강화학습 논문 리뷰 스터디 10기

Dynamic multi-objective scheduling for flexible job shop by deep reinforcement learning

김용희(Kim Yong Hae)

Agenda

- 강화학습 논문 스터디와 함께한 시간
- 논문 내용 정리

강화학습 논문 스터디와 함께한 시간

8th


Available online at www.sciencedirect.com

ScienceDirect

Procedia CIRP 72 (2018) 1264–1269

www.elsevier.com/locate/procedia

51st CIRP Conference on Manufacturing Systems

Optimization of global production scheduling with deep reinforcement learning

Bernd Waschneck, André Reichstaller, Lenz Belzner, Thomas Altenmüller, Thomas Bauernhansl, Alexander Knapp, Andreas Kyek

Abstract

Industrie 4.0 introduces decentralized, self-organizing and self-learning systems for production control. At the same time, new machine learning algorithms are getting increasingly powerful and solve real world problems. We apply Google DeepMind's Deep Q Network (DQN) agent algorithm for Reinforcement Learning (RL) to production scheduling to achieve the Industrie 4.0 vision for production control. In an RL environment cooperative DQN agents, which utilize deep neural networks, are trained with user-defined objectives to optimize scheduling. We validate our system with a small factory simulation, which is modeling an abstracted frontend-of-line semiconductor production facility.

© 2018 The Authors. Published by Elsevier B.V.

Keywords: Production Scheduling, Reinforcement Learning, Machine Learning in Manufacturing

1. Introduction

- Proceeding -

9th


COMPLEX SYSTEM MODELING AND SIMULATION
ISSN 2096-9929 01/06 pp 257–270
Volume 1, Number 4, December 2021
DOI: 10.23919/CSMS.2021.0027

A Review of Reinforcement Learning Based Intelligent Optimization for Manufacturing Scheduling

Ling Wang*, Zixiao Pan, and Jingjing Wang

Abstract: As the critical component of manufacturing systems, production scheduling aims to optimize objectives in terms of profit, efficiency, and energy consumption by reasonably determining the main factors including processing path, machine assignment, execute time and so on. Due to the large scale and strongly coupled constraints nature, as well as the real-time solving requirement in certain scenarios, it faces great challenges in solving the manufacturing scheduling problems. With the development of machine learning, Reinforcement Learning (RL) has made breakthroughs in a variety of decision-making problems. For manufacturing scheduling problems, in this paper we summarize the designs of state and action, tease out RL-based algorithm for scheduling, review the applications of RL for different types of scheduling problems, and then discuss the fusion modes of reinforcement learning and meta-heuristics. Finally, we analyze the existing problems in current research, and point out the future research direction and significant contents to promote the research and applications of RL-based scheduling optimization.

Key words: Reinforcement Learning (RL); manufacturing scheduling; scheduling optimization

1 Introduction

- Review Paper -

10th


Computers & Industrial Engineering

journal homepage: www.elsevier.com/locate/caie

Dynamic multi-objective scheduling for flexible job shop by deep reinforcement learning

Shu Luo, Linxuan Zhang*, Yushun Fan

ARTICLE INFO

Keywords:
Flexible job shop scheduling
Multi-objective
New job insertion
Dispatching rules
Deep reinforcement learning

ABSTRACT

1. Introduction

https://doi.org/10.1016/j.cie.2021.107489

- Full Paper -

논문 내용 정리

논문 개요

Computers & Industrial Engineering 159 (2021) 107489

Contents lists available at ScienceDirect

Computers & Industrial Engineering

journal homepage: www.elsevier.com/locate/caie

Dynamic multi-objective scheduling for flexible job shop by deep reinforcement learning

Shu Luo, Linxuan Zhang*, Yushun Fan

National Engineering Research Center for Computer Integrated Manufacturing systems, Department of Automation, Tsinghua University, Beijing 100084, China

ARTICLE INFO

Keywords:
Flexible job shop scheduling
Multi-objective
New job insertion
Dispatching rules
Deep reinforcement learning

ABSTRACT

In modern volatile and complex manufacturing environment, dynamic events such as new job insertions and machine breakdowns may randomly occur at any time and different objectives in conflict with each other should be optimized simultaneously, leading to an urgent requirement of real-time multi-objective rescheduling methods that can achieve both time efficiency and solution quality. In this regard, this paper proposes an on-line rescheduling framework named as two-hierarchy deep Q network (THDQN) for the dynamic multi-objective flexible job shop scheduling problem (DMOFJSP) with new job insertions. Two practical objectives including total weighted tardiness and average machine utilization rate are optimized. The THDQN model contains two deep Q network (DQN) based agents. The higher-level DQN is a controller determining the temporary optimization goal for the lower DQN. At each rescheduling point, it takes the current state features as input and chooses a feasible goal to guide the behaviour of the lower DQN. Four different goals corresponding to four different forms of reward functions are suggested, each of which optimizes an indicator of tardiness or machine utilization rate. The lower-level DQN acts as an actuator. It takes the current state features together with the higher optimization goal as input and chooses a proper dispatching rule to achieve the given goal. Six composite dispatching rules are developed to select an available operation and assign it on a feasible machine, which serve as the candidate action set for the lower DQN. A novel training framework based on double DQN (DDQN) is designed. The trained THDQN is compared with each proposed composite dispatching rule, existing well-known dispatching rules as well as other reinforcement learning based scheduling methods on a wide range of test instances. Results of numerical experiments have confirmed both the effectiveness and generality of the proposed THDQN.

