Journal of the Korean Institute of Industrial Engineers
Vol. 34, No. 3, pp. 318-327, September 2008.

주기정비를 고려한 단일기계 일정계획을 위한 휴리스틱 알고리즘 설계

이상헌[1†] · 이인구[2]

[1]국방대학교 운영분석학과 / [2]공군본부 전투발전단

Heuristic Algorithm for the Single-machine Scheduling with Periodic Maintenance

Sang-Heon Lee[1] · In-Gu Lee[2]

[1]Department of Operations Research, Korea National Defense University, Seoul 122-875, Korea
[2]Studies and Analyses Wing, ROK Air Force Hq, Chungnam 321-929, Korea

This paper considers a single-machine scheduling problem with availability constraints. In many realistic situations, machines may not be always available due to various reasons such as maintenance, breakdown and repair. However, most literature on scheduling assume that the machines are available at all times. This paper deals with a single-machine scheduling problem with periodic maintenance. If the maintenance decision is made jointly with the job scheduling, the system will be more effective. The objective is to minimize the total completion time of jobs. This problem is proved to be NP-hard in the strong sense. The proposed breaking heuristic(BH) algorithm rule is established by some theorems and conditions. Our computational results show that the BH algorithm is much more efficient than existing heuristic.

Keywords: Scheduling, Availability Constraint, Maintenance, Total Completion Time, Breaking Heuristic

1. 서 론

일정계획은 순차적으로 해야 하는 작업을 계획하고 우선순위를 결정하는 것으로서 가용한 자원의 활용을 최적화하기 위한 방법이다. 효율적인 일정계획 수립은 현대의 극도로 경쟁적인 기업환경 하에서 생산관리의 매우 중요한 요소 중 하나이다. 단일기계 일정계획 모델은 제조공정의 유형 중에서 비교적 단순하고 특별한 사례이지만, 이에 대한 분석을 통하여 병렬기계 또는 흐름공정에서는 얻을 수 없는 통찰력과 더불어 좀 더 복잡한 모델에 적용하기 위한 휴리스틱 알고리듬을 얻을 수 있다. 전형적인 단일기계 일정계획 연구에서는 '기계는 항상 가용(available)하다'는 것을 가정한다. 그러나 현실적으로 많은 산업현장에서는 다음과 같은 기계 가용성 제약이 발생할 수 있다(Leung, 2004). 첫째, 일정계획 대상기간의 처음에 기계를 사용할 수 없는 경우로 이는 이전 대상기간에 계획된 작업이 완료되지 않았거나, 중요한 작업이 예약되어 있는 경우에 발생한다. 둘째, 기계에 대한 정비로 가용성 제한(availability constraints)이 발생하는 경우이다. 만약 정비에 대한 결정이 사전에 이루어진다면, 일정계획에서 불가동 기간은 제약조건이 될 것이고, 생산계획과 정비계획의 조화를 통해서 생산성을 높일 수 있을 것이다. 셋째, 기계 고장과 수리에 관련된 경우로 잠재적인 혹은 부분적인 기계 고장 시 즉각적인 수리를 할 것인지, 아니면 비효율적인 작업처리 속도로 좀 더 사용할 것인지에 관한 문제 등이 그 예가 될 수 있다. 또한, 기계 가용성 제약에 관련된 연구는 일정계획 대상기간 동안 가용성 제약 발생기간이 확정적인(deterministic) 경우와 결정변수(decision variable)인 경우로 구분할 수 있다. 가용성 제약이 결정변수인 경우 그 기간은 작업 일정계획과 함께 결정되어야 한다.

† 연락저자 : 이상헌, 122-875 서울시 은평구 수색동 205번지 국방대학교 운영분석학과, Tel : 02-300-2374, Fax : 02-309-6233, E-mail : leesangh@kndu.ac.kr
2008년 1월 21일 접수; 2008년 3월 28일 수정본 접수; 2008년 3월 30일 게재 확정.

주기정비(Periodic Maintenance)를 고려한 단일기계 일정계획에 관한 연구는 일정계획 대상기간 동안 수행되는 정비의 횟수 및 정비 시작시간의 유동성 여부에 따라 다음과 같은 네 가지 유형으로 분류할 수 있다(Chen, 2006a). 첫째, 각 기계는 단일한 불가동 기간을 가지며 이 기간의 시작시간이 고정되어 있는 경우(Problem for Single Unavailability and Constant : Problem SC)이다(Leon and Wu, 1992; Lee and Liman, 1992; Lee, 1996; Sadfi *et al*., 2005). Lee *et al*.(1997), Sanlaville and Schmidt (1998), Schmidt(2000) 등은 이와 관련된 연구조사를 하였다. 둘째, 각 기계는 단일한 불가동 기간을 가지며 이 기간의 시작시간이 결정변수인 경우(Problem for Single Unavailability and Decision Variable : Problem SV)이다(Lee and Leon, 2001; Lee and Lin, 2001; Yang *et al*., 2002; Kacem *et al*., 2008). Yang *et al*. (2002)부터 정비주기가 결정변수인 주기정비를 작업 일정계획과 혼합하여 연구하기 시작하였다. Kacem *et al*.(2008)은 가중 총 작업완료시간(weighted sum of completion times)을 최소화하기 위한 분지한계 모형, 혼합 정수계획 모형, 동적계획 모형을 제안하고, 제안된 분지한계 모형과 동적계획 모형이 혼합 정수계획 모형보다 우수함을 보였다. 셋째, 각 기계는 복수의 불가동 기간을 가지며 이 기간의 시작시간이 고정되어 있는 경우(Problem for Multiple Unavailability and Constant : Problem MC)이다(Leon and Chen, 2003; Chen, 2006b; Ji *et al*., 2007). Chen (2006b)은 총 흐름시간(total flow time)을 최소화하기 위한 분지한계 모형과 휴리스틱 알고리듬을 제안하였고, Ji *et al*.(2007)은 최종작업 완료시간(makespan) 최소화 문제에서 전형적인 최장 처리시간(LPT : Longest Processing Time) 알고리듬의 worst-case ratio가 2임을 보이고 $P = NP$가 아니라면 LPT 알고리듬이 가능한 최선의 방법임을 보였다. 마지막으로 각 기계는 복수의 불가동 기간을 가지며 이 기간의 시작시간이 결정변수인 경우(Problem for Multiple Unavailability and Decision Variable: Problem MV)이다(Qi *et al*., 1999; Graves and Lee, 1999; Aktruk *et al*., 2003, 2004; Chen, 2006a). Chen(2006a)은 평균 흐름시간(mean flow time)을 최소화하기 위한 네 가지 수리모형과 대상작업의 수가 큰 경우에 적용하기 위한 휴리스틱 알고리듬을 제시한 바 있다.

본 연구는 일정계획 대상기간 동안 복수의 주기정비가 수행되고, 그 기간의 시작시간이 결정변수인 Problem MV에 관한 문제 중 총 작업완료시간(total completion time)을 최소화하는 방법에 관한 것이다. 이러한 유형의 문제는 NP-hard(Qi *et al*., 1999)이기 때문에, 본 연구에서는 대상 작업의 수가 큰 경우에 유용하게 적용할 수 있는 휴리스틱 알고리듬을 제시한다.

