유전알고리즘과 시뮬레이션을 이용한 유연생산시스템에서의 최적 버퍼 할당에 관한 연구

The Study for Optimal Buffer Allocation in FMS Using Genetic Algorithm and Simulation

이용균*, 김경섭**
Yong Kyun Lee, Kyung Sup Kim

Abstract

This study presents a heuristic algorithm for buffer allocation in FMS(Flexible Manufacturing System). The buffers, which are finite resources in FMS, are responsible for improvement of an overall system utilization. But, until now, the study for buffer allocation in FMS are rarely conducted because of the complexity in FMS. Most studies for buffer allocation had been addressed to the simple production line system. The presented algorithm uses a simulation for the description of system complexity and uses a genetic algorithm for finding better buffer allocation. Lastly, we compare performance of the presented algorithm with that of a simple heuristic, and analyze the experiment results.

* 연세대학교 컴퓨터과학·산업시스템공학과
** 연세대학교 기계전자공학부 부교수

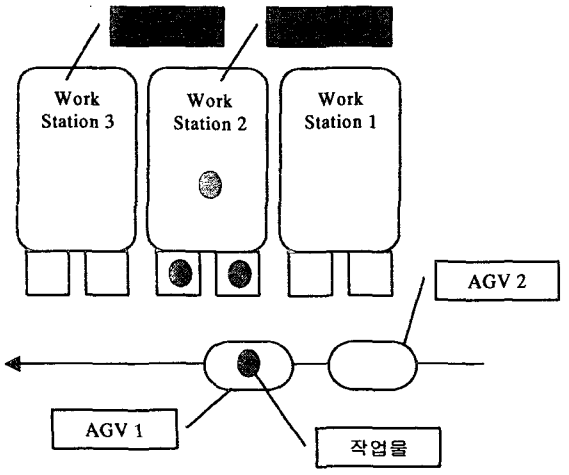
1. 서론

현대의 생산 형태는 소비자 기호의 다양화와 기업 간 경쟁의 심화에 의하여 소품종 다량생산에서 다품종 소량생산으로 변화하고 있다. 또한, 생산 공정 내에서 작업물의 실제 가공시간은 전체 사이클 타임의 5%를 넘지 않고 대부분의 시간을 작업장내 이동이나 다음 공정을 대기하는데 사용하는 것으로 알려져 있다. 이에 따라 생산 환경의 변화에 쉽게 대응가능하고 대기시간과 이동시간을 감소시킬 수 있는 유연 생산시스템(FMS : Flexible Manufacturing System)이 제안되었고 그 효용이 많은 연구를 통해 증명되어 왔다. 유연 생산 시스템은 자동 창고 시스템(AS/RS), 무인 운반차(AGV), 자동화 생산 기계(NC Machine)등으로 이루어지며 공장의 자동화와 유연성에 그 목적을 두고 있다. 유연 생산 시스템에서 버퍼는 시스템의 최종 Output을 증가시키고 생산라인의 안정성을 보완하는 역할을 한다. 특정한 시스템의 경우 버퍼가 없어도 시스템의 운영에 문제가 발생하지 않을 수 있지만 대부분의 시스템에서는 버퍼가 시스템의 성능을 크게 향상시킨다. 특히, 시스템 내에 병목현상을 보이는 Workstation이 존재한다면 버퍼가 시스템의 성능에 더 많은 영향을 미치게 된다. 또한, 버퍼는 생산라인 중 한 Workstation의 고장이 발생할 시에 전체 라인이 멈추게 되는 것을 방지하는 안정성 보완을 위해서도 필요하다. 기존의 FMS에 대한 연구는 무인 운반차의 발주 방식, 작업 스케줄링, 설비 계획 등에 대하여 주로 이루어져 왔다. 그러나, FMS 내에서 공간적인 제약과 비용적인 측면을 고려하면 버퍼도 하나의 제한된 자원이고 이를 어떻게 할당하느냐에 따라 시스템의 Output이 크게 달라질 수 있으므로 이를 최적으로 할당하여 시스템의 Output을 높이는 것은 가치가 있는 것이라 할 수 있다. 이와는 반대로, 이미 존재하는 시스템에 대해서는 시스템 설계 시에 생산 능력 계획 등에 의하여 원하는 Output Level이 존재하므로 이를 만족시키는 최소의 총 버퍼 수를 가지는 Buffer Profile을 찾아내는 것도 의미가 있다. 또 한, 기존의 연구와는 달리 본 연구에서는 여러 가지 타입의 작업물을 고려하므로 각 타입의 Value가 다를 수 있다는 것은 충분히 타당하며 전체의 Value를 최대화 할 수 있는 Buffer Profile을 찾아내는 것도 하나의 의미를 가진다.

