https://doi.org/10.1016/j.jmsy.2021.08.002

# Towards Self-X Cognitive Manufacturing Network: An Industrial Knowledge Graph-Based Multi-Agent Reinforcement Learning Approach

Pai Zheng[1*], Liqiao Xia[1], Chengxi Li[1,2], Xinyu Li[3**], Bufan Liu[3]

[1]*Department of Industrial and Systems Engineering, The Hong Kong Polytechnic University, Hung Hom, Hong Kong Special Administrative Region, China*

[2]*Laboratory for Artificial Intelligence in Design, Hong Kong Science Park, New Territories, Hong Kong Special Administrative Region*

[3]*School of Mechanical and Aerospace Engineering, Nanyang Technological University, Singapore*

**Abstract:** Empowered by the advanced cognitive computing, industrial Internet-of-Things, and data analytics techniques, today's smart manufacturing systems are ever-increasingly equipped with cognitive capabilities, towards an emerging Self-X cognitive manufacturing network with higher level of automation. Nevertheless, to our best knowledge, the readiness of 'Self-X' levels (e.g., *self-configuration, self-optimization,* and *self-adjust/adaptive/healing*) is still in the infant stage. To pave its way, this work stepwise introduces an industrial knowledge graph (IKG)-based multi-agent reinforcement learning (MARL) method for achieving the Self-X cognitive manufacturing network. Firstly, an IKG should be formulated based on the extracted empirical knowledge and recognized patterns in the manufacturing process, by exploiting the massive human-generated and machine-sensed multimodal data. Then, a proposed graph neural network-based embedding algorithm can be performed based on a comprehensive understanding of the established IKG, to achieve semantic-based self-configurable solution searching and task decomposition. Moreover, a MARL-enabled decentralized system is presented to self-optimize the manufacturing process, and to further complement the IKG towards Self-X cognitive manufacturing network. An illustrative example of multi-robot reaching task is conducted lastly to validate the feasibility of the proposed approach. As an explorative study, limitations and future perspectives are also highlighted to attract more open discussions and in-depth research for ever smarter manufacturing.

**Keywords:** Industrial knowledge graph; graph embedding; cognitive manufacturing; graph neural network; reinforcement learning

## 1. Introduction

During the past two centuries, manufacturing paradigms have rapidly shifted from craft production, mass production, mass customization to today's mass personalization model, so as to proactively adapt to the high variety and low volume manufacturing in mass efficiency [1,2]. To achieve it, smart manufacturing systems play a critical role, which enable the execution of on-demand manufacturing processes smoothly and intelligently [3]. Nevertheless, there still lacks a semantic-based organization of massive heterogeneous manufacturing resources, which blocks a free flow of ever-evolving knowledge among machining modules, information systems, and stakeholders, and hence inhibits the effective knowledge exploitation in the manufacturing scenarios [4].

[*] *Corresponding authors*: pai.zheng@polyu.edu.hk; cash.li@ntu.edu.sg

Meanwhile, with increased flexibility and scalability of manufacturing resources leveraged in the personalized production process, their efficient management becomes ever critical to achieve a higher level of automation, prescribed by Self-X capabilities (e.g., self-configure, self-optimize, and self-adjust/adaptive/healing) in the 5C architecture model [5].

To address those issues, an emerging paradigm of cognitive manufacturing [6] has been brought up, which embraces human-level information processing of cognitive computing, the industrial Internet-of-Things (IIoT), and advanced data analytics to drive and optimize the manufacturing processes towards mass personalization [7]. Cognitive computing [8], as a revolutionary AI concept, not only can perform human-level perceptions, but also emulate the human brain's reasoning process, which is progressively flourishing the existing manufacturing systems with more cognitive intelligence. It is by nature all about exploiting data and knowledge from diverse multimodal resources (e.g., sensory data, social sensors) [10] to generate manufacturing values out of "rational" or "perceptual" methods [11]. Meanwhile, via the prevailing implementation of IIoT [12,13], manufacturing resources become ubiquitously connected and interoperable, and the massive human-generated and machine-sensed 5V big data can be effectively exploited with advanced data analytics and deep learning techniques [14,15]. Therefore, in this context, there lies great potential to enable the cognitive communication and management of flexible and scalable manufacturing tasks with Self-X capabilities, which however, is still in its infant stage and has been seldom discussed to-date.

