

# Disassembly Scheduling Problem Based on Reinforcement Learning

①

Deep RL-based energy-aware disassembly planning for end-of-life products with stimuli-activated self-disassembly

②

Dynamic Balancing of U-Shaped Robotic Disassembly Lines Using an Effective Deep RL Approach

③

Modelling and condition-based control of a flexible and hybrid disassembly system with manual and autonomous workstations using RL

④

RL for Hybrid Disassembly Line Balancing Problems

⑤

RL-Based Selective Disassembly Sequence Planning for the End-of-Life Products With Structure Uncertainty

①

**Fully Decentralized Multiagent  
Communication via Causal Inference**

RL training mechanisms under  
high-dimensional/sparse rewards

⑤

**Communication-Efficient Decentralized  
Multi-Agent Reinforcement Learning for  
Cooperative Adaptive Cruise Control**

②

**Dynamic Balancing of U-Shaped Robotic  
Disassembly Lines Using an Effective Deep  
RL Approach**

Heterogeneous Resource Collaboration

③

**Modelling and condition-based control of a  
flexible and hybrid disassembly system with  
manual and autonomous workstations using RL**

④

**RL for Hybrid  
Disassembly Line  
Balancing Problems**

## Deep reinforcement learning-based energy-aware disassembly planning for end-of-life products with stimuli-activated self-disassembly

Di Wang · Jing Zhao · Muyue Han · Lin Li





**This paper was published in:** Journal of Intelligent Manufacturing (IF: 7.4)

**Published Time:** 2024

**Reference:** Wang D, Zhao J, Han M, et al. Deep reinforcement learning-based energy-aware disassembly planning for end-of-life products with stimuli-activated self-disassembly[J]. Journal of Intelligent Manufacturing, 2024: 1-20.







## Contents

01		<b>Problem Description</b>
02		<b>Method Design</b>
03		<b>Experiment Results</b>
04		<b>Conclusion</b>



## Contents

01		<b>Problem Description</b>
02		Method Design
03		Experiment Results
04		Conclusion

# Problem Description





## Challenge:

- End-of-life (EOL) products exhibit **high structural uncertainty** (e.g., missing parts, quality variation), making static disassembly plans ineffective.
- Traditional manual disassembly is labor-intensive, slow, and energy-inefficient, especially for complex products.

## Method:

- Proposes a hybrid system combining **manual workstations and stimuli-activated** self-disassembly (enabled by 4D printing) for non-destructive, parallel part removal.
- Develops a modified DDPG algorithm with parameter embedding to generate Pareto-optimal sequences that balance **profit and energy recovery** under uncertainty, using a compact grid-world product representation.



01		Problem Description
02		Method Design
03		Experiment Results
04		Conclusion

# 1 Problem Definition





$$d_i = 1, \forall i \in M \quad (3)$$

$$\begin{cases} z_{i1} = 1, & c_i(1 - d_i\beta_0) \geq \bar{C}_1 \\ z_{i2} = 1, & \bar{C}_1 > c_i(1 - d_i\beta_0) \geq \bar{C}_2 \\ z_{i3} = 1, & \bar{C}_2 > c_i(1 - d_i\beta_0) \geq \bar{C}_3 \\ z_{i4} = 1, & \bar{C}_3 > c_i(1 - d_i\beta_0) \end{cases} \quad (4)$$

$$\sum_{j=1}^4 z_{ij} = 1, \forall i \in N \quad (5)$$

$$r_{i3} = \sum_{j=1}^{o_i} v_j^c u_j \quad (6)$$

$$R = \sum_{i=1}^n \sum_{k=1}^4 z_{ik} r_{ik} \quad (7)$$

$$DC = C_L \left( \sum_{i=1}^n \tilde{T}_i d_i \sum_{i=1}^n \sum_{j=1}^n \sum_{a=1}^{n-1} \tilde{T}_{ij} d_i d_j x_{ia} x_{j(a+1)} H(a, a+1) \right) + C_S \left( \sum_{i=1}^n \tilde{T}_S (1 - d_i) \right) \quad (8)$$

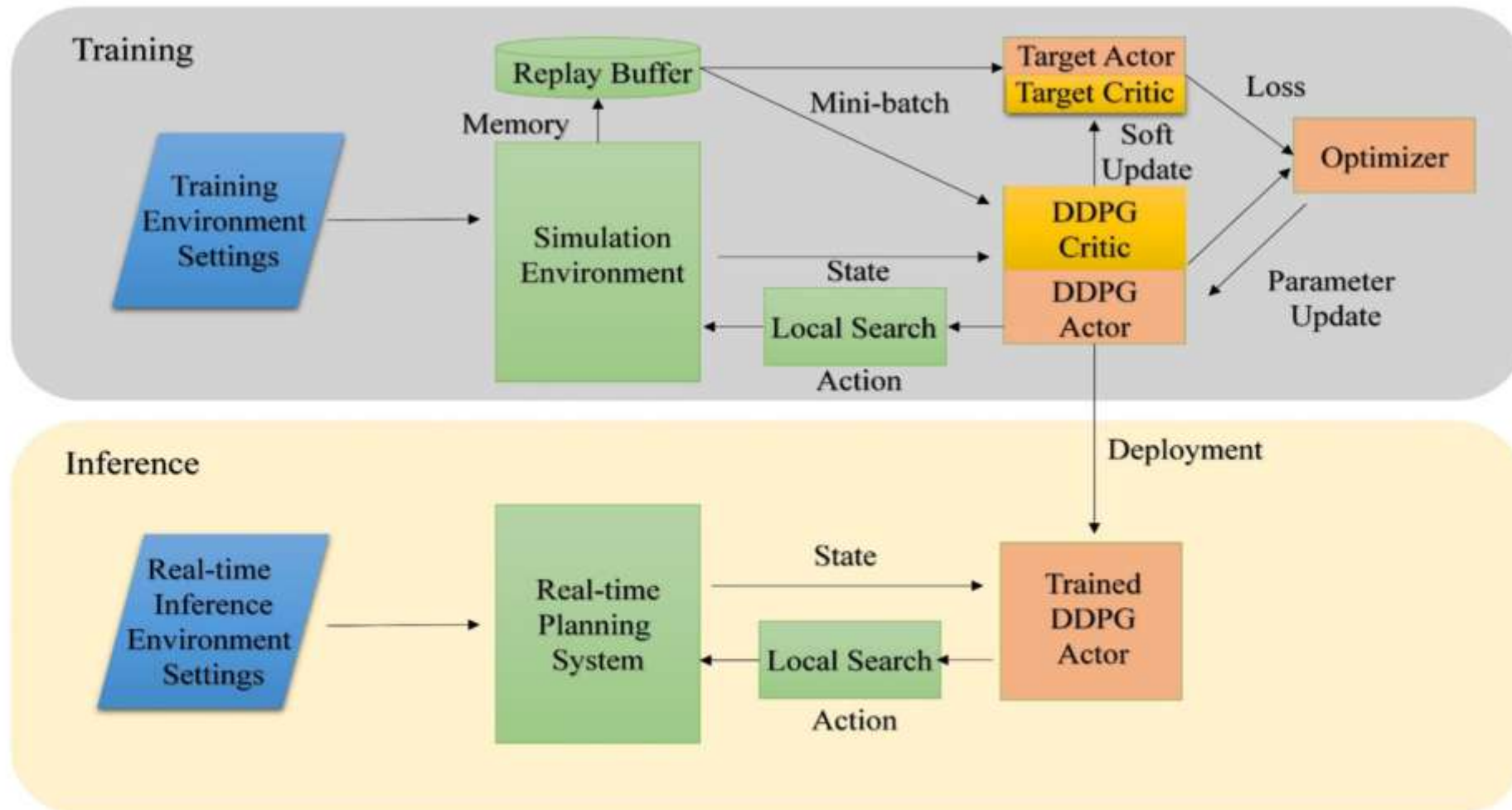
$$RC = \sum_{i=1}^n \sum_{k=1}^4 r_{ik}^c (1 - d_i) \beta_1 + \sum_{i=1}^n \sum_{k=1}^4 r_{ik}^c d_i \quad (9)$$

$$f_1 = R - DC - RC \quad (10)$$

The saved energy of component  $i$  is formulated as Eq. (11).

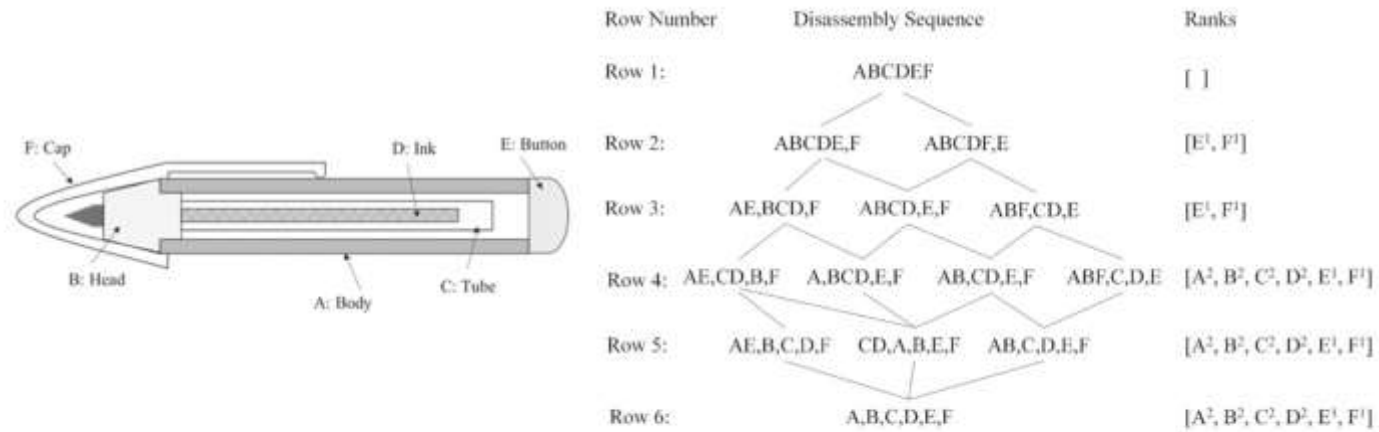
$$E_i = \begin{cases} E_i^e - E_i^s(1 - d_i) & z_{i1} = 1 \\ E_i^e - E_{i2}^c(1 - d_i)\beta_2 - E_{i2}^c d_i - E_i^s(1 - d_i) & z_{i2} = 1 \\ E_i^m - E_{i3}^c(1 - d_i)\beta_2 - E_{i3}^c d_i - E_i^s(1 - d_i) & z_{i3} = 1 \\ -E_i^s(1 - d_i) & z_{i4} = 1 \end{cases} \quad (11)$$

# Method Design



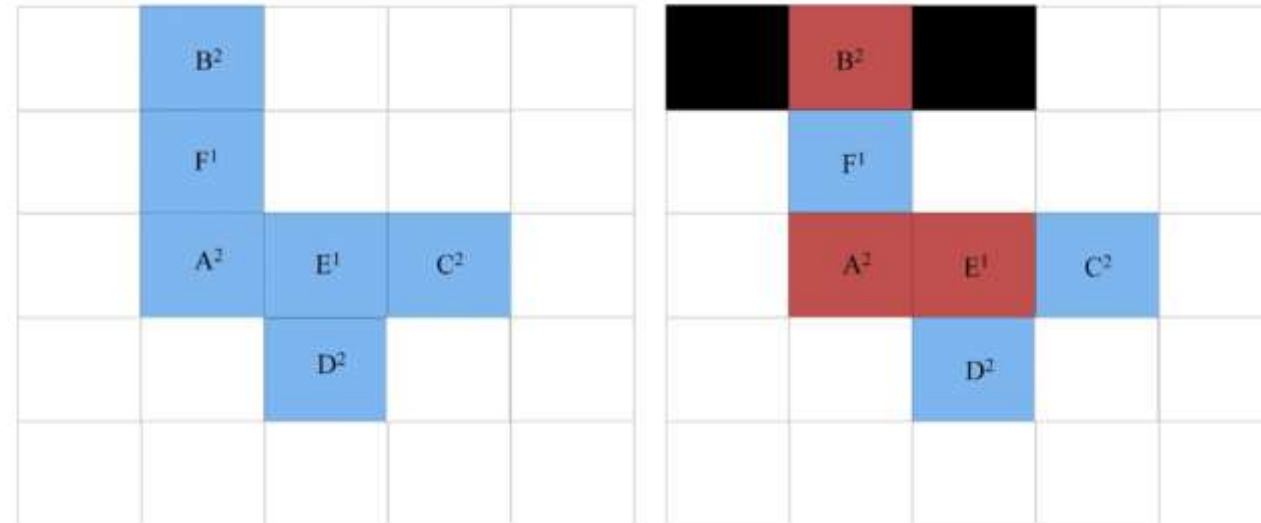
**Fig. 3** The training and real-time inference frameworks of the proposed DDPG method

# Method Design



**Fig. 4** Ballpoint pen example (left) and the generation of grid world (right) from the undirected state-diagram graph representation model

**Fig. 5** Grid-world representations of a Ballpoint pen product, showing the manual disassembly scheme only (left) and the hybrid disassembly scheme (right)



