을 동시에 고려하고, 기계에서 발생하는 작업 공구 등의 제약(Restriction) 사항을 추가하여 ATCS 규칙을 상황에 맞도록 사용하는 개선된 규칙이다.

RATCS는 작업의 우선순위가 높고, 가공시간이 짧으며, 메이저와 마이너의 작업 준비시간이 작을수록, 특히 반복 공정에서 작업 투입 가능 시간이 빠르고, 병렬기계에서 사용할 수 있는 작업 공구가 가능한 작업이 높은 인덱스를 가진다.

식 (1)에서, t는 작업 l의 가공 완료시간($C_{l_{o'i'}}$), $\overline{p}$와 $\overline{S}$는 각각 작업 순서가 결정되지 않는 작업들의 평균 가공시간과 평균 작업 준비시간을 의미한다.

k_1, k_2는 조정모수(Scaling Parameter)로서 문제 특성에 맞게 주어지며 식 (1)의 두 번째 항과 세 번째 항은 전체 인덱스 값에 미치는 영향을 조절해주는 역할을 한다.

식 (1)의 두 번째 항은 빠른 납기를 가지는 작업들에 대해 높은 인덱스 값을 주며, 특히 납기를 어긴 작업들에 대해서는 납기 지연정도 따라 더 큰 인덱스 값을 준다. 이렇게 하므로 작업 순서 결정 시, 납기 지연 정도의 차이를 분별할 수 있게 한다.

식 (1)의 세 번째 항에서는 작업의 투입시간을 순서 의존적인 작업 준비시간인 $S_{l_{o'i'(m,n)}j_{oi(m,n)}}$에 반영하였다. $S_{l_{o'i'(m,n)}j_{oi(m,n)}}$를 계산하기 위해 작업의 투입 시간을 계산하여야 한다. 먼저 작업 j의 투입 가능 시간은 제품의 종류와 작업의 사이클에 따라 제품 사이클의 작업 공구 투입 가능 시간(R_{Type_i})과 선행 작업의 완료시간($C_{j_{oi-1}}$)중 큰 값을 가진다.

R_{Type_i}는 공구가 기계에서 사용중일 때는 기계의 완료시간의 값을 가지고 사용하지 않을 때는 R_{Type_i}은 0이 된다. 그리고 작업 j의 투입 가능 시간이 작업 l의 종료시간인 t보다 작거나 같다면, 즉 작업 j가 t 시간에 투입 가능(Available)하다면 작업 l과 작업 j의 메이저 종류(Major Type)를 비교한다. 만일, 같으면 마이너 준비시간을 그대로 사용하고 다르면 메이저와 마이너 준비시간의 합을 사용한다. 반대로 작업 l의 종료시간과 작업 j의 가공 시작시간 사이에 유휴시간(Idle Time)이 발생하게 된다면 $S_{l_{o'i'(m,n)}j_{oi(m,n)}}$값에 유휴시간을 더하고 메이저 종류를 비교한 값을 사용한다. 따라서 $S_{l_{o'i'(m,n)}j_{oi(m,n)}}$는 불필요한 기계 유휴시간과 순서 의존적인 작업 준비시간을 함께 고려한 작업 준비시간이 된다. RATCS 규칙에서 사용하는 인덱스 계산 수식은 (1) 에 정리하였다.

$$I_{j_{oi}(m,n)}(t, l_{o'i'(m.n)}) = \frac{w_{j_{oi}}}{p_{j_{oi}}} \exp\left(-\frac{d_{j_{oi}} - p_{j_{oi}} - t}{k_1 \overline{p}}\right) \exp\left(-\frac{S_{l_{o'i'(m,n)}j_{oi(m,n)}}}{k_2 \overline{S}}\right) \quad (1)$$

Subject to,

$$S_{l_{o'i'(m,n)}j_{oi(m,n)}} = \begin{cases} f_{l_{o'i'(m,n)}=j_{oi(m,n)}} & r_{j_{oi}} < t \\ r_{j_{oi}} - t + f_{l_{o'i'(m,n)}=j_{oi(m,n)}} & otherwise \end{cases}$$

$$f_{l_{o'i'(m,n)}=j_{oi(m,n)}} = \begin{cases} (m_{l_{o'i'(m,n)}j_{oi(m,n)}} + s_{0j_{oi(m,n)}}) & l_{o'i'(m)} \neq j_{oi(m)} \\ s_{l_{o'i'(m,n)}j_{oi(m,n)}} & l_{o'i'(m)} = j_{oi(m)} \end{cases}$$

$$r_{j_{oi}} = Max(R_{Type_i}, C_{j_{oi-1}})$$

$$d_{j_{oi}} = d_o - \sum_{y=i+1}^{total\ cycle} p_{j_{oy}}$$

Numerical formula of RATCS index

[Notation]

$j_{oi(m,n)}$: 주문이 o, 사이클이 i, 메이저 종류(Major Type) m, 마이너 종류(Minor Type) n인 작업. 이하 작업 J

$I_{j_{oi}(m,n)}(t, l_{o'i'(m,n)})$: l 작업이 t 시간에서 끝났을 때 다음 작업 j가 갖는 인덱스(Index)

$w_{j_{oi}}$: 작업 j가 갖는 가중치(Weight)

$p_{j_{oi}}$: 작업 j가 갖는 가공시간(Processing Time)

$d_{j_{oi}}$: 작업 j의 납기(Due Date)

$\overline{p}$: 시간 t 이후의 잔여 작업들이 갖는 평균 가공시간 (Average Processing Time)

$S_{l_{o'i'(m,n)}j_{ok(m,n)}}$: 작업 l에서 작업 j로 올 때 걸리는 총 작업 준비시간(Setup Time)

$s_{l_{o'i'(m,n)}j_{oi(m,n)}}$: 작업 l에서 작업 j로 올 때 걸리는 마이너 준비시간(Minor Setup Time)

$m_{l_{o'i'(m,n)}j_{oi(m,n)}}$: 작업 i에서 작업 j로 올 때 걸리는 메이저 준비시간(Major Setup Time)

$\overline{S}$: 시간 t 이후의 잔여 작업들이 갖는 메이저와 마이너의 평균 준비시간(Average Major and Minor Setup Time)

$r_{j_{ok}}$: 작업 j의 투입 가능 시간(Release Time)

$C_{j_{oi-1}}$: 주문이 o이고 사이클 $i-1$인 작업의 완료시간

R_{Type_i} : 제품 사이클이 i의 작업 공구 투입 가능 시간

$f_{l_{o'i'(m,n)}=j_{oi(m,n)}}$: 두 작업의 메이저와 마이너의 종류를 비교하여 작업 준비시간을 결정하는 함수

RATCS는 위에서 설명한 작업의 우선순위와 작업의 가공시간, 순서 의존적인 작업 준비시간 및 작업 투입 가능 시간과 작업 공구의 사용 가능 여부를 계산하고 작업의 유휴시간까지 고려하여 작업들이 더 빠른 작업 완료시간을 가질 수 있도록 한다. 따라서 초기해 생성 시에 최적해에 근접한 좋은 해를 만들 수 있게 한다.