2. 기존해법 고찰

주기정비를 고려하지 않는 일반적인 단일기계 일정계획에서 총 작업완료 시간을 최소화하는 방법은 최단처리 시간(SPT : Shortest Processing Time) rule에 의하여 작업처리시간이 적은 작업부터 우선 처리하는 방식이다. Chen(2006a)은 이러한 SPT rule을 바탕으로 Problem MV에서 평균흐름시간을 최소화하기 위한 다음과 같은 휴리스틱 알고리즘을 제안한 바 있다. 첫째, 모든 작업을 SPT rule에 의하여 정렬하고, 첫 번째 작업부터 첫 번째 batch의 첫 번째 위치에 배열을 시작한다. 이 batch는 차기 주기정비가 수행될 때 까지 처리되는 작업들의 집합으로 Problem MV에는 복수 회의 주기정비가 포함되기 때문에 전체 작업을 부분집합인 batch들로 구분할 수 있다. 둘째, 추가되는 작업으로 인하여 현재 batch의 처리시간이 정비주기 상한선에서 정비 소요시간을 뺀 값보다 크게 되면, 그 작업은 다음 batch의 첫 번째 위치에 배열한다. 셋째, 마지막 작업까지 배열이 완료되면 종료한다. 그러나 위의 휴리스틱 알고리듬에서 생성된 batch 수는 최적해의 batch 수보다 클 수 있으며, Example 1은 이에 대한 간단한 예(counter example)를 나타낸다.

Example 1 : 단일기계가 있는 작업장에서 처리해야 하는 네 개의 작업(J_1, J_2, J_3, J_4)에 대한 일정계획을 수립하려고 한다. 각 작업의 처리시간은 각각 p_1, p_2, p_3, p_4 이고 $p_1 = p_2 = 1, p_3 = p_4 = 4$이다. 기계는 정비주기 하한선($u$) 이후에 정비를 시작하여 정비주기 상한선($v$) 이전에 정비를 완료하여야 하고, 이러한 주기정비에 소요되는 시간(w)은 일정한 것으로 가정한다. 여기서 $u = 4, v = 7, w = 2$이다.

<Figure 1>의 schedule A는 Chen(2006a)의 휴리스틱 알고리듬에 의해 생성된 계획이고, schedule B는 최적해를 나타낸 것이다. Schedule A에서는 작업시간이 적은 작업부터 먼저 배치되는데 만약 J_2 다음에 J_3을 바로 배치한다면, J_3의 작업 완료시

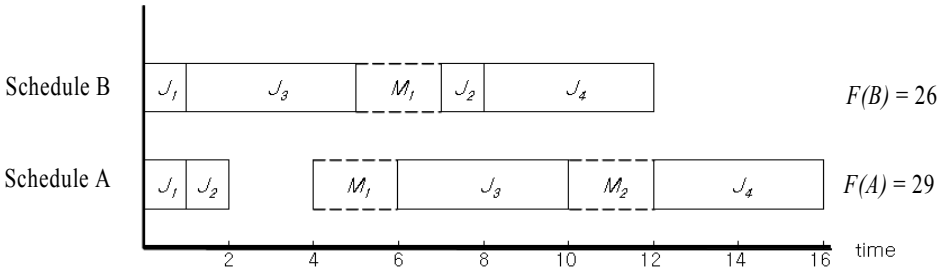

Figure 1. Counter-example of scheduling for Example 1

간은 6이 되고 이때 바로 정비를 시작한다고 해도 정비 완료시간은 8이 되어 정비주기 상한선을 초과하게 된다. 따라서 첫 번째 정비(M_1)는 J_2 다음에 실시하게 되고, 이때 정비 시작시간은 정비주기 하한선인 4가 된다. 이에 따라 기계에는 단위시간 1의 유휴시간이 발생한다. J_3은 첫 번째 정비 완료시간인 6에 시작해서 10에 끝나는데 J_4도 J_3 다음에 바로 처리될 수 없기 때문에 두 번째 정비(M_2)가 끝나는 12에 시작해서 16에 완료되고 이것으로 schedule A는 종료된다. 이와 대비하여 최적해인 schedule B에는 한 번의 정비만 소요되고, 기계 유휴시간도 발생하지 않음을 알 수 있다. 각 스케줄의 총 작업 완료시간을 $F(A), F(B)$라고 할 때 <Table 1>과 같이 $F(A) = 29 > F(B) = 26$로 최적해의 총 작업완료 시간이 더 적음을 알 수 있다.

Table 1. The total completion time of two schedules in <Figure 1>

Job position		1	2	3	4	Total completion time
Schedule A	Job	J_1	J_2	J_3	J_4	29
	Completion time	1	2	10	16	
Schedule B	Job	J_1	J_3	J_2	J_4	26
	Completion time	1	5	8	12	

본 연구에서는 Chen(2006a)의 SPT 휴리스틱 알고리듬에서 나온 스케줄을 기반으로 정비횟수와 기계 유휴 시간을 감소시킴으로써 총 정비완료 시간을 감소시킬 수 있는 breaking heuristic(BH) 알고리듬을 제안한다.

3. 문제 정의 및 수리모형

3.1 문제 및 용어 정의

본 연구에서 다루는 문제는 정비주기가 결정변수인 복수의 주기정비를 고려한 단일기계 일정계획에 관한 것이다. 즉, 작업 일정계획과 기계 정비계획을 동시에 고려하는 문제이다. 기계에 대한 차기 주기정비는 직전 주기정비 완료시점(단, 첫 번째 정비는 일정계획 시작시간)에서 정비주기 하한선 이상 경과한 이후에만 시작할 수 있고, 반드시 정비주기 상한선 경과 이전에 완료되어야 한다. 각각의 정비를 수행하는데 소요되는 정비 소요시간은 모두 동일하고 확정되어 있다. 또한 특정 작업의 선취(preemption)는 허용되지 않고, 기계 가용성 제약의 세 가지 유형(Leung, 2004) 중 가용성 제약 발생 전에 완료되지 못한 작업은 처음부터 다시 시작해야 하는 nonresumable한 기계를 대상으로 한다. 또한, 모든 작업은 일정계획 시작시간에 수행될 수 있고 각 작업을 수행할 때 소요되는 준비시간은 0이며, 각 작업의 처리시간은 정비주기 하한선보다 크지 않다.

<Figure 2>는 본 연구에 대한 일정계획의 전형적인 예시를 나타낸 것으로 일정계획 대상주기 동안 복수회의 주기정비가 이전 주기정비 이후 시점부터 정비주기 하한선과 상한선 사이에서 반드시 수행되어야 하고, 각각의 정비 사이에는 1개 이상의 작업들로 구성된 전체 작업의 부분집합이 되는 batch들이 있다. 본 연구 구현을 위해 필요한 용어를 정의하면 다음과 같다.

<Notation>
- N : 작업 수
- u : 정비주기 하한선
- v : 정비주기 상한선
- w : 정비 소요시간
- M : Big M[매우 큰 양(+)의 수]
- J_i : 작업번호 $i(i = 1, 2, \cdots, N)$
- p_i : J_i 처리시간
- C_i : J_i 작업 완료시간
- L : batch 수
- B_j : batch 번호 $j(j = 1, 2, \cdots, L)$
- t_j : B_j내의 모든 작업에 대한 처리시간의 합
- gb_j : B_j에서 t_j가 정비주기 하한선보다 작은 경우 그 차이 값 [즉, $max\{0, u-t_j\}$]
- rb_j : B_j에서 정비주기 상한선과 정비 소요시간의 차에서 t_j를 뺀 값(즉, $v-w-t_j$)
- nb_j : B_j내의 작업 수

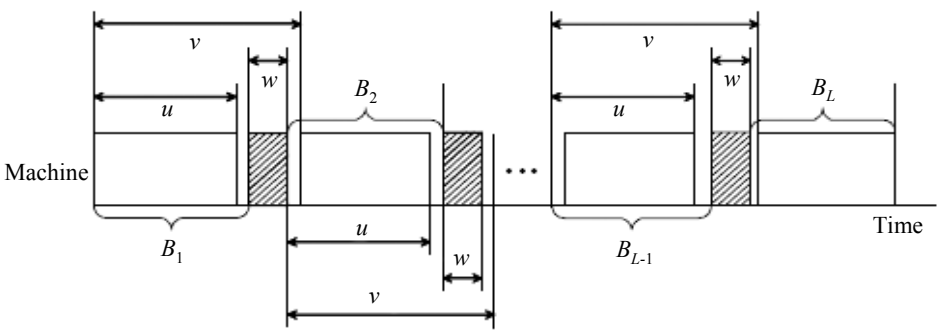

Figure 2. An example for the periodic maintenance

$p_{j(k)}$: j번째 batch의 k번째에 배치된 작업의 처리시간 ($j = 1, 2, \cdots, L, k = 1, 2, \cdots, nb_j$)

k_j : j번째 배치된 작업 다음에 바로 정비가 수행되면 1, 아니면 0

b_j : j번째 배치된 작업 다음에 바로 정비가 수행될 때(즉, kj = 1일 때) 그 작업이 포함된 batch 내의 모든 작업 처리시간의 합.
만일 $k_j = 0$ 이면 $b_j = 0$

g_j : j번째 배치된 작업 다음에 바로 정비가 수행될 때(즉, kj = 1일 때) 그 작업의 완료시간에서 차기정비 시작 가능시간 중 최소까지의 시간. 만일 $k_j = 0$ 이면 $g_j = 0$

r_j : j번째 배치된 작업의 완료시간에서 차기정비 시작 가능시간 중 최대까지의 시간

x_{ij} : J_i가 j번째 배치된 작업이면 1, 아니면 0

M_j : j번째 주기정비 수행기간($j = 1, 2, \cdots, L-1$)

CM_j : j번째 주기정비 완료시간($j = 1, 2, \cdots, L-1$)

$F(S)$: 스케줄 S의 총 작업완료시간[즉, $F(S) = \sum_{j \in S} C_j$]

3.2 수리모형

본 연구에서 제안하는 휴리스틱 알고리듬을 평가하기 위한 수리모형은 Chen(2006a)의 혼합 정수계획법(MIP : Mixed Integer Programming)을 이용한 모형 중 총 작업완료시간 산출에 부합되게 변형한 것이다. 총 작업완료 시간은 식 (1)과 같이 네 가지 요소로 구성될 수 있다. 첫째, 각 작업 batch별 포함 작업들에 대한 완료시간의 합, 둘째, 각 작업 batch 이후에 오는 작업의 수에 해당 batch 작업 처리시간을 곱한 값들의 합, 셋째, 각 작업 batch 이후에 오는 작업의 수에 정비 소요시간을 곱한 값들의 합, 넷째, 각 작업 batch 이후에 오는 작업의 수에 해당 batch의 작업완료시간에서 차기 정비시간을 뺀 값을 곱한 값들의 합이다. 즉 각 작업의 완료시간은 해당 batch내에서 그 작업의 총 작업완료시간에 이전 batch들의 처리시간, 수행된 정비시간과 기계 불 가동 시간 등을 모두 합한 값이 된다.

총 작업완료시간(total completion time)

$$= \sum_{j=1}^{N}(v-w-r_j) + \sum_{j=1}^{N-1}(N-j)b_j + w\sum_{j=1}^{N-1}(N-j)k_j + \sum_{j=1}^{N-1}(N-j)g_j \quad (1)$$

각 작업은 식 (2)와 같이 한번만 수행되고, 각 위치에는 식 (3)과 같이 한 개의 작업만 배치될 수 있다.

$$\sum_{j=1}^{N} x_{ij} = 1, \ i = 1, 2, \cdots, N \quad (2)$$

$$\sum_{i=1}^{N} x_{ij} = 1, \ j = 1, 2, \cdots, N \quad (3)$$

각 위치에 배치된 작업의 완료시간에서 차기 주기정비 시작 가능시간 중 최대까지의 시간은 정비주기 상한선과 정비 소요시간의 차와 직전에 정비가 수행되었는지 여부에 따라 식 (4), 식 (5), 식 (6)과 같은 제약이 있고, 여기서 식 (5)와 식 (6)의 등식은 각 위치 직전에서 주기정비의 수행(즉, $k_{j-1} = 0$) 여부에 따라 결정된다.

$$r_1 + \sum_{i=1}^{N} p_i x_{i1} = v - w \quad (4)$$

$$r_j + \sum_{i=1}^{N} p_i x_{ij} \le r_{j-1} + Mk_{j-1}, \ j = 2, 3, \cdots, N \quad (5)$$

$$r_j + \sum_{i=1}^{N} p_i x_{ij} \le v - w + M(1-k_{j-1}), \ j = 2, 3, \cdots, N \quad (6)$$

각 batch 내의 모든 작업 처리시간의 합과 각 batch에서 마지막 작업의 완료시간에서 차기정비 시작 가능시간 중 최소까지의 시간에 대한 제약은 식 (7) 및 식 (8)과 같고, 등식은 각 위치 다음에 정비가 수행될 때 성립한다.

$$b_j \ge v - w - r_j - M(1-k_j), \ j = 1, 2, \cdots, N-1 \quad (7)$$

$$g_j \ge u - b_j - M(1-k_j), \ j = 1, 2, \cdots, N-1 \quad (8)$$

각 변수의 비음조건 및 이진 변수에 대한 조건은 식 (9)와 같다.

$$r_j \ge 0, \ j = 1, 2, \cdots, N \quad (9)$$
$$g_j \ge 0, \ j = 1, 2, \cdots, N-1$$
$$x_{ij} \ is \ binary, \ i = 1, 2, \cdots, N, \ j = 1, 2, \cdots, N$$
$$k \ is \ binary, \ j = 1, 2, \cdots, N-1$$

4. Breaking Heuristic 알고리듬

4.1 Breaking rule을 위한 관련 Theorem

Theorem 1 : 서로 다른 작업개수로 구성된 두 개의 batch에서 만일 적은 작업개수를 가진 batch를 뒤에 배치하면, 더 적은 총 작업완료시간을 갖는다.