# Method Design

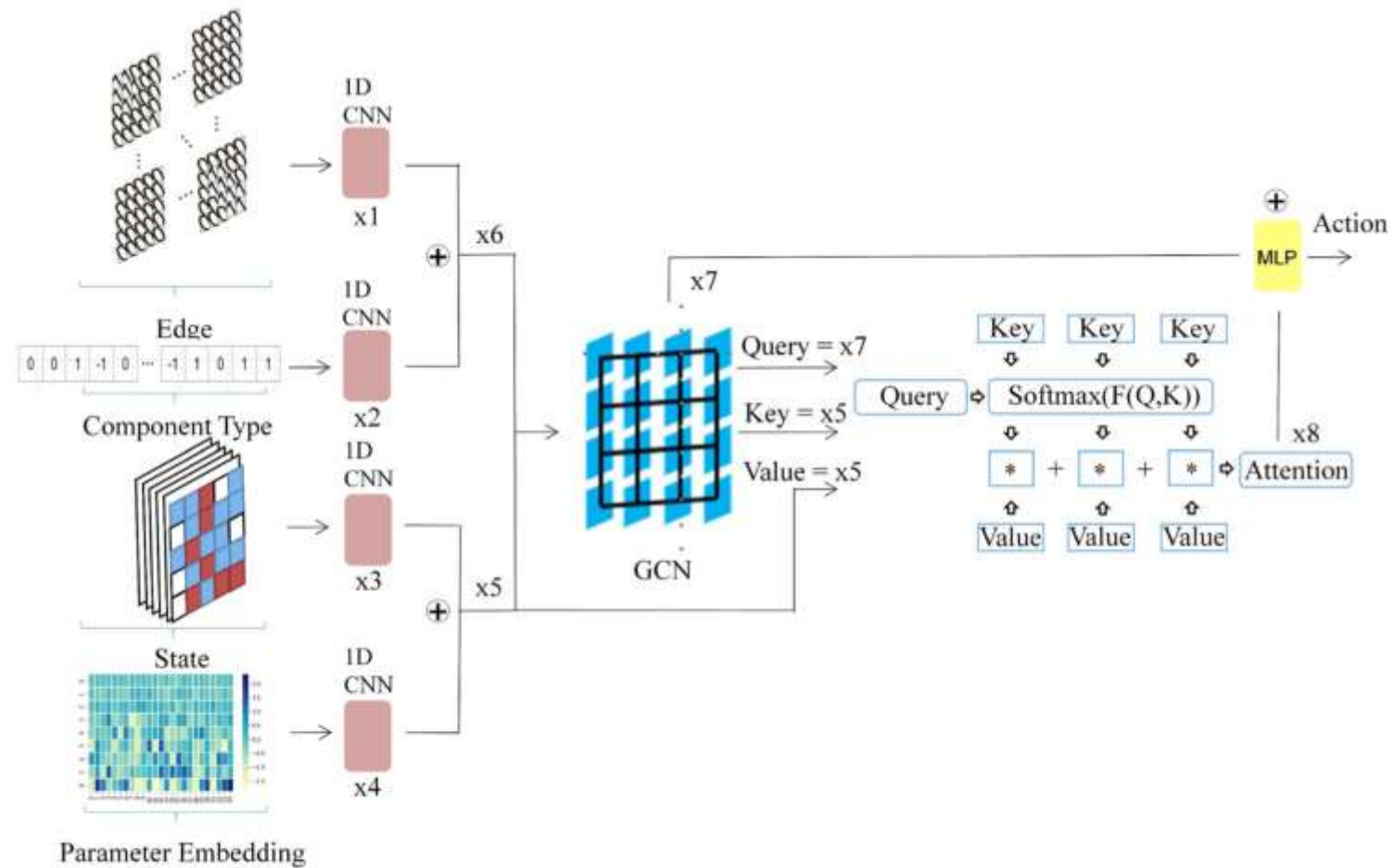






Fig. 6 The architecture of the actor network with the GCN encoder



01		Problem Description
02		Method Design
03		Experiment Results
04		Conclusion

# Experiment Results

## Dataset

Fig. 7 Disassembly instructions for the LCD Television

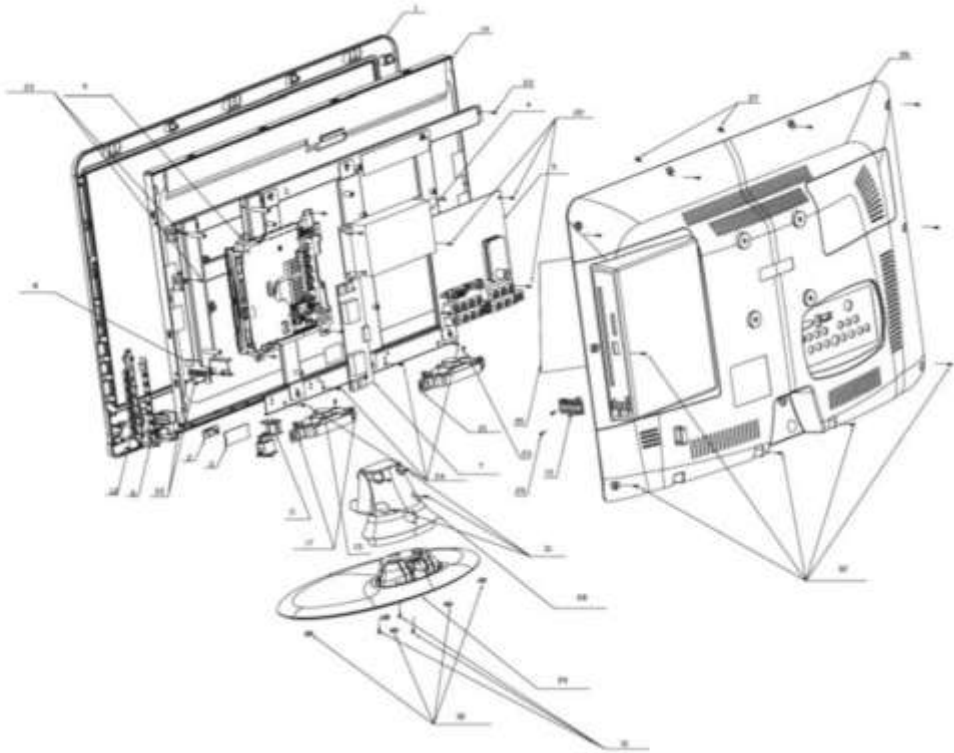
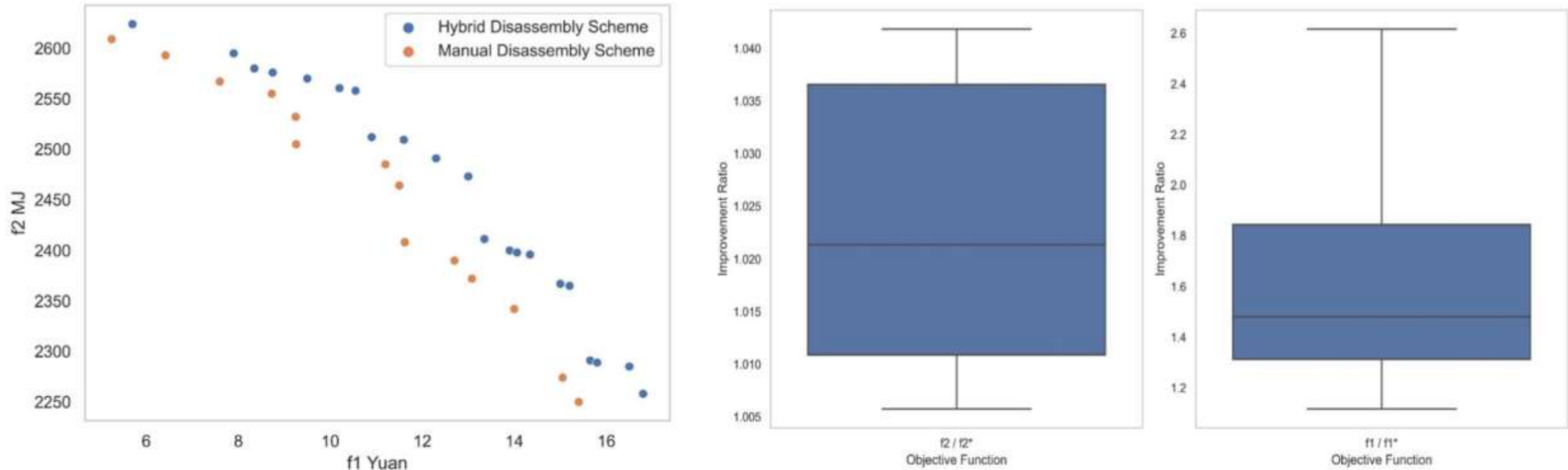


Table 1 General information of the LCD TV

Components	Manual disassembly time (seconds)	Residual value (Yuan)	Mass (g)
Metal fixing plate	18.0	0.2970	15.0
Metal washer 1	2.4	0.0660	10.0
Metal washer 2	2.4	0.0660	10.0
Top metal support	21.0	0.4950	25.0
Cylindrical support 1	6.0	0.2400	30.0
Cylindrical support 2	6.0	0.1600	20.0
Toughened glass seat	13.8	0.2380	150.0
Steel plate	12.0	0.3300	50.0
Rubber gasket	6.0	0.0200	20.0
Control button	4.8	0.0100	9.2
Power switch	4.8	0.0100	5.0
Side loudspeaker	21.0	0.6000	152.0
Control receiver board	6.0	0.4000	3.0
Positive loudspeaker	15.0	0.3071	77.8
Power supply board	42.0	0.6466	118.0
Main board	42.0	0.7908	196.0
Metal board	35.4	1.2078	183.0
Metal mounting plate	109.2	4.2174	639.0
Surface frame	73.8	1.1000	270.8
LCD screen	85.2	9.6684	2900.0
Back cover	99.0	1.7904	723.8
Cover plate	1.8	0.2280	25.0
Support	2.4	0.0169	15.6

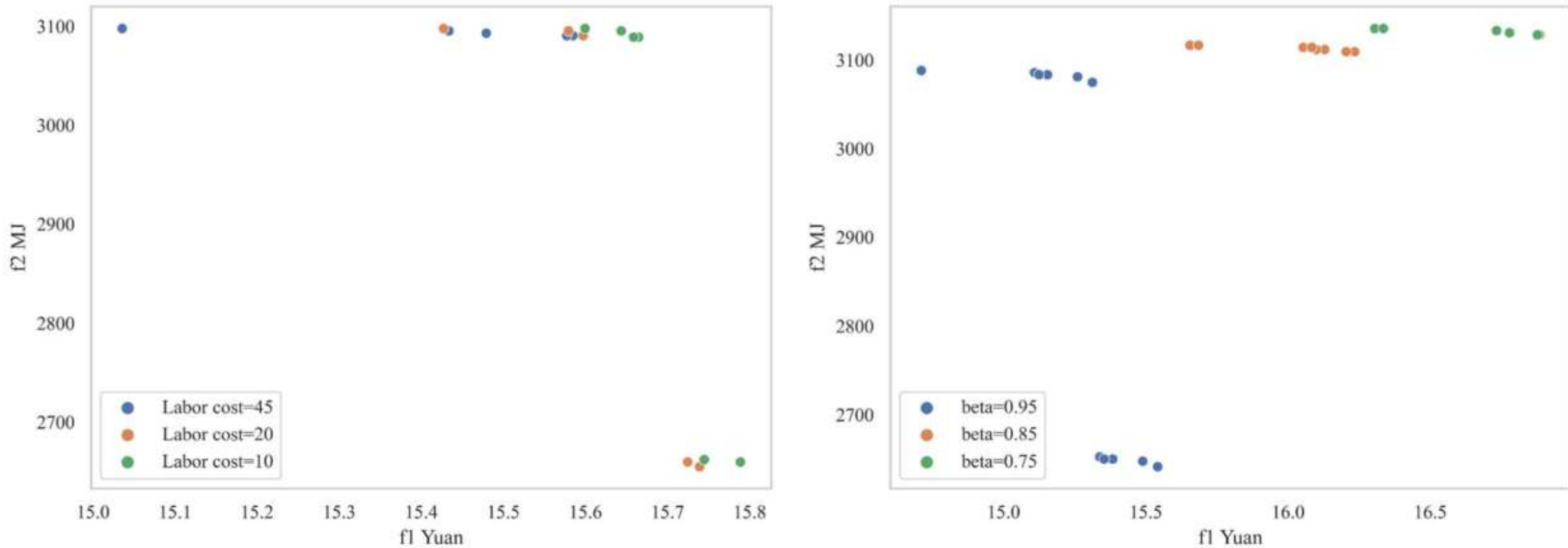
# Experiment Results



**Fig. 8** Comparison of Pareto frontiers between hybrid and manual disassembly scheme (left), and the recovery profit and energy improvement ratio achieved with the hybrid disassembly scheme (right)



# Experiment Results



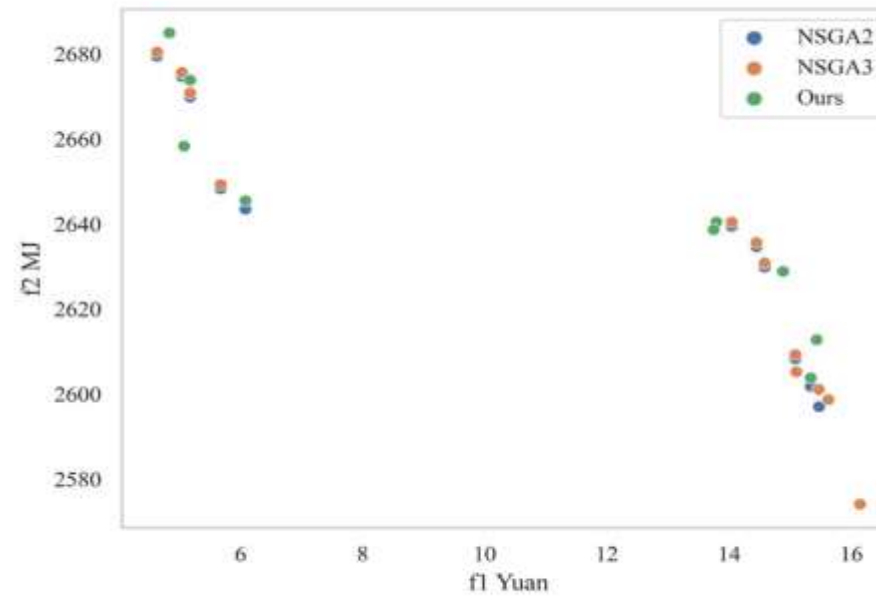
**Fig. 9** Sensitivity analysis of the influence of labor cost (left) and disassembly damage (right) on Pareto frontiers



# Experiment Results

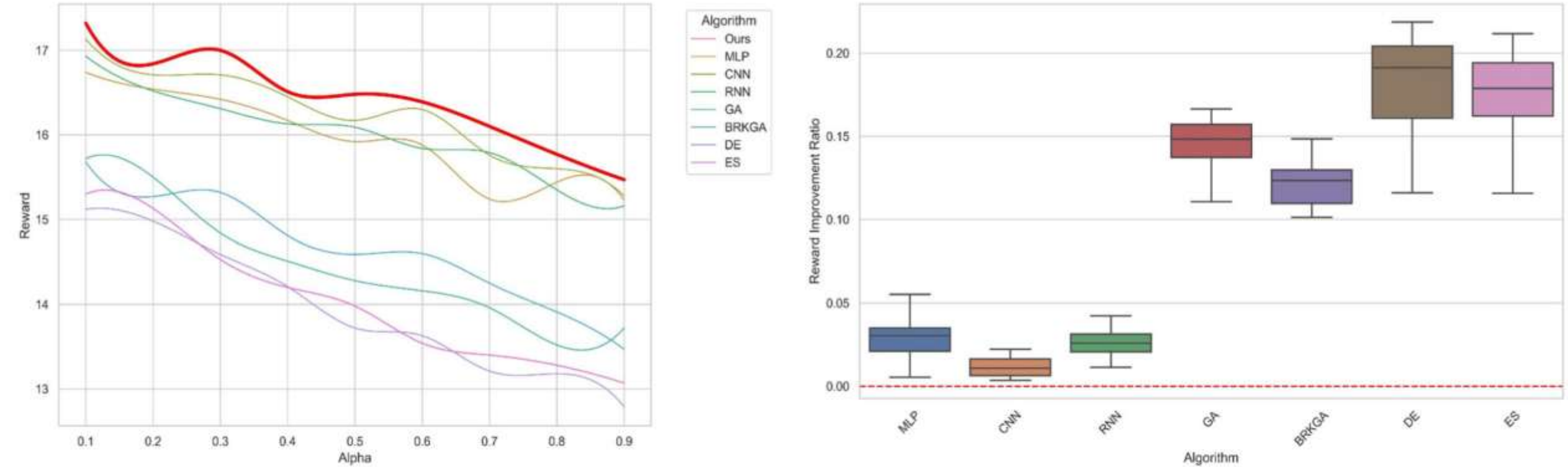
**Table 2** Performance evaluation of algorithms on the Pareto frontiers

Algorithm	Number of nondominated solutions	CPU time (seconds)
NSGA2	11	258.64
NSGA3	12	213.75
Ours	6	0.059



**Fig. 10** Comparison of different algorithms' Pareto frontiers

# Experiment Results



**Fig. 11** Comparison of collected rewards for different algorithms across varying  $\alpha$  values (left), and the reward improvement ratio of our algorithm over benchmarks across  $\alpha$  values (right)

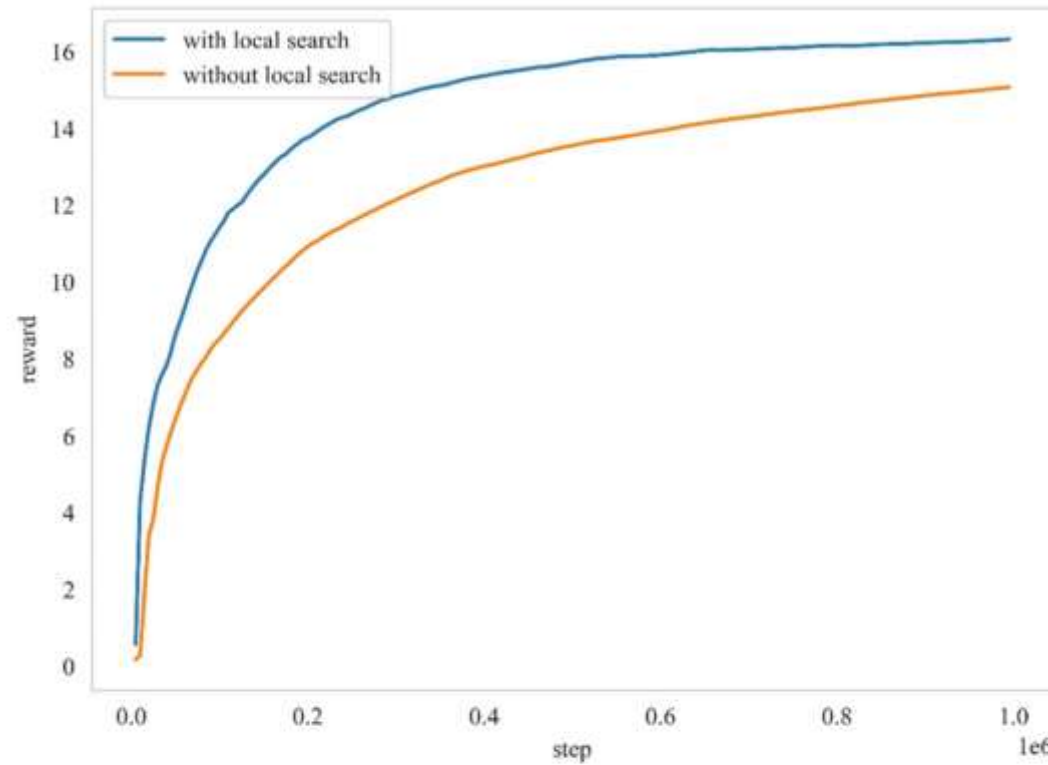
# Experiment Results

**Table 3** Comparison between our proposed method and state-of-art heuristic methods

$\alpha$	0.1	0.3	0.5	0.7	0.9	CPU time (seconds)
Algorithm	Reward	Reward	Reward	Reward	Reward	
Ours	<b>17.32</b>	<b>17.00</b>	<b>16.48</b>	<b>16.10</b>	<b>15.47</b>	<b>0.059</b>
MLP (Kutschenreiter-Praszkiewicz, 2020)	16.74	16.42	15.92	15.24	15.23	0.055
CNN (Lee et al., 2022)	17.13	16.71	16.17	15.76	15.28	0.071
RNN (Jeunet et al., 2021)	16.93	16.31	16.09	15.79	15.16	0.088
GA (Giudice & Fargione, 2007)	15.72	14.84	14.28	13.96	13.72	1212.9
BRKGA (Biajoli et al., 2019)	15.68	15.32	14.59	14.25	13.47	1865.6
DE (Wang et al., 2020)	15.12	14.59	13.72	13.21	12.79	654.13
ES (Chien et al., 2014)	15.30	14.53	13.98	13.40	13.07	1503.5

Bold values indicate the superior results obtained by our proposed method

# Experiment Results



**Fig. 12** Received rewards during the training with and without local search in our algorithm

# Experiment Results

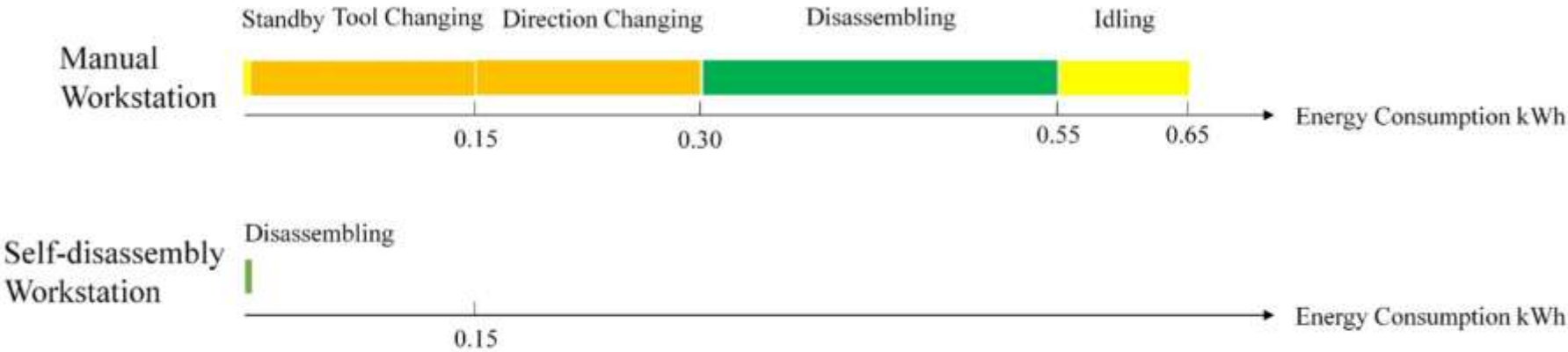






Fig. 13 Energy consumption comparison between manual and self-disassembly workstations

**Table 4** Supplementary results for comparison between our proposed method and state-of-art heuristic methods

$\alpha$	0.2	0.4	0.6	0.8
Algorithm	Reward	Reward	Reward	Reward
Ours	<b>16.84</b>	<b>16.51</b>	<b>16.39</b>	<b>15.77</b>
MLP (Kutschenreiter-Praszkiewicz, 2020)	16.54	16.17	15.88	15.44
CNN (Lee et al., 2022)	16.71	16.45	16.30	15.60
RNN (Jeunet et al., 2021)	16.52	16.13	15.84	15.35
GA (Giudice & Fargione, 2007)	15.50	14.51	14.16	13.52
BRKGA (Biajoli et al., 2019)	15.27	14.81	14.60	13.91
DE (Wang et al., 2020)	14.98	14.21	13.63	13.18
ES (Chien et al., 2014)	15.13	14.20	13.54	13.28



01		Problem Description
02		Method Design
03		Experiment Results
04		Conclusion

①

The proposed hybrid disassembly system—combining manual workstations and stimuli-activated self-disassembly (enabled by 4D printing)—significantly improves disassembly efficiency, profit recovery, and energy savings.