2.4.2 초기해 생성 절차

초기해 생성 절차는 앞에서 설명된 RATCS 규칙을 사용하여 기계의 종료시간에 할당되지 않은 작업들은 기계에서 가공했던 제품의 종류를 기준으로 인덱스를 계산하고 높은 인덱스의 작업을 그 기계에 할당한다. 기계의 종료시간을 기준으로 할당되지 않은 작업이 없을 때까지 계산하여 할당함으로써 초기해를 생성한다. <Figure 5>는 초기해 생성 시에 기계의 종료시간에서 기계에서 가공중인 제품의 종류를 기준으로 RATCS로 계산된 높은 인덱스의 작업을 기계에 할당하는 모습이다.

[초기해 생성 절차]

Step 1 : 기계를 선택

병렬기계에서 가장 빠른 작업 완료시간을 가지는 기계를 선택 한다.

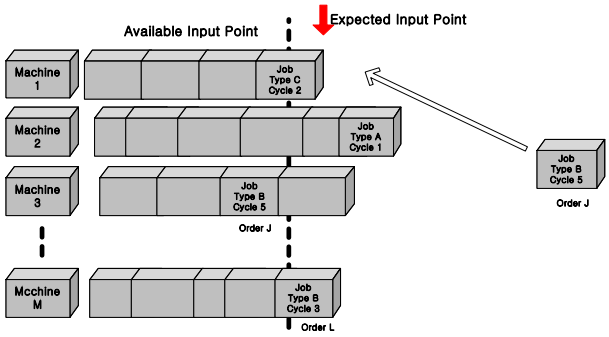

Figure 5. Job Allocations.

Step 2 : 작업의 투입시간 결정

각 기계에서 사용하는 동일한 사이클의 작업 가능 시간을 남은 작업들의 투입 가능 시간과 비교하여 높은 값을 작업의 투입시간으로 정한다.

Step 3 : RATCS로 정렬

현재 기계에서 작업중이던 작업 종류를 기준으로 남은 작업의 인덱스를 RATCS 규칙을 이용하여 구하고 가장 높은 인덱스별로 정렬한다.

Step 4 : 작업을 할당

가장 높은 인덱스를 가지는 작업을 선택하여 현재 기계에 할당하고 작업 완료시간을 구한다.

Step 5 : 완료시간 계산

동일한 사이클 작업 가능 시간을 Step 4에서 구한 완료시간으로 넣어준다.

Step 6 : 작업 투입 가능 시간 변경

현재 선택된 기계에 작업을 할당하고 Step 4에서 구한 완료시간을 현재 할당된 작업의 남은 사이클 작업의 작업 투입 가능 시간으로 변경한다.

Step 7 : 작업을 반복

남은 대상 작업들에 대하여 할당이 끝날 때까지 Step 1 ~ Step 6까지 반복한다.

2.5 해의 개선단계

2.5.1 타부탐색 알고리듬

초기해 생성단계에서 RATCS를 이용하여 생성된 초기해는 단순한 할당 규칙보다는 좋은 해를 보여주지만 할당 규칙의

근시안적 한계를 벗어나지는 못한다. 근시안적 한계를 극복하여 최적해에 근접한 해(Near Optimal Solution)에 도달하기 위하여 타부탐색 알고리듬을 통해 초기해 생성단계에서 얻은 초기해를 개선한다. 기존의 타부탐색 알고리듬은 문제의 크기가 커질수록 – 작업수와 기계수가 늘어날수록 – 생성되는 이웃해의 수가 급격히 증가하여 계산시간이 급증한다. 그러므로 이웃해의 수를 적절하게 제어하는 것이 알고리듬 성능에 큰 영향을 미친다. 따라서 본 연구에서는 이웃해 생성에서 탐색에 소요되는 계산시간을 단축시키기 위해 Rolling Horizon 방법을 사용한다. Rolling Horizon 타부탐색 알고리듬은 전체 일정 계획에 대하여 타부탐색을 수행하지 않고 타부탐색을 위해 일정 계획의 시작시간부터 일정한 간격으로 전체 구간을 나누어 탐색하는 것이다.

Rolling Horizon 타부탐색 알고리듬의 이웃해는 이동방법, 결정모수(Decision Parameter)인 탐색 깊이(Search Depth)와 탐색 반복 횟수(Search Iteration Number), 탐색 구간(Search Term)에 의해 결정된다. 이러한 결정모수들은 이웃해의 생성을 일정 수준 이하로 제약함과 동시에 이웃해를 찾는 과정을 집중적이면서 다양하게 수행할 수 있게 한다.

Rolling Horizon 방법을 사용하는 것은 대상 작업들이 시간축을 따라서 계획되어 있고, 반복 공정에 따른 작업의 선·후행 관계가 있기 때문이다.

또한 병렬기계의 공구 사용에 대한 제약에 의해 작업이 이동할 수 있는 구간의 크기가 결정되어 있기 때문이다. 이것은 작업이 이동 시에 기계에서 사용하는 공구의 제약을 가지므로 이동할 수 있는 작업의 수가 제한적이며, 선행 작업의 이동에 따라 후행 작업의 투입 가능 시간에 영향을 주어 이동 시점 이후의 일정 계획을 재조정해야 할 필요가 있기 때문이다.

따라서 Rolling Horizon 방법으로 구간을 제한하면 이동할 수 있는 대상 작업의 수를 제한하여 탐색하므로 탐색 속도를 향상시킬 수 있고, 이웃해의 종류를 줄여 적절한 이웃해를 생성하므로 탐색의 속도를 높일 수 있다. 또한 일반 타부탐색 알고리듬의 해를 보다 빠르고 집중적으로 탐색하여 같은 개선 상황에서 좋은 탐색 속도를 가질 수 있다.

2.5.2 Rolling Horizon 타부탐색 알고리듬 구조

Rolling Horizon 타부탐색 알고리듬은 한 구간에서 탐색이 완료되면 그 다음 구간으로 이동하고 마지막 구간까지 타부탐색을 수행하며 해를 개선하는 알고리듬이다.

Rolling Horizon 타부탐색 알고리듬을 사용하기 위해서는 탐색 영역인 구간을 나누어야 한다. 구간을 나누기 위해서는 계획시평(Scheduling Horizon)과 결정모수(Decision Parameter)를 사용한다. 계획시평(T)은 계획 대상 작업의 초기 할당된 시간부터 마지막 작업의 일정 계획 완료시간($C_{\max}$)까지이다. 구간의 크기는 계획시평을 결정모수(λ)로 나누어준 값이다. 여기서 결정모수(λ)는 우리가 나누고 싶어 하는 구간의 개수이다. 이를 수식으로 나타내면 다음과 같다.

$$\text{Search Term } T=(C_{\max}-Min(r_{oi}))/\lambda$$

구간의 개수는 결정모수인 λ와 같고 구간 $k(k=1\sim\lambda)$를 k는 1부터 시작하여 정해진 탐색 횟수 $i(i=1\sim r)$를 수행하게 되면 k를 하나씩 증가시킨다. 결국 마지막 구간인 λ까지 탐색 횟수만큼 탐색을 완료하면 타부탐색을 종료한다.

탐색 깊이(Search Depth)는 나누어진 탐색 구간을 몇 개의 구간을 포함하여 탐색을 실시할 것인지를 결정한다. 탐색 깊이는 계획 시평 상에 선·후행 작업이 길게 걸쳐 있으므로 해의 성능을 올리기 위해서는 후행 작업의 지연시간을 선행 작업에서 미리 파악하여 후행 작업의 투입 가능 시간을 앞당겨 전체 해의 성능을 높이기 위해 사용한다.

Rolling Horizon 타부탐색 알고리듬 절차는 현재 기계에 계획된 작업 중 현재 구간에 가중치 납기 지연의 합에 가장 크게 기여하는 작업($w_{j_{ok}}\times Max(C_{j_{ok}}-d_{j_{ok}},0)$ 값이 가장 큰 작업)을 선택한다. 현재 구간 안에서 가중치 납기 지연이 가장 큰 작업의 선행 작업들을 찾아서 투입 가능 시간까지 앞으로 옮기고 다시 일정 계획을 수립한다. 즉, 선행 작업들의 작업 종료시간을 단축시키고 이동된 시점 후부터는 다시 일정 계획을 수립하는 것이다. 만일 해의 향상이 있으면 기존의 최적해를 개선된 해로 바꾸고 현재 구간부터 탐색 과정을 반복한다. 그렇지 않으면 다시 원래의 해로 돌아가서 앞에서 이동한 작업의 후행 작업을 선택하고 다시 탐색한다.

이렇게 선·후행 관계로 탐색하다가 결국 가장 큰 가중치 지연을 보이는 작업까지 오면 그 작업의 탐색은 끝내고 다음으로 납기 지연시간이 큰 작업을 선택하여 위의 방법으로 다시 탐색을 시킨다. 이렇게 탐색하여 탐색 구간 λ까지 탐색을 완료하면 가중치 지연시간이 큰 작업들의 투입 가능 시간과 종료시간이 빨라져서 결국 지연시간이 줄어들게 된다. 이렇게 큰 가중치 납기 지연 작업들의 납기 지연 정도가 줄어들게 되면 전체 가중치 납기 지연시간의 합은 감소한다.

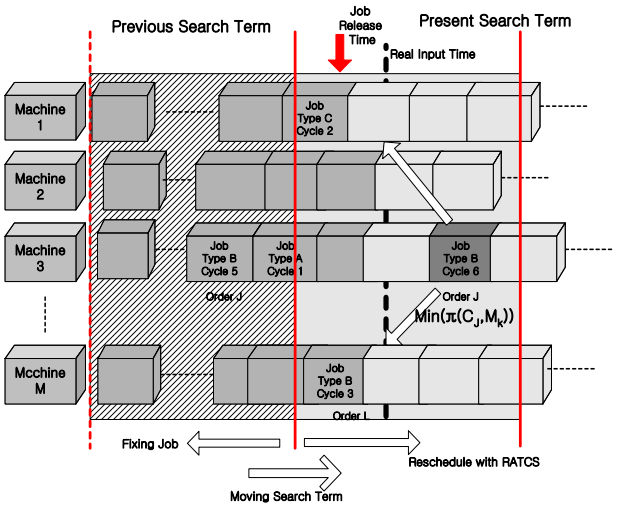

Figure 6. Movement of the job in the tabu search.

<Figure 6>은 타부탐색에서 이웃해를 찾아가는 모습을 보여준다. 탐색 구간 내에서 이동할 작업이 이동 가능한 시간에서 낮은 지연시간을 가진 곳에 할당하도록 하는 것이다.

[Rolling Horizon 타부탐색 알고리듬 절차]

Step 1 : 최우수해 설정

RATCS로 생성한 초기해 Π와 목적함수 값 $L_w(\Pi)$을 현재해 Π_{cur}와 $L_w(\Pi_{cur})$ 및 최우수해(Best Solution) Π_{Best}와 $L_w(\Pi_{Best})$로 설정한다.

Step 2 : 탐색 구간 설정

초기해의 전체 구간인($Min(r_{oi}) \sim C_{\max}$)을 λ로 나눈다. 즉, λ개의 구간을 만들고 탐색 깊이를 정한다. 탐색 구간 $\alpha\,(\alpha = 1 \sim \lambda)$와 탐색 반복 횟수 $\beta\,(\beta = 1 \sim \rho)$를 초기화하고 현재 탐색 구간과 탐색 횟수 값을 1로 설정한다.