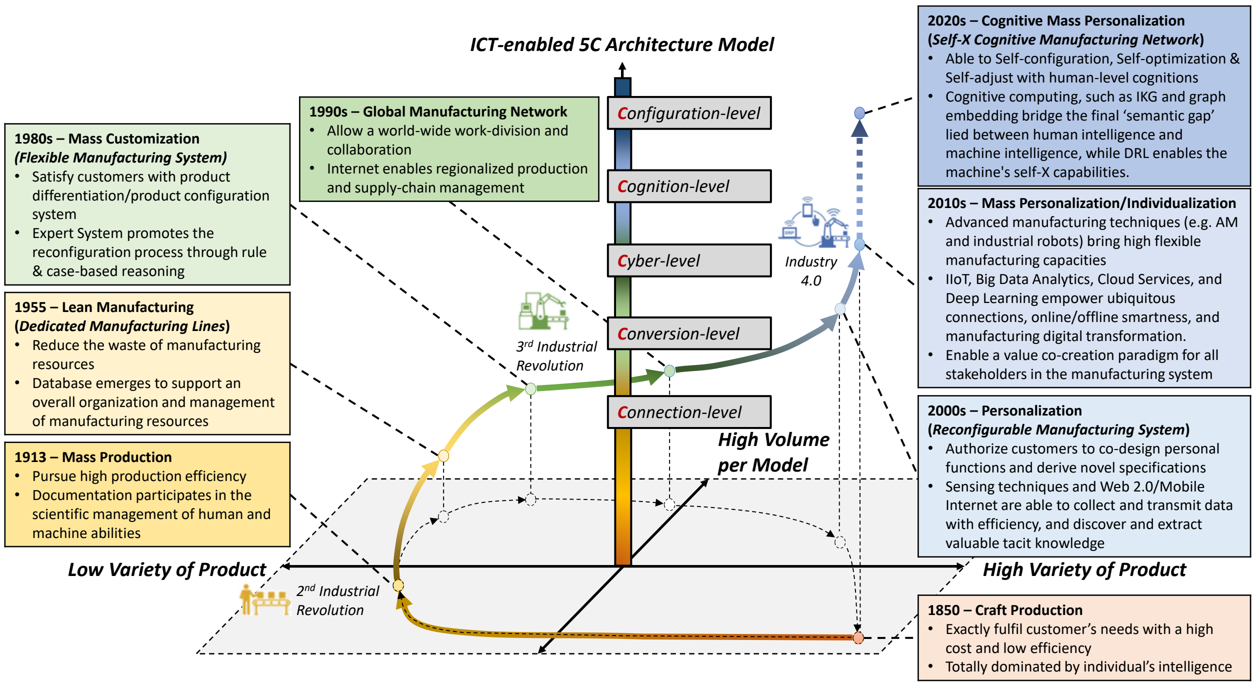
To pave its way, this work proposes an industrial knowledge graph (IKG)-based multi-agent reinforcement learning (MARL) approach to realizing the so-called Self-X cognitive manufacturing network, which is defined as *"a type of cognitive manufacturing system, of which all the manufacturing 'things' are organized and managed in a network (graph)-based manner, with high-level Self-X capabilities, including self-configuration, self-optimization and self-adjust/adaptive/healing*". To achieve the Self-X capabilities of the defined cognitive manufacturing network, the rest of this paper is organized as follows. Section 2 reviews the roadmap towards Self-X cognitive manufacturing network, and the related fundamental works to achieve it. Section 3 describes the proposed method for approaching the Self-X cognitive manufacturing network in a self-configurable and self-optimized manner. Firstly, an IKG is formulated based on the extracted empirical knowledge and recognized patterns in the manufacturing process by leveraging the massive human-generated and machine-sensed multimodal data, which serves as the foundation of the cognitive manufacturing network. Then, a proposed graph neural network-based embedding algorithm is performed based on a comprehensive understanding of the IKG, to achieve semantic-based self-configurable solution searching and task decomposition. Furthermore, the MARL-enabled decentralized multi-agent system is introduced to self-optimize the manufacturing process, and further to complement the IKG towards the Self-X cognitive manufacturing network lastly. An illustrative example of a multi-robot reaching task is further given in Section 4, to validate the feasibility of the proposed approach, with its implementation scope and limitations discussed as well. At last, main contributions and future perspectives of this research are highlighted in Section 5 to attract more in-depth research in this promising field.

## 2. Literature review

This section discusses the main roadmap of typical manufacturing paradigms with its enabling manufacturing systems, and reviews related works of IKG, graph embedding, and deep reinforcement learning (DRL) adopted in the manufacturing domain, all of which serve as the fundamental basis for realizing the Self-X capabilities of cognitive manufacturing systems.

## 2.1 The roadmap towards Self-X cognitive manufacturing network

The manufacturing paradigm has evolved much during the past two centuries along with its representative manufacturing systems, in regard to three evaluation criteria, namely product variety (i.e., x-axis), volume (i.e., y-axis) per model and automation level derived from 5C architecture model [5] (i.e., z-axis), as shown in Figure 1. It started with 'Craft Production' in the first industrial revolution, which delivered a 'design for customer' manner based on the individual hand maker's intelligence in high cost and low efficiency. This is followed by 'Mass Production' and "Lean Manufacturing" in the second industrial revolution with dedicated manufacturing lines. By offering a very limited variety of products at high efficiency with low waste, the 'design of customer' manner was conducted. Then, with the prosperity of Internet and Mobile Internet in the third industrial revolution, 'Mass Customisation' and 'Global Manufacturing Network' paradigms became dominant, of which 'design with customer' can be achieved at an affordable cost, through agile/flexible/reconfigurable manufacturing system together with online configurations [16] [17]. More recently in the 2010s, the voice of 'design by customer' in the 'Mass Personalization' [1][18] or 'Mass Individualization' [2] paradigm becomes one of the ultimate goals of Industry 4.0, enabled by the advanced manufacturing technologies (e.g., additive manufacturing and industrial robots), IIoT, big data analytics and cognitive computing, in a cyber-physical integrated manner.



**1980s – Mass Customization**
***(Flexible Manufacturing System)***
- Satisfy customers with product differentiation/product configuration system
- Expert System promotes the reconfiguration process through rule & case-based reasoning

**1955 – Lean Manufacturing**
**(*Dedicated Manufacturing Lines*)**
- Reduce the waste of manