FlexSim 소프트웨어를 이용한 강화학습 기반 작업 할당 모형 개발

박진성[O], 김준우[*]
[O]동아대학교 산업경영공학과,
[*]동아대학교 산업경영공학과
e-mail: pjs0958@donga.ac.kr[O], kjunwoo@dau.ac.kr[*]

Developing Reinforcement Learning based Job Allocation Model by Using FlexSim Software

Jin-Sung Park[O], Jun-Woo Kim[*]
[O]Dept. of Industiral and Management Systems Engineering, Dong-A University,
[*]Dept. of Industiral and Management Systems Engineering, Dong-A University

● 요 약 ●

병렬 기계 작업장에서 자원을 효율적으로 활용하기 위해서는 처리할 작업을 적절한 기계에 할당해야 한다. 특정 작업을 처리할 기계를 선택할 때 휴리스틱을 사용할 수도 있으나, 특정 작업장에 맞춤화된 휴리스틱을 개발하는 것은 쉽지 않다. 반면, 본 논문에서는 이종 병렬 기계 작업장을 위한 작업 할당 모형을 개발하는데 강화학습을 응용하고자 한다. 작업 할당 모형을 학습하는데 필요한 에피소드들은 상용 시뮬레이션 소프트웨어인 FlexSim을 이용하여 생성하였다. 아울러, stable-baseline3 라이브러리를 이용하여 강화학습 알고리즘을 생성된 에피소드들에 적용하였다. 실험 결과를 통해 시뮬레이션과 강화학습이 작업장 운영관리에 유용함을 알 수 있었다.

키워드: 이종 병렬 기계(unrelated parallel machines), 작업 할당(job allocation), 강화학습(reinforcement learning), 시뮬레이션(simulation), FlexSim 소프트웨어(FlexSim software)

I. Introduction

어떤 공정을 처리할 수 있는 기계가 여러 대 존재하여, 개별 작업물이 이들 중 1대만 방문하게 되는 경우, 해당 기계들을 병렬 기계(parallel machines)라고 지칭한다. 처리할 작업물이 다수인 경우에는 이들을 주어진 기계들에 적절히 분배해야 총처리시간(makespan)을 최소화하고, 기계를 비롯한 자원들을 보다 효율적으로 활용할 수 있다[1]. 나아가, 처리할 수 있는 공정의 유형은 동일하나, 기계들의 성능이나 특성이 상이하여, 특정 작업물의 처리 시간이 할당된 기계에 따라 달라지는 이종 병렬 기계(unrelated parallel machines) 작업장에서는 개별 작업물에 대한 기계별 처리시간까지 고려하여 작업물을 할당할 기계를 결정하여야 한다[2].

다수의 작업물을 병렬 기계에 적절히 배분하여 총처리시간을 최소화하는 것은 NP-hard 문제이기 때문에, 휴리스틱이나 메타 휴리스틱 같은 근사해법을 적용하는 경우가 많다[1-2]. 다만, 개별 현장에 맞춤화된 근사해법을 개발하는 것은 일반적으로 까다롭고, 공정의 특성이나 생산일정계획에 대한 배경지식을 요구하는 것으로 알려져 있다[3].

한편, 최근에는 이 같이 복잡한 문제를 해결하기 위한 방법으로 강화학습(Reinforcement Learning, RL)을 비롯한 인공지능(Artificial Intelligence, AI) 기법들이 각광을 받고 있다[4]. 이에, 본 논문에서는 이종 병렬 기계 작업장에서 개별 작업물을 할당할 기계를 선정하는데 사용할 수 있는 작업 할당 모형을 개발하는데 RL 알고리즘을 응용해보고, 그 과정과 결과에 대해 토의해보고자 한다.