증명 : 전체 일정계획의 부분 스케줄 $S = (M_k, J_p, M_l, J_q, J_r)$이라고 가정하자. 이때 J_p는 batch B_i내의 작업이고, J_q와 J_r은 batch B_j내의 작업이다. 스케줄 S'를 batch B_i와 B_j을 교환해서 발생하는 것이라고 하면 $S' = (M_k, J_q, J_r, M'_l, J_p)$이고, 스케줄 S와 S'의 총 작업완료시간은 식 (10) 및 식 (11)과 같이 된다.

$$F(S) = CM_k + p_p + (CM_k + p_p + g_p + w + p_q) + (CM_k + p_p + g_p + w + p_q + p_r) \quad (10)$$

$$F(S') = CM_k + p_q + (CM_k + p_q + p_r) + (CM_k + p_q + p_r + g_r + w + p_p) \quad (11)$$

여기서 $F(S) - F(S') = 2(p_p + g_p) - (p_q + p_r + g_r) > 0$ 이므로, 스케줄 S'가 더 적은 총 작업완료시간을 갖는다. □

Theorem 2 : 서로 같은 작업개수로 구성된 두 개의 batch에서 만일 batch내 모든 작업에 대한 처리시간의 합이 큰 batch를 뒤에 배치하면, 더 적거나 같은 총 작업완료시간을 갖는다.

증명 : 전체 일정계획의 부분 스케줄 $S = (M_k, J_p, J_q, M_l, J_r, J_s)$이라고 가정하자. 이때 J_p와 J_q는 batch B_i내의 작업이고, J_r과 J_s는 batch B_j내의 작업이다. $p_p + p_q > p_r + p_s$ 이고, 스케줄 S'를 batch B_i와 B_j을 교환해서 생기는 것이라고 하면 $S' = (M_k, J_r, J_s, M'_l, J_p, J_q)$이고, 스케줄 S와 S'의 총 작업완료시간은 식 (12) 및 식 (13)과 같이 된다.

$$F(S) = CM_k + p_p + (CM_k + p_p + p_q) + (CM_k + p_p + p_q + g_q + w + p_r) + (CM_k + p_p + p_q + g_q + w + p_r + p_s) \quad (12)$$

$$F(S') = CM_k + p_r + (CM_k + p_r + p_s) + (CM_k + p_r + p_s + g_s + w + p_p) + (CM_k + p_r + p_s + g_s + w + p_p + p_q) \quad (13)$$

여기서 $F(S) - F(S') = 2(p_p + p_q + g_q) - 2(p_r + p_s + g_s) \geq 0$ 이므로, 스케줄 S'가 더 적거나 같은 총 작업완료 시간을 갖는다. □

4.2 Breaking Heuristic 알고리듬

Problem MC의 경우에는 총 작업완료 시간을 최소화하기 위하여 전체 작업을 처리하는데 소요되는 정비횟수를 최소화하여야 한다(Chen, 2006b). 그러나 Problem MV의 경우에는 정비주기가 결정변수이기 때문에 특수한 경우에는 총 작업완료시간을 최소화하는 최적 스케줄에서 수행되는 주기정비 횟수가 임의의 스케줄의 주기정비 횟수보다 많을 수도 있다. Example 2는 이 경우를 예시할 수 있는 간단한 예이다.

Example 2 : 단일기계가 있는 작업장에서 처리해야 하는 네 개의 작업(J_1, J_2, J_3, J_4)에 대한 일정계획을 수립하려고 한다. 각 작업의 처리시간은 각각 p_1, p_2, p_3, p_4이며 $p_1 = p_2 = 1$, $p_3 = p_4 = 2$이다. 기계의 정비주기 하한선(u)은 2, 정비주기 상한선(v)은 4이고, 정비 소요시간(w)은 $0 < w < 1$ 사이의 임의의 고정된 수이다.

스케줄 π_1을 Example 2에서 최소 주기정비 횟수를 갖는 스케줄이고 스케줄 π_2를 최적해라고 하면, $\pi_1 = (J_1, J_3, M_1, J_2, J_4)$이고, $\pi_2 = (J_1, J_2, M_1, J_3, M_2, J_4)$이다. 각 스케줄의 총 정비완료시간 $F(\pi_1)$, $F(\pi_2)$는 <Table 2>와 같이 $F(\pi_1) = \sum_{j \in \pi_1} C_j = 14 + 2w$, $F(\pi_2) = \sum_{j \in \pi_2} C_j = 13 + 3w$이다. 따라서 $F(\pi_2) < F(\pi_1)$이고, 여기서 최적해 π_2의 주기정비 횟수는 최소 주기정비 횟수보다 크다.

그러나 일반적으로 일정계획 대상기간에 수행되는 주기정비를 줄일 수 있다면 Example 1과 같이 makespan 및 기계 유휴시간이 감소하게 되어 총 작업완료시간의 감소를 기대할 수 있다. 본 연구에서 제시하는 휴리스틱 알고리듬은 Chen(2006a)의 SPT 휴리스틱 알고리듬으로 구한 스케줄을 초기해로 하여 그 스케줄에서 주기정비를 감소시킴으로써 해의 개선을 도모한다. 주기정비의 감소는 전체 일정계획에서 특정 batch 내의 모든 작업을 다른 batch에 배치시킬 수 있을 때 가능하고, 이는 정비주기 상한선과 정비 소요시간의 차에서 각 batch 내의 모든 작업 처리시간의 합(t_j)을 뺀 값의 총합$\{\sum_{j=1}^{L-1} rb_j = \sum_{j=1}^{L-1}(v - w - t_j)\}$이 batch 당 최대 작업처리 가능시간($v$-$w$)보다 큰 경우에 시도할 수 있다. 다음에 제시하는 breaking rule은 이러한 주기

Table 2. The total completion time of two schedules for Example 2

Job position		1	2	3	4	Total completion time
Schedule π_1 (The number of required maintenance activities : 1)	Job	J_1	J_3	J_2	J_4	14 + 2w
	Processing time	1	2	1	2	
	Completion time	1	3	4 + w	6 + w	
	Accumulated completion time	1	4	8 + w	14 + 2w	
Schedule π_2 (The number of required maintenance activities : 2)	Job	J_1	J_2	J_3	J_4	13 + 3w
	Processing time	1	1	2	2	
	Completion time	1	2	4 + w	6 + 2w	
	Accumulated completion time	1	3	7 + w	13 + 3w	

정비 감소 방법의 시행 조건 및 절차 등을 구체화한 것이다.