Step 3 : 지연 작업 선택

현재 구간에서 가중치 납기 지연 정도 ($w_{j_{ok}} \times Max(C_{j_{ok}} - d_{j_{ok}}, 0)$)가 가장 큰 작업을 선택한다.

Step 4 : 선행 작업 검색

현재 구간 α에 선택된 작업의 선행 작업이 포함되어 있는지 검색한다. 선행 작업이 포함되어 있다면 Step 5로 없다면 Step 6으로 이동한다.

Step 5 : 선행 작업 이동 후 RATCS로 정렬

선행 작업 중 가장 앞선 작업을 선택하고 선택된 작업을 이동 조건에 맞추어서 이동한 후 삽입된 위치 다음의 작업들에 대해서는 RATCS 규칙으로 다시 정렬하여 이웃해를 생성한다.

Step 6 : 현재 작업 이동 후 RATCS로 정렬

현재 작업을 이동 조건에 맞추어서 이동한 후 삽입된 위치 다음의 작업들에 대해서는 RATCS 규칙으로 다시 정렬하여 이웃해를 생성한다.

Step 7 : 목적함수 값을 비교

이동된 작업은 타부 목록에 저장하고 생성된 이웃해 중 목적함수 값이 최우수해(Best Solution) Π_{Best}의 $L_w(\Pi_{Best})$보다 작으면 최우수해 Π_{Best}와 $L_w(\Pi_{Best})$를 생성된 이웃해로 바꾼다.

Step 8 : 탐색 횟수 증가

탐색 횟수 β를 1 증가시키고 탐색 횟수가 정해진 탐색 횟수보다 많으면 구간 α를 1 증가시키고 탐색 횟수 β는 1로 설정하고 Step 3으로 이동한다. 만일 α값이 정해진 구간의 값보다 크면 탐색을 종료한다.

2.5.3 이웃해 생성 방법

타부탐색 알고리듬은 많은 이웃해를 생성하고 비교하여 보다 좋은 해를 찾아간다. 본 Rolling Horizon 타부탐색 알고리듬에 사용한 이웃해의 생성 방법은 이동 가능한 작업을 투입 가능 시간까지 삽입 이동을 시킨다. 그리고 그 시점 이후의 모든 작업을 RATCS 방법으로 다시 일정 계획을 수립하는 두 단계로 이루어진다.

삽입 이동은 한 작업을 선택하여 일정 계획상의 다른 위치에 삽입하여 이웃해를 생성해 내는 방법이다. 삽입 이동에 의해 이웃해를 생성하기 위해서는 옮길 작업과 옮겨갈 위치를 결정해 주어야 한다. 옮길 작업은 탐색 구간 내의 모든 작업이 해당되고, 옮겨갈 위치는 다음의 삽입 이동 조건을 만족하는 모든 위치가 된다. 삽입 이동의 조건은 다음과 같다.

Ⅰ) $C_{M_i} \le r_{oi}$

Ⅱ) $C_{j_{oi}} = Min(\Pi(C_{j_{oi}}, M_k))$

Ⅲ) 타부 목록에 의해 금지된 목록이 아님.

병렬기계에서 이동할 기계를 M_k라 하고 기계의 완료시간을 C_{M_k}라 하고 옮길 작업을 j_{oi} 라고 하면 옮겨갈 위치 x는 j_{oi}의 시작 가능 시간이 된다. 작업이 이동할 수 있는 곳은 $Min(\Pi(C_{j_{oi}}, M_k))$으로 현재해 Π 안에서 이동하려는 작업 j_{oi}가 M_k의 기계에서 이동 후에 완료시간 $C_{j_{oi}}$이 최소가 되는 기계이고 작업 j_{oi}의 완료시간이 된다.

첫 번째 조건은 옮겨갈 위치는 선택된 작업의 시작 가능 시간 이후가 되어야 한다. 따라서 기계들의 완료시간은 작업의 시작 가능 시간 전에 끝나야 한다. 이것은 기계에 어떤 작업도 진행되지 않아야 작업 투입 가능 시간에 바로 투입할 수 있기 때문이다. 두 번째 조건은 옮겨갈 위치들 중에서 기계에 할당되어 있는 작업에 따라서 메이저와 마이너 작업 준비시간을 비교하여 이동 후의 작업 완료시간이 가장 작은 위치로 이동해야한다. 세 번째 조건은 타부목록에 의해 금지된 목록이 아니어야 이동할 수 있다.

3. 결과 분석 및 결론

3.1 비교 대안 및 실험

지금까지 발표된 연구들 중에서 본 연구 환경과 같은 조건을 갖는 일정 계획 문제에 대한 연구는 없었다. 따라서 알고리듬에 대한 객관적인 성능 평가를 위해 ATCS 규칙과 비교하였

다. 또한 기존의 할당 규칙(Dispatching Rule)인 MS와 EDD는 본 연구의 상황에 맞게 수정하여 비교하였다. 따라서 ATCS 규칙과 수정된 M-MS(Modified MS)와 M-EDD(Modified EDD) 규칙을 본 연구에서 제안한 RATCS 규칙과 동일한 실험 환경에서 모의실험을 통하여 비교한다.

동일한 환경에서의 실험을 위해 선행단계, 초기해 생성단계를 거친다. 다만 초기해 생성단계에서의 작업 투입 순서를 위한 우선순위만 위의 알고리듬에 의해 실험하였다. 따라서 본 연구에서 제안한 RATCS와의 동일한 환경에서의 실험을 통해 비교하였다. 비교를 위해 수정된 M-MS 규칙과 M-EDD 규칙은 다음과 같다.

• M-MS(Modified Minimum Slack)

본 연구에서 사용한 M-MS는 RATCS에서 작업 투입 순서를 결정하는 인덱스 값을 MS 규칙을 수정하여 작업의 우선순위를 결정하였다. 각 작업의 우선순서를 결정하는 인덱스는 작업 준비시간을 제외한 작업 여유시간을 계산하고 작업 여유시간의 정도에 따른 가중치로 계산되었다. 그리고 계산된 인덱스 값은 순서 의존적인 메이저 마이너 준비 작업시간과 여유시간이 고려되어 계산된다. 또한 반복 작업에 따른 선·후행 관계에서 작업 투입 가능 시간의 변경과, 기계에서 작업 공구의 사용 제약을 고려하였다.