기존의 버퍼에 관한 연구는 주로 FMS가 아닌 간단한 라인 생산 시스템에 대하여 이루어지거나, 복잡한 시스템을 대상으로 한 연구들은 버퍼 할당의 특성인 Combinatorial 특성을 반영하지 못하였다. 본 연구에서는 시스템의 복잡성을 모델링하기 위하여 시뮬레이션을 사용하고 버퍼 할당 문제의 특성인 Combinatorial 특성을 반영하기 위하여 수정된 유전 알고리즘을 사용한다. 유전 알고리즘은 현재의 해 집합을 더 개선 할 수 있는 방향으로 이끌어 나가는 메타 휴리스틱의 한 기법이고 이 유전 알고리즘에서의 해의 평가를 위해 상용 시뮬레이터인 Arena를 이용한 시뮬레이션을 수행한다. Arena가 제공하는 Add-In 형식인 VBA블록을 사용하여 시뮬레이션의 정해진 반복 수행마다 자동적으로 Buffer Profile을 바꾸어 시뮬레이션을 수행하게 하고, 한 세대의 시뮬레이션이 끝날 때에는 자동적으로 유전 알고리즘을 수행하여 새로운 해 집합을 생성하게 된다. 위에서 언급한 각각의 문제에 따라 알고리즘이 약간씩 수정되어야 하며 각각의 수행결과에 대해 기존의 알고리즘과 비교, 평가를 수행한다.

2. 버퍼개념 및 연구배경

일반적으로 생산 공정은 기계의 고장이나 각기 다른 생산시간, 작업 할당의 잘못, 물류 시스템의 제약 등으로 인해 공정의 효율이 낮아지게 되고 각 기계의 가동률도 떨어지게 된다. 이를 보완하기 위하여 주로 사용되는 것이 버퍼이다. 생산 공정에서 버퍼의 부족으로 일어나는 현상은 Blocking, Deadlock, Starving이 있다. 이들을 <그림 1>에 그림으로 표현했다. Blocking은 버퍼의 부족 등으로 인해 작업물이 선행 작업물의



<그림 1> Blocking, Deadlock and Starving

작업시간이 끝날 때까지 대기하는 것을 말한다. 그림의 예로 설명하면 1번 무인 운반차가 2번 기계에 작업물을 운반하려 하는데 2번 기계가 작업중이고 입력버퍼에 대기하고 있는 작업물이 있다면 1번 무인 운반차는 기계의 작업이 끝나서 작업물이 출력버퍼로 이동할 때까지 기다려야 한다. 이러한 상황을 Blocking이라 하며 위에에서 각 버퍼의 용량은 1이라 가정했고 앞으로의 예에서도 1이라 가정한다. Deadlock은 Blocking보다 더 심각한 상황을 발생시키는 데 이는 사람이 개입하지 않으면 결국 전체 시스템이 정지되어 버리는 현상이다. Blocking의 예에서 만약 출력 버퍼에도 하나의 작업물이 다음 기계로 옮겨가기 위해 무인 운반차를 대기하고 있고 그 작업물에 할당 된 무인 운반차가 1번 무인 운반차 뒤에 오게 된다면 기계에서 작업이 끝나더라도 작업물이 빠져나가지 못하게 되고 두 대의 무인 운반차도 계속 기계 2번 앞에서 머무르게 된다. 이러한 상황이 Deadlock이며 이것은 사람이 개입하지 않거나 미리 방지를 해주지 않으면 결국 시스템 전체가 멈추게 된다. 마지막으로 Starving은 Blocking이나 Deadlock이 일어난 후속 기계가 작업이 없이 기다리는 현상을 가리킨다. 이러한 현상이 일어나지 않거나 일어나는 시간과 회수를 줄이는 것이 시스템 전체의 성능을 증가시키는데 많은 역할을 한다.

기존의 버퍼 할당 관련 연구를 살펴보면 크게 수학적 접근방법을 사용한 연구, 휴리스틱을 사용한 연구 그리고 버퍼의 최적 할당 특성을 분석한 연구 이렇게 세 가지로 나누어 볼 수 있다. 우선 수학적 접근방법을 사용한 연구들은 주로 마코프 체인, 동적 계획법, 변형된 선형 계획법 등을 사용해서 문제에 접근하였다. 그러나 시스템이 복잡해지거나 커지면 상태 공간 폭발 등의 문제가 발생하기 때문에 대부분 간단한 생산라인에 대해서만 연구가 수행되었다. 이들 중 Jafri와 Shanthikumar는 동적 계획법을 사용하여 문제에 접근했다. 그들은 생산라인을 각각 하나의 기계를 가지는 Stage로 구분하고 각 Stage의 중간에 버퍼를 두는 시스템으로 구성하였다. 그들은 동적 계획법으로 문제를 Formulation하기 위하여 시스템의 Throughput을 구하는 휴리스틱을 제안하였고 이 휴리스틱에서 분할 알고리즘을 사용하여 Throughput의 근사치를 계산하였다. 그들은 간단한 생산라인에서 Greedy Search로 구해낸 최적 값과 자신들이 제시한 알고리즘에 의한 값과의 비교를 통해 알고리즘의 정확성을 검증했다. 이 외에 Yamashita와 Altok등도 동적 계획법을 이용했고 Ho등은 Gradient Technique를 사용하여 최적 버퍼 할당 문제에 접근했다. 수학적 접근 방법의 단점을 보완하려는 노력으로 휴리스틱 접근방법이 최근에 많이 제시되고 있다. Lutz 등은 Tabu Search와 시뮬레이션을 이용하여 문제에 접근하였고 Knowledge Base를 이용한 Vorous의 방법과 Dimension Reduction과 Beam Search를 이용한 Park의 연구 등 많은 방법이 시도되고 있다. 마지막으로 버퍼의 최적 할당 특성을 분석한 연구들은 Inverted Bowl 현상을 분석하고 작업 시간 분포의 변화나 기계의 고장 그리고 이어지는 작업간의 Coefficient Of Variation의 영향에 Inverted Bowl 현상이 어떤 식으로 변화하는가에 대한 관찰과 분석을 수행한 연구가 대부분이다.