Breaking rule

1. <시작 조건> breaking은 SPT rule에 의하여 생성된 초기해의 정비주기 상한선과 정비 소요시간의 차에서 batch내 모든 작업의 처리시간의 합을 뺀 값의 총 합($\sum_{j=1}^{L} rb_j$)이 정비주기 상한선과 정비 소요시간의 차(v-w)보다 크거나 같을 때 시행한다(breaking rule의 기본조건).
2. <가능 조건> nonresumable한 작업이기 때문에 breaking 대상 batch(B_j)의 모든 작업이 $B_k(k = j+1, \cdots, L-1)$에 배치될 수 있어야 breaking을 시행한다(SPT rule에 의한 breaking rule의 기본조건).
3. <우선순위 조건> B_i, B_j ($i<j$)에서 두 batch 모두 breaking이 가능하면, B_j부터 breaking을 시도한다(SPT rule에 의한 breaking rule의 기본조건).
4. <시행 방법> breaking 시에는 대상 batch의 작업 중 작업 처리시간{$p_{j(k)}$}이 적은 작업부터 배치 가능한 앞쪽 batch의 여유 공간(rb_j)부터 SPT rule에 따라 배치한다(SPT rule에 의한 breaking rule의 기본조건).
5. <수락 조건> breaking은 실시 후 총 작업완료시간[$F(S)$]이 개선될 경우에만 실시한다(breaking rule의 기본조건).
6. <재배치 조건 Ⅰ> breaking 실시 후 batch내 작업의 수(nb_j)의 내림차순으로 각 batch를 재배치한다(Theorem 1에 의거).
7. <재배치 조건 Ⅱ> batch내 작업의 수(nb_j)가 같은 경우에는 batch내 모든 작업의 처리시간의 합(t_j)이 작은 batch를 앞쪽에 배치한다(Theorem 2에 의거).

위의 breaking rule을 적용하여 정비주기가 결정변수인 복수의 주기정비를 고려한 단일기계 일정계획에서 총 정비완료 시간을 최소화하기 위한 breaking heuristic 알고리듬은 다음과 같다.

Breaking Heuristic(BH) 알고리듬(<Figure 3 참조>)

Step 1 : 모든 작업을 작업 처리시간의 오름차순으로 정렬한다.

Step 2 : SPT rule을 이용하여 초기 스케줄을 구성한다.

(2-1) : 작업 처리시간이 가장 작은 작업부터 첫 번째 batch의 첫 번째 position에 배치한다.

(2-2) : 배치한 작업의 C_i 및 해당 batch의 t_j, nb_j, rb_j, gb_j 등을 계산한다.

(2-3) : 다음 작업(J_{i+1})이 현재 batch에 위치 가능(즉, $p_{i+1}<v-w-t_j$)하면, 현재 batch내의 다음 position에 위치시키고, (2-5)로 이동한다.

(2-4) : 정비주기 하한선을 고려하여 다음 batch의 시작시간을 계산하고, (2-2)로 이동한다.

(2-5) : 마지막 작업(즉, 작업 처리시간이 가장 큰 작업)까지 배치가 완료되었으면 step 4로 이동하고, 아니면 (2-2)로 이동한다.

Step 3 : 총 작업완료시간[$F(S)$]을 산출한다.

Step 4 : breaking 시작 조건을 만족하면 step 5, 아니면 step 6으로 이동한다.

Step 5 : breaking rule을 이용하여 개선된 스케줄 구성을 시도한다.

(5-1) : 마지막 주기정비 직전 batch(B_{L-1})부터 한 단계씩 앞쪽 batch로 이동하면서 breaking 가능 조건 만족하는 batch를 찾는다(반복 시에는 미 조사된 마지막 batch부터 검사한다).

(5-2) : 시행 방법에 따라 해당 batch에 대한 breaking을 실시하고, 재배치 조건을 적용한다.

(5-3) : 수락 조건을 검사하여 만족하면 (5-4), 그렇지 않으면 (5-5)로 이동한다.

(5-4) : breaking을 확정하고, (5-6)으로 이동한다.

(5-5) : breaking 시도를 취소한다.

(5-6) : 첫 번째 batch까지 조사가 완료되었으면 STEP 6, 아니면 (5-1)로 이동한다.

Step 6 : 알고리듬을 종료한다.

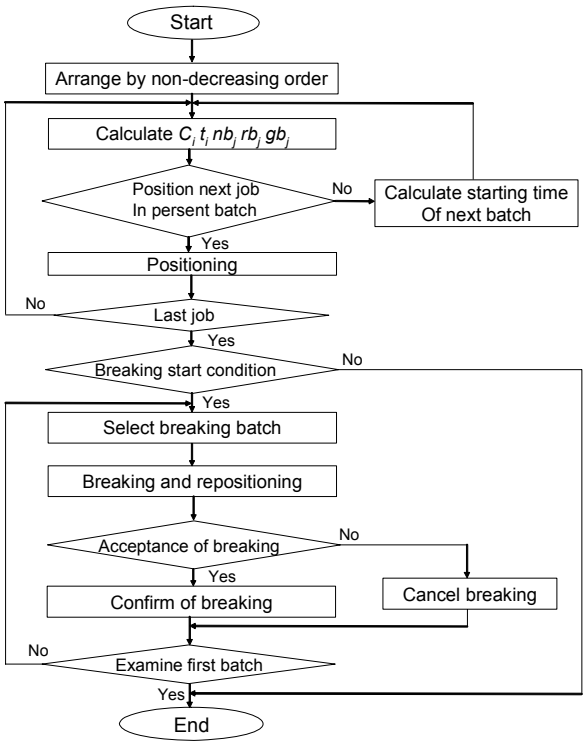

Figure 3. Flow chart of BH algorithm.

4.3 수치 예제

Example 3은 BH 알고리듬과 Chen(2006a)의 SPT 휴리스틱 알고리듬을 비교하기 위한 간단한 수치예제로 Chen(2006b)에 사용된 예제를 정비주기 상한선과 하한선을 설정하여 본 연구에 부합되게 적용한 것이다. 만약 $w = v$-u로 제한한다면, Problem MC 문제가 된다. 각 알고리듬의 성능은 제 3장에서 살펴본 MIP 모형과의 차이를 나타내는 식 (14)의 PED(Percentage Error De-

viation)로 평가한다.

$$PED = \frac{[\text{알고리듬의 총 작업완료시간} - MIP\text{모형의 총 작업완료시간}]}{MIP\text{모형의 총 작업완료시간}} \times 100 \quad (14)$$

Example 3 : 대상작업 10개(<Table 3 참조>), *u = 7, v = 10, w = 2.*

Table 3. Data for Example 3

$J_{(i)}$	$J_{(1)}$	$J_{(2)}$	$J_{(3)}$	$J_{(4)}$	$J_{(5)}$	$J_{(6)}$	$J_{(7)}$	$J_{(8)}$	$J_{(9)}$	$J_{(10)}$
$p_{(i)}$	1	1	2	2	3	3	3	4	4	5

<Table 3>은 10개의 작업을 SPT 순으로 배열한 것이다. $\sum_{i=1}^{10} p_{(i)} = 28$로 정비를 고려하지 않으면, makespan C_{max}=*28*이다. 그러나 기계의 차기 주기정비까지의 최대 가동 가능시간(*v-w*)이 8이기 때문에 이 스케줄에는 최소 3회 이상의 주기정비가 포함되어야 한다. <Table 4>는 작업 처리시간이 작은 작업부터 정비주기에 맞추어 배열한 것으로, SPT 휴리스틱 알고리듬의 최종해이면서 BH 알고리듬의 초기해이다. 여기에는 4회의 정비를 포함하여 *C_{max} = 41, F(S) = 158*이고, $\sum_{j=1}^{4} rb_j = 9$, $\sum_{j=1}^{4} gb_j = 5$가 된다.