II. 연구 배경

생산일정계획의 수립을 위해서는 처리할 작업들에 대한 시퀀싱이나 배분과 같은 조합 최적화가 필요한 경우가 많다. 또한, 조합 최적화 문제들은 보통 NP-hardness를 갖기 때문에 정확한 최적해를 찾기보다는 근사 해법을 적용하여 단시간 내에 근사 최적해를 찾는 것이 일반적이다[1-2][5]. 그러나, 개별 현장의 특성이 반영된 효과적인 근사 해법을 개발하는 것은 까다롭고 상당한 시간을 요구한다[3]. 따라서, 강화학습과 같은 AI 기법을 활용할 수 있다면 보다 효과적으로

생산일정계획을 수립할 수 있을 것이다.

강화학습이란 가상의 에이전트(agent)가 주어진 상황에 대한 효과적인 대응 방법을 학습하도록 하는데 사용할 수 있는 AI 기법 중 하나이다. 에이전트는 의사결정이 필요할 때마다 현재 상황에 대한 관찰(observation)을 실시하고, 어떤 행동(action)을 취하게 된다. 이후에는 자신의 행동에 대한 보상(reward)을 받는다. 아울러, 에이전트가 가급적 많은 보상을 받도록 적절한 행동을 취하도록 하는 것이 강화학습의 목표이다[6].

일련의 관찰, 행동 및 보상 획득이 이루어지는 과정을 에피소드(episode)라고 하는데, 강화학습을 사용하기 위해서는 에이전트가 적절한 환경에서 에피소드들을 경험하게 하면서 그 결과에 기반하여 행동 선택 방식을 갱신하도록 해야 한다. 본 연구에서는 강력한 3D 시각화 기능과 다양한 부가 기능들을 제공하는 상용 이산 사건 시뮬레이션 소프트웨어인 FlexSim[7]을 이용하여 강화학습에 필요한 환경을 구성하였다. 이처럼 시뮬레이션과 강화학습을 연계하는 것은 근래 다양한 분야에서 시도되고 있는 접근법이기도 하다[8].

생성된 에피소드를 분석하여 에이전트를 학습시키는 데는 Python 기반의 Stable-Baselines3 라이브러리를 사용하였다. 이 라이브러리는 다양한 심층 RL 알고리즘들을 지원할 뿐만 아니라, 문서화가 잘 되어 있고, 많은 테스트를 통해 안정성이 검증되어 있다는 점 등의 장점을 갖는다[9].

III. 강화학습을 위한 FlexSim 모형

<Fig. 1>은 FlexSim 소프트웨어를 이용하여 작성한 이종 병렬 기계 작업장 모형의 전체적인 구조를 보여준다. 작업물들의 도착 간격은 평균이 10초인 지수 분포를 따르며, 도착한 작업물들은 처음에 Queue_init을 거쳐, Queue1~3 중 한 곳을 향하게 된다. Queue1~3는 각각 이종 병렬 기계 Processor1~3에 투입될 작업물들의 대기 장소이며, Processor1~3에서 처리가 끝난 작업물은 Sink1 개체로 보내어 모형에서 삭제한다.

나아가, 작업물들은 1~3번 타입의 3가지 유형으로 구분하며, 특정 작업물이 i(i=1, 2, 3)번 타입일 확률은 1/3이다. 또한, 특정 작업물의 타입은 이전 또는 이후 작업물들의 타입과 무관하다. 기본적으로, Processor1~3은 모두 모든 유형의 작업물을 처리할 수 있다. 다만, 개별 작업물의 처리 시간은 <Table 1>과 같이 어떤 기계에 할당되는지에 따라 달라진다. 좀 더 자세히 말해, 작업물의 처리 시간은 일양분포를 따르는데, 타입 번호와 기계 번호가 같을 경우에는 10~20초 범위의 값을 갖는 반면, 그렇지 않은 경우에는 30~40초 범위의 값을 가져 더 오랜 시간이 소요된다. 따라서, 가능하면 i번 타입 작업물은 Processor$i$에 할당하되, 만약 Queue$i$에서 대기 중인 작업물이 과도하게 많은 경우에는 Processor$j$ ($i \neq j$)에 할당하는 것을 고려할 수도 있겠으나, 언제 이 같은 판단을 내려야 하는지는 명확하지 않은 상황이다.