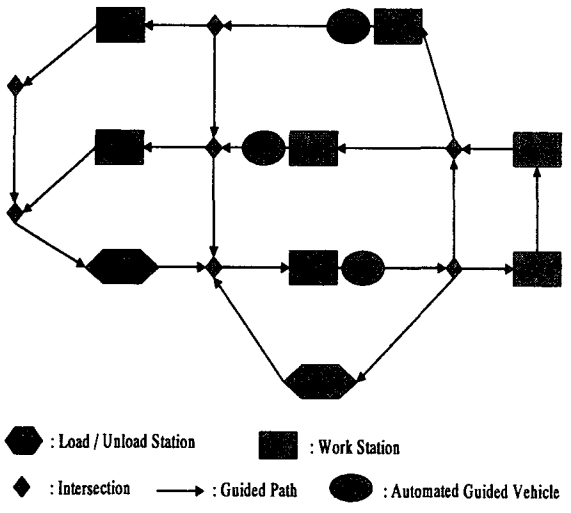
그러나 위의 기존 연구들은 버퍼 할당 문제가 갖고 있는 Combinatorial 특성만을 반영하여 복

잡한 모델에 적용하기 힘들거나, 모델의 복잡성은 반영을 했으나 특정한 몇 가지 해 집합만을 고려하여 Combinatorial 특성을 반영하지 못한 단점이 있다. 본 연구에서는 시뮬레이션을 사용하여 복잡한 모델의 문제도 다룰 수 있고 수정된 유전 알고리즘을 사용하여 버퍼 할당 문제의 특성인 Combinatorial특성도 다룰 수 있는 알고리즘을 제시한다.

3. 알고리즘

3.1 System Configuration

<그림 2>에 제시된 것처럼 시스템은 7개의 Workstation과 3대의 무인 운반차, 그리고 Load/Unload Station으로 이루어져 있는 유연생산 시스템이다. 각 무인 운반차는 Guided Path를 따라 움직이며 모든 Guided Path는 단방향 Path이고 각 Work-station의 앞뒤에는 입/출력 버퍼가 있다. 시스템에 투입되는 각 제품의 양의 부족 때문에 생기는 Throughput의 영향을 제거하기 위하여 투입 기간을 짧게 두고 Load Station의 용량을 무한대로 두었고 또한, 각 제품이 시스템에서 빠져나갈 때의 지연에 의한 영향



<그림 2> 대상 생산시스템

을 제거하기 위하여 Unload Station의 용량을 무한대로 설정하였다. 제품 타입에 따른 생산순서는 <표 1>에 제시되어 있으며 한 제품의 같은 Workstation에서의 작업도 생산 순서 상 다른 작업이라면 작업시간이 틀리다.

<표 1> 제품 정보

제품타입	발생 간격	생산 순서
A	Expo(600)	1->2->1->2->3->4
B	Expo(400)	4->5->6->7->5->7
C	Expo(500)	1->2->4->2->4->6
D	Expo(600)	4->5->3->6->7->5

기존의 연구들이 하나의 제품 타입만을 고려하는 반면 이 시스템에서 생산되는 제품의 타입은 4가지이며 각 타입은 다른 생산 순서를 가진다. 여러 가지 제품 타입과 생산순서를 고려함으로써 최적 버퍼 할당문제는 더 복잡해지고 기존의 생산라인 시스템에서 제시된 알고리즘들과 다른 알고리즘을 필요로 하게 된다.

3.2 Problem Definition

본 연구에서는 모두 세 가지 문제에 대하여 알고리즘을 제시한다. 생산 시스템에서 총 버퍼의 크기는 버퍼의 추가에 따라 부담되는 비용측면이나 차지하는 면적을 고려할 때 하나의 유한한 자원이라 볼 수 있다. 따라서 총 버퍼가 제한되어 있는 상황은 타당하며 이미 존재하는 시스템에 대해서 제한이 있는 버퍼를 최적으로 할당하여 시스템 전체의 Throughput을 최대화하는 문제는 타당하다. 이와 같이, 본 연구에서 고려하는 첫 번째 문제는 기존의 연구와 같이 시스템의 총 버퍼 수에 제한이 있을 때 Throughput의 최대화를 목적함수로 하는 문제이다. 이 문제의 제약으로는 시스템 전체에 할당 가능한 총 버퍼의 제한과 각 Workstation의 입/출력 버퍼에 할당 가능한 버퍼의 제한이 있다. 시스템에서 Throughput에 영향을 미치는 요소들 중, 각 Workstation의 Processing Time과 각 타입 작업

물의 투입 간격 그리고 AGV Dispatching Rule이 결정된 것이라면 그 시스템의 Throughput은 각 Workstation의 입/출력 버퍼의 크기에 따라 결정된다. 각 버퍼의 크기를 하나의 벡터로 표현한 것이 B이며 B = (b1, b2,....., bN)으로 표시되고 시스템의 Throughput은 T(B)라고 표시할 수 있다. 여기서 함수 T는 버퍼 벡터에 따른 시스템의 Throughput을 표현하는 함수이다. 위 첫 번째 문제를 수식적으로 표현하면 다음과 같다.

$$Max\ T(B)$$
$$s.t\ \sum_{i=1}^{N} b_i \le Total\ Buffer\ Capacity$$
$$b_i \le Each\ Buffer\ Capacity,\ i=1..N$$
$$B=(b_1, b_2, \ldots\ldots, b_N)$$
$$N: Total\ Buffer\ Location$$