The modified DDPG algorithm with parameter embedding, grid-world product representation, and a custom loss function enables real-time, adaptive planning under multiple uncertainties (e.g., missing parts, quality variation, product type).

②

Develop a multi-agent RL system to jointly optimize line balancing and human–robot collaboration in hybrid disassembly lines.

Investigate energy characteristics of different shape-memory materials to refine self-disassembly modeling.

Integrate Pareto optimality directly into policy learning (e.g., via multi-objective DDPG) to better approximate the true Pareto frontier without linear scalarization.

## Dynamic Balancing of U-Shaped Robotic Disassembly Lines Using an Effective Deep Reinforcement Learning Approach

Kaipu Wang , Yibing Li , Jun Guo , Liang Gao , Senior Member, IEEE, and Xinyu Li , Member, IEEE

**This paper was published in:** IEEE Transactions on Industrial Informatics (IF: 9.9)





**Published Time:** 2024

**Reference:** Wang K, Li Y, Guo J, et al. Dynamic balancing of U-shaped robotic disassembly lines using an effective deep reinforcement learning approach[J]. IEEE Transactions on Industrial Informatics, 2024, 20(4): 6855-6865.









## Contents

01		<b>Problem Description</b>
02		<b>Method Design</b>
03		<b>Experiment Results</b>
04		<b>Conclusion</b>



## Contents

01		<b>Problem Description</b>
02		Method Design
03		Experiment Results
04		Conclusion

# Problem Description





## Challenge

- Real-world disassembly lines face dynamic disturbances (e.g., corrosion, missing parts, deformation), causing disassembly times to change unpredictably and rendering static balancing solutions ineffective.
- Traditional optimization methods (e.g., MILP, metaheuristics) are computationally slow and require full replanning when conditions change, making them unsuitable for real-time control.

## Method

- Proposes a **deep Q-network (DQN)** approach that treats task assignment as a Markov Decision Process, enabling fast, adaptive rebalancing under uncertainty.
- Designs **8 informative state features** and **10 heuristic-based actions** (e.g., shortest processing time, fewest predecessors) to guide intelligent, feasible decisions without enumerating all task assignments.



01		Problem Description
02		Method Design
03		Experiment Results
04		Future Work

# Method Design

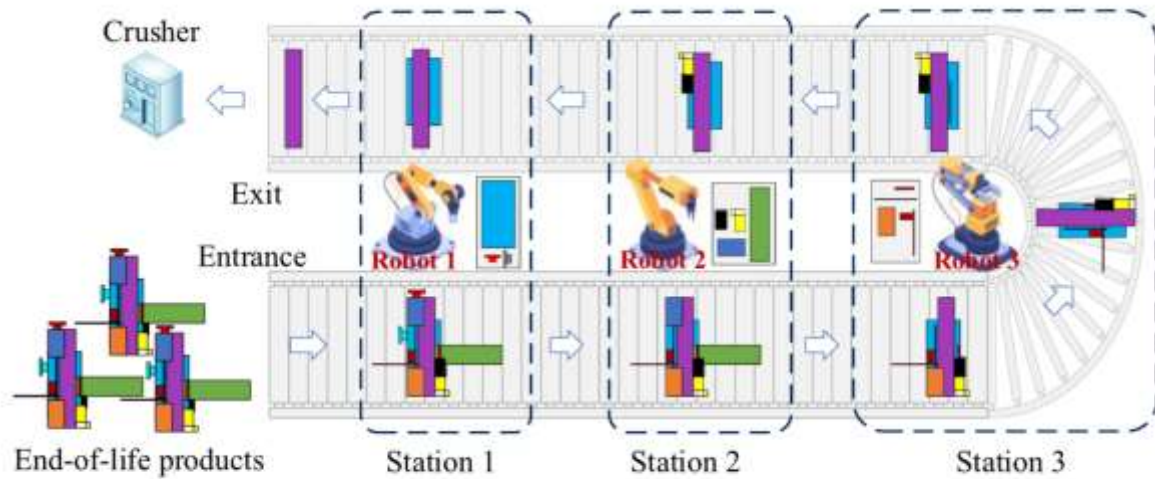


Fig. 1. Layout of the U-shaped robotic disassembly line.

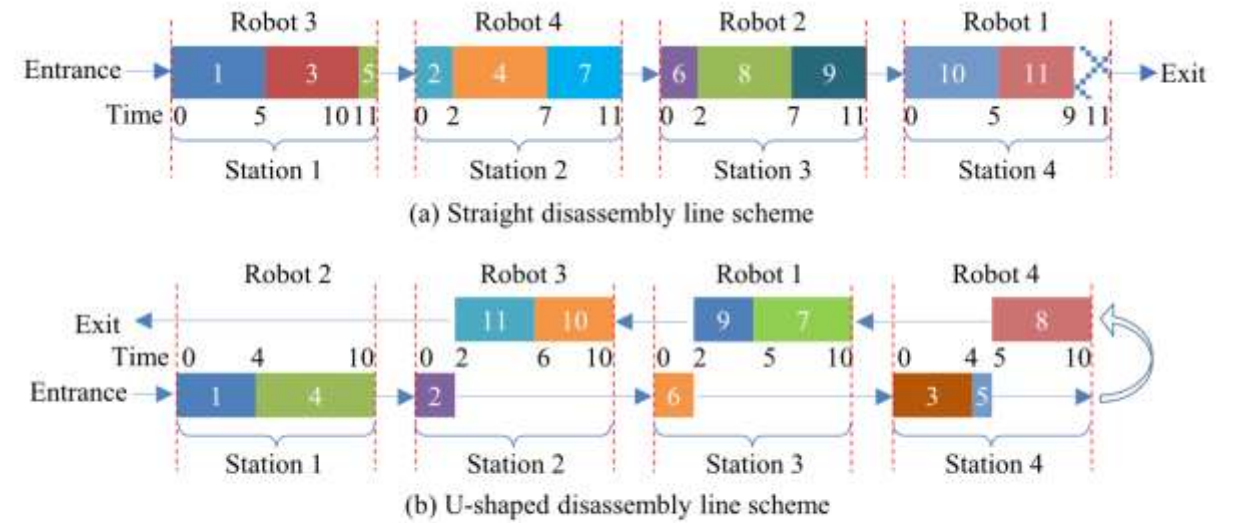


Fig. 2. Gantt chart of the disassembly scheme of the two layout forms.

# Method Design

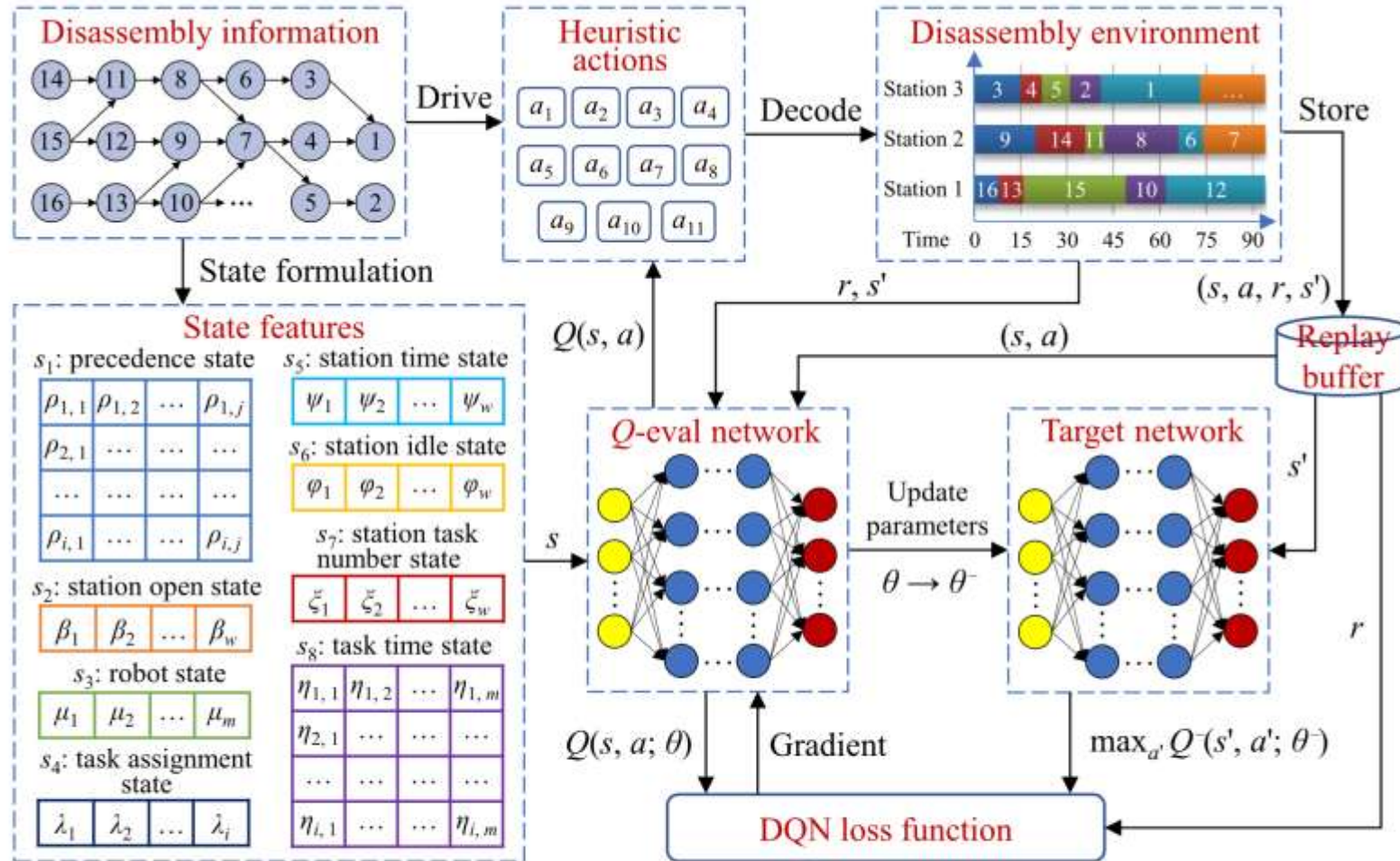


Fig. 3. Framework of the proposed DQN.



TABLE II

STATE FEATURES OF THE DISASSEMBLY ENVIRONMENT

No.	Expression	State description
$s_1$	$\rho_{ij} = \sum_{w \in W} (1 - x_{iw}) p_{ij}, \forall i, j \in I$	Precedence relationship state
$s_2$	$\beta_w = y_w, \forall w \in W$	Station open state
$s_3$	$\mu_m = \sum_{w \in W} z_{mw}, \forall m \in M$	Robot state
$s_4$	$\lambda_i = \sum_{w \in W} x_{iw}, \forall i \in I$	Task assignment state
$s_5$	$\psi_w = \sum_{m \in M} \sum_{i \in I} x_{iw} z_{mw} t_{im} / \sum_{m \in M} \sum_{i \in I} t_{im}, \forall w \in W$	Station time state
$s_6$	$\varphi_w = 1 - \sum_{m \in M} \sum_{i \in I} x_{iw} z_{mw} t_{im} / \sum_{m \in M} \sum_{i \in I} t_{im}, \forall w \in W$	Station idle state
$s_7$	$\xi_w = \sum_{i \in I} x_{iw} /  I , \forall w \in W$	Station task number state
$s_8$	$\eta_{im} = \sum_{w \in W} (1 - x_{iw}) t_{im} / \sum_{m \in M} \sum_{i \in I} t_{im}, \forall i \in I, m \in M$	Task time state

$$r_k = \int_{\tau=t_{k-1}}^{t_k} \delta_k(\tau) d\tau \quad (27)$$

$$\delta_k(\tau) = \left( T_C^* - \min_{m \in M} \left\{ \sum_{i \in I} x_{iw} z_{mw} t_{im} \right\} \right) / T_C^* \quad (28)$$

TABLE III

HEURISTIC ACTIONS FOR THE DISASSEMBLY SYSTEM

No.	Expression	Action description	Symbol
$a_1$	$i = \arg \max_{i \in Set, m \in M} \{t_{im}\}$	Select the task with the longest disassembly time	TLT
$a_2$	$i = \arg \min_{i \in Set, m \in M} \{t_{im}\}$	Select the task with the shortest disassembly time	TST
$a_3$	$i = \arg \max_{i \in Set} \{t_{im} + \sum_{m \in M} \sum_{j \in I} x_{jw} z_{mw} t_{jm}\}$	Select the task that maximizes the station time	TXT
$a_4$	$i = \arg \min_{i \in Set} \{t_{im} + \sum_{m \in M} \sum_{j \in I} x_{jw} z_{mw} t_{jm}\}$	Select the task that minimizes the station time	TNT
$a_5$	$i = \arg \max_{i \in Set} \{n_i^p\}$	Select the task with the maximum number of predecessors	TXP
$a_6$	$i = \arg \min_{i \in Set} \{n_i^p\}$	Select the task with the minimum number of predecessors	TNP
$a_7$	$i = \arg \max_{i \in Set} \{n_i^s\}$	Select the task with the maximum number of successors	TXS
$a_8$	$i = \arg \min_{i \in Set} \{n_i^s\}$	Select the task with the minimum number of successors	TNS
$a_9$	$i = \arg \max_{i \in Set} \{T_C^* - t_{im} - \sum_{m \in M} \sum_{j \in I} x_{jw} z_{mw} t_{jm}\}$	Select the task that keeps the station time away from the $T_C^*$	TAT
$a_{10}$	$i = \arg \min_{i \in Set} \{T_C^* - t_{im} - \sum_{m \in M} \sum_{j \in I} x_{jw} z_{mw} t_{jm}\}$	Select the task that keeps the station time close to the $T_C^*$	TCT
$a_{11}$	$i, \forall i \in Set$	Select a task randomly	RAT

# Method Design

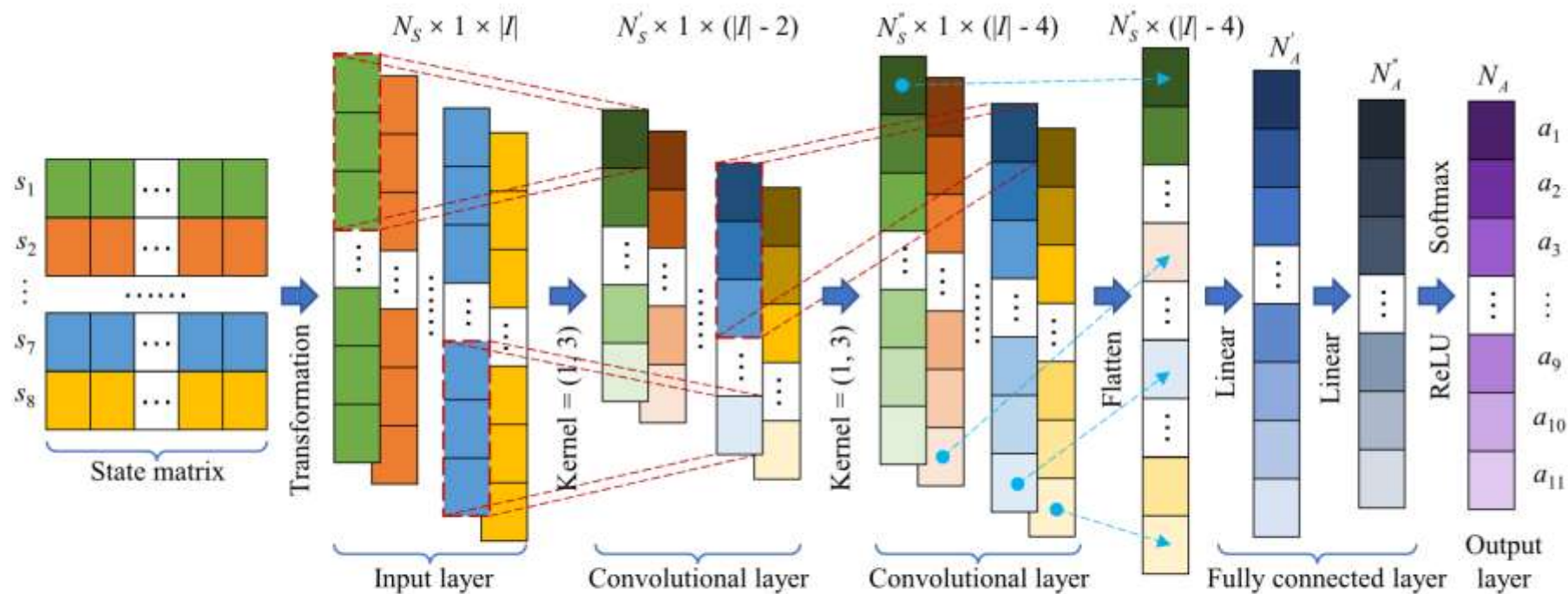






Fig. 4. Neural network of DQN.