• M-EDD(Modified Earliest Due Date)

본 연구에서 사용한 M-EDD는 위에서 사용한 M-MS와 동일한 제약 사항을 고려하여 구성하였다.

각 작업의 우선 순서를 결정하는 인덱스는 작업을 투입할 때의 시간에서 각 작업의 납기를 기준으로 납기의 순서와 납기의 지연 정도를 계산하고 가중치를 적용하여 계산된다. 계산된 인덱스 값은 순서 의존적인 메이저 마이너 준비 작업시간과 여유시간이 고려되어 계산된다.

또한 본 연구에서는 해의 개선을 위해 Rolling Horizon 타부탐색 알고리듬을 제안하였다. 제안한 Rolling Horizon 타부탐색 알고리듬의 성능을 평가하기 위해 일반 타부탐색 알고리듬과 비교하였다. 알고리듬의 성능 평가는 해의 향상 정도와 알고리듬 수행시간으로 하였다.

3.2 실험 설계

본 연구에서 제시한 알고리듬의 효과를 검증하기 위하여 동일한 실험 환경 상태에서 모의실험을 통해 제시된 대안과의 비교를 실시하였다.

실험을 위해 주어진 문제에는 동종 병렬기계(Identical Machine)만 존재한다. 주문은 병렬기계를 주문된 제품 종류의 사이클 수만큼 반복 공정을 거쳐야 작업이 끝난다. 병렬기계는 같은 제품의 동일한 사이클 작업은 한 대에서만 가능하며 가공중인 작업이 끝날 때까지 같은 제품의 같은 사이클 작업은 대기한다.

알고리듬의 성능을 비교하기 위한 척도로는 작업의 평균 가중치 납기 지연, 개선 비율, 알고리듬 수행시간이다. 실험을 통해 나온 결과값으로 대안 알고리듬과 성능을 비교하였다. 실험에 필요한 실험 환경 데이터는 <Table 1>에 정리하였다.

Table 1. Simulation experiment data

Experiment data List	Data values
Number of orders	100, 200, 300
Number of major types	2, 4
Number of minor types	3, 6
Number of parallel machines	3, 4, 5
Distribution of cycle	U[5, 10]
Distribution of cycle processing time	U[30, 60]
Distribution of major setup time	U[80, 100]
Distribution of minor setup time	U[25, 30]
Distribution of order	U[1,550]
Number of search terms(λ)	3, 6
Search depths	2
Search iteration numbers	9, 18
Number of problems	Order(3) *major type (2) *minor type (2) *parallel machine(3) = 36 Problems
Number of simulations	10 times of each Problem 36*10 = Total 360 times

알고리듬의 성능 비교는 결정모수를 변화시켜 가면서 나온 결과값 중에서 경향이 아주 다른 것들을 제외한 평균값으로 수행하였다.

본 연구에서 제시한 알고리듬은 모두 C++를 이용하여 구현하였고, 펜티엄4 2GHz 컴퓨터에서 실험하였다

3.3 실험 결과 분석

3.3.1 RATCS의 성능 분석

3.2에서 언급한 총 360회의 실험에 대한 결과를 <Table 2>, <Table 3>, <Table 4>에 정리하였다.

먼저 초기해 생성에서 제안한 RATCS 알고리듬과 대안 알고리듬을 비교한다.

초기해 생성 시에 RATCS 알고리듬과 대안 알고리듬을 비교하기 위해 주문수, 기계수, 메이저 종류수, 마이너 종류수를 기준으로 평가한다.

Table 2. Mean TWT about 100 orders

Order	Majors	Minors	Machines	Jobs	M-EDDTWT	M-MSTWT	ATCSTWT	RATCSTWT
100	4	6	5	747	104445	83422	72651	52450
100	4	6	4	747	250139	246159	151404	69659
100	4	6	3	747	983550	966766	819483	205479
100	2	6	5	749	82257	71967	52420	40326
100	2	6	4	749	149651	128841	92322	47657
100	2	6	3	749	555018	486971	319515	102792
100	4	3	5	746	158643	145858	106930	59164
100	4	3	4	746	320629	307868	208990	87127
100	4	3	3	746	1310522	1233758	1070513	257260
100	2	3	5	716	57110	50525	30314	24601
100	2	3	4	716	110931	93062	53746	25445
100	2	3	3	716	267264	251060	165090	37936

Table 3. Mean TWT about 200 orders

Order	Majors	Minors	Machines	Jobs	M-EDDTWT	M-MSTWT	ATCSTWT	RATCSTWT
200	4	6	5	1491	270849	261987	196757	123498
200	4	6	4	1491	619028	609880	355624	153481
200	4	6	3	1491	2098366	1927686	1708415	431265
200	2	6	5	1502	188953	162950	122305	74458
200	2	6	4	1502	307813	288942	183800	78944
200	2	6	3	1502	973064	891839	627574	146314
200	4	3	5	1480	361262	341035	267623	129356
200	4	3	4	1480	779517	762311	603657	158885
200	4	3	3	1480	4626556	4183884	3059670	883556
200	2	3	5	1498	176208	163372	109777	57070
200	2	3	4	1498	280587	264579	217395	65428
200	2	3	3	1498	1016083	865972	759491	130881

Table 4. Mean TWT about 300 jobs

Order	Majors	Minors	Machines	Jobs	M-EDDTWT	M-MSTWT	ATCSTWT	RATCSTWT
300	4	6	5	2271	524069	494068	333269	217069
300	4	6	4	2271	1160367	1078355	864673	274334
300	4	6	3	2271	5335706	4820435	4433407	1330835
300	2	6	5	2306	271185	263011	164743	120370
300	2	6	4	2306	437955	432483	255337	125022
300	2	6	3	2306	1501017	1222440	974432	253208
300	4	3	5	2287	487146	423697	298987	197696
300	4	3	4	2287	849343	790202	543975	247952
300	4	3	3	2287	3228744	3165079	3095025	653321
300	2	3	5	2338	252886	245388	172561	104357
300	2	3	4	2338	574211	539815	339861	128565
300	2	3	3	2338	2427538	2352680	1991377	332335

주문수에 따른 성능은 <Figure 7>에서 보듯이 제안한 알고리듬이 대안 알고리듬보다 좋은 성능을 나타내고 있다. 기계의 능력이 고정된 상태에서 작업수가 증가하기 때문에 <Figure 7>에서처럼 주문수가 증가할수록 TWT의 합은 증가한다.