Table 4. Final solution of Example 3 by the SPT heuristic algorithm

$J_{(i)}$	$J_{(1)}$	$J_{(2)}$	$J_{(3)}$	$J_{(4)}$	M_1	$J_{(5)}$	$J_{(6)}$	M_2	$J_{(7)}$	$J_{(8)}$	M_3	$J_{(9)}$	M_4	$J_{(10)}$
$p_{(i)}$	1	1	2	2	(1+2)	3	3	(1+2)	3	4	(2)	4	(3+2)	5
$C_{(i)}$	1	2	4	6	(9)	12	15	(18)	21	25	(27)	31	(36)	41
$F_{(S)}$	1	3	7	13		25	40		61	86		117		158

한편, $\sum_{j=1}^{4} rb_j = 9 \geq v - w = 8$로 breaking 시작조건을 만족하고 이에 따라 마지막 주기정비 직전 batch인 4번째 batch부터 breaking 가능여부를 조사한다. 첫 번째 batch의 모든 작업이 다른 batch들의 여유 공간(rb_j) 내에 배치가 가능하기 때문에 breaking을 시행하면 <Table 5>와 같이 되고, 여기서 재배치를 시행하면 <Table 6>과 같이 BH 알고리듬의 최종해가 된다.

Table 5. Breaking of first batch of the final solution for Example 3 by SPT heuristic algorithm(before repositioning)

$J_{(i)}$	$J_{(1)}$	$J_{(2)}$	$J_{(5)}$	$J_{(6)}$	M_1	$J_{(7)}$	$J_{(8)}$	M_2	$J_{(3)}$	$J_{(4)}$	$J_{(9)}$	M_3	$J_{(10)}$
$p_{(i)}$	1	1	3	3	(2)	3	4	(2)	2	2	4	(2)	5
$C_{(i)}$	1	2	5	8	(10)	13	17	(20)	21	23	27	(29)	34
$F_{(S)}$	1	3	8	16		29	46		67	90	117		151

Table 6. Final solution of Example 3 by BH algorithm

$J_{(i)}$	$J_{(1)}$	$J_{(2)}$	$J_{(5)}$	$J_{(6)}$	M_1	$J_{(3)}$	$J_{(4)}$	$J_{(9)}$	M_2	$J_{(7)}$	$J_{(8)}$	M_3	$J_{(10)}$
$p_{(i)}$	1	1	3	3	(2)	2	2	4	(2)	3	4	(2)	5
$C_{(i)}$	1	2	5	8	(10)	12	14	18	(20)	23	27	(29)	34
$F_{(S)}$	1	3	8	16		28	42	60		83	110		144

이 스케줄은 *C_{max} = 34, F(S) = 144*이고, $\sum_{j=1}^{3} rb_j = 1$, $\sum_{j=1}^{4} gb_j = 0$으로 SPT 알고리듬의 해에서 1개의 batch를 breaking 하여 주기정비를 1회 줄이고, 기계 유휴시간을 5단위 시간 감소시키면서 총 정비 완료시간을 14단위 시간 감소시킨다. <Table 7>은 Example 3에 대한 최적해로 이것으로 SPT 휴리스틱 알고리듬과 BH 알고리듬의 PED를 계산하면, 각각 12.88과 2.88로 BH 알고리듬의 최종해가 최적해에 더욱 근사함을 알 수 있다.

Table 7. Optimal solution of Example 3

$J_{(i)}$	$J_{(1)}$	$J_{(2)}$	$J_{(5)}$	$J_{(6)}$	M_1	$J_{(3)}$	$J_{(4)}$	$J_{(9)}$	M_2	$J_{(7)}$	$J_{(8)}$	M_3	$J_{(10)}$
$p_{(i)}$	1	1	2	3	(2)	2	3	3	(2)	4	4	(2)	5
$C_{(i)}$	1	2	4	7	(9)	11	14	17	(19)	23	27	(29)	34
$F_{(S)}$	1	3	7	14		25	39	56		79	106		140

5. 실험 및 결과 분석

5.1 실험 설계

Chen(2006a)의 SPT 휴리스틱 알고리듬 및 MIP 모형과의 비교분석을 통하여 BH 알고리듬의 효용성을 평가한다. BH 알고리듬과 SPT 휴리스틱 알고리듬은 VC++로 구현하고, MIP 모형은 ILOG OPL 4.0으로 구현하였으며, 세 가지 모형 모두 Pentium-Ⅳ CPU 3.00GHz, RAM 504MB 컴퓨터를 사용하여 실험하였다. Chen(2006a)은 주기정비에 관련된 입력변수를 다음과 같이 설정하여 6개의 실험 Group을 구성하였다. 정비주기 하한선(*u*)은 $\max\{0.25\sum p_i, \max p_i\}$, $\{0.5\sum p_i, \max p_i\}$ 또는 $\max\{0.75\sum p_i, \max p_i\}$의 세 가지, 정비 소요시간(*w*)은 구간 [1, 15] 또는 [16, 30]의 두 가지에서 uniform 분포를 사용하여 random하게 생성하고, 정비주기 상한선(*v*)은 *u+30*, 각 작업 소요시간은 구간 [1, 15]에서 이산 uniform 분포를 사용하여 총 작업 수만큼 random하게 생성한다. 그러나 이와 같은 실험 데이터 구성은 작업 수가 증가하면, 각 실험 Group에서 정비주기 하한선이 전체 작업 처리시간의 합의 0.25, 0.5 또는 0.75배가 되기 때문에 일정계획에 포함되는 정비횟수가 일정한 수 이하로 고정되는 제약이 있다.