01		Problem Description
02		Method Design
03		Experiment Results
04		Conclusion

# Experiment Results

## 1 Cellphone Dataset

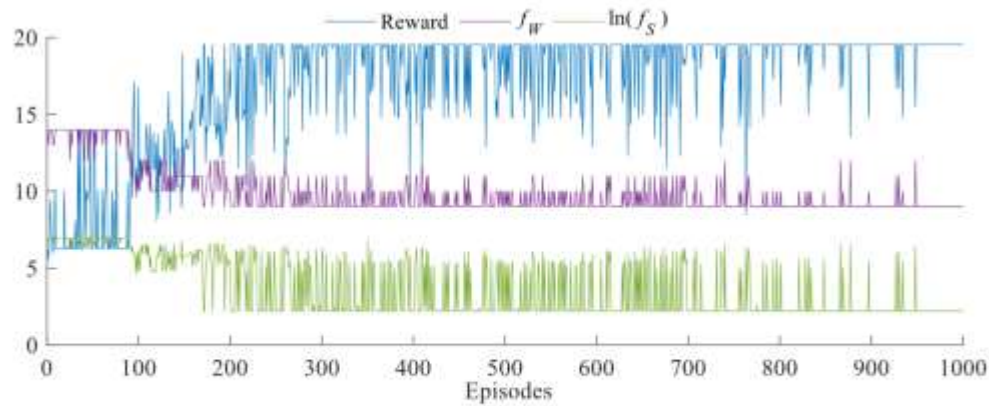


Fig. 5. Training process of DQN on the cell phone disassembly case.

$$f_S = \sum_{w \in W} (T_C - T_w)^2$$

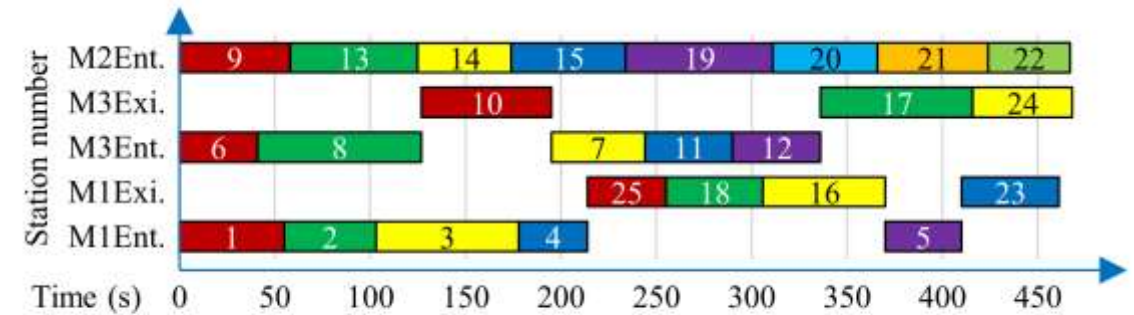


Fig. 6. Gantt chart of the disassembly scheme of case 1.

# Experiment Results

## 2 Laptop Dataset

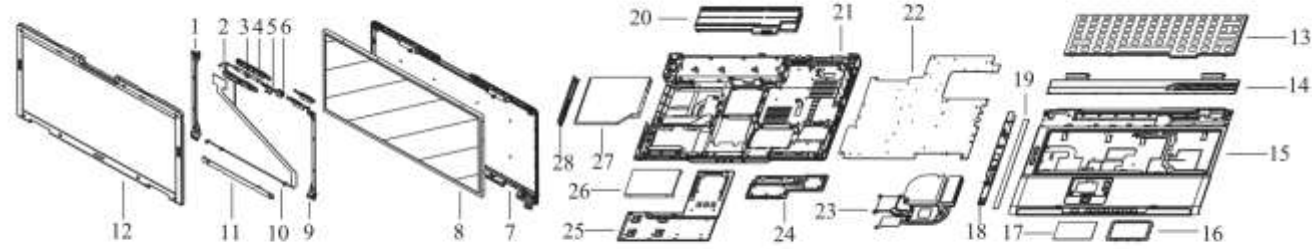


Fig. 8. Exploded view of the laptop.

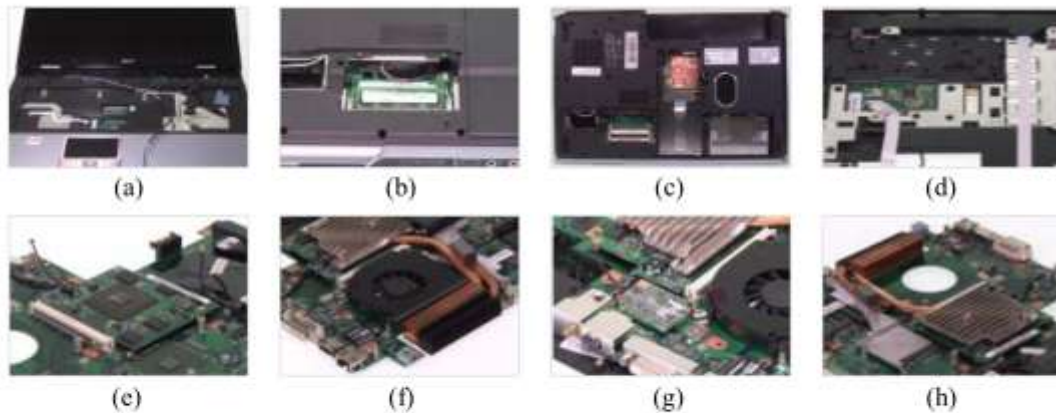


Fig. 7. Some components of the laptop.

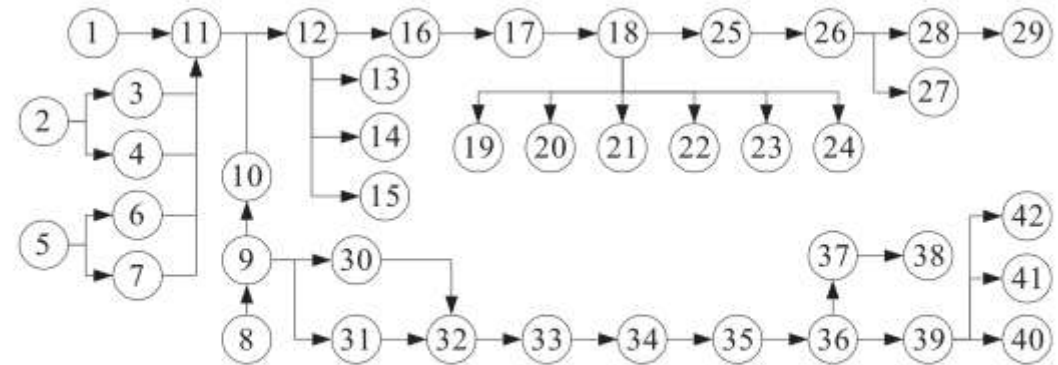


Fig. 9. Precedence relationships among the 42 disassembly tasks.

# Experiment Results

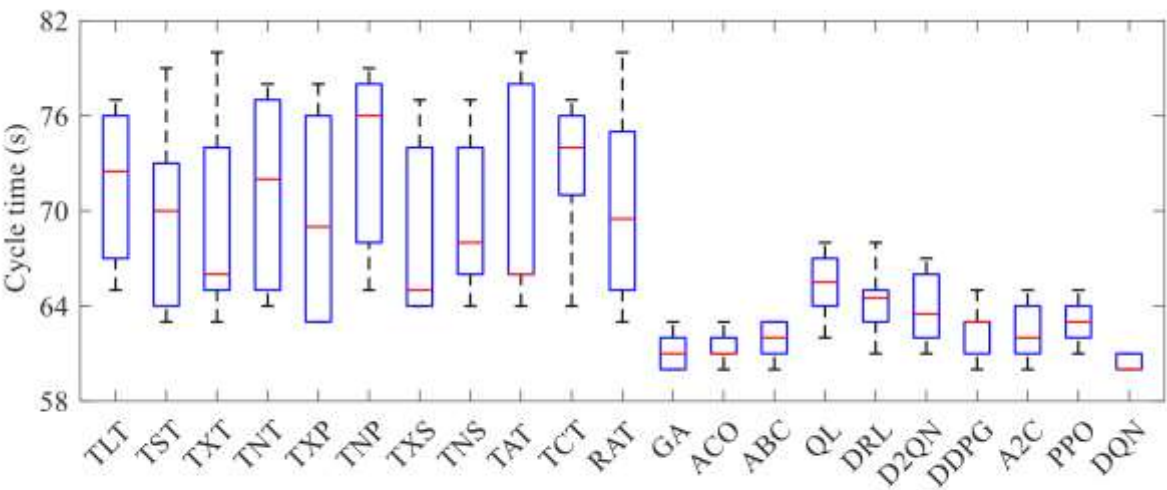


Fig. 10. Boxplot of the 21 algorithms.

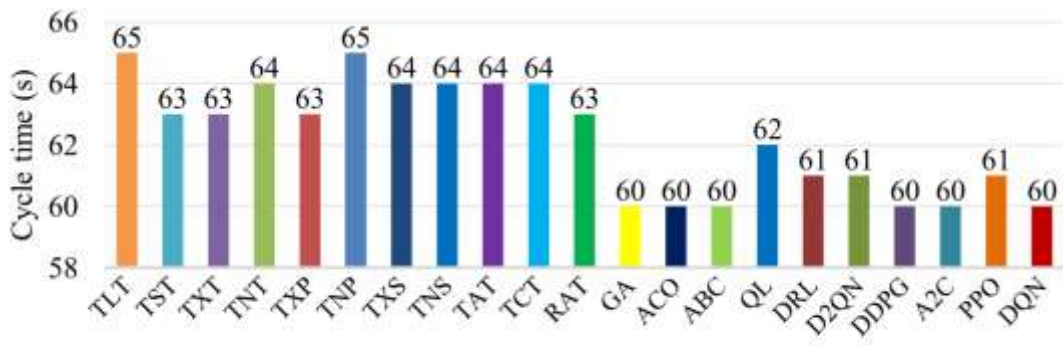


Fig. 11. Best cycle time of each algorithm.

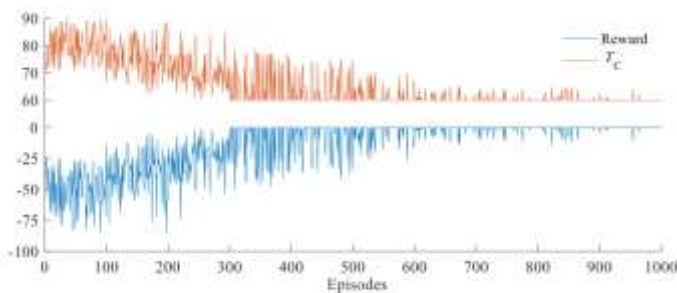


Fig. 12. Training process of cycle time and reward.

TABLE VI  
RESULTS OF THE NEW PROBLEM

Algorithm	GA	ACO	ABC	QL	DRL	D2QN	DDPG	A2C	PPO	DQN
$T_c$	62	63	62	67	67	66	65	66	66	64
Running time	83.52	96.35	77.89	0.039	0.048	0.053	0.059	0.051	0.054	0.042

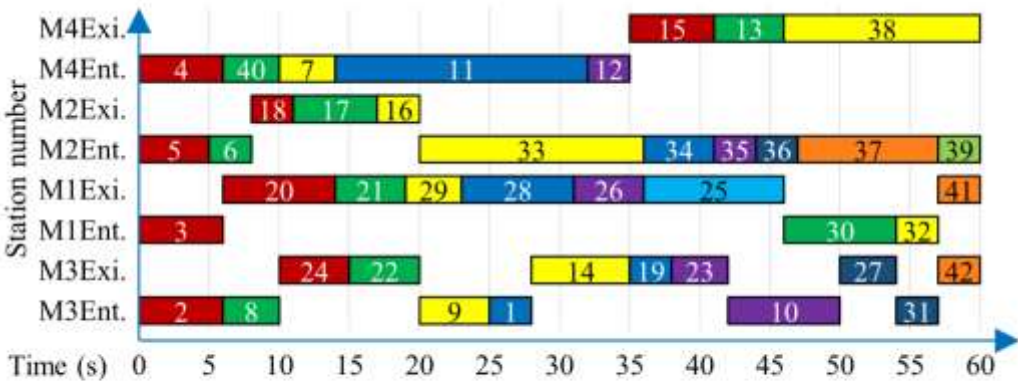






Fig. 13. Gantt chart of the optimal strategy  $\pi^*$ .



01		Problem Description
02		Method Design
03		Experiment Results
04		Conclusion

# Conclusion

①

The proposed DQN with 8 state features and 10 heuristic-based actions effectively minimizes cycle time and achieves optimal task assignment for U-shaped robotic disassembly lines.

In a real laptop disassembly case, the method achieves the optimal cycle time of 60 s, with full robot utilization and zero idle time.

Under dynamic disturbances (e.g., corrosion, missing parts), the trained DQN replans in  $<0.05$  s, far outpacing metaheuristics in responsiveness.

②

Extend the DRL framework to other line configurations: linear, two-sided, parallel, or mixed-model disassembly/assembly lines.

Apply the approach to broader combinatorial optimization problems, such as sequence planning or multi-objective balancing.

Incorporate more realistic uncertainties, such as robot failures, tool wear, or variable product mix, to enhance robustness in industrial settings.



## Modelling and condition-based control of a flexible and hybrid disassembly system with manual and autonomous workstations using reinforcement learning

Marco Wurster · Marius Michel · Marvin Carl May · Andreas Kuhnle · Nicole Stricker · Gisela Lanza





**This paper was published in:** Journal of Intelligent Manufacturing (IF: 7.4)

**Published Time:** 2022

**Reference:** Wurster M, Michel M, May M C, et al. Modelling and condition-based control of a flexible and hybrid disassembly system with manual and autonomous workstations using reinforcement learning[J]. Journal of intelligent manufacturing, 2022, 33(2): 575-591.







## Contents

01		<b>Problem Description</b>
02		<b>Method Design</b>
03		<b>Experiment Results</b>
04		<b>Conclusion</b>





## Contents

01		<b>Problem Description</b>
02		Method Design
03		Experiment Results
04		Conclusion

# Problem Description





## Challenge:

Disassembly systems in remanufacturing face high uncertainty due to the **variable condition, type, and quality of returned products**. Integrating autonomous workstations into production increases flexibility but introduces unpredictable failures, making conventional planning and control methods inadequate for hybrid systems that combine manual and autonomous stations.

## Method:

A hybrid disassembly factory model is established using Disassembly **Petri Nets** to represent dynamic process states and failure probabilities. A Deep Q-Network–based reinforcement learning controller is developed to perform real-time, condition-based dispatching of disassembly tasks, jointly optimizing resource utilization, makespan, and operation reliability.