하지만 주문수에 상관없이 본 연구에서 제안한 RATCS 알고리듬은 대안 알고리듬에 비해 TWT가 작다. 또한 주문수가 늘어남에 따라 대안 알고리듬보다 TWT의 증가 비율은 작다. 따라서 본 연구에서 제안한 알고리듬이 대안 알고리듬보다 효율적이다.

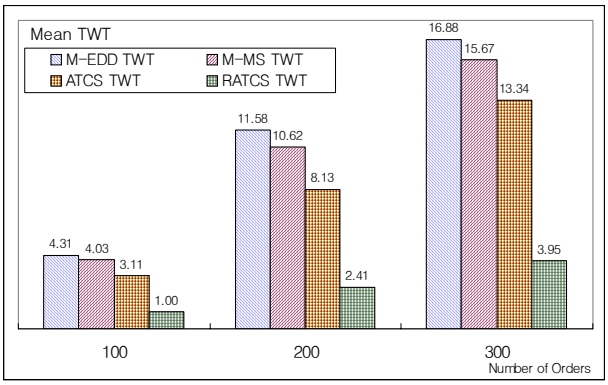

Figure 7. TWT about the number of orders.

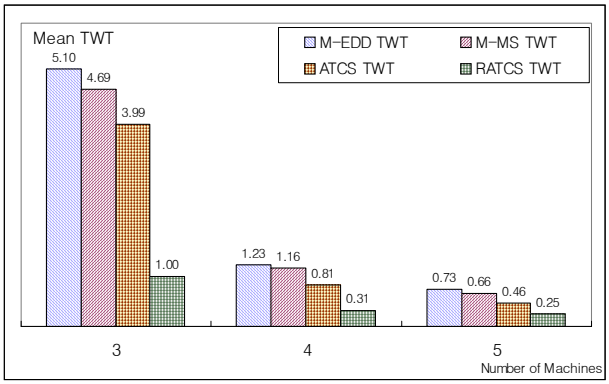

Figure 8. Mean TWT about the number of machines.

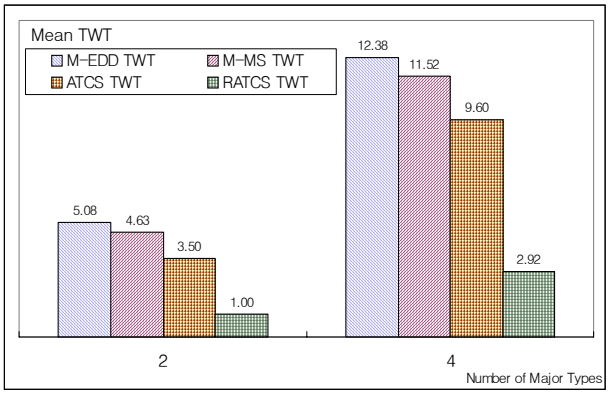

Figure 9. Mean TWT about the number of major types.

기계수에 따른 성능은 <Figure 8>에서 보듯이 대안 알고리듬보다 좋은 성능을 나타내고 있다. 기계의 수가 증가하면 고정된 주문을 처리하는 기계 능력이 커지기 때문에 본 알고리듬과 대안 알고리듬 모두 TWT는 줄어든다. 하지만 기계수에 상관없이 본 연구에서 제안한 RATCS 알고리듬은 대안 알고리듬에 비하여 TWT가 작고 효율적이다.

메이저 종류수에 따른 성능은 <Figure 9>에서 보듯이 대안 알고리듬보다 좋은 성능을 나타내고 있다. 메이저 종류가 작을수록 작업을 준비시간 중에서 큰 부분을 차지하는 메이저 준비시간이 줄어들기 때문에 본 알고리듬과 대안 알고리듬 모두 TWT는 줄어든다. 메이저 종류가 늘어나면 메이저 종류의 수만큼 다양하고 많은 준비시간이 필요하여 TWT는 증가한다. 하지만 메이저 종류수에 상관없이 RATCS 알고리듬은 대안 알고리듬에 비하여 TWT가 작고 효율적이다.

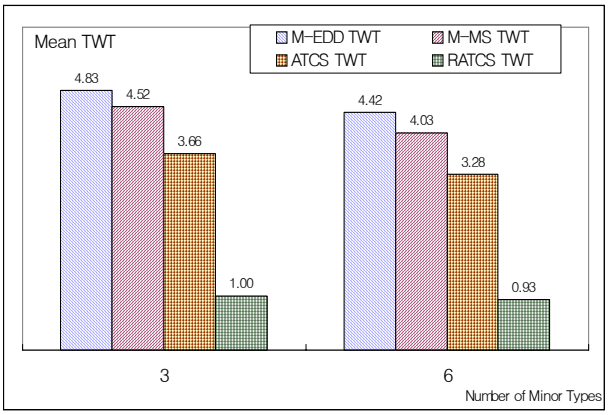

Figure 10. Mean TWT about the number of minor types.

마이너 종류수에 따른 성능은 <Figure 10>에서 보듯이 대안 알고리듬보다 좋은 성능을 나타내고 있다. 또한 마이너 종류가 증가하여도 TWT는 거의 변화가 없고 약간 감소하였다. 메이저 종류수가 고정일 때 마이너 종류수가 증가하면 작업 공구 제한 등의 제약 사항이 줄어들게 되어 기계의 활용도(Utilization)가 늘어나기 때문에 TWT는 오히려 감소하게 된다. 작업 공구 제한이 줄어들면 같은 제품의 동일한 사이클을 작업하는 상황이 줄어들어 기계가 유휴(Idle) 상태임에도 작업 공구 제한에 의해 대기해야 하는 작업의 수가 줄어들게 되고 따라서 기계의 활용도가 높아지고 TWT는 감소한다.