1. Introduction

Nowadays, with the dramatic growth of complexity and uncertainty in manufacturing systems, there has been an urgent need of dynamic multi-objective rescheduling methods with the ability of handling random disturbances such as dynamic demands, machine breakdowns and uncertain processing times in real time while simultaneously considering different objectives like makespan and total tardiness. Without lose of generality, most of scheduling problems existing in modern multi-type and low-volume discrete manufacturing systems can be regarded as dynamic multi-objective flexible job shop scheduling problem (DMOFJSP). It has been proved to be NP-hard (Garey, Johnson, & Sethi, 1976) and is more intractable than classical job shop scheduling problem (JSP) since each operation can be processed on one or more compatible machines. Due to its high complexity and universality, the DMOFJSP is of great significance to be researched for both academia and industry.

To address the dynamic events, traditional scheduling methods for the DMOFJSP can be mainly divided into two kinds, namely metaheuristics and dispatching rules. Metaheuristics, such as genetic algorithm (GA) (Kundakcı & Kulak, 2016), particle swarm optimization (PSO) (Tang, Dai, Salido, & Giret, 2016) and ant colony optimization algorithm (ACO) (Zhang, Li, Zhang, & Wang, 2020), always decompose a dynamic scheduling problem into a sequence of static scheduling problems and solve them separately. They can obtain near-optimal solutions but suffer from poor time efficiency. Dispatching rules, like first in first out (FIFO), earliest due date (EDD) and most remaining processing time (MRT) (Rajendran & Holthaus, 1999), react to the dynamic disturbances exactly in real time. They can adjust the schedule in the shortest time but the obtained results are far from optimal in the long

* Corresponding author.
E-mail addresses: luos17@mails.tsinghua.edu.cn (S. Luo), lxzhang@mail.tsinghua.edu.cn (L. Zhang), fanyus@tsinghua.edu.cn (Y. Fan).

https://doi.org/10.1016/j.cie.2021.107489
Received 19 August 2020; Received in revised form 4 May 2021; Accepted 15 June 2021
Available online 18 June 2021
0360-8352/

- **Dynamic multi-objective scheduling for flexible job shop by deep reinforcement learning**
- **2021. 06**
- **Full Paper**
- **주요 내용**

  - 현대의 복잡한 제조 환경에서는 새로운 작업 추가(new job insertion)되거나 설비 고장 등과 같은 동적 이벤트가 얼마든지 무작위로 발생될 수 있으며, 충돌하는 서로 다른 목표를 동시에 최적화해야 하는 니즈가 있어 시간 효율성과 스케줄링 품질을 모두 달성할 수 있는 실시간 다목적 리스케줄링 방안 필요

  - 논문에서 새로운 작업 삽입이 있는 DMOFJSP(Dynamic Multi-Objective Flexible Job Shop Scheduling Problem)를 위한 THDQN(two-hierarchy deep Q network)이라는 온라인 리스케줄링 프레임워크(On-line rescheduling framework)를 제안

  - THDQN 모델은 두 개의 DQN 기반의 Agent가 포함되어 있으며, 상위 레벨 DQN은 하위 DQN에 대한 임시 최적화 목표를 결정하는 컨트롤러 역할을, 하위 DQN은 목표 달성을 위한 적절한 디스패칭 룰을 선택하는 액추에이터 역할 수행

논문 구성

1. Introduction

- 오늘날 제조 시스템에 존재하는 대부분의 스케줄링 문제는 DMOFJSP(dynamic multi-objective flexible job shop scheduling problem)로 간주될 수 있고 이 것은 NP-hard로 입증되었으며 각 작업이 하나 이상의 호환 가능한 시스템에서 처리될 수 있기 때문에 고전적 JSP(Job Shop Scheduling Problem) 보다 다루기 어렵기 때문에 학계와 산업계 모두에서 연구하는데 큰 의미가 있음
- Dynamic Event를 처리하기 위해 DMOFJSP에 대한 전통적인 스케줄링 방법은 주로 Meta-heuristics과 Dispatching rule 이 있음

  - 메타-휴리스틱 : 동적 스케줄링 문제를 일련의 정적 스케줄링 문제로 분해하여 개별적으로 해결함으로써 최적에 가까운 솔루션을 얻을 수 있으나 시간 효율성이 좋지 않음
  - 디스패칭-룰 : 최단 시간에 스케줄링 조정이 될 수 있지만 근시안적으로 장기적 최적화 달성이 어려울 뿐만 아니라, 모든 목표에 대해 동시에 최적화할 단일 규칙을 선택하는 것 역시 어려움
- DMOFSP의 리스케줄링 과정은 MDP로 모델링 될 수 있는데 이를 해결하기 위한 과거에는 항상 동적 프로그래밍(DP)(Howard, 1960)에 의존했는데, 이는 상태 전이 함수의 정확한 모델링을 요구하는 모델 기반 방법이나 다양한 불확실성을 지닌 현대 제조 시스템에서는 상태 전이 함수를 미리 정확하게 모델링할 수 없음
- 최근 몇 년 동안 강화학습이 MDP를 처리하는 효과적이 방법으로 부상했으며 이는 시행착오를 통해서 모델 없이 최적의 행동 전략을 학습할 수 있기 때문에 실제 동적 스케줄링 문제에서 많은 성공적인 응용을 달성함

1. Introduction

- 그럼에도 불구하고 고전적인 RL 기반 동적 스케줄링은 두 가지 문제점이 있음
  - 실제 제조 시스템은 대부분 연속적이므로 Q-table을 메모리에 유지하는 것이 불가능
    -> 연속 상태 공간을 이산화 하여 각 간격이 상태에 해당되게 처리 : 모델 정확도가 감소
    -> 심층 신경망을 Q-function approximator로 사용하는 DRL 활용
  - 다중 목표 리스케줄링 문제는 단일 RL agent가 모든 목표를 동시에 최적화하도록 설계하는게 어려움
    -> HRL(hierarchical reinforcement learning) 활용
- 이러한 요소는 DRL과 HRL의 장점을 결합하여 두 가지 딜레마를 해결하도록 하였으며 DMOFJSP를 해결하기 위해 계층적 심층 강화 학습 기반 방법을 채택한 이전 연구는 없었음
- 새로운 작업 추가되는 DMOFJSP를 해결하기 위해 THDQN(two-hierarchy deep Q network)라는 two-hierarchy deep reinforcement learning model을 제안함

Table 1
Existing RL based methods for dynamic job shop scheduling problem.

Work	State space	Algorithm	Agent	Objective	Dynamic events	Problem
Zhang and Dietterich (1995)	Continuous	Temporal difference algorithm	Single−agent	Makespan	None	Job shop scheduling
Riedmiller and Riedmiller (1999)	Continuous	Q-learning	Multi-agent	Summed tardiness	None	Job shop scheduling
Aydin and Öztemel (2000)	Discrete	Q-learning	Single−agent	Mean tardiness	New job insertions	Job shop scheduling
Wang and Usher (2004)	Discrete	Q-learning	Single−agent	Mean tardiness	New job insertions	Job shop scheduling
Yingzi and Mingyang (2004)	Discrete	Q-learning	Single−agent	Mean tardiness	New job insertions	Job shop scheduling
Chen et al. (2010)	Discrete	Q-learning	Single−agent	Mean flow time; Mean tardiness	Fluctuation of work in process	Job shop scheduling
Gabel and Riedmiller (2012)	Discrete	Policy gradient	Multi-agent	Makespan	None	Job shop scheduling
Bouazza et al. (2017)	Discrete	Q-learning	Multi-agent	Makespan; Total weighted completion time; Weighted average waiting time	New job insertions	Flexible job shop scheduling
Shahrabi et al. (2017)	Discrete	Q-learning	Single−agent	Mean flow time	New job inserions; Machine breakdowns	Job shop scheduling
Wang (2018)	Discrete	Q-learning	Multi-agent	Earliness and tardiness punishment	New job insertions	Job shop scheduling
Waschneck et al. (2018)	Continuous	Deep Q-learning	Multi-agent	Uptime utilization	Machine breakdowns	Flexible job shop scheduling
Kuhnle et al. (2019)	Continuous	Trust region policy optimization	Single−agent	Machine utilization; Lead time of orders	None	Job shop scheduling
Liu et al. (2020)	Continuous	Deep deterministic policy gradient	Multi-agent	Makespan	Processing time variations	Job shop scheduling
Altenmüller et al. (2020)	Continuous	Deep Q-learning	Single−agent	Time constraint violations	Machine breakdowns; New job insertions	Job shop scheduling
Our method	Continuous	Hierarchical deep Q-learning	Single−agent	Total weighted tardiness; Average machine utilization rate	New job insertions	Flexible job shop scheduling

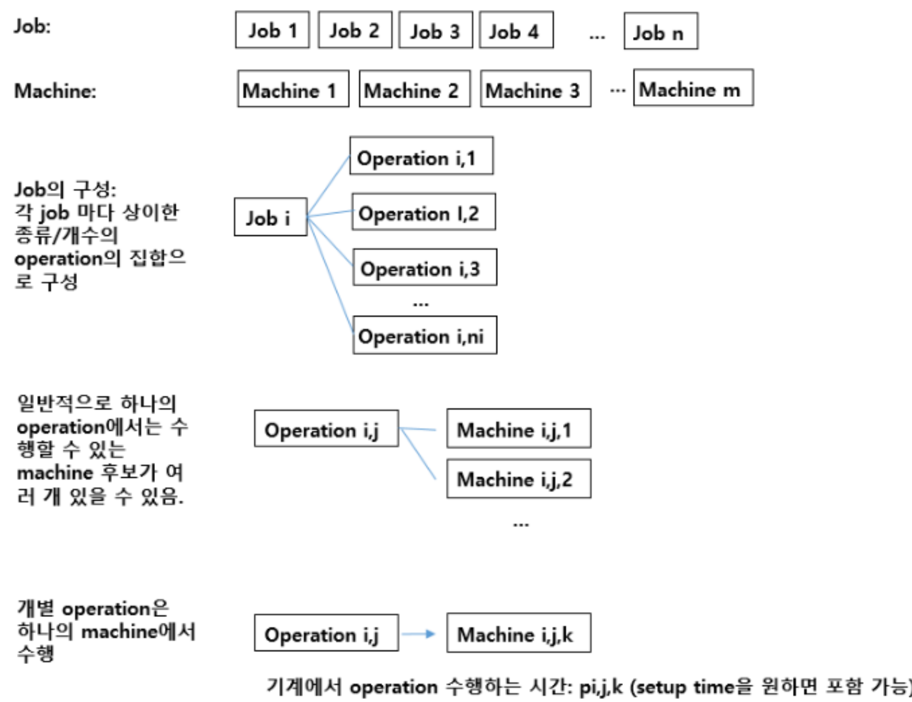
4. Problem formulation

The DMOFJSP with new job insertions considered in this paper can be defined as follows. There are n successively arriving jobs $\boldsymbol{J} = \{J_1, J_2, \ldots, J_n\}$ to be processed on m machines $\boldsymbol{M} = \{M_1, M_2, \ldots, M_m\}$. Each job J_i consists of n_i operations where $O_{i,j}$ is the jth operation of job J_i. Each operation $O_{i,j}$ can be processed on any machine M_k selected from a compatible machine set $\boldsymbol{M}_{i,j}$ ($\boldsymbol{M}_{i,j} \subseteq \boldsymbol{M}$). The processing time of operation $O_{i,j}$ on machine M_k is denoted by $t_{i,j,k}$. The arrival time and due date of job J_i is A_i and D_i, respectively. $C_{i,j}$ represents the actual completion time of operation $O_{i,j}$. The urgency degree of job J_i is denoted by Pr_i, where a higher urgency degree indicates a higher punishment on tardiness. The objective is to minimize the total weighted tardiness and maximize the average machine utilization rate simultaneously. To simplify the problem at hand, several predefined constraints should be satisfied as follows.

(1) Each machine can process at most one operation at a time (capacity constraint).
(2) All operations belonging to the same job should be processed one after another in a fixed order (precedence constraint).
(3) Each operation should be processed nonpreemptively without interruption.
(4) Transportation times and setup times are negligible.

2. Job shop scheduling 문제의 정의

-Job shop scheduling 문제는 Job, operation, machine 간 관계를 아래와 같이 설정하고, 개별 Job들을 수행할 수 있도록 구성 operation을 machine에 할당해주는 문제라고 정의할 수 있음.



Job shop scheduling 문제 정의

※ 출처 : https://m.blog.naver.com/PostView.naver?isHttpsRedirect=true&blogId=ppepajbw432&logNo=221532219346

4. Problem formulation

The notations used for problem formulation are listed below.
(1) Parameters:

n: total number of jobs
m: total number of machines
J_i: the ith job.
n_i: total number of operations belonging to job J_i.
M_k: the kth machine.
$O_{i,j}$: the jth operation of job J_i.
$\boldsymbol{M}_{i,j}$: the available machine set for operation $O_{i,j}$.
$t_{i,j,k}$: the processing time of operation $O_{i,j}$ on machine M_k.
A_i: the arrival time of job J_i.
D_i: the due date of job J_i.
Pr_i: the urgency degree of job J_i.
i, h: index of jobs, $i, h = 1, 2, \ldots, n$.
j, g: index of operations belonging to job J_i and J_h, $j = 1, 2, \ldots, n_i$, $g = 1, 2, \ldots, n_h$.
k: index of machines, $k = 1, 2, \ldots, m$.

(2) Decision variables:

$C_{i,j}$: the completion time of operation $O_{i,j}$.

$$X_{i,j,k} = \begin{cases} 1 & \text{if } O_{i,j} \text{ is assigned on machine } M_k \\ 0 & \text{otherwise} \end{cases}$$

$$Y_{i,j,h,g} = \begin{cases} 1 & \text{if } O_{i,j} \text{ is a predecessor of } O_{h,g} \\ -1 & \text{if } O_{i,j} \text{ is a successor of } O_{h,g} \end{cases}$$

$X_{i,j,k}$ determines which machine an operation is assigned on, while $Y_{i,j,h,g}$ determines the relative processing priority between any two operations.

Based on the notations above and the model developed in Lu, Li, Gao, Liao, and Yi (2017), the DMOFJSP addressed in this paper can be described mathematically as follows.

$$\text{Minimize}\begin{cases} TWT = \sum_{i=1}^{n} \max\left(C_{i,n_i} - D_i, 0\right) \cdot Pr_i & (6) \\ 1\Big/U_{ave} = 1\Bigg/\left(\frac{1}{m}\sum_{k=1}^{m}\frac{\sum_{i=1}^{n}\sum_{j=1}^{n_i} t_{i,j,k}X_{i,j,k}}{\max\limits_{i} C_{i,n_i} \cdot X_{i,n_i,k}}\right) & (7) \end{cases}$$

$$s.t.\begin{cases} C_{i,0} = 0, \quad C_{i,j} > 0, \quad \forall i, j & (8) \\ \sum_{k \in \boldsymbol{M}_{i,j}} X_{i,j,k} = 1, \quad \forall i, j & (9) \\ \left(C_{i,1} - t_{i,1,k} - A_i\right)X_{i,1,k} \geqslant 0, \quad \forall i, k & (10) \\ \left(C_{i,j} - t_{i,j,k} - C_{i,j-1}\right)X_{i,j,k} \geqslant 0, \quad \forall i, j, k & (11) \\ \left(C_{h,g} - t_{h,g,k} - C_{i,j}\right)X_{i,j,k}X_{h,g,k}\left(Y_{i,j,h,g} + 1\right) & \\ +\left(C_{i,j} - t_{i,j,k} - C_{h,g}\right)X_{i,j,k}X_{h,g,k}\left(1 - Y_{i,j,h,g}\right) \geqslant 0, \quad \forall i, j, h, g, k & (12) \end{cases}$$

Objective (6) is total weighted tardiness of all jobs, where the urgency degree is used as weight factor (i.e., penalty factor) of tardiness. Objective (7) is the reciprocal of average machine utilization rate. Eq. (8) indicates that the completion time of each operation must be non-negative. Eq. (9) suggested that each operation can be assigned on only one available machine. Eq. (10) makes sure that a job can only be processed after its arrival time. Precedence constraint is ensured in Eq. (11). Capacity constraint is guaranteed in Eq. (12).