01		Problem Description
02		Method Design
03		Experiment Results
04		Conclusion

# Method Design

## 1 Disassembly Petri Net

A **Petri** net is a directed graph with two different types of nodes: places and transitions. A place describes a state and is represented by a circle. A transition describes a process and is represented by a bar. Places and transitions are connected by directed edges. An edge never connects two places or two transitions, but always a place with a transition or vice versa. Dynamic systems can be represented by moving so-called tokens through the system. Tokens occupy places and stand by their position for the current state of the system. By “firing” the transitions, the tokens are moved through the system and thus model the dynamic behavior of a system. (Reisig 2013)

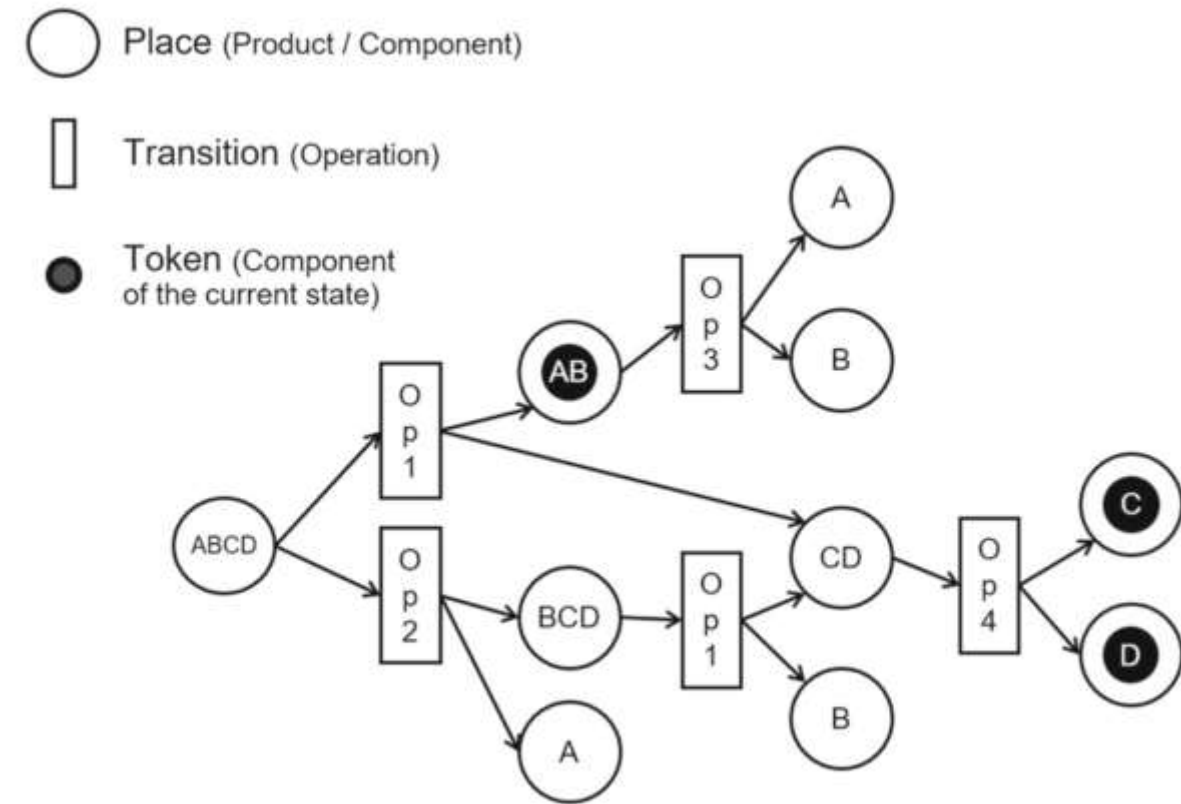


Fig. 1 Disassembly Petri Net of an assembly including 4 components

## 1 Disassembly Petri Net

A disassembly **Petri** net is defined as 6-tuples:

$$DPN = (P, T, I, O, m_0, \rho)$$

with

1. Let  $P = \{p_i\}$  be a finite set of places,  $i = 1, \dots, m$ . Here  $p_1$  is the root representing the product and has no incoming edges. Let the subset of  $P' \subset P$  of the set of places be the set of all places without outgoing edges. These are called leaves and represent the components. The remaining places correspond to subassemblies.
2. Let  $T = \{t_j\}$  be a finite set of transitions,  $j = 1, \dots, n$ ; A transition corresponds to a disassembly operation. Each transition has at least one incoming and one outgoing edge.
3. Let  $I : P \times T \rightarrow \{0,1\}$  be an input function defining the set of directed edges from  $P$  to  $T$ . Let  $I_{ij} = 1$ , if there is an edge from location  $p_i$  to transition  $t_j$ ; otherwise  $I_{ij} = 0$ .
4. Let  $O : T \times P \rightarrow \{0,1\}$  be an output function describing the directed edges from  $T$  to  $P$ . Let  $O_{ij} = 1$ , if  $p_i$  is the initial point of transition  $t_j$ ; otherwise  $O_{ij} = 0$ .
5. Let  $m_0$  be the initial mark with  $m_0(p_1) = 1$  and  $m_0(p_i) = 0 \forall p_i \in P \setminus \{p_1\}$
6. Let  $\rho : T \rightarrow [0,1]$  be a probability value indicating the success probability of a transition.

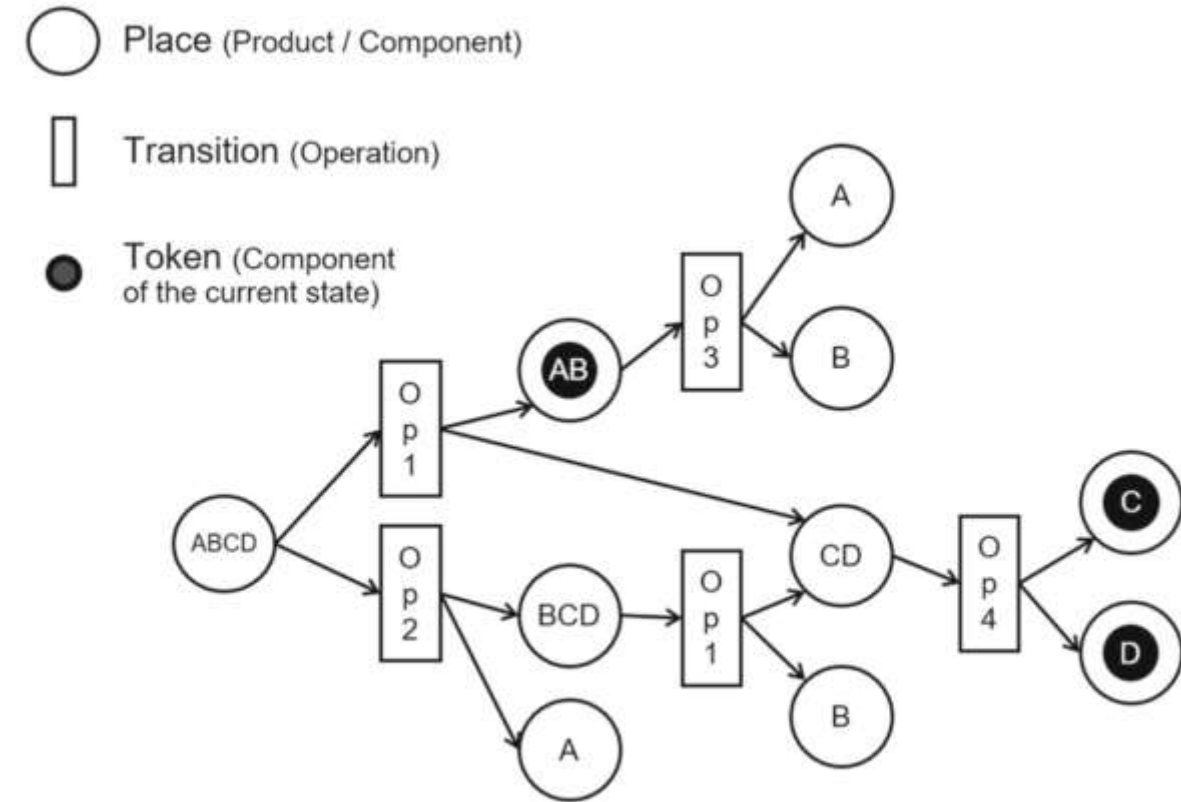
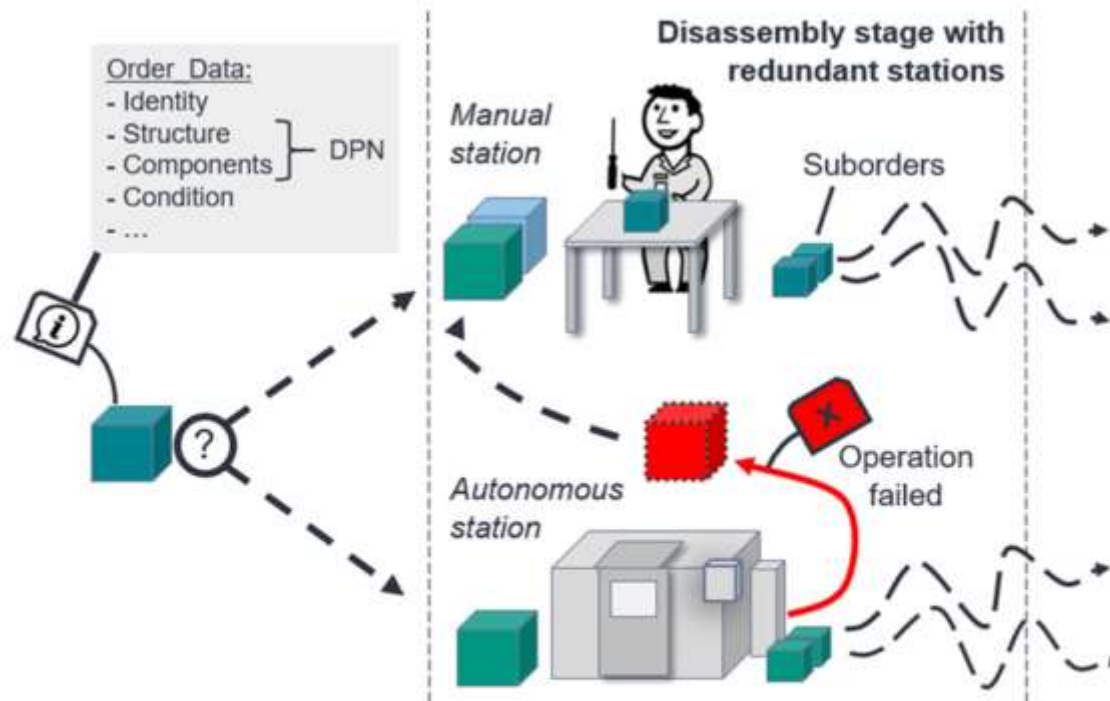


Fig. 1 Disassembly Petri Net of an assembly including 4 components

# Method Design

## 2 Order Define



$$q = \begin{pmatrix} q_1 \\ q_2 \\ \dots \\ q_p \end{pmatrix} \quad (1)$$

**Fig. 2** Illustration of characteristic entities and properties of the model including product condition, manual and autonomous stations with failing operations and the resulting diverging material flow



# Method Design

## 3 MDP Design

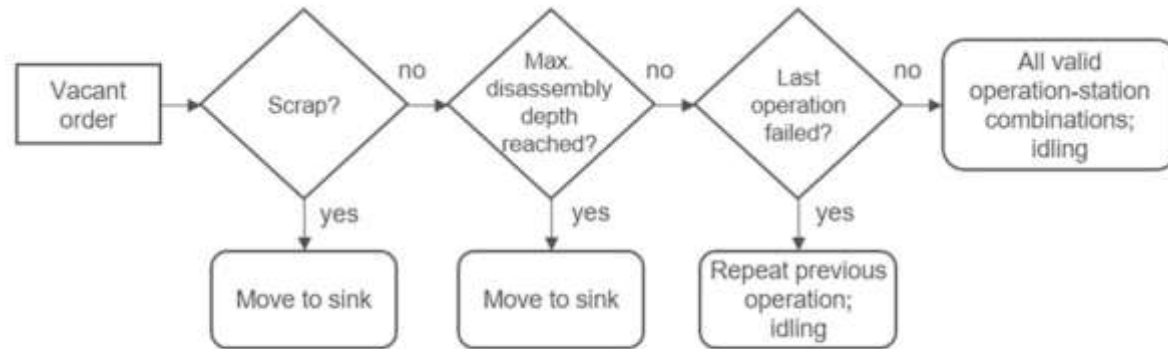


Fig. 3 Specification of the valid action space

$$r_{RC} = \sum_{k=1}^K -t_{k,diff} c_{k,working} \quad (5)$$

$$r_{time} = -(t_{Allocation,x} - t_{Allocation,x-1}) \quad (6)$$

$$r_{fail} = \begin{cases} -1, & \text{if last operation failed} \\ 0, & \text{otherwise} \end{cases} \quad (7)$$

$$r_{total} = w_{RC} r_{RC} + w_{time} r_{time} + w_{fail} r_{fail} \quad (8)$$

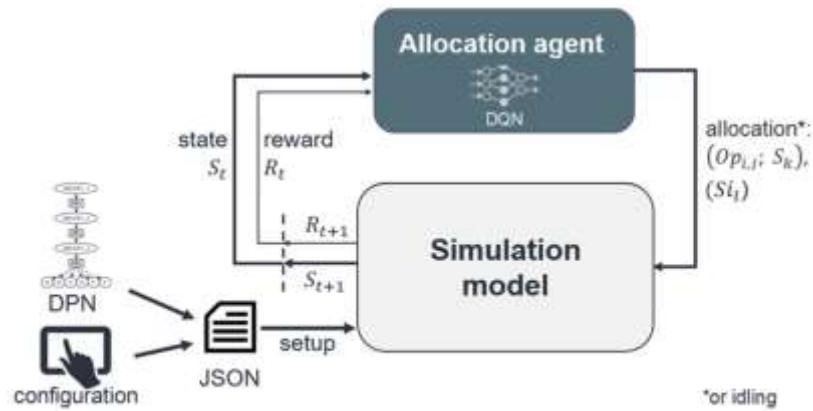






Fig. 4 System overview

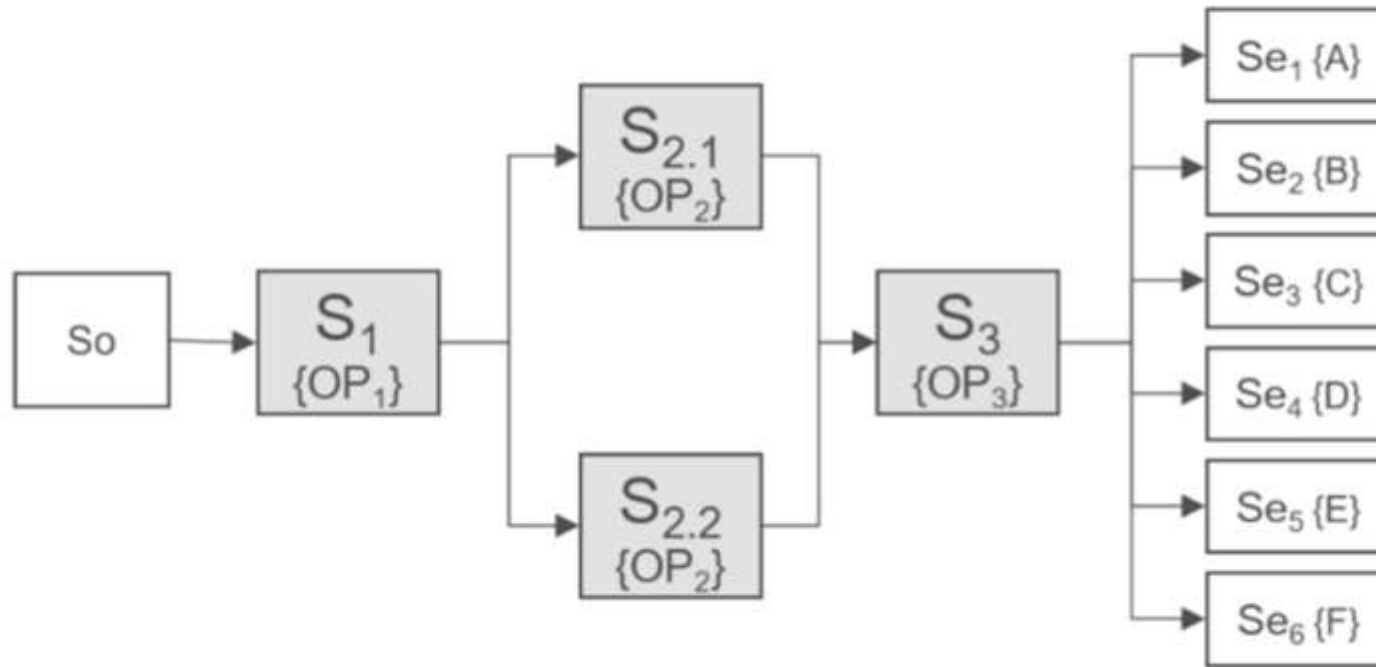




01		Problem Description
02		Method Design
03		Experiment Results
04		Conclusion

# Experiment Results

(a) Structure



(b) DPN

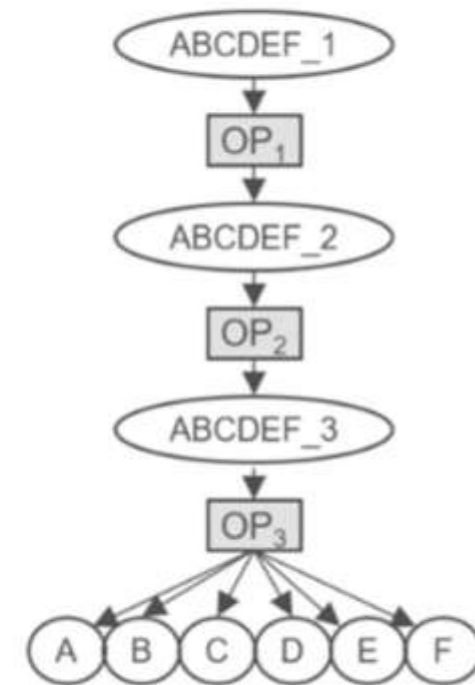
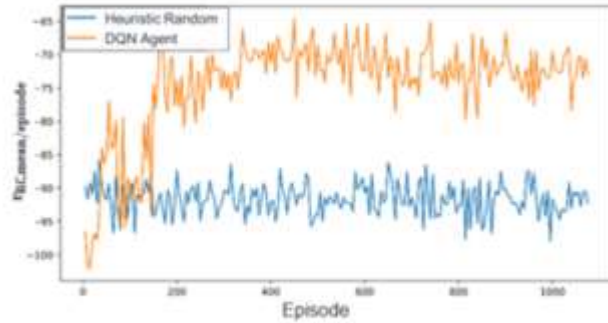