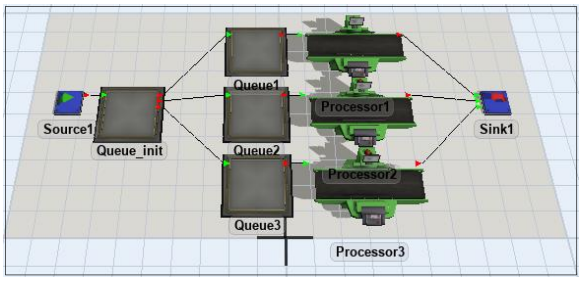

Fig. 1. FlexSim Model for 3 Unrelated Parallel Machines

Table 1. Processing Times of Given Jobs

Type	Processor1	Processor2	Processor3
1	Uniform(10, 20)	Uniform(30, 40)	Uniform(30, 40)
2	Uniform(30, 40)	Uniform(10, 20)	Uniform(30, 40)
3	Uniform(30, 40)	Uniform(30, 40)	Uniform(10, 20)

상기한 모형에서 Queue_init에 진입한 작업물을 Queue1~3 중 적절한 개체로 전달하는데 사용할 수 있는 작업 할당 모형을 개발하기 위해, 강화학습 구성요소들을 아래와 같이 설정하였다.

•상태 S_t=[$Type_t$, W_1, W_2, W_3], 단, $Type_t$=t 시점에 Queue_init에서 내보낼 작업물의 타입 번호, W_i=t 시점에 Queue$i$에서 대기 중인 작업물 개수 ($i = 1, 2, 3$)

•행동 : Queue_init에서 내보낼 작업물의 다음 행선지로 Queue$i$를 선택 ($i = 1, 2, 3$)

•보상 r=K / FT, 단, FT=처리를 마치고 Sink1에 진입한 작업물의 흐름 시간(Sink1 진입 시간 - Queue_init 이탈 시간)

나아가, 보상을 계산할 때 상수 K의 값으로는 작업물 처리 시간 평균값에 해당하는 25를 사용하였고, Queue_init에서 작업물 1개를 내보내기 위해 다음 행선지를 결정할 때마다 상태에 대한 관찰 및 행동이 이루어지도록 구성하였다.

IV. 실험 결과

작업 할당 모형을 얻기 위해 먼저, Stable-Baselines3 라이브러리가 제공하는 것들 중, 폭넓은 범위의 문제에서 좋은 성능을 보이는 것으로 알려진 정책 기반 알고리즘인 PPO (Proximal Policy Optimization)[10]를 이용하여 timestep=50,000회에 대한 학습을 실시하였다. 이후에는 학습된 모형의 성능을 분석하기 위해, Queue_init에서 작업물을 내보낼 때 Queue1~3중 무작위로 1개를 선택하는 "임의 선택" 시나리오와 학습된 RL 모형을 이용하여 작업 할당을 실시하는 "RL 모형 이용" 시나리오, 2가지를 비교하였고, 그 결과는 <Table 2>에 요약하였다.

2가지 시나리오에 대해 FlexSim 소프트웨어를 이용하여 실행시간 10,000초 동안에 걸쳐 시뮬레이션 실험을 진행하면서 Queue1~3에서의 평균 대기시간을 집계한 결과, <Table 2>에서처럼 학습된 RL 기반 작업 할당 모형을 이용했을 때, 대기시간들이 훨씬 짧고, 결과적으로 작업물의 흐름이 원활한 것을 볼 수 있었다.

Table 2. Comparisons of Performance Measures

성능 평가지표	임의 선택	RL 모형 이용
Queue1 평균 대기시간	202.8	7.9
Queue2 평균 대기시간	95.4	16.6
Queue3 평균 대기시간	379.7	16.0

V. 결론

본 연구를 통해 다음과 같은 점들을 살펴볼 수 있었다.