두 번째로, 시스템이 새로 설계될 시에는 수요 예측치, 혹은 그와 동일한 Desired Through-put을 고려해서 생산 능력 계획 등에 의하여 설계된다. 따라서 위에서 제시한 시스템 Through-put을 최대화시키는 것보다는 Desired Output을 만족시키는 최소의 Buffer수와 Profile을 찾아내는 것이 더 의미 있는 문제가 된다. 이 문제에서는 목적함수가 각 Workstation에 할당된 총 버퍼 수의 최소화가 될 것이다. 그리고, 제약으로는 시스템의 Throughput이 Desired Throughput을 만족하는 여부와 각 Work-station의 입, 출력 버퍼에 할당 가능한 버퍼의 제한이 있고 이를 수식적으로 표현하면 다음과 같다.

$$Min\ \sum_{i=1}^{N} b_i$$
$$s.t\ T(B) \ge Desired\ Throughput$$
$$b_i \le Each\ Buffer\ Capacity,\ i=1..N$$
$$B=(b_1, b_2, \ldots\ldots, b_N)$$
$$N: Total\ Buffer\ Location$$

마지막으로, 본 연구에서는 여러 타입의 작업물을 고려하기 때문에 각 타입의 Value가 다를 수 있다는 가정은 당연하다. 따라서 목적함수가 위에서 고려한 첫 번째 목적인 Throughput의 최대화가 아닌 전체 Value의 최대화가 된다. 여기서 Tpart type 함수는 버퍼 벡터 B에 대하여 각 타입의 Throughput을 나타내는 함수이다. 이 문제에서 제약은 각 버퍼의 설치에 따른 비용이 총 가용한 비용보다 작은가의 여부가 될 수도 있으나 이는 총 버퍼 수에 따른 제약과 의미상 같기 때문에 첫 번째 목적의 제약과 같은 제약식으로 설정하였다.

$$Max \sum_{PartType=1}^{4} (T_{part\ type}(B) \times Value_{part\ type})$$
$$s.t\ \sum_{i=1}^{N} b_i \le Total\ Buffer\ Capacity$$
$$b_i \le Each\ Buffer\ Capacity\ \ i=1..N$$
$$B = (b_1, b_2, \ldots\ldots, b_N)$$
$$N : Total\ Buffer\ Location$$

위의 세 가지 문제들은 모두 버퍼 할당의 문제이므로 Combinatorial 특성을 지니며 시스템이 유연 생산 시스템과 같이 복잡한 시스템이면 기존의 알고리즘으로는 해결할 수 없다. 따라서 본 연구에서는 기본적으로 시뮬레이션을 사용하여 시스템을 모델링하여 Throughput을 측정하고 수정된 유전 알고리즘을 사용하여 각 문제에서 최적 버퍼 할당을 찾는 알고리즘을 제안한다. 세 가지 문제에 따라 약간씩 알고리즘의 변형이 필요하며 본 연구에서는 각각의 변형을 제시하고 실험을 수행한다.

3.3 Algorithm Step

본 연구에서는 해의 형태를 14개의 각 버퍼를 의미하는 유전자를 갖는 하나의 개체로 구성한다. 예를 들어 하나의 해가 (0, 1, 2, 1, 1, 0, 0, 2, 2, 1, 0, 0, 2, 1)의 형태를 갖는 다면 1번 Workstation의 입/출력 버퍼의 수가 0과 1이 되고 2번 Workstation의 입/출력 버퍼의 수가 2와 1이 된다는 것을 의미한다. 따라서 하나의 해는 총 7개 Workstation, 14개 입/출력 버퍼의 크기

를 나타낸다. 지금부터 알고리즘의 각 단계를 설명하도록 하겠다.

Step 1. Buffer Profile의 초기 모집단을 생성한다.

유전 알고리즘은 해의 집합을 운영하여 해를 개선시키므로 항상 모집단을 구성해야 하고 모집단의 크기와 초기 모집단 생성 알고리즘은 알고리즘의 성능에 많은 영향을 미친다. 본 연구에서는 모집단의 크기를 10으로 하고 초기 모집단은 임의의 수를 생성하는 방법으로 구성하였다. 초기 모집단 생성에 다른 휴리스틱을 사용하면 해가 빨리 수렴할 수 있지만 유전 알고리즘의 기존 연구에서 국부 해에 빠질 가능성도 커짐을 보였다. 또 한 초기 모집단을 생성하기 위해 휴리스틱을 수행하는 시간도 무시할 수 없으므로 본 연구에서는 임의 생성방법을 사용하였다. 초기 모집단을 구성할 때 임의 생성 방법을 사용하기 때문에 초기해에서 비가능해가 나올 수 있다는 것도 고려해야 한다. 따라서 초기해에서는 각 Workstation 입/출력 버퍼 용량의 합이 시스템 전체의 버퍼 제한 용량을 넘는 비가능해가 나오지 않도록 임의 생성 방법을 적절히 조정하는 방법을 취하였다.

Step 2. 시뮬레이션을 수행한다.

본 연구에서는 시뮬레이션의 Random특성을 고려하여 하나의 Buffer Profile당 30번의 반복시행 결과 값의 평균을 시스템의 Throughput으로 사용한다.