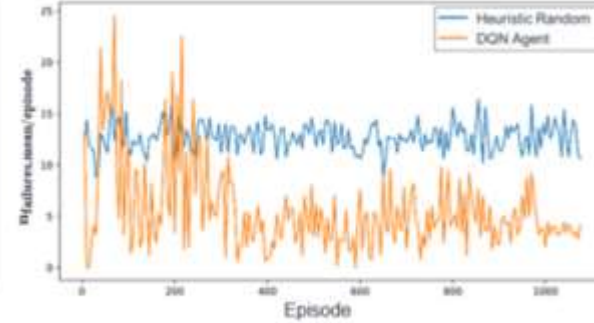
Fig. 5 a Structure of the test instance b Disassembly Petri Net of the product ABCDEF

# Experiment Results

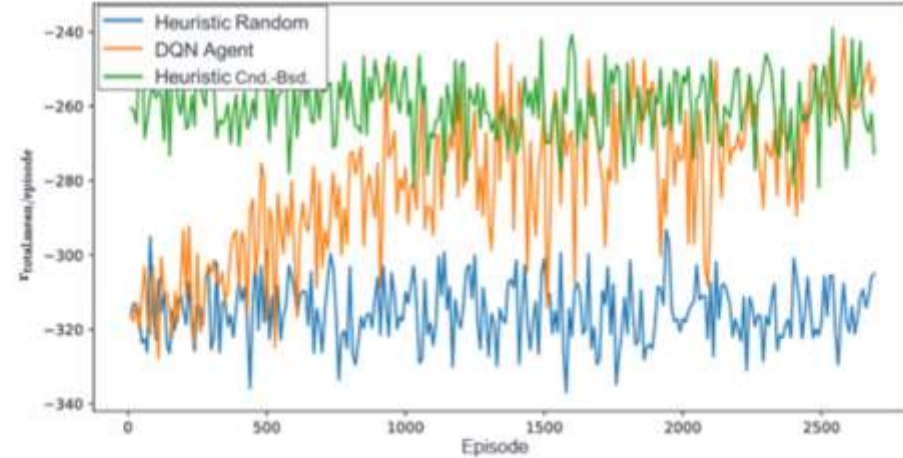
(a) Mean episodic resource cost reward (test case 1)



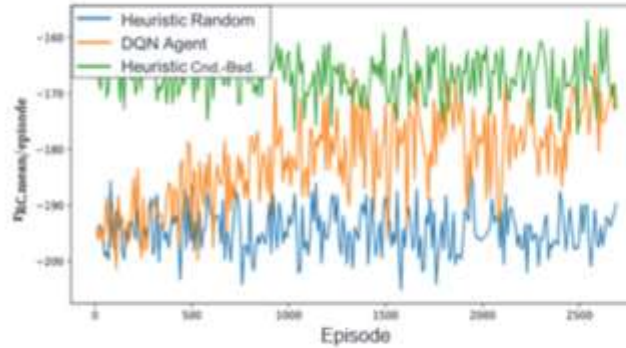
(b) Mean # of failed operations per episode (test case 1)



(c) Mean episodic total reward (test case 2)



(d) Mean episodic resource cost reward (test case 2)



(e) Mean episodic time reward (test case 2)

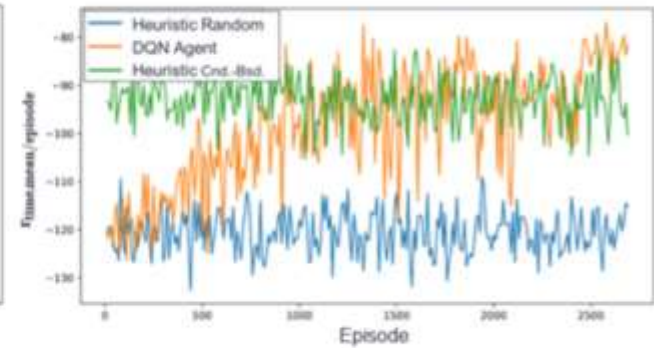






Fig. 6 Computational results and learning progress of the DQN-agent and the benchmark heuristics



01		Problem Description
02		Method Design
03		Experiment Results
04		Conclusion

# Conclusion

①

The study demonstrates that reinforcement learning can effectively manage the dynamic and uncertain nature of hybrid disassembly systems that include both manual and autonomous workstations. By modeling the system through a Disassembly Petri Net and implementing a Deep Q-Network-based controller, the approach achieves adaptive decision-making and improved performance in terms of utilization, reliability, and makespan.

②

Future research will focus on integrating multi-agent reinforcement learning to coordinate multiple autonomous stations and on incorporating real-time sensory data to enhance condition-based control accuracy. Moreover, the scalability of the proposed framework will be examined for larger and more complex industrial disassembly lines.

## Reinforcement Learning for Hybrid Disassembly Line Balancing Problems

Jiacun Wang, GuiPeng Xi, XiWang Guo, Shixin Liu, ShuJin Qin, Henry Han





**This paper was published in:** IEEE Transactions on Intelligent Vehicles (IF: 14.3)

**Published Time:** 2024

**Reference:** Wang J, Xi G P, Guo X W, et al. Reinforcement learning for hybrid disassembly line balancing problems[J]. Neurocomputing, 2024, 569: 127145.







## Contents

01		<b>Problem Description</b>
02		<b>Method Design</b>
03		<b>Experiment Results</b>
04		<b>Conclusion</b>





## Contents

01		<b>Problem Description</b>
02		Method Design
03		Experiment Results
04		Conclusion

# Problem Description

## Challenge

The hybrid disassembly line—**combining linear and U-shaped layouts**—introduces complex task and line assignment decisions under **multi-skill worker constraints**, making the problem highly combinatorial and NP-hard.





Traditional optimization methods struggle with dynamic product mixtures and skill-matching requirements, and cannot adapt in real time to changing disassembly demands.

## Method

Formulates the problem as a Markov Decision Process and proposes a Soft Actor–Critic (SAC) algorithm—originally for continuous control—to solve this discrete combinatorial problem via interval mapping (continuous actions  $\rightarrow$  discrete decisions).

Designs a custom Gym environment that integrates **product precedence graphs**, worker-skill matrices, and hybrid line layouts, enabling end-to-end RL-based task and line assignment.



01		Problem Description
02		Method Design
03		Experiment Results
04		Conclusion

# Method Design

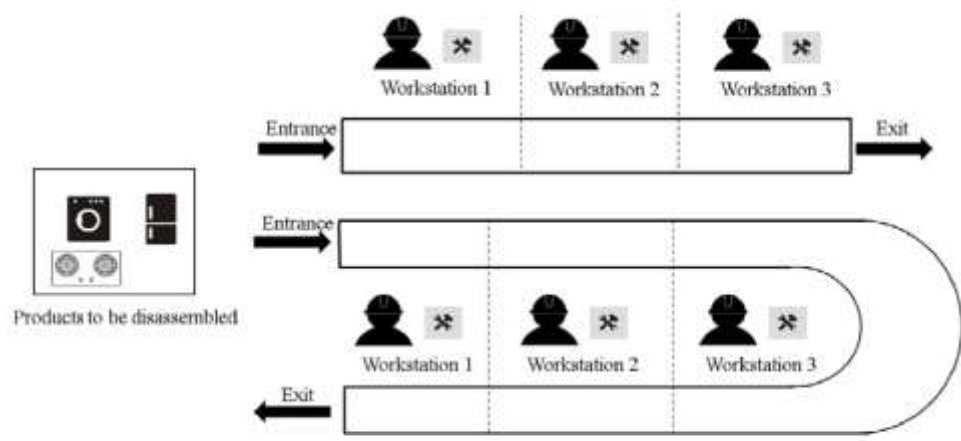


Fig. 1. An example of HDLBP.

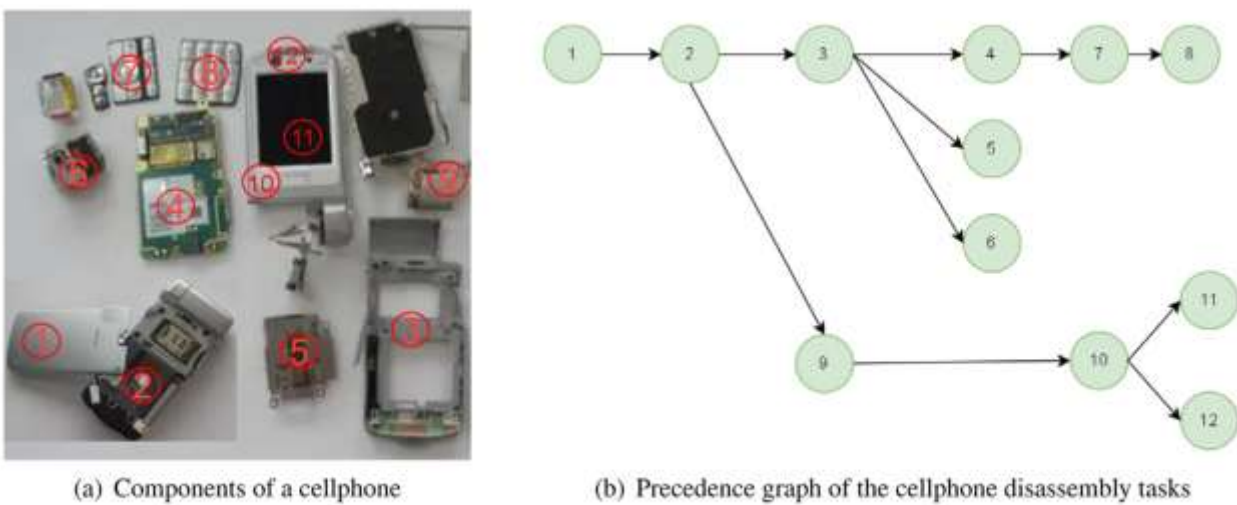


Fig. 2. Cellphone components and precedence graph of disassembly tasks.

**Table 1**  
Cell phone disassembly tasks.

Task index	Task name	Task index	Task name
1	remove battery cover	7	remove the main button
2	remove battery	8	remove number button
3	remove back case	9	remove junction
4	remove board	10	remove front case
5	remove microphone	11	remove LCD
6	remove camera	12	remove speaker

**Table 2**  
Relationship between tasks and skills.

Task index	Skill		Task index	Skill	
	1	2		1	2
1	0	0	7	0	0
2	0	1	8	0	0
3	0	0	9	1	0
4	1	0	10	0	0
5	1	0	11	1	0
6	1	0	12	1	0

**Table 3**  
Relationship between workers and skills.

Disassembly line type	Worker	Skill	
		1	2
Linear	1	0	1
	2	0	0
	3	1	0
U-shaped	1	1	1
	2	0	0
	3	1	1

$$a_{wn}^l = \begin{cases} 1, & \text{if Worker } w \text{ has Skill } n \text{ on line } l \\ 0, & \text{if Worker } w \text{ does not have Skill } n \text{ on line } l \end{cases}$$

# Method Design

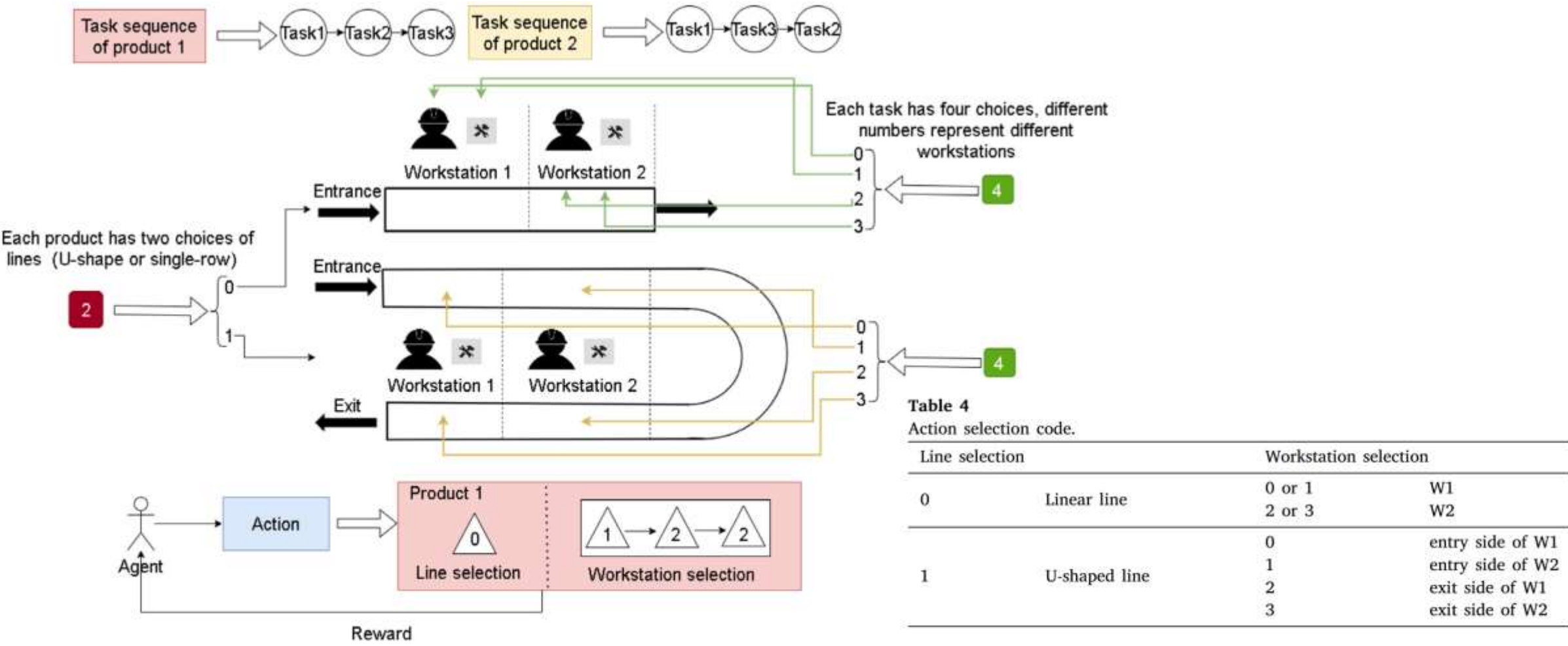






Fig. 3. Action design for an HDLBP instance with a linear disassembly line and a U-shape disassembly line. There are two products to be disassembled. Each disassembly line has two workstations and each product has three disassembly tasks. The action is to select a disassembly line first and then a workstation. For a workstation in the U-shape line, an entry side or exit side must also be selected. The example action code (0, 1, 2, 2 for product 1 shown in the pink box means product 1 is to be disassembled on the single-row linear disassembly line, its task 1 is assigned to workstation 1, and its tasks 2 and 3 are assigned to workstation 2.



01		Problem Description
02		Method Design
03		Experiment Results
04		Conclusion

# Experiment Results

**Table 5**

Products for disassembly test.

Product	Number of tasks	Number of required skills
TV	12	5
Cellphone	12	5
Tesla battery	37	5

**Table 6**

Test instances.

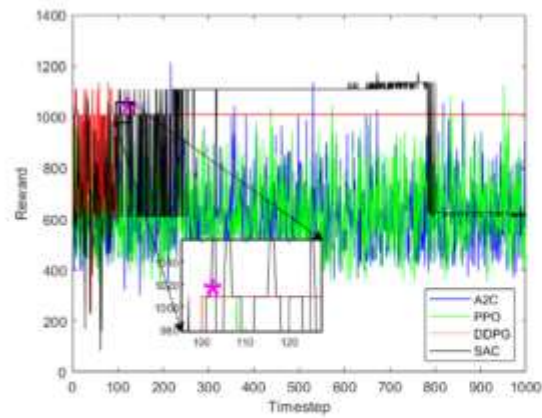
Instance ID	Number of products			Number of tasks
	TV	Cellphone	Tesla battery	
1	1	1	0	24
2	1	2	0	36
3	0	1	1	49
4	1	1	1	61
5	1	2	2	110
6	2	2	3	159

*Disassembly profit:* The profit of the product under the current disassembly scheme.

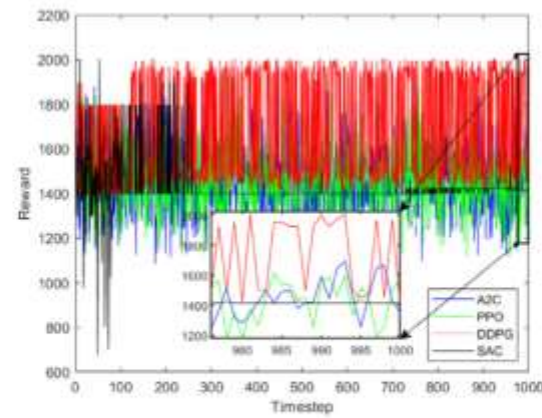
*Runtime:* Running time of the algorithm after completing the specified number of training timesteps.