5. Proposed methods for the DMOFJSP

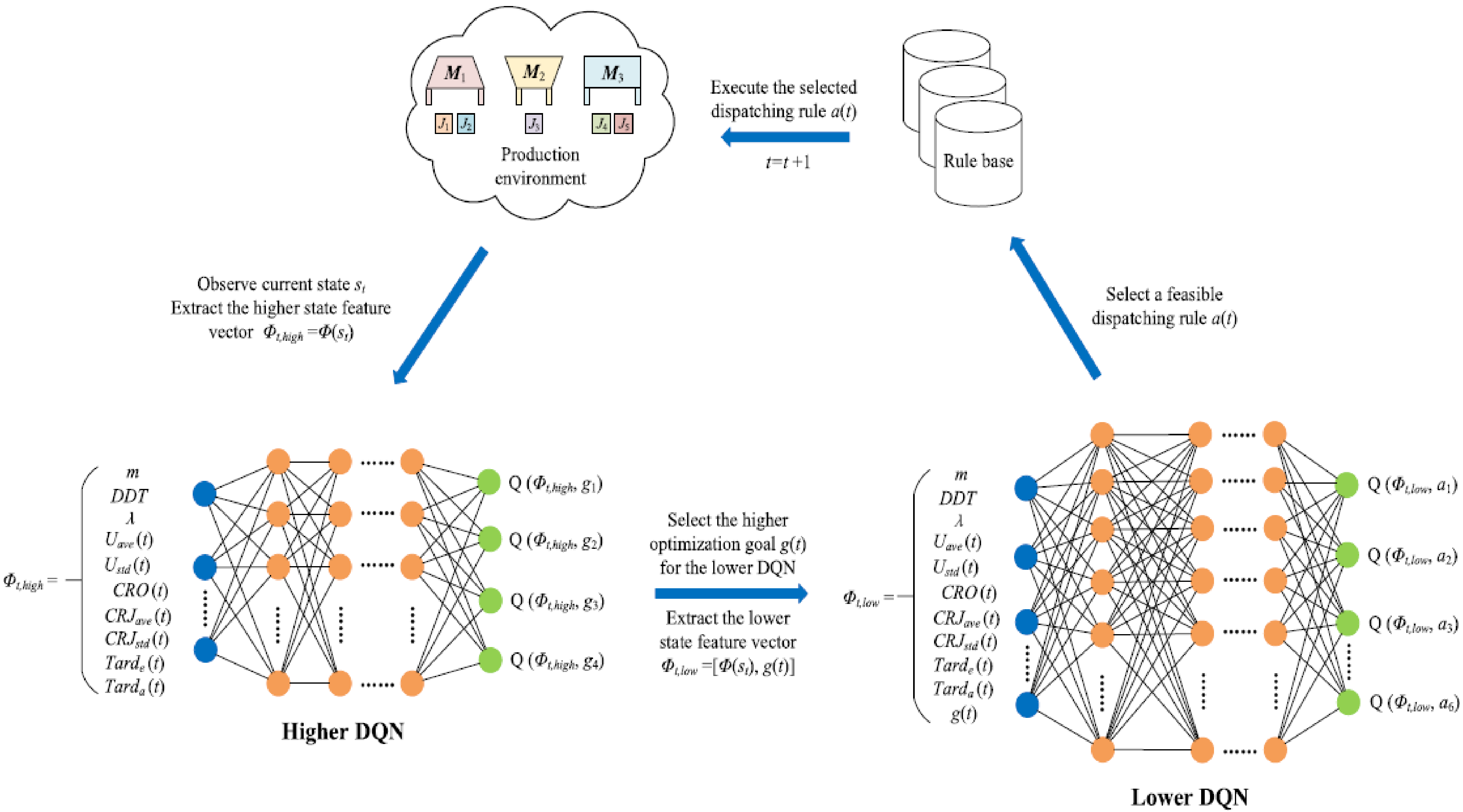


Fig. 1. Structure of the proposed THDQN.

5. Proposed methods for the DMOFJSP

Algorithm 12. The DDQN based training algorithm for the THDQN

Initialize the higher replay memory D_{high} with capacity N_{high} and the lower replay memory D_{low} with capacity N_{low}
Initialize higher online DQN Q_{high} with random weights θ_{high}
Initialize higher target DQN $\widehat{Q}_{high}$ with weights $\theta_{high}^{-} = \theta_{high}$
Initialize lower online DQN Q_{low} with random weights θ_{low}
Initialize lower target DQN $\widehat{Q}_{low}$ with weights $\theta_{low}^{-} = \theta_{low}$
for epoch $= 1 : L$ **do**
  Initialize a new production environment with random settings of m, DDT and λ
  Observe the initial state s_0 with state feature vector $\phi(s_0)$
  Set the initial higher feature vector $\phi_{0,high} = \phi(s_0)$
  Select higher goal $g_0 = \varepsilon\text{-}greedy(Q_{high}, \phi_{0,high})$
  for $t = 0 : T$ (t is the rescheduling point at which an operation has been completed or a new job arrives, T is the terminal time when all operations have been scheduled) **do**
    Observe the current state s_t with state feature vector $\phi(s_t)$
    Set $\phi_{t,low} = [\phi(s_t), g_t]$
    Select dispatching rule $a_t = \varepsilon\text{-}greedy(Q_{low}, \phi_{t,low})$
    Execute rule a_t, observe the new state s_{t+1} with state feature vector $\phi(s_{t+1})$, obtain the immediate reward r_t by Algorithm 11
    Set $\phi_{t+1,high} = \phi(s_{t+1})$
    Select higher goal $g_{t+1} = \varepsilon\text{-}greedy(Q_{high}, \phi_{t+1,high})$
    Set $\phi_{t+1,low} = [\phi(s_{t+1}), g_{t+1}]$
    Store transition $(\phi_{t,high}, g_t, r_t, \phi_{t+1,high})$ in D_{high}
    Store transition $(\phi_{t,low}, a_t, r_t, \phi_{t+1,low})$ in D_{low}
    Sample random minibatch of transitions $(\phi_{j,high}, g_j, r_j, \phi_{j+1,high})$ from D_{high}
    Set $y_{high} = r_j + \gamma \widehat{Q}_{high}(\phi_{j+1,high}, \text{argmax}_{g'} Q_{high}(\phi_{j+1,high}, g'; \theta_{high}); \theta_{high}^{-})$
    Perform a gradient descent step on $(y_j - Q_{high}(\phi_{j,high}, g_j; \theta_{high}))^2$ with respect to the parameters θ_{high} of higher online network Q_{high}
    Sample random minibatch of transitions $(\phi_{j,low}, a_j, r_j, \phi_{j+1,low})$ from D_{low}
    Set $y_j = r_j + \gamma \widehat{Q}_{low}(\phi_{j+1,low}, \text{argmax}_{a'} Q_{low}(\phi_{j+1,low}, a'; \theta_{low}); \theta_{low}^{-})$
    Perform a gradient descent step on $(y_j - Q_{low}(\phi_{j,low}, a_j; \theta_{low}))^2$ with respect to the parameters θ_{low} of lower online network Q_{low}
    Every C steps reset $\widehat{Q}_{high} = Q_{high}$, $\widehat{Q}_{low} = Q_{low}$
  end for
end for