이상에서 분석 해 본 바와 같이 RATCS 알고리듬과 대안 알고리듬을 비교해 보면 주문수, 기계수, 메이저 종류의 수, 마이너 종류의 수에 상관없이 좋은 결과를 얻고 있다는 것을 알 수 있다. 따라서 연구에서 제안한 초기해 생성 알고리듬인 RATCS가 대안 알고리듬보다 우수한 초기해를 생성하는 것을 볼 수 있다.

3.3.2 Rolling Horizon 타부탐색 알고리듬 분석

본 연구에서는 RATCS 규칙에 의한 초기해를 생성하여 Rolling Horizon 타부탐색 알고리듬을 통하여 해를 개선한다. 따라서 해의 개선을 위해 타부탐색 알고리듬에서 사용한 Rolling Horizon 방법의 적합성을 평가하기 위하여 일반적인 타부탐색 알고리듬과 비교하였다.

<Table 5>, <Table 6>은 주문수 100개일 때, RATCS로 만든 초기해를 기준으로 Rolling Horizon 타부탐색(RTS) 알고리듬과 일반적인 타부탐색(GTS) 알고리듬에서 개선 횟수를 9번과 18번으로 하여 비교한 것이다.

Table 5. Mean TWT and Time about 9 times

Majors	Minors	Machines	Jobs	RATCSTWT	RATCSTime	RTSTWT	RTSTime	GTSTWT	GTSTime
4	6	5	747	52450	639	25076	45506	26657	221680
4	6	4	747	69659	655	37316	77680	47575	219770
4	6	3	747	205479	689	137461	154891	146393	150600
2	6	5	749	40326	833	8905	3409	12057	20444
2	6	4	749	47657	655	16769	21512	17816	68700
2	6	3	749	102792	625	77921	96330	81832	226691
4	3	5	746	59164	635	20807	28331	33673	132924
4	3	4	746	87127	684	53486	79316	58983	104621
4	3	3	746	257260	678	170459	127686	189830	281042
2	3	5	716	24601	574	4693	6003	6912	44024
2	3	4	716	25445	567	8216	14028	8549	29928
2	3	3	716	37936	572	19863	26462	23372	64459

Table 6. Mean TWT and Time about 18 times

Majors	Minors	Machines	Jobs	RATCSTWT	RATCSTime	RTSTWT	RTSTime	GTSTWT	GTSTime
4	6	5	747	52450	639	18927	57648	25263	305667
4	6	4	747	69659	655	35634	101419	44227	364817
4	6	3	747	205479	689	125258	182919	132029	435922
2	6	5	749	40326	833	3215	8461	3513	41446
2	6	4	749	47657	655	11347	23797	14318	158834
2	6	3	749	102792	625	75200	112086	75963	321994
4	3	5	746	59164	635	14553	36928	24666	209698
4	3	4	746	87127	684	44138	90289	52695	221431
4	3	3	746	257260	678	164976	155338	171905	443013
2	3	5	716	24601	574	4111	14205	5474	58672
2	3	4	716	25445	567	6823	19006	6945	71498
2	3	3	716	37936	572	16654	40311	18517	77100

Rolling Horizon 타부탐색 알고리듬의 개선 횟수는 구간×탐색 횟수이므로 일반 타부탐색 알고리듬의 개선 횟수와 동일하게 하기 위하여 <Table 5>에서는 구간의 수를 3으로 구간당 탐색 횟수는 3으로 설정하여 일반 타부탐색 알고리듬의 개선 횟수 9와 동일하게 하였다. 또한 <Table 6>에서는 구간을 6으로 구간당 탐색 횟수를 3으로 설정하여 일반 타부탐색 알고리듬과 동일한 개선 횟수를 갖게 하였다.

<Table 5>, <Table 6>에서 타부탐색의 효율과 타부탐색에 의한 개선된 해의 TWT 값을 비교해 보면, 기계수나 주문수에 상관없이 RATCS 초기해보다 일정한 개선 비율로 TWT가 작아진 것을 볼 수 있고 타부탐색 알고리듬이 해의 성능을 향상시킨 것을 확인할 수 있다.

<Figure 11>, <Figure 12>에서 보듯이 일반적인 타부탐색 알고리듬의 수행시간이 본 연구에서 제안한 Rolling Horizon 타부탐색보다 훨씬 크다는 것을 알 수 있다. 또한 TWT도 Rolling Horizon 타부탐색 알고리듬이 더 좋은 것을 알 수 있다. 이것은 제안한 Rolling Horizon 타부탐색 알고리듬은 전체 구간을 일정한 탐색 구간으로 나누어 탐색 구간 안에서 집중적인 탐색을 수행하기 때문이다. 따라서 Rolling Horizon 타부탐색 알고리듬이 일반적인 타부탐색 알고리듬 방법보다 같은 탐색 횟수에서 빠른 속도로 개선된 해를 찾을 수 있다. 따라서 본 연구에서 제안하는 타부탐색 알고리듬을 이용하는 것이 더 효과적이라고 생각된다.

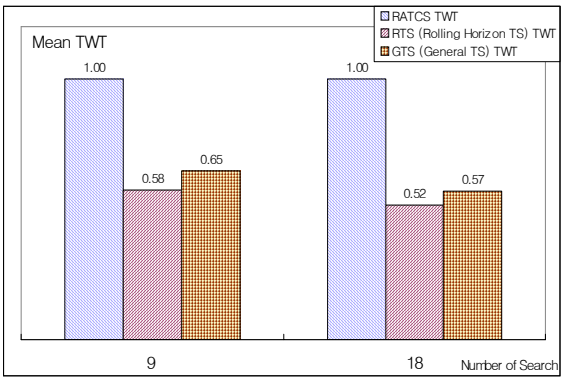

Figure 11. Mean TWT.

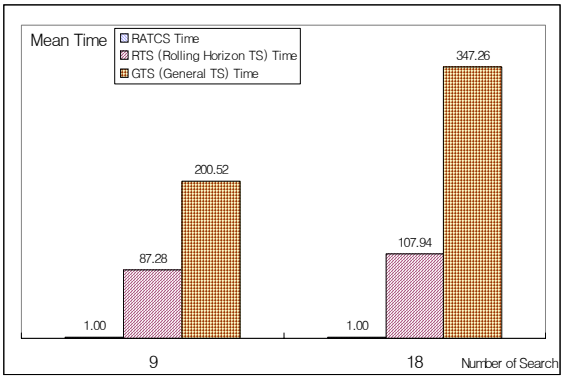

Figure 12. Mean Time.