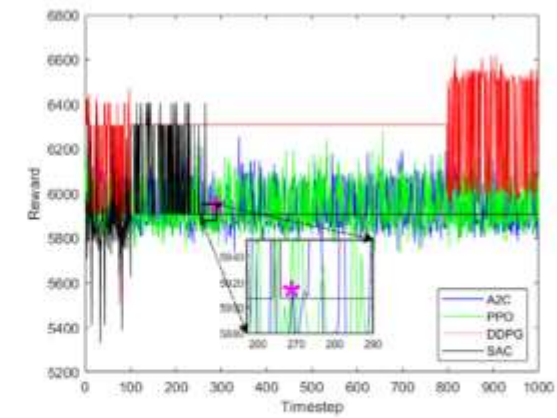
# Experiment Results



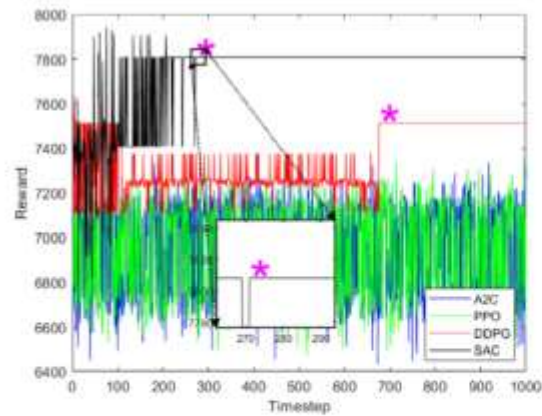
(a) Test instance 1.



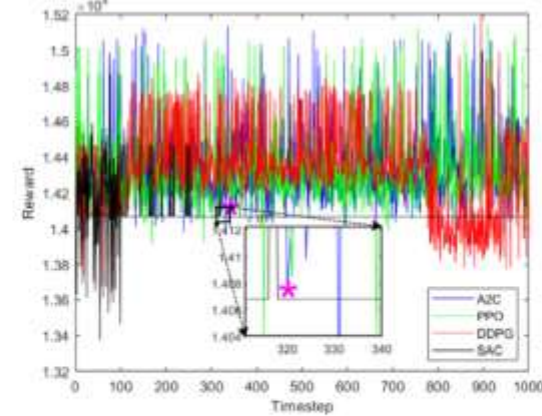
(b) Test instance 2.



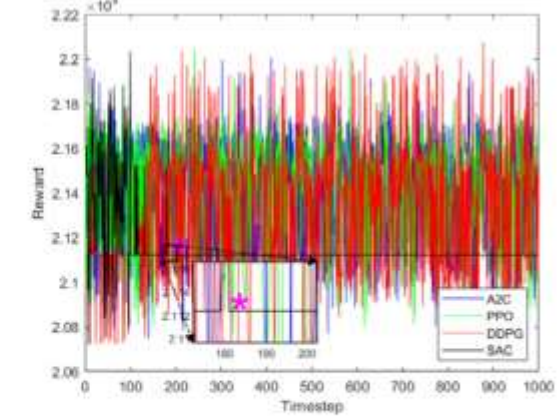
(c) Test instance 3.



(d) Test instance 4.



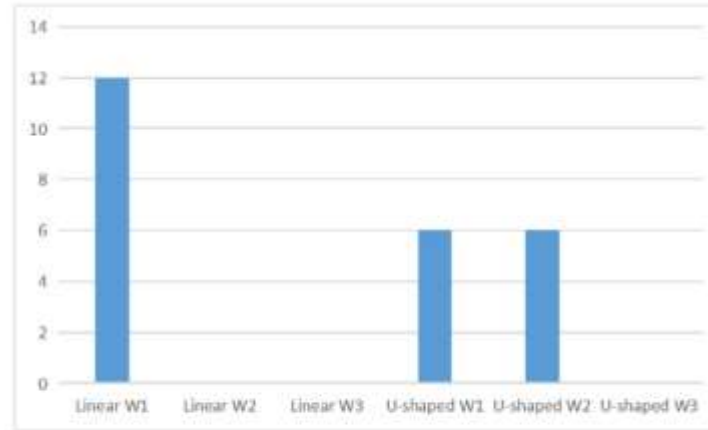
(e) Test instance 5.



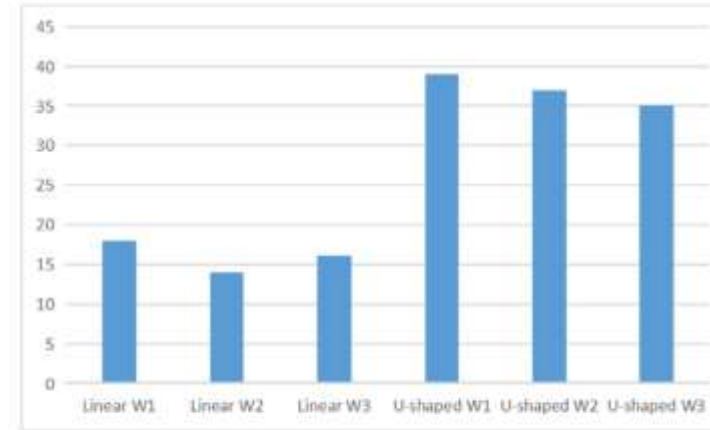
(f) Test instance 6.

Fig. 5. Trend of reward throughout training.

# Experiment Results



(a) Small-scale tasks distribution in configuration 1



(b) Large-scale tasks distribution in configuration 1

Fig. 4. Different disassembly scales distribution in configuration 1.

**Table 7**

Workstation skill configuration.

Config. ID	Number of skills					
	Linear			U-shaped		
	W1	W2	W3	W1	W2	W3
1	1	0	0	1	0	0
2	2	0	0	2	0	0
3	3	0	0	3	0	0
4	4	0	0	4	0	0
5	5	0	0	5	0	0

# Experiment Results

**Table 8**  
Results of different configurations.

Config. ID	Instance ID	The number of tasks on each workstation					
		Linear			U-shaped		
		W1	W2	W3	W1	W2	W3
1	1	12	0	0	6	6	0
	2	14	10	0	5	3	4
	3	37	0	0	5	3	4
	4	37	0	0	9	8	7
	5	17	18	0	26	25	23
	6	18	14	16	39	37	35
2	1	12	0	0	6	6	0
	2	5	2	5	13	11	0
	3	37	0	0	5	3	4
	4	37	0	0	9	7	8
	5	9	15	12	26	23	25
	6	17	16	15	39	35	37
3	1	6	0	6	6	0	6
	2	9	3	0	13	0	11
	3	37	0	0	8	4	0
	4	10	4	10	24	13	0
	5	26	22	26	12	12	11
	6	30	27	29	27	23	23
4	1	6	0	6	6	0	6
	2	5	7	0	13	0	11
	3	12	0	0	24	0	13
	4	12	0	12	24	0	13
	5	26	23	25	12	13	11
	6	34	32	32	23	18	20
5	1	6	0	6	6	0	6
	2	5	7	0	13	0	11
	3	12	0	0	24	0	13
	4	12	0	12	24	0	13
	5	26	23	25	12	13	11
	6	34	32	32	23	18	20





**Table 9**  
The profit after training.

Instance ID	A2C	DDPG	PPO	SAC	CPLEX
1	638	1009	396	1403	1518
2	1534	1983	1498	2098	2429
3	5871	6555	5836	6600	6864
4	6560	7513	7169	7808	8135
5	14 221	14 316	14 197	14 754	15 507
6	21 688	21 939	21 510	22 063	23 040

**Table 10**  
Computation time of algorithms.

Instance ID	Computation time (seconds)			
	A2C	DDPG	PPO	SAC
1	6.167	12.633	12.357	19.614
2	6.292	12.756	12.922	19.482
3	14.045	12.479	27.510	20.547
4	14.686	13.742	28.696	20.780
5	15.696	13.976	30.454	21.254
6	16.898	14.238	32.931	22.641



01		Problem Description
02		Method Design
03		Experiment Results
04		Conclusion

# Conclusion

①

This work study the Hybrid Disassembly Line Balancing Problem (HDLBP), combining linear and U-shaped lines with multi-skilled workers, and formulates a mathematical model to maximize disassembly profit (including skill training costs).

A custom Gym environment and a Soft Actor–Critic (SAC) algorithm with interval mapping (continuous  $\rightarrow$  discrete actions) are proposed to solve this combinatorial problem in real time.

Experiments on small- to large-scale cases (TV, cellphone, Tesla battery) show that SAC outperforms DDPG, A2C, and PPO in solution quality and stability, and achieves results closest to CPLEX.

②

Incorporating external uncertainties (e.g., equipment failures, supply disruptions);

Developing multi-agent or multi-objective SAC for HDLBP;

Extending to dynamic, multi-goal balancing in real-world remanufacturing settings.

## Reinforcement Learning-Based Selective Disassembly Sequence Planning for the End-of-Life Products With Structure Uncertainty

Jiajun Chai , Graduate Student Member, IEEE, Yuanheng Zhu , Senior Member, IEEE, and Dongbin Zhao , Fellow, IEEE





**This paper was published in:** IEEE Robotics and Automation Letters (IF: 5.3)

**Published Time:** 2021

**Reference:** Zhao X, Li C, Tang Y, et al. Reinforcement learning-based selective disassembly sequence planning for the end-of-life products with structure uncertainty[J]. IEEE Robotics and Automation Letters, 2021, 6(4): 7807-7814.







## Contents

01		<b>Problem Description</b>
02		<b>Method Design</b>
03		<b>Experiment Results</b>
04		<b>Conclusion</b>





## Contents

01		<b>Problem Description</b>
02		Method Design
03		Experiment Results
04		Conclusion

# Problem Description





## Challenge

- End-of-life (EOL) products often have structural uncertainty—such as **missing, damaged, or corroded** parts—making pre-planned disassembly sequences infeasible or suboptimal.
- Traditional heuristic or metaheuristic methods require full re-optimization when product structure changes, lacking real-time adaptability.

## Method

- Proposes a **multi-level selective disassembly hybrid graph model (MSDHGM)** that encodes contact, precedence, and hierarchical relationships, and dynamically updates under structural uncertainty.
- Formulates the problem as a Markov Decision Process (MDP) and solves it with a Deep Q-Network (DQN-SDSP) that uses Pareto-based rewards, tool/direction change penalties, and level-based incentives to generate adaptive, near-optimal sequences.



01		Problem Description
02		Method Design
03		Experiment Results
04		Future Work

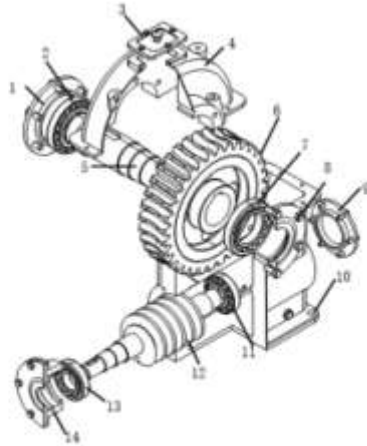


Fig. 1. Assembly drawing of a reducer.

HGM is used to illustrate the structure of EOL products, which can be defined as a 4-tuple:  $HGM = \{VF_n, VC_c, PR\}$ , where  $VF_n = \{VF_1, VF_2, \dots, VF_n\}$  is the part that cannot be further disassembled.  $VC_c = \{VC_{c1}, VC_{c2}, \dots, VC_{cf}\}$  is an undirected edge that represents the contact constraint between two parts.  $PR$  is precedence relationships among the parts of the EOL products, which consists of  $E_p$  and  $DE_u$  where  $E_p = \{E_{p1}, E_{p2}, \dots, E_{pe}\}$  illustrates the precedence relationships among the contact parts of the EOL products, which can be represented by a directed solid edge.  $DE_u = \{DE_{u1}, DE_{u2}, \dots, DE_{um}\}$

MSDHGM can be defined as follows:

$$MSDHGM = \{VF_n, VC_c, PR_h, L_n\} \quad (1)$$

where  $L_n = \{L_1, L_2, \dots, L_n\}$  is the disassembly level of each part of EOL products. According to its composition characteristics, the  $MSDHGM = \{VF_n, VC_c, PR, L_n\}$  can be categorized into  $M_p = \{VF_n, PR\}$ ,  $M_c = \{VF_n, VC_c\}$ , and  $M_l = \{VF_n, L_n\}$ . Then, the following defines precedence matrix  $M_p$ , contact matrix  $M_c$ , and level matrix  $M_l$ , respectively.

# Method Design

1) *Precedence Matrix  $M_p$* : Generally, the precedence relationships include AND and OR relationships among parts [16]. The AND relationship indicates the part  $i$  can be disassembled after completing all its prior disassembly parts. For instance, in Fig. 2, part 4 can be obtained after part 1, part 3, and part 8 are released. The OR relationship means that the part  $i$  can be removed after one of its prior disassembly parts is disassembled. The precedence matrix  $M_p$  can be defined as follows.

$$M_p = \begin{bmatrix} a_{1,1}^{DE} & a_{1,2}^{DE} & \cdots & a_{1,n}^{DE} \\ a_{2,1}^{DE} & a_{2,2}^{DE} & \cdots & a_{2,n}^{DE} \\ \vdots & \vdots & & \vdots \\ a_{n,1}^{DE} & a_{n,2}^{DE} & \cdots & a_{n,n}^{DE} \end{bmatrix} \quad (2)$$

$$a_{i,j}^{DE} = \begin{cases} 1, & \text{if part } i \text{ is the AND predecessor of part } j \\ -1, & \text{if part } i \text{ is the OR predecessor of part } j \\ 0, & \text{otherwise} \end{cases} \quad (3)$$

where  $a_{i,j}^{DE}$  indicates precedence relationship between the parts  $i$  and  $j$ .

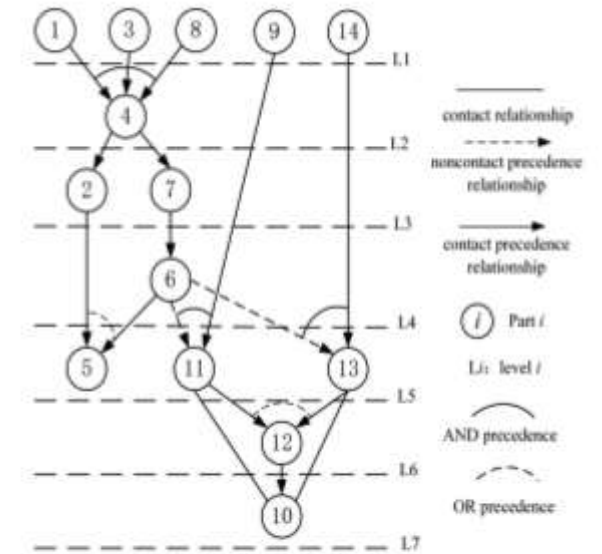


Fig. 2. MSDHGM of the reducer.

# Method Design

2) **Contact Matrix  $M_c$** : The contact relationships of parts consist of two primary parts: direct contact constraints and indirect contact among parts. For instance, in Fig. 2, part 10 has a contact relationship with part 11, which is marked by the solid line. The contact matrix  $M_c$  is shown in Eq. 4 and Eq. 5 and the contact matrix  $M_c$  of a reducer is given in Fig. 4.

$$M_c = \begin{bmatrix} a_{1,1}^E & a_{1,2}^E & \cdots & a_{1,n}^E \\ a_{2,1}^E & a_{2,2}^E & \cdots & a_{2,n}^E \\ \vdots & \vdots & & \vdots \\ a_{n,1}^E & a_{n,2}^E & \cdots & a_{n,n}^E \end{bmatrix} \quad (4)$$

$$a_{i,j}^E = \begin{cases} 1, & \text{if there is a contact between part } i \text{ and } j \\ 0, & \text{otherwise} \end{cases} \quad (5)$$

where  $a_{i,j}^E$  indicates the contact relationship between part  $i$  and part  $j$ .

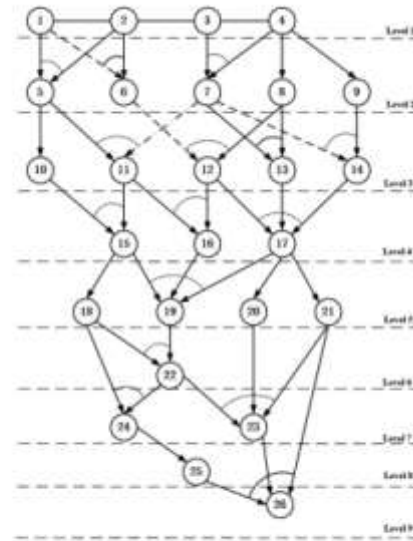


Fig. 4. The MSDHGM of the case study.

3) **Level Matrix  $M_l$** : The layered process is designed to obtain the disassembly level of each part, which is given in **Algorithm 1**. For instance, in Fig. 2, it is obvious that parts 13, 89, and 14 are in level 1 at the beginning of the disassembly process. The level matrix  $M_l$  is shown in Eq. 6 and Eq. 7.

$$M_l = \begin{bmatrix} a_{1,1}^L & a_{1,2}^L & \cdots & a_{1,n}^L \\ a_{2,1}^L & a_{2,2}^L & \cdots & a_{2,n}^L \\ \vdots & \vdots & & \vdots \\ a_{n,1}^L & a_{n,2}^L & \cdots & a_{n,n}^L \end{bmatrix} \quad (6)$$

$$a_{i,j}^L = \begin{cases} 1, & \text{if part } j \text{ is in level } i \\ 0, & \text{otherwise} \end{cases} \quad (7)$$

**Algorithm 1:** Layered Process for MSDHGM.

**Input:** Precedence matrix  $M_p$  and contact matrix  $M_c$

**Output:** The level set  $L$  of parts

```

1  Def Generate Level ()
2  Initialize Level set  $L$ ,  $l = 1$ ,  $j = 1$ ;
3  While (1)
4  Searching precedence matrix  $M_p$  and contact matrix
    $M_c$ 
5  Obtain all parts  $\{p_1, p_2, \dots, p_i\}$  that can be removed
   simultaneously
6  If  $\{p_1, p_2, \dots, p_i\} \neq \emptyset$ 
7   $\{p_1, p_2, \dots, p_i\} \rightarrow L = \{a_{j1}, a_{j2}, \dots, a_{ji}, l\}$ 
8  Update  $M_p^{i+1}, M_c^{i+1} \leftarrow M_p^i, M_c^i$ 
9   $l \leftarrow l + 1$ 
10  $j \leftarrow j + 1$ 
11 Else
12 Break
13 End If
14 End While
15 Return Level set  $L$ 

```



---

**Algorithm 2:** Dynamic Adjustment of  $M_p$  and  $M_c$ 

---

**Input:** Precedence matrix  $M_p$ , contact matrix  $M_c$  and  $MP$  ( $mp_1, mp_2, \dots, mp_v$ ). The number of missing parts of EOL products  $V$ .