5. Proposed methods for the DMOFJSP

$$\text{Minimize}\begin{cases} TWT = \sum_{i=1}^{n} \max\left(C_{i,n_i} - D_i, 0\right) \cdot Pr_i & (6) \\ 1 \Big/ U_{ave} = 1 \Bigg/ \left(\frac{1}{m} \sum_{k=1}^{m} \frac{\sum_{i=1}^{n} \sum_{j=1}^{n_i} t_{i,j,k} X_{i,j,k}}{\max_i C_{i,n_i} \cdot X_{i,n_i,k}} \right) & (7) \end{cases}$$

- estimated total weighted tardiness *ETWT*
- actual tardiness rate $Tard_a$,
- estimated tardiness rate $Tard_e$
- average machine utilization rate U_{ave}

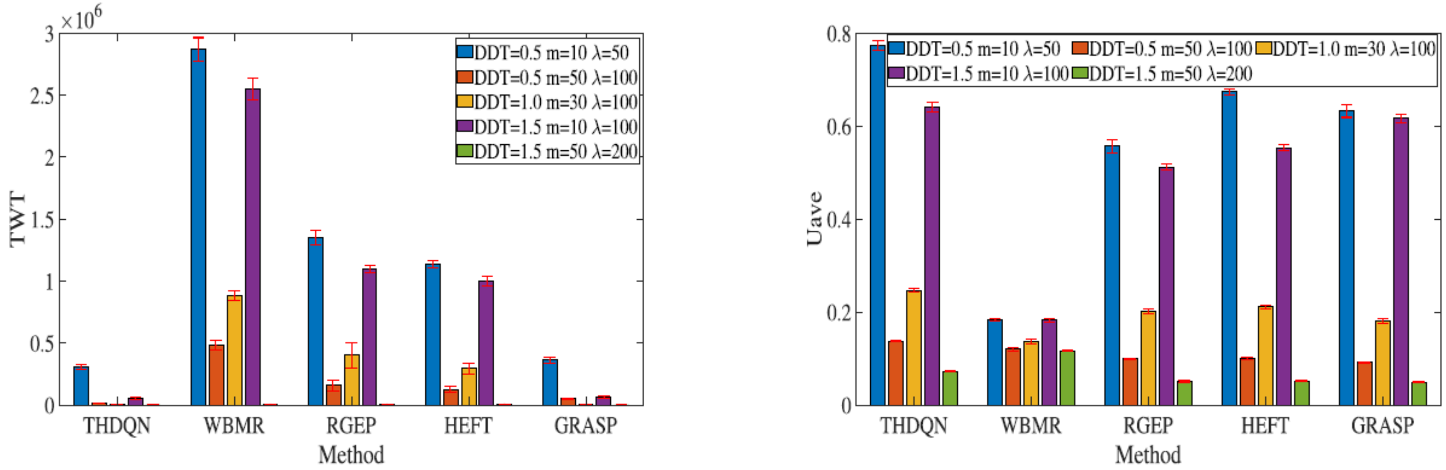
Algorithm 11. Definition of the reward r_t for the state-action pair (s_t, a_t) at each rescheduling point t

---

Input: The current higher goal g_t, the values of goal indicators before and after taking action a_t
Output: The reward r_t for rescheduling point t