Step 2.1 VBA를 이용하여 Arena Model의 각 버퍼 용량을 설정한다.

VBA란 Arena와 Visual Basic을 연결해주는 Add-In으로 Visual Basic에서 임의 생성 방법으로 만들어진 초기 모집단의 Buffer Profile에 맞게 Arena Model의 버퍼 용량을 설정한다.

Step 2.2 하나의 버퍼 프로파일, 즉 모집단에서 하나의 해의 시뮬레이션을 주어진 반복회수만큼 반복하여 시스템의 Throughput을 결정하고 Step 2.3으로 이동한다.

이 단계에서 만약 Buffer Profile의 총 합이 시스템 전체의 제한된 Buffer 크기보다 크다면 비 가능해이기 때문에 시스템의 Throughput을 반으로 줄여준다. 이것은 Step 3에서 비 가능해가 다음 세대로 유전되지 않게 하기 위함이다.

Step 2.3 한 해집합 내의 모든 Buffer Profile에 대한 시뮬레이션이 끝나면 Step3으로 이동하고 아니라면 다음 Buffer Profile로 Arena Model을 설정하고 Step 2.2로 이동한다.

Step 3. 시뮬레이션의 결과 값들을 사용하여 새로운 해집합을 생성한다.

Step 3.1 다음 세대에 유전될 개체를 선별한다.

선별의 방법에는 여러 가지 방법이 있지만 본 연구에서는 토너먼트 선별방식을 사용한다. 여러 해에서의 시뮬레이션을 통해 나온 Throughput이 24시간을 운영해서 나온 결과이므로 크게 차이가 나지 않을 수도 있다. 따라서 각 해의 Through-put의 비교로 선별을 하는 토너먼트 선별방식이

<표 2> 각 Workstation의 순위 인덱스

	WS 1	WS 2	WS 3	WS 4	WS 5	WS 6	WS 7
포함된 회수	3	4	2	5	4	3	3
평균 Process Time	(133,173)	(152,181)	(165,192)	(126,160)	(82,100)	(106,133)	(130,150)
순위 인덱스	5	6	7	3	1	2	4

확률을 주는 등의 다른 방식보다는 본 연구에 더 합당하다. 유전 알고리즘에서는 선별압력이 너무 강하면 해가 조기 수렴하여 국부 해에 빠질 경우가 있고 너무 약하면 임의 탐색법 같은 알고리즘이 되어 버리므로 선별압력을 적절히 조정하는 것이 중요하다. 본 연구에서는 10개의 Buffer Profile 중 5개를 먼저 선정하고 그 다섯 Buffer Profile중 최대의 Throughput을 보이는 것을 다음 세대로 유전되는 집합에 포함시킨다. 위의 과정을 모집단의 총 수인 10개가 선정될 때까지 반복하여 우선 10개의 유전되는 개체의 집합을 만들어 낸다.

Step 3.2 교차연산을 수행한다.

유전 알고리즘에서 교차연산은 이전 세대의 좋은 해의 형태를 이어가기 위한 연산이다. 본 연구에서는 여러 가지 교차연산 방법 중 일점교차 방법을 사용한다. 해 집합 내의 모든 Buffer Profile에 대하여 0부터 1사이의 임의의 수를 생성한 후 만약 임의의 수가 설정한 교차확률보다 작다면 교차 집합에 추가한다. 교차연산은 특성상 교차되어야 하는 개체의 수가 항상 짝수여야 하기 때문에 만약 홀수의 개체가 교차 집합에 추가되었다면 추가되지 않은 것 중 하나의 개체를 선택하여 교차집합에 추가한다. 일점 교차의 예를 본 연구에서 쓰이는 해의 형태를 사용하여 예를 들면 다음과 같다.

Profile1 : (0,0,1,1,2,2,0 | 0,1,0,1,2,3)
Profile2 : (1,1,0,0,1,1,1 | 1,2,2,0,0,0)

이렇게 두 개의 교차집합에 들어간 Buffer Profile이 있고 교차점을 위의 Buffer Profile에서 | 로 표시 한 대로 7번째 Gene의 뒤라고 하면 교차되어 나오는 두 개의 Buffer Profile은 다음과 같다.

Profile1′ : (0,0,1,1,2,2,0 | 1,2,2,0,0,0)
Profile2′ : (1,1,0,0,1,1,1 | 0,1,0,1,2,3)

Step 3.3 돌연변이연산을 수행한다.

생태계에서 돌연변이는 대개 나쁜 영향을 미치나 때로는 유익한 돌연변이가 일어나기도 한다. 유전 알고리즘에서 돌연변이는 해의 탐색중 국부해에 빠지는 것을 방지하는 역할을 한다. 돌연변이 율은 각 유전자가 돌연변이 될 확률이며 이 확률이 너무 커지면 임의 탐색법과 같아질 위험이 많아지고 너무 작으면 국부해에 빠질 위험이 많아지므로 세심한 조정이 필요하다. 본 연구에서는 최적해의 개선이 있을 때에는 돌연변이 율을 줄여주고 개선이 없을 시에는 일정한 확률까지 단계적으로 높여주는 방법을 사용하고 있다. 이 단계에서도 두 개의 0부터 1까지의 임의의 수와 1부터 3까지의 임의의 수 모두 세 개를 생성한다. 첫 번째 임의의 수는 해당되는 유전자가 교차 연산을 수행하는지의 여부를 교차확률과 비교하여 결정하고 두 번째 임의의 수는 교차 연