**Output:**  $M_p$  and  $M_c$

---

```
1  Def Update ()
2  For  $i \leq V$  do:
3     $mp = MP[0][i]$ 
4    For  $j < n$  do:
5       $a_{mp,j}^{DE} = 0$ 
6       $a_{mp,j}^E = 0$ 
7       $a_{j,mp}^E = 0$ 
8    End For
9  End For
10 Return  $M_p, M_c$ 
```

---

## Reward

---

a) *Disassembly Time*: Disassembly time is the time consumption of operators in the disassembly process. Note that the disassembly time varies with the number of exchanges of disassembly tool and direction. So, the total disassembly time  $f_1$  mainly contains the basic disassembly time, the penalty time for disassembly direction and tool changes, as shown in Eq. 8.

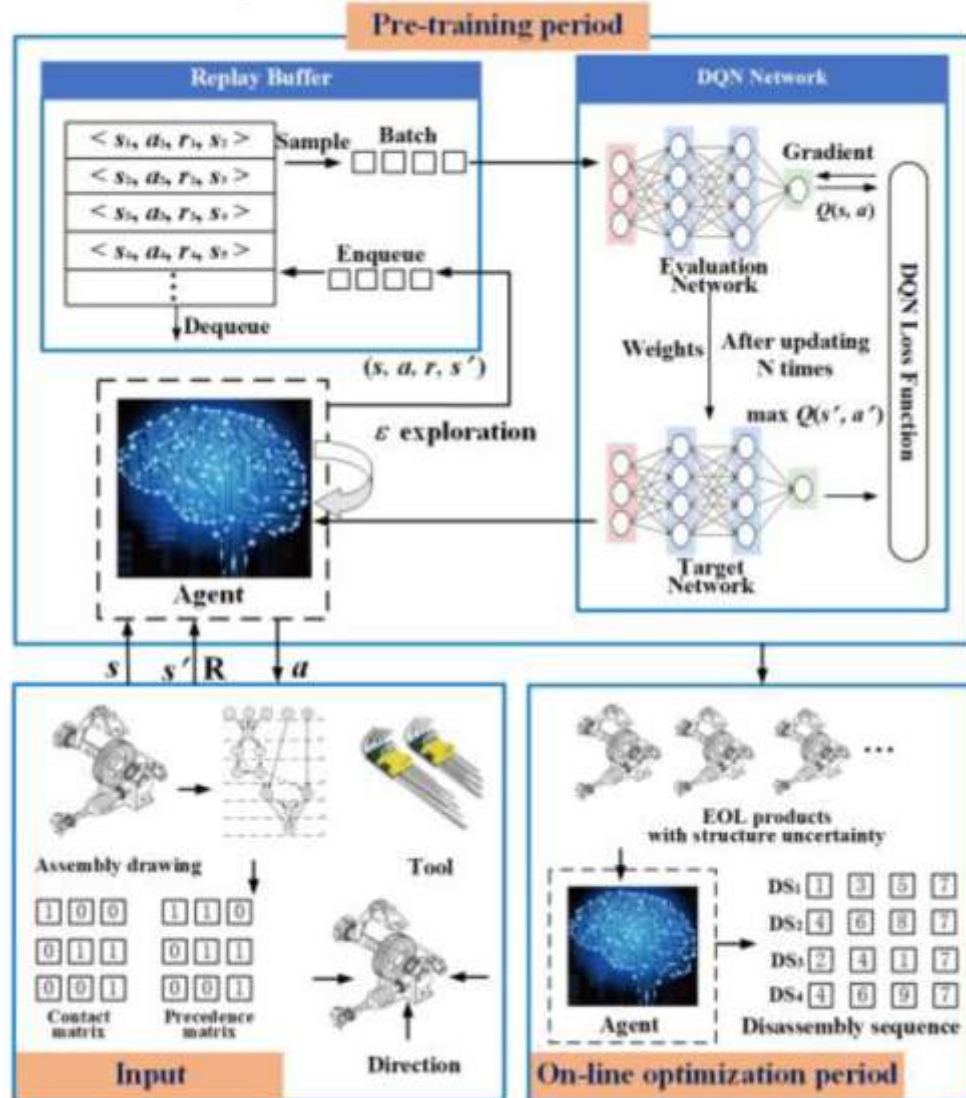
$$f_1 = \sum_{i=1}^N (t^i + x_i t_d^i + y_i t_{tool}^i) \quad (8)$$

b) *Disassembly Profit*: Another objective is disassembly profit  $f_2$ , which is formulated as follows:

$$f_2 = v_{TP} - \sum_{j=1}^N (c_j d_j) - h_m f_1 \quad (9)$$



# Method Design







**Algorithm 3:** Action and State for Disassembly Sequence Planning.

**Input:** Precedence matrix  $M_p$ , Contact matrix  $M_c$ ,  $s_t = \{M_p, M_c, M_l, a_{t-1}\}$

**Output:** The available parallel disassembly parts  $a_t = \{a_1, a_2, a_3, \dots, a_n\}$   
 $s_{t+1} = \{M_p, M_c, M_l, a_t\}$

- 1 **Def** Generate Action ()
- 2 **For**  $i < N$  **do**:
- 3 Search precedence matrix  $M_p$  and connection matrix  $M_c$
- 4 Calculate the number of precedence and contact constraint
- 5 **If**  $\sum_{i=1}^N a_{i,j}^{DE} = 0$  and  $\sum_{i=1}^N a_{i,j}^E \geq 1$  **do**:
- 6 Store  $a_i$  in  $a_t = \{a_1, a_2, a_3, \dots, a_n\}$
- 7 **End If**
- 8 **End For**
- 9 **Return**  $a_t = \{a_1, a_2, a_3, \dots, a_n\}$
- 10 **Def** Update state ()
- 11 **If** Select  $a_i$  from  $a_t = \{a_1, a_2, a_3, \dots, a_n\}$  **do**:
- 12 Update  $M_p^i, M_c^i \rightarrow M_p^{i+1}, M_c^{i+1}$  where the constraints are released
- 13 **End If**
- 14 Obtain  $s_{t+1} = \{M_p, M_c, M_l, a_t\}$
- 15 **Return**  $s_{t+1} = \{M_p, M_c, M_l, a_t\}$



01		Problem Description
02		Method Design
03		Experiment Results
04		Conclusion

# Experiment Results

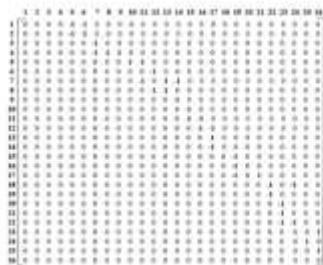


Fig. 5. The precedence matrix of case study.



Fig. 6. The connection matrix of case study.

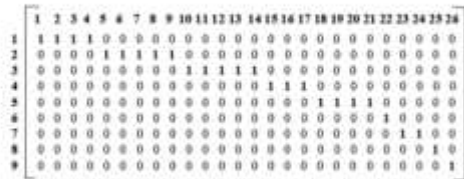


Fig. 7. The level matrix of case study.

TABLE I  
THE VALUE OF EACH PART

Part index	Disassembly time (s)	Tool (T)	Direction (D)	$C_i$ (¥)
1	1.2	T2	+X	1.2
2	1.3	T2	+Y	2.3
3	2.6	T2	+X	0.35
4	0.8	T4	+Z	2.6
5	3.1	T2	+X	2.1
6	1.1	T3	+X	0.7
7	2.3	T2	+Y	0.6
8	6.5	T1	+X	1.6
9	2.7	T1	+Y	3.1
10	3.5	T1	+X	3.2
11	2.36	T2	-Y	0.3
12	2.55	T4	+X	2.2
13	6.6	T2	-Y	3.6
14	4.3	T4	-Y	1.36
15	2.42	T2	+X	0.8
16	5.6	T3	+X	2.3
17	0.2	T3	+X	0.9
18	4.2	T2	+X	3.34
19	2.65	T3	-Y	1.75
20	4.5	T3	+Z	4.564
21	7.6	T3	+X	7.932
22	2.65	T2	-Y	2.65
23	3.4	T2	+X	1.4
24	2.41	T3	+Y	2.1
25	6.7	T3	+X	1.7
26	3.5	T4	-Y	3.5

# Experiment Results

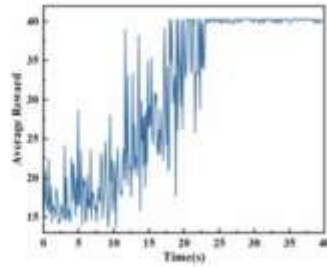


Fig. 9. Average reward of DQN-SDSP.

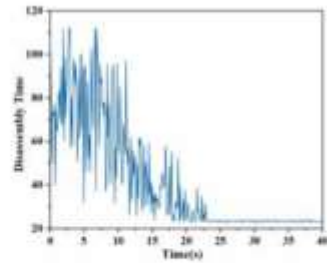


Fig. 10. Convergence curves of disassembly time.

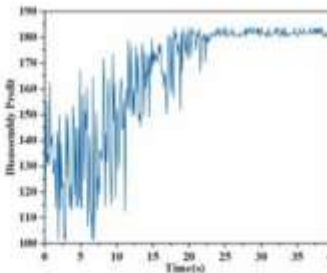


Fig. 11. Convergence curves of disassembly profit.

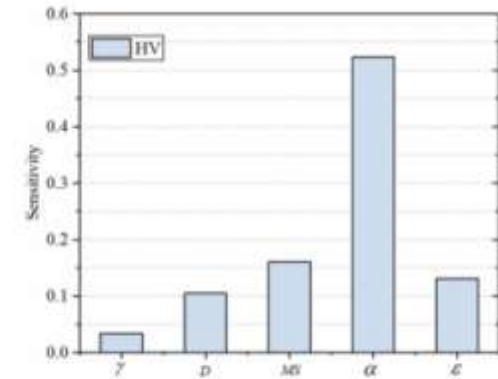


Fig. 8. Results of sensitivity analysis.

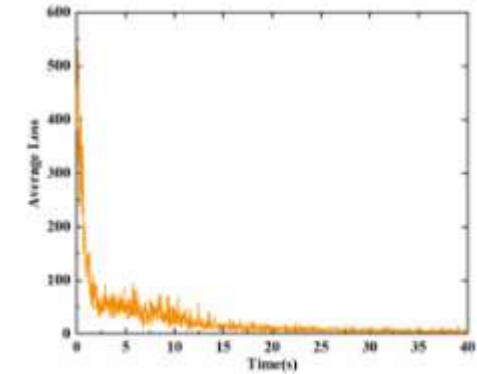


Fig. 12. Track of loss with network structures.

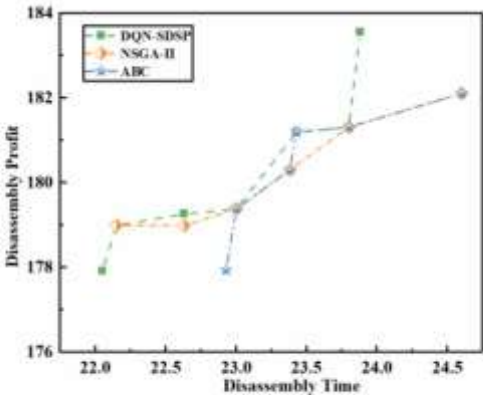
# Experiment Results

TABLE III  
PARETO SOLUTIONS FOR EOL PRODUCTS WITH TARGET PARTS IS 23

No.	Sequence	Tool change	Direction change	$f_1(s)$	$f_2(¥)$
1	3 → 7 → 11 → 15 → 19 → 22 → 23	2	5	23.88	183.55
2	4 → 7 → 11 → 15 → 18 → 22 → 23	1	5	22.63	179.26
3	3 → 7 → 11 → 15 → 18 → 22 → 23	0	5	23.43	181.19
4	3 → 7 → 14 → 17 → 20 → 23	3	2	23.80	181.31
5	4 → 7 → 14 → 17 → 20 → 23	4	5	23.00	179.38
6	1 → 2 → 6 → 12 → 17 → 20 → 23	5	4	22.05	177.92
7	2 → 1 → 6 → 12 → 17 → 19 → 22 → 23.	5	3	22.15	178.99

TABLE IV  
COMPARISON RESULTS FOR OPTIMIZATION USING DQN-SDSP, NSGA-II AND ABC

Method	Objective	Pareto solution							HV	SP
DQN-SDSP	Time	23.88	22.63	23.43	23.80	23.00	22.05	22.15	<b>1.55</b>	0.89
	Profit	183.55	179.26	181.19	181.31	179.38	177.92	178.99		
NSGA-II	Time	23.38	24.6	23.8	23.00	22.63	22.15		1.52	0.52
	Profit	180.30	182.1	181.31	179.38	179.26	178.99			
ABC	Time	24.6	23.80	23.43	23.38	22.93	23.00		1.38	<b>0.49</b>
	Profit	182.10	181.31	181.19	180.30	177.94	179.38			



Pareto solution of DQN-SDSP, NSGA-II and ABC.

# Experiment Results

TABLE V  
RESULTS OF PERFORMANCE TEST BETWEEN DQN-SDSP, NSGA-II AND ABC

NO.	Method	Removed Parts	Selective Disassembly Sequences	$(f_1, f_2)$	CPU (s)
Scenario 1	DQN-SDSP	Part 1 Part 2 Part 4 Part 9	6→12→17→20→23	(16.15,183.78)	2.31
			14→17→20→23	(16.50,185.18)	
			14→17→19→22→23	(17.30,185.97)	
	NSGA-II		6→12→17→21→23	(17.85,179.73)	7.82
			14→3→17→20→23	(20.10,183.69)	
			14→17→21→23	(18.20,181.13)	
			7→11→15→19→22→23	(20.58,185.22)	
	ABC		14→6→17→19→22→23	(18.40,184.83)	7.56
			14→6→17→19→20→23	(17.60,184.04)	
			6→12→17→20→23	(16.15,183.78)	
			7→11→15→19→22→23	(20.58,185.22)	
Scenario 2	DQN-SDSP	Part 1 Part 2 Part 3 Part 4 Part 5 Part 7 Part 9	11→15→19→22→23	(17.58,187.02)	2.14
			14→17→20→23	(16.50,185.18)	
			6→12→17→20→23	(16.15,183.78)	
			14→17→19→22→23	(17.30,185.97)	
	NSGA-II		11→6→15→19→22→23	(20.68,185.08)	6.54
			10→15→19→22→23	(19.02,183.54)	
			10→15→18→22→23	(18.57,181.18)	
	ABC		6→12→17→20→23	(16.15,183.78)	6.15
			14→6→17→20→23	(17.6,184.04)	
			11→6→15→19→22→23	(20.68,185.08)	

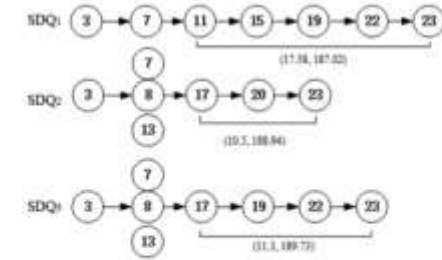


Fig. 14. Performance test of DQN-SDSP.

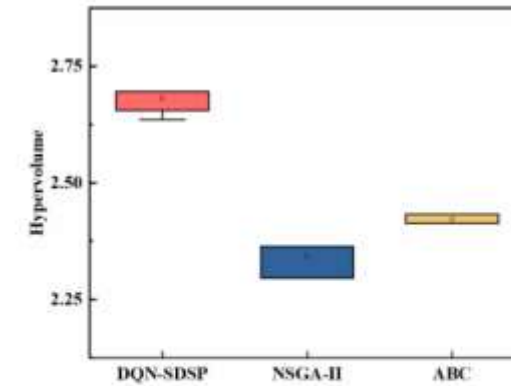






Fig. 15. Box plot of three methods on hypervolume value.





01		Problem Description
02		Method Design
03		Experiment Results
04		Conclusion



# Conclusion

①

A multi-level hybrid graph model (MSDHGM) is introduced to compactly represent precedence, contact, and hierarchical relationships, enabling dynamic updates under uncertainty.

The DQN-SDSP method effectively handles structure uncertainty (e.g., missing or damaged parts) and generates adaptive, near-optimal disassembly sequences in real time.

The method outperforms NSGA-II and ABC in solution quality (higher hypervolume) and adapts instantly to structural changes—no re-optimization needed.

②

Validate with real-world data, extend to multiple target parts, and integrate multi-objective RL to learn the true Pareto front without scalarization.

State feature coding can take various forms and can be represented by different matrices or composite grid worlds. It integrates complex constraints such as product priority graphs, worker-skill matrices, and mixed line layouts.

# Thank you for listening