따라서 본 연구의 실험 데이터는 총 작업 수에 비례하여 정비횟수가 증가하는 Chen(2006b)의 실험 데이터를 현실에 보다 부합된 본 연구에 적합하도록 다음과 같이 변형하여 실험하였

다. 입력변수의 종류에는 정비주기 하한선, 정비주기 상한선, 정비소요 시간, 작업 처리시간, 총 작업 수 등이 있으며, 입력변수에 따라 <Table 8>과 같이 6개의 실험 Group으로 설정하였다. 즉, 정비주기 하한선은 10, 15, 20 단위시간의 세 가지, 정비소요시간은 3, 6 단위시간의 두 가지로 하고, 정비주기 상한선은 $u+2w$로 설정하였다. 또한, 작업 처리시간은 [1, 10] 사이의 정수에서 random하게 생성하고, 총 작업 수(N)는 10, 20, 30, 100개의 4가지 경우로 설정하였다. 실험 반복횟수는 실험 Group별로 각 작업 수 당 10회이다. 또한, 실험입력 변수 중 random하게 생성하는 작업 처리시간은 BH 알고리듬에서 산출된 결과를 MIP 모형에 적용하였다. 각 알고리듬의 성능은 MIP 모형에 대한 BH 알고리듬과 SPT 휴리스틱 알고리듬의 산출결과에 따른 평균 PED와 평균 실행시간으로 평가한다. 단, $N = 30$이상인 경우 ILOG OPL 4.0을 통하여 수리모형의 해를 구할 수 없거나 실행시간이 과다하게 소요되어 BH 알고리듬을 기준으로 하여 평균 PED를 측정하였다.

5.2 실험 결과분석

<Table 9>은 각 Group 별 실험결과를 종합한 것이다. 정비주기가 결정변수인 복수의 주기정비를 고려한 단일기계 일정계획에서 총 정비완료시간 최소화에 관한 SPT 휴리스틱 알고리

Table 8. Classification of experimental groups by input variables

Classification	Group 1	Group 2	Group 3	Group 4	Group 5	Group 6
The earliest maintenance start time(u)	10	10	15	15	20	20
The maintenance time of the machine(w)	3	6	3	6	3	6
The latest maintenance completion time(v)	16	22	21	27	26	32
The processing time of jobs(p_i)	randomly generated from a discrete uniform distribution over[1, 10]					

Table 9. Computational results

Classification		SPT heuristic algorithm			BH algorithm			MIP model			
		Mean total completion time	Mean batch numbers	Mean PED	Mean total completion time	Mean batch numbers	Mean PED	Mean total completion time	Mean batch numbers	Mean PED	Comp. time (second)
Group 1	*N = 10*	269.9	5.7	4.9436	261.9	5.4	1.9436	256.9	4.9	0	1.297
	N = 20	1013.7	11.6	6.4811	956.7	10.2	0.4937	952.0	9.1	0	153.569
	N = 30	2349.5	16.5	4.0569	2257.9	14.7	0	-	-	-	-
	N = 100	27249.8	58.6	7.7189	25424.6	50.3	0	-	-	-	-
Group 2	*N = 10*	340.5	5.1	2.9012	334.7	4.8	1.1484	330.9	4.6	0	1.502
	N = 20	1148.0	9.6	3.4327	1122.6	8.8	1.1442	1109.9	8.6	0	60.812
	N = 30	2713.1	14.2	1.7057	2667.6	12.8	0	-	-	-	-
	N = 100	28254.6	44.9	1.9628	27710.7	39.7	0	-	-	-	-
Group 3	*N = 10*	261.0	4.1	3.4483	258.5	3.9	2.4574	252.3	3.5	0	1.184
	N = 20	1055.2	8.0	3.1577	1045.2	7.5	2.1801	1022.9	4.6	0	31.88
	N = 30	2128.6	11.3	1.1836	2103.7	10.0	0	-	-	-	-
	N = 100	24033.6	37.9	1.3494	23713.6	33.1	0	-	-	-	-
Group 4	*N = 10*	246.1	3.2	0.5721	246.1	3.2	0.5721	244.7	3.1	0	1.126
	N = 20	1058.4	6.5	1.9457	1058.4	6.5	1.9457	1038.2	6.1	0	25.037
	N = 30	2285.1	9.2	0	2285.1	9.2	0	-	-	-	-
	N = 100	26582.3	30.1	0.0497	26569.1	29.4	0	-	-	-	-
Group 5	*N = 10*	252.8	3.0	2.1414	252.8	3.0	2.1414	247.5	2.8	0	1.097
	N = 20	991.2	6.0	1.9019	988.5	5.9	1.6243	972.7	5.5	0	8.613
	N = 30	2047.6	8.5	0.1565	2044.4	8.4	0	-	-	-	-
	N = 100	22153.5	27.6	2.4776	21617.9	25.4	0	-	-	-	-
Group 6	*N = 10*	258.1	2.8	0.9781	258.1	2.8	0.9781	255.6	2.7	0	0.793
	N = 20	930.8	5.2	1.5049	926.6	5.1	1.0469	917.0	4.6	0	10.367
	N = 30	2216.9	7.8	0	2216.9	7.8	0	-	-	-	-
	N = 100	24919.8	25.5	0.8874	24700.6	23.5	0	-	-	-	-

Note) 1. The computational times of SPT heuristic algorithm and BH algorithm are nearly 0 second.
2. "-" means that there are no computational results in cases of N = 30, 100.

듬의 성능은 MIP 모형 대비 평균 PED가 5% 미만으로 비교적 효율적이다(Chen, 2006a). 그러나 <Table 8>의 실험 결과처럼 본 연구에서 제시한 BH 알고리듬은 SPT 휴리스틱 알고리듬보다 최대 약 7.7%, 평균 약 1.5%의 개선된 해를 얻을 수 있다.

또한 , 실험 Group별 입력변수 변화에 따른 BH 알고리듬의 개선효과를 SPT 휴리스틱과 비교하여 분석하였다. 이때 평가기준인 평균 PED는 BH 알고리듬에서 얻어진 해를 기준으로 SPT 휴리스틱 알고리듬의 해를 평가한 것으로, 평균 PED가 클수록 BH 알고리듬의 개선효과가 높은 것을 나타낸다. <Figure 4>는 정비 소요시간 3, 정비주기 상한선은 u+2w로 고정하고, 정비주기 하한선을 각각 10, 15, 20으로 변화하는 Group 1, 3, 5의 실험결과를 나타낸 것이다. 각 Group 모두 총 작업수가 증가할수록 BH 알고리듬의 개선효과가 증가하는 경향이 있으며, 정비주기 하한선이 작을수록 개선효과가 뛰어남을 알 수 있다. <Figure 5>는 정비주기 하한선 10, 정비주기 상한선은 u+2w로 고정하고, 정비 소요시간이 각각 3, 6으로 변화하는 Group 1, 2의 실험결과를 나타낸 것이다. 정비 소요시간이 작고 이에 따라 정비주기 상한선이 작게 설정되는 경우 BH 알고리듬의 개선효과가 뛰어남을 알 수 있다.