<표 3> 실험 결과

균등할당 결과			제안 알고리즘 결과		
총 버퍼양	Buffer Profile	Throughput	Buffer Profile	Throughput	차이 (%)
12	1,1,1,1,0,0,1,1,1,1,1,1,1,1	290	1,1,2,1,0,1,1,1,0,1,0,1,1,1	404	39.3
14	1,1,1,1,1,1,1,1,1,1,1,1,1,1	410	3,1,1,1,0,1,2,1,0,1,0,1,1,1	424	3.41
16	1,1,1,1,1,1,1,1,2,2,1,1,1,1	416	2,1,2,1,1,1,2,1,0,1,0,1,2,1	437	5.04
20	1,1,1,1,1,1,2,2,2,2,2,2,1,1	427	2,1,2,1,1,1,3,1,1,2,1,1,2,1	454	6.32
25	2,2,2,1,1,1,2,2,2,2,2,2,2,2	445	3,1,3,1,1,1,5,2,1,1,1,1,3,1	470	5.61
30	2,2,2,2,2,2,2,2,3,3,2,2,2,2	460	5,1,3,1,1,1,5,2,1,1,1,3,4,1	475	3.26

산이 수행될 때 각 유전자에 버퍼를 추가할 것인가 감소시킬 것인가를 결정한다. 마지막 임의의 수는 유전자에 추가 혹은 제거하는 버퍼의 크기를 설정한다. 본 연구에서 돌연변이는 비 가능해를 생성할 확률이 높아 좋다고 볼 수는 없지만 이미 밝힌 바와 같이 돌연변이 연산으로 국부해를 빠져나갈 수 있고 또 수정된 돌연변이 연산은 해의 개선 속도를 높일 수 있기 때문에 돌연변이 연산을 사용한다. 여기서 수정된 돌연변이 연산이란 모델에서 각 버퍼의 효율을 측정해 효율이 크면 버퍼를 추가하는 방향으로 돌연변이 연산을 하고 작으면 제거하는 방향으로 돌연변이 연산을 수행함을 의미한다. 이렇게 시스템 모델에서 의미 있는 정보들을 돌연변이 연산에 추가함으로써 유전 알고리즘의 성능 개선도 이룰 수 있다.

Step 4. 종료 조건을 검사한다.

유전 알고리즘에서 쓰이는 종료조건의 종류에는 최대의 세대수를 사용하거나 원하는 목적함수 값을 달성하는 등 여러 가지가 있으나 본 연구에서는 최적해의 개선이 40세대 동안 일어나지 않으면 종료하도록 종료 조건을 설정하였다.

3.4 알고리즘 변형

본 연구에서 고려하는 다른 두 문제, 즉 Desired Throughput을 만족하는 최소의 버퍼 용량과 Buffer Profile을 찾는 문제와, 시스템에서 생산되는 각 타입의 Value가 다를 때 전체 Value의 최대화에 대한 Buffer Profile을 찾는 문제에 대해서는 이제까지 설명한 기본 알고리즘에 대한 변형이 있어야 한다.

우선 Desired Throughput을 만족하는 최소의 버퍼 용량과 Buffer Profile을 찾는 문제에 대하여 먼저 설명하도록 한다. 위 문제에 대한 알고리즘으로 만들기 위해서는 우선 Step 2.2에서 비가능해를 구분하는 부분부터 변경해야 한다. 즉 비가능해를 현 Buffer Profile에 의해 나온 시뮬레이션 결과 값이 Desired Throughput보다 큰가의 여부에 따라 비가능해를 구분하여야 한다. 그리고 목적함수가 변경되므로 Step 3.1의 선별 부분에서도 현 Buffer Profile의 각 유전자 값의 합이 최소인 개체를 선별하도록 변경하여야 한다.

두 번째로, 전체 Value의 최대화에 대한 Buffer Profile을 찾는 문제에 대해서는 먼저 목적함수의 변경에 따른 알고리즘의 변경이 필요하다. 따라서 Step 3.1의 선별 부분에서 각 타입의 Value와 각 타입의 Throughput을 곱한 것을 전체 타입에 대해 합을 한 값을 목적함수 값으로 써야 한다. 그리고 시뮬레이션 모델에서도 전체의 Throughput을 계산하여 유전 알고리즘에 전달하는 것이 아니라 각 타입의 Throughput을 따로 전달하도록 변경하여야 한다.

본 연구에서는 위의 단계들을 Arena와 VBA를 이용하여 지정된 반복시행이 끝날 때마다 자동적으로 수정된 유전 알고리즘이 수행되게 설계하였다.

4. 실험 결과 및 분석

본 연구에서는 Throughput 최대화 문제에 대한 실험을 위해서 시스템의 총 버퍼 용량을 12에서 30으로 늘여가면서 실험을 수행하였고 각 Workstation의 입, 출력 버퍼의 용량은 5로 제한하였다. 총 시뮬레이션 시간은 초를 기본단위로 하여 24시간을 실험하였다. 기존의 유연생산 시스템을 위한 버퍼 할당 알고리즘이 제시되지 않았기 때문에, 제안한 알고리즘의 성능검증을 위하여 Workload와 Processing Time을 고려한 균등할당 방법과 비교를 수행하였다. 시스템에서 버퍼는 Workload가 많고 Processing Time이 짧은 곳에 많은 할당을 하는 것이 기본적으로는 좋다. Work-load가 많고 Processing Time이 짧은 곳이 병목 현상을 일으킬 우려가 있는 곳이기 때문이다. 따라서, 균등할당에 Workload와 Processing Time을 고려한 알고리즘을 비교대상 알고리즘으로 선정하였다. 우선 각각의 타입에 대하여 1번 Workstation부터 7번 Workstation까지 생산 순서 내의 포함된 회수를

계산한다. 그리고, 그에 따른 각각의 프로세싱타임을 더하여 회수로 나누어줘서 순위 인덱스를 설정한다. 회수와 그에 따른 순위 인덱스는 <표 2>에 제시되어 있다.