본 연구의 주어진 환경에 대해 효과적인 일정 계획을 수립하기 위해서는 RATCS에 의한 알고리듬이 비교 대안보다 좋은 성능을 가졌으며 제안한 타부탐색 알고리듬은 비교적 짧은 시간 안에 해의 성능을 향상시킬 수 있었다.

4. 결론 및 추후 연구

본 연구는 병렬기계에서 작업 공구의 제약 사항이 존재하며, 순서 의존적인 메이저와 마이너 작업 준비시간과 반복 공정을 고려한 병렬기계에서 TWT를 최소화하는 병렬기계 일정 계획 수립 방법을 제안하였다. 본 연구에서 제시된 방법은 주문을 작업 단위로 나누는 선행 계획단계와 RATCS를 이용한 초기해 생성단계 그리고 타부탐색 알고리듬을 이용한 해의 개선단계로 구성하였다. 초기해 생성단계에서의 RATCS 규칙은 기존의 ATCS 규칙을 개선하여 반복 공정에서 순서 의존적인 작업 준비시간, 작업의 시작 가능 시간, 작업 공구 제한 등의 제약 사항 등을 고려할 수 있도록 제안된 규칙이다. 또한 타부탐색 알고리듬의 속도를 높이기 위해 타부탐색의 대상 기간을 제한하는 Rolling Horizon 탐부탐색 방법을 사용하였다.

초기해 생성에서 제시한 RATCS 알고리듬의 객관적인 평가를 위하여 작업 공구의 제약을 가진 ATCS와 기존의 할당 규칙인 EDD와 MS를 사용하였고 작업수와 기계수의 변화에 우수한 초기해를 생성하는 것을 입증하였다. 또한 일반적인 타부탐색 알고리듬보다 빠른 시간에 좋은 성능을 갖는 Rolling Horizon 타부탐색 알고리듬도 제안하였다.

본 연구에서 제안한 알고리듬은 반도체 팹 공정과 유사한 특성을 갖기 때문에 반도체 팹의 포토 공정으로 적용이 가능할 것으로 생각된다. 그러나 일반적인 팹 공정에서는 포토 공정과 포토 외 공정이 존재하기 때문에 사이클과 사이클 사이의 지연시간을 고려한 확장된 일정 계획 수립 알고리듬에 대한 연구가 필요하다.

참고문헌

Baker, K. R. and Su, Z. S.(1974), “Sequencing with due-dates and early start times to minimize maximum tardiness”, *Naval Research Logistics Quarterly*, **21**, 171-176.

Bitran, G. R. and Gilbert, S. M.(1990), “Sequencing Production on Parallel Machine with Two Magnitudes of Sequence-dependant Setup Cost”, *Journal of Manufacturing and Operations Management*, **3**, 190-206

Eom, D-H., Shin, H-J., Kwun, I-H., Shim J-K., and Kim, S-S.(2002), “Scheduling Jobs on Parallel Machines with Sequence-Dependent Family Set-up Times”, I*nternational Journal of Advanced Manufacturing Technology*, **19**, 926-932.

Garey, M. R. and Johnson, D. S.(1979), Computers and intractability: A guide to the theory of NP completeness, San Francisco.

Islam A. and Eksioglu, M.(1997), “A tabu search approach for the single-machine mean tardiness problem,” *Journal of the Operational Research Society*, **48**(7), 751- 755.

James, R. J. W.(1997), “Using tabu search to solve the common due date early/tardy machine scheduling problem,” *Computers and Operations research*, **24**(3), 199-208.

Kim, S. Y., Lee Y. H. and Agnihotri, D.(1995), “A hybrid approach to sequencing jobs using heuristic rules and neural networks,”

Production Planning & Control, **6**, 445-454.

Kim S. C. and Bobrowski, P. M.(1994), Impact on sequence-dependent setup time on job shop scheduling performance, *International Journal of Production Research*, **32**(7), 1503-1520.

Laguna, M., Barnes, J. W. and Glover, F.(1991), "Tabu search methods for single machine scheduling problem," *Journal of Intelligent Manufacturing*, **2**, 63-74.

Lee, Y. H., Bhaskran, K. and Pinedo, M.(1997), "A heuristic to minimize the total weighted tardiness with sequence-dependent setups," *IIE Transactions*, **29**, 45-52.

Lee, Y. H. and Pinedo, M.(1997), "Scheduling jobs on parallel machines with sequence-dependent setup times," *European Journal of Operations Research*, **100**, 464-474.

Monma, C, and Potts, C. N.(1994), "Analysis of huristic for preemptive parallel machine scheduling with batch setup times," *Operation Research*, **32**, 1243-1263.

Raman, N., Rachamadugu, R. V. and Talbot, F. B.(1989), "Real-time scheduling on an automated machine center," *European Journal of Operational Research*, **40**, 222-242.

Schutten, J. M. J. and Leussink, R. A. M.(1996), "Parallel machine scheduling with release dates, due dates and family setup times," *International Journal of Production Economics*, **46**, 119-126.

Shin H.-J., Kim S. S. and Ko K. S.(2001), "A Tabu Search Algorithm for single machines scheduling Problem with Job Release Times and Sequence-dependent Set up Times" *Journal of korean Institute of Industrial Engineering* , **27** , 2, 158-168.

So K. C.(1990), "Some heuristics for scheduling jobs on parallel machines with setup," *Management Science*, **36**, 467-475.

Tang, C. S.(1990), "Scheduling batches on parallel machines with major and minor setups," *European Journal of Operation Research*, **46**, 28-3.

Tang, C. S. and Wittrock R. J.(1985), "Parallel machines scheduling with major and minor setups", York Town Heights, NY, RC 11412 IBM TJ Watson Research Center.

Vepsalainen, A. and Morton, T. E.(1987), "Priority rules and lead time estimation for job shop scheduling with weighted tardiness costs," *Management Science*, **33**, 1036-1047.