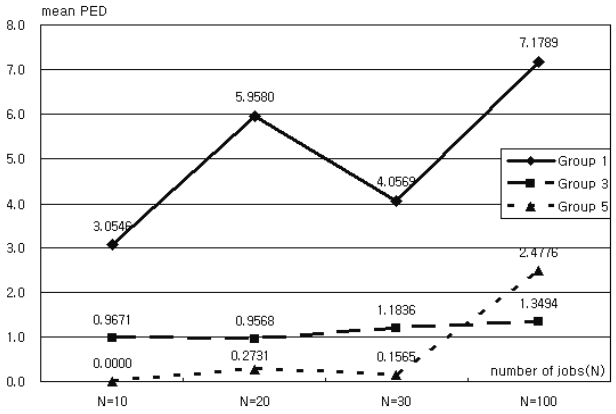

Figure 4. Improvement of mean PED by the variation of the earliest maintenance start time(u)

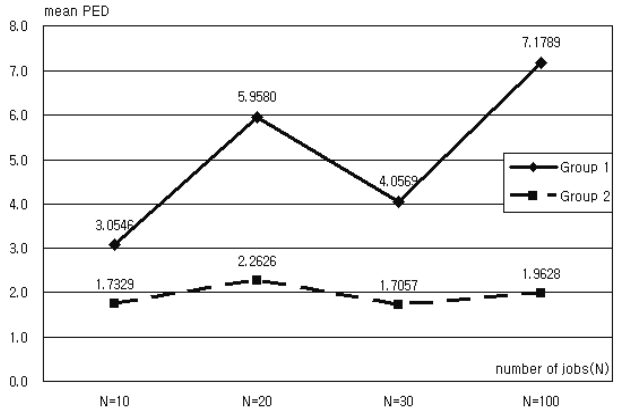

Figure 5. Improvement of mean PED by the variation of the maintenance time of the machine(w)

6. 결론 및 향후 연구방향

오늘날 많은 산업현장에서는 장비 가동률과 정비 효율성 등을 고려하여 계획적인 주기정비가 수행되고 있고, 일반적으로 개별 작업보다는 기계의 주기정비가 중요하기 때문에 효율적인 작업 일정계획은 이러한 주기정비를 고려하여 수립되어야 한다. 본 연구에서는 정비주기가 결정변수인 복수회의 주기정비를 고려한 단일기계 일정계획에서 총 작업완료시간을 감소시키기 위한 BH 알고리듬을 제안하였다. 이를 위하여 먼저 몇 가지 관련 theorem을 정의하고, 이를 바탕으로 breaking rule을 설정하여 알고리듬을 수립하였다. BH 알고리듬의 핵심 내용은 SPT rule을 바탕으로 수립한 초기해에서 특정 batch 내의 모든 작업을 다른 batch들에 배치시켜서 일정계획에 포함되는 정비 횟수 및 기계 유휴시간을 감소함으로써 총 작업완료시간을 줄이는 것이다.

BH 알고리듬을 기존 알고리듬 및 수리모형과 비교하여 실험한 결과 기존 알고리듬에 비하여 해에 대한 개선효과가 우수하며, 수리모형을 적용할 수 없는 대상 작업의 수가 큰 문제에도 쉽게 적용할 수 있음을 보였다. 하지만 BH 알고리듬은 일정계획 대상기간에 포함되는 batch의 수가 적은 경우에는 특정 batch 내의 모든 작업을 다른 batch들에 배치시키는 것이 불가능하여 기존 알고리듬에서 개선된 해를 구하지 못하는 한계가 있다.

향후 다양한 입력변수를 바탕으로 본 알고리듬의 활용 가능 분야를 폭넓게 확인하고, makespan이나 최대 지연의 최소화 등의 다른 목적함수에 대한 연구도 이루어져야 할 것이다. 더 나아가 단일기계 모형에서 얻어진 결과를 바탕으로 병렬기계 또는 흐름공정에 적용하는 연구가 필요한 실정이다.

참고문헌

Akturk, M. S., Ghosh, J. B., and Gunes, E. D. (2003), Scheduling with Tool Changes to Minimize Total Completion Time : A Study of Heuristics and Their Performance, *Naval Research Logistics*, **50**, 15-30.

Akturk, M. S., Ghosh, J. B., and Gunes, E. D. (2004), Scheduling with Tool Changes to Minimize Total Completion Time : Basic Results and SPT Performance, *European Journal of Operational Research*, **157**, 784-790.

Chen, J. S. (2006a), Single-machine Scheduling with Flexible and Periodic Maintenance, *Journal of Operational Research Society*, **57**, 703-710.

Chen, W. J. (2006b), Minimizing Total Flow Time in the Single-machine Scheduling Problem with Periodic Maintenance, *Journal of Operational Research Society*, **57**, 410-415.

Graves, G. H. and Lee, C. Y. (1999), Scheduling Maintenance and Semi-resumable Jobs on a Single Machine, *Naval Research Logistics*, **46**, 845-863.

Ji, Min, He, Yong, and Cheng, T. C. E. (2007), Single-machine Scheduling with Periodic Maintenance to Minimize Makespan, *Compu-*

ters and Operations Research, **34**, 1764-1770.

Kacem, I., Chu, C., and Souissi, A. (2008), Single-machine Scheduling with an Availability Constraint to Minimize the Weighted Sum of the Completion Times, *Computers and Operations Research*, **35**, 827-844.

Lee, C. Y. (1996), Machine Scheduling with Availability Constraint, *Journal of Global Optimization*, **9**, 395-416.

Lee, C. Y., Lei, L. and Pinedo, M. (1997), Current Trend in Deterministic Scheduling, *Annals of Operations Research*, **70**, 1-41.

Lee, C. Y. and Leon, L. (2001), Machine Scheduling with a Rate-modifying Activity, *European Journal of Operational Research*, **128**, 119-128.

Lee, C. Y. and Liman, S. D. (1992), Single-Machine Flow-time Scheduling with Scheduled Maintenance, *Acta Inform*, **29**, 375-382.

Lee, C. Y. and Lin, C. S. (2001), Single-Machine Scheduling with Maintenance and Repair Rate-modifying Activity, *European Journal of Operational Research* , **135**, 493-513.

Leon, C. J. and Wu, S. D (1992), On Scheduling with Ready-times, Due-dates and Vacations, *Naval Research Logistics*, **39**, 53-65.

Leung, Joseph Y-T. (2004), *Handbook of Scheduling*, Chapman and Hall/CRC.

Liao, C. J. and Chen, W. J (2003), Single-machine Scheduling with Periodic Maintenance and Nonresumable Jobs, *Computers and Operations Research*, **30**, 1335-1347.

Qi, X., Chen, T. and Tu, F. (1999), Scheduling the Maintenance on a Single Machine, *Journal of Operational Research Societ*y, **50**, 1071-1078.

Sadfi, Cherif, Penz, Bernard, Rapine, Christophe, Blazewicz, Jacek and Formanowicz, Piotr. (2005), An Improved Approximation Algorithm for the Single Machine Total Completion Time Scheduling Problem with Availability Constraints, *European Journal of Operational Research*, **161**, 3-10.

Sanlaville, E. and Schimdt, G. (1998), Machine Scheduling with Availability Constraints, *Acta Inform*, **35**, 795-811.

Schmidt, Günter (2000), Scheduling with Limited Machine Availability, *European Journal of Operational Research*, **121**, 1-15.

Yang, D. L., Hung, C. L., Hsu, C. J. and Chern, M. S. (2002), Minimizing the Makespan in a Single Machine Scheduling Problem with a Flexible Maintenance, *Journal of the Chinese Institute of Industrial Engineers*, **19**, 63-66.