즉, 가장 짧은 Processing Time을 가지는 5번 Workstation에 최고의 인덱스를 주고 가장 긴 Processing Time을 가지는 3번 Workstation에 최저의 인덱스를 두어서 차등화 된 균등할당을 수행하였다.

<표 3>에 그 결과가 제시되어 있다. 실험결과를 보면 제안된 알고리즘의 성능이 전체적으로 약 10.5%의 향상이 되었음을 알 수 있다. 그리고 성능의 차이가 가장 큰 총 버퍼 용량이 12일 때를 제외하면 전체적으로 약 4.7%의 성능향상이 되었음을 보인다

특이한 점은 총 버퍼 용량이 12일 때 균등할당 알고리즘과 제안된 알고리즘의 차이가 다른 때 보다 훨씬 큰 것을 볼 수 있다는 것이다.

이는 전체 시스템의 성능이 총 버퍼 용량이 작을 때에 버퍼의 추가나 감소에 훨씬 더 민감하다는 기존의 연구와 같은 맥락을 보인다. 그러나 본 연구에서는 버퍼의 수도 중요하지만 버퍼가 있는 위치도 전체 시스템 성능을 위해서 중요하며 같은 수의 총 버퍼 용량이라도 위치에 따라 크게 성능의 차이를 보일 수 있음을 보였다. 기존의 연구와 같은 결과를 보이는 것은 균등할당 알고리즘에서 총 버퍼 용량이 12에서 14로 늘어날 때 Throughput이 크게 늘어나는 것이다. 이것은 앞에서 설명한 시스템 성능이 총 버퍼 용량이 작을 때 더 민감하다는 것을 보여주는 것이다. 또 한, 제안된 알고리즘은 총 버퍼 용량이 25를 넘으면 더 이상 성능의 향상이 없음을 보이며 이는 모든 Workstation의 입/출력 버퍼를 상위 제한인 5로 설정해 놓고 실험을 수행했을 때에 470정도의 Throughput을 보이는 것으로 입증할 수 있다.

두 번째로, 각 타입의 Value가 다르고 전체 Value를 최대화하는 문제에 대해서는 <표 4>에 그 결과가 나타나 있다. 이와 비교하기 위하여 <표 4>의 최대의 Throughput을 보이는 Buffer Profile과 변형된 알고리즘에 의해 나온 Buffer Profile의 전체 Value의 차이를 보여주었다. 본 연구에서는 4가지 타입의 작업물을 고려하며 각 타입의 Value는 차례대로 15, 3, 5, 7 로 설정하였다.

총 버퍼량을 12에서 25로 늘여가면서 실험을 한 결과 전체적으로 7.2%의 성능 향상이 있음을 보여주었다. 이는 각 타입의 Value가 다를 때는 당연히 Buffer Profile도 달라져야 한다는 것을 의미한다.

마지막으로, 총 버퍼 수를 최소화하는 문제에 대해서는 Desired Throughput을 360으로 설정하였고 <표 5>에서 보이듯 이 총 8개의 버퍼만 존재하면 Desired Throughput을 만족할 수 있음을 보였다.

이는 균등할당 알고리즘에서 총 버퍼 용량이 12일 때 300을 넘지 않는 Throughput을 나타내는 것에 비하면 버퍼의 위치가 시스템 성능에 크게 영향을 미치며 균등할당 알고리즘보다 많은 성능의 향상을 보일 수 있음을 나타낸다.

<표 4> Value가 다른 실험 결과

총 버퍼양	Buffer Profile	Throughput	차이 (%)
12	1,1,0,1,1,1,1,1,0,1,1,1,1,1	2810	6.27
14	1,1,3,1,0,2,1,1,0,1,0,1,1,1	3030	11.6
16	1,1,4,1,1,1,2,1,0,1,0,1,1,1	3170	7.02
20	2,1,3,1,0,1,2,1,1,1,1,1,3,2	3220	5.74
25	4,1,5,1,1,2,2,2,1,1,1,1,2,1	3440	8.48

<표 5> Min Total Buffer Capacity 결과

Buffer Profile	총 버퍼 양
0,1,1,0,0,1,0,1,0,1,0,1,1,1	8

5. 결론 및 향후 연구과제

본 연구에서 유연 생산 시스템 같은 복잡한 시스템에서 Combinatorial 특성을 지니는 버퍼 할당 문제에 대한 근사 최적해를 시뮬레이션과 수정된 유전 알고리즘을 사용하여 개발된 알고리즘으로 제시했다. 기존의 연구들과 달리 제시된 알고리즘은 시스템의 복잡성과 Combinatorial 특성을 모두 다룰 수 있고 Park이 주장한 Stagnat Area에도 빠지지 않음을 보였다. 제시된 알고리즘의 성능검증을 위하여 Workload와 Processing Time을 고려한 균등할당 알고리즘과 성능을 비교하였고 전체적으로 10.5%의 성능 향상이 있음을 보였다. 그리고 제안된 Output이 존재하는 경우의 총 버퍼 최소화에 대한 문제에 대해서는 첫 번째 문제에서 변형된 알고리즘을 사용하였고 실험을 통하여 알고리즘의 타당성을 제시하였다. 마지막으로 각 타입이 Value를 가지는 문제에 대해서도 알고리즘 변형을 통하여 실험을 수행하였고 그 결과를 제시하였다.