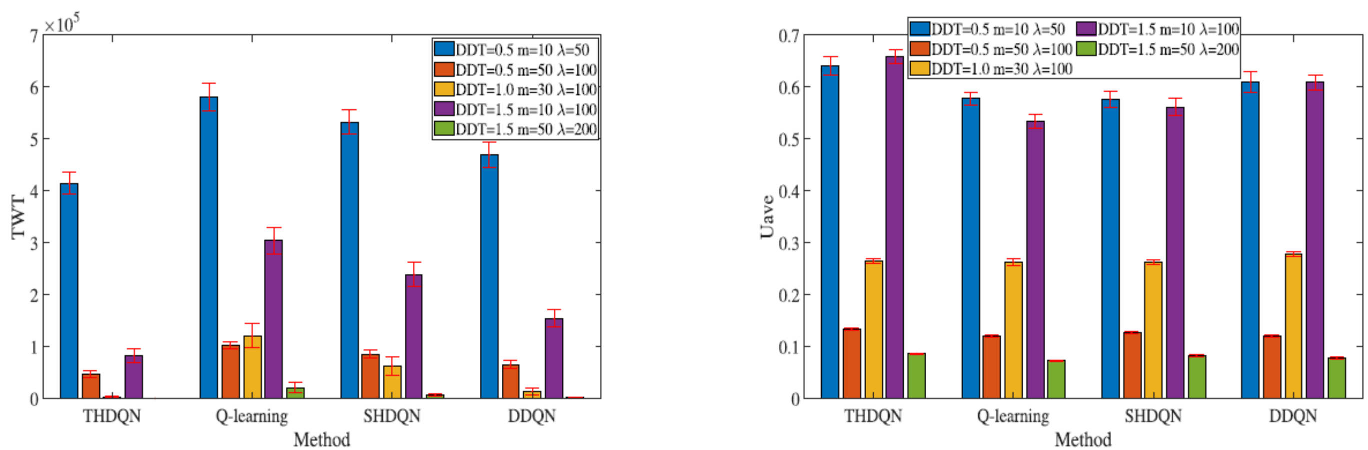
```
if g_t == 1 then
    if ETWT(t+1) < ETWT(t) then
        r_t ← 1
    else if ETWT(t+1) > ETWT(t) then
        r_t ← −1
    else
        r_t ← 0
else if g_t == 2 then
    if Tard_a(t+1) < Tard_a(t) then
        r_t ← 1
    else if Tard_a(t+1) > Tard_a(t) then
        r_t ← −1
    else
        r_t ← 0
else if g_t == 3 then
    if Tard_e(t+1) < Tard_e(t) then
        r_t ← 1
    else if Tard_e(t+1) > Tard_e(t) then
        r_t ← −1
    else
        r_t ← 0
else if g_t == 4 then
    if U_ave(t+1) > U_ave(t) then
        r_t ← 1
    else if U_ave(t+1) > U_ave(t)·0.95 then
        r_t ← 0
    else
        r_t ← −1
End if
Return r_t
```

6. Numerical experiments



(a) TWT (b) U_{ave}

Fig. 4. Average values of two objectives obtained by the THDQN and other existing composite rules on some representative instances.



(a) TWT (b) U_{ave}

Fig. 5. Average values of two objectives obtained by the THDQN and other RL based scheduling methods on some representative instances.

7. Conclusion

- TWT, Uave를 최적화하는 것을 목표로 하는 New Job Insertion과 함께 DMOFJSP(dynamic multi-objective flexible job shop scheduling problem)를 위해 2계층 심층 Q 네트워크(THDQN)를 방식 제안
- 제안된 THDQN은 상위 컨트롤러와 하위 액추에이터를 포함하는 두 개의 DQN 기반 에이전트를 포함
- 각 rescheduling 지점에서 higher DQN은 현재 상태를 기반으로 lower DQN에 대한 임시 최적화 목표를 결정, lower DQN은 high 최적화 목표와 현재 상태 기능을 모두 입력으로 사용하고 주어진 목표를 달성하기 위해 실행 가능한 디스패칭 규칙을 선택
- traning 과정에서 4가지 형태의 보상 기능에 해당하는 4가지 목표가 설계되었으며, 각 목표는 TWT 또는 Uave 지표를 최적화
- 처리되지 않은 작업을 동시에 선택하고 사용 가능한 시스템에 할당하기 위해 6개의 복합 디스패칭 규칙을 개발
- training 하기 위해 이중 DQN 기반 훈련 프레임워크를 제안
- scheduling을 실시간으로 수행할 수 있을 뿐만 아니라 장기적으로 여러 목표 간에 좋은 절충안을 만들 수 있음
- 수치 실험을 통해 제안된 THDQN이 composite dispatching rule, 기존의 잘 알려진 디스패칭 규칙 및 다른 RL 기반 스케줄링 방법보다 훨씬 더 잘 수행됨을 보여줌

7. Conclusion

- 후속 연구는 기계 고장 및 다양한 처리 시간과 같은 보다 불확실한 이벤트를 조사할 예정
- 평균 흐름 시간, 에너지 소비 및 생산 비용과 같은 다른 목표도 다른 목표에 대한 THDQN의 일반성을 검증하기 위해 고려할 가치가 있음
- 한편, 많은 features를 THDQN에 대한 입력으로 간주되며, 그 결합으로 인해 네트워크가 오도될 수 있다는 점에 유의해야 합니다. 따라서 features selection algorithm을 도입하면 심층 Q 네트워크의 성능을 향상시키는 데 유용할 수 있음
- 제안하는 THDQN은 본질적으로 정책을 통해 직접 최적화할 수 없는 가치 기반 방법이므로 DMOFJSP를 해결하기 위해 actor-critic algorithm 및 proximal policy optimization 같은 다른 최신 정책 기반 방법을 적용할 예정


감사합니다.