첫째, 강화학습을 통해 이종 병렬 기계 작업장에서 현장 상황을 고려하여 작업물을 적절한 기계에 할당하기 위한 작업 할당 모형을 개발할 수 있었다. 이처럼, 강화학습을 적절히 활용하면 생산운영관리와 관련된 다양한 의사결정모형을 개발할 수 있을 것으로 생각된다. 특히, 조합 최적화에 해당하는 의사결정에 이 같은 모형이 유용할 것이다.

둘째, 상용 시뮬레이션 소프트웨어와 오픈소스 강화학습 라이브러리의 연동을 통해 매우 편리한 방식으로 강화학습을 실시하는데 필요한 환경을 구축하고, 학습을 진행한 후 그 성과를 평가할 수 있었다.

셋째, 외부 라이브러리가 제공하는 강화학습 알고리즘을 사용하더라도 상태 및 행동 공간, 보상 함수 등 세부 내용들은 시뮬레이션 소프트웨어 내에서 정의되어야 한다. 따라서, 강화학습을 실행하는데 시뮬레이션을 활용하고자 하는 경우에는 유연한 로직 작성 도구와 외부 애플리케이션 연동을 지원하는 시뮬레이션 소프트웨어를 선택해야 할 것이다. 특히, FlexSim 소프트웨어는 이 외에도 3D 모델링 개체나 개체 지향적 구조 등 다양한 장점을 가져, 강화학습 환경으로 사용하는데 매우 적합하였다. 앞으로도 저자들은 이 같은 방식으로 생산운영관리 분야의 다양한 문제들을 해결하는데 사용할 수 있는 의사결정 모형을 개발하기 위한 연구를 지속적으로 수행하고자 한다.

ACKNOWLEDGEMENT

This research was supported by the Ministry of Education of the Republic of Korea and National Research Foundation(NRF) (NRF-2022S1A5C2A03093301)

REFERENCES

[1] M. L. Pinedo, "*Scheduling: Theory, Algorithms, and Systems*," Springer, 2016.

[2] M. Đurasević, and D. Jakobović, "Heuristic and Metaheuristic Methods for the Parallel Unrelated Machines Scheduling Problem: A Survey," Artificial Intelligence Review, forthcoming.

[3] E. K. Burke, M. Gendreau, M. Hyde, G. Kendall, G. Ochoa, E. Özcan, and R. Qu, "Hyper-heuristics: A Survey of the State of the Art," Journal of the Operational Research Society, Vol. 64, No. 12, pp. 1695-1724, 2013.

[4] S. Levine, "Reinforcement Learning and Control as Probabilistic Inference: Tutorial and Review," arXiv Preprint:1805.00909, 2018.

[5] T. Zhou, D. Tand, H. Zhu, and L. Wang, "Reinforcement Learning with Composite Rewards for Production Scheduling in a Smart Factory," IEEE Access, Vol. 9, pp. 752-766, 2020.

[6] R. S. Sutton, and A. G. Barto, "Reinforcement Learning: An Introduction (2nd ed.)," Bradford Books, 2018.

[7] M. Beaverstock, A. Greenwood, and W. Nordgren, "*Applied Simulation – Modeling and Analysis using FlexSim*," BookBaby, 2018.

[8] N. Feldkamp, S. Bergmann, and S. Strassburger, "Simulation-based Deep Reinforcement Learning for Modular Production Systems," Proceedings of the 2020 Winter Simulation Conference, pp. 1596-1607, 2020.

[9] A. Raffin, A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, and N. Dormann, "Stable-baselines3: Reliable Reinforcement Learning Implementations," Journal of Machine Learning Research, Vol. 22, No. 268, pp. 1-8, 2021.

[10] H. Lee, T. Kim, H. Kim, and S.-H. Hwang, "Reinforcement Learning based Autonomous Emergency Steering Control in Virtual Environment," Journal of Drive and Control, Vol. 19, No. 4, pp. 110-116, 2022.