본 연구의 의의로는 시스템의 복잡성으로 인하여 수학적 접근이 어려운 문제를 시뮬레이션과 유전알고리즘을 사용하여 근사 최적해를 찾을 수 있는 방법론을 제시하였다는 것이다. 그리고 시스템의 상황에 따라 버퍼 할당의 여러 문제에 적용 가능하도록 쉽게 알고리즘을 수정할 수 있는 방법론을 제시하였다는 것도 하나의 의의라 하겠다. 마지막으로 메타 휴리스틱과 시뮬레이션을 쉽게 연결할 수 있는 Arena와 Visual Basic의 연결에 관한 Guide Line을 제시하였다는 것도 의미를 가질 수 있다.

실제 유연 생산 시스템에 본 연구의 알고리즘을 적용 할 시에는 상용 시뮬레이터만 가지고 있으면 쉽게 시스템을 분석하고 근사 최적해를 찾아낼 수 있을 것이다.

지금까지의 연구에서는 유연 생산 시스템의 다른 구성요소와 버퍼의 연관성을 고려하지 않았다. 향후의 연구에서는 무인 운반차의 발주 방식이나 무인 운반차의 수 등과 함께 버퍼 할당을 고려하여 유연 생산 시스템의 여러 요소의 상관관계를 알아보는 연구도 이루어져야 한다고 생각한다. 즉, 시스템에 무인 운반차가 적정량 보다 적을 때나 많을 때 버퍼 할당이 어떻게 변하는지와 무인 운반차의 발주 방식이 달라 질 때 버퍼 할당의 변화와 시스템 Output의 변화를 알아보는 연구가 필요하다. 그리고 이번 연구의 대상이 된 Traditional System이 아닌 요즘 성능의 검증이 되고 있는 Tandem System이나 양방향 무인 운반차, 혹은 Multi-Load AGV 등의 시스템에 대해서도 제시된 알고리즘이 적용될 수 있도록 알고리즘을 발전시키는 것도 의의가 있을 것이다.

참고문헌

(1) H. Pierreval and L. Tautou (1997), Using evolutionary algorithms and simulation for the optimization of manufacturing systems. IIE Transactions, Vol 29, 181-189.

(2) Hideaki Yamashita and Tayfur Altiok (1998), Buffer capacity allocation for a desired throughput in production lines. IIE Transactions, Vol 30, 883-891.

(3) Mohsen A.Jafari and J. George Shanthikumar (1989), Determination of Optimal Buffer Storage Capacities and Optimal Allocation in Multistage Automatic Transfer Lines. IIE Transactions, Vol 21, 130-135.

(4) Christian M. Lutz, K.Roscoe Davis, Minghe Sun (1998), Determining buffer location and size in production lines using tabu search. European Journal of Operational Research, 301-316.

(5) G. A. Vorous and H. T. Papadopoulos (1998), Buffer Allocation In Unreliable Production Lines Using A Knowledge Based System. Computers Ops Res, Vol 25, 1055-1067.

(6) T. Park (1993), A two-phase heuristic algorithm for determining buffer sizes of

production lines. International Journal Of Production Research, Vol 31, 613-631.
(7) Mark S. Hillier (2000), Characterizing the optimal allocation of storage space in production line systems with variable processing times. IIE Transactions, Vol 32, 1-8.
(8) Frederick S. Hillier and Kut C. So (1991), The Effect of the Coefficient of Variation of Operation Times on the Allocation of Storage Space in Production Line Systems. IIE Transactions, Vol 23, 198-206.
(9) 김여근, 윤복식, 이상복 (1997), 메타 휴리스틱, 영지 문화사.
(10) KUT C. So (1997), Optimal buffer allocation strategy for minimizing work in process inventory in unpaced production lines. IIE Transactions. Vol 29. 81-88.
(11) Neil B, Marks (1987), Output analysis of an automated serial assembly line. International Journal of Production Research. Vol 25, No 8. 1197-1207.
(12) D.Seong, S.Y. Chang and Y.Hong (1995), Heuristic algorithms for buffer allocation in a production line with unreliable machines. International Journal of Production Research. Vol 33, No 7. 1989-2005.

저자소개

이용균
1999년 한성대학교 산업공학과 졸업(학사)
2001년 8월 연세대학교 컴퓨터과학·산업시스템공학과 졸업(석사)
관심분야 : SCM, Buffer Allocation in FMS

김경섭
1982년 연세대학교 기계공학과 졸업(학사)
1986년 University of Nebraska-Lincoln 산업공학과 졸업(석사)
1993년 North Carolina State University 산업공학과 졸업(박사)
1994년 삼성데이타시스템 선임연구원
1994년~현재 연세대학교 기계전자공학부 부교수
관심분야 : 물류시스템, 시뮬레이션 모델링 및 분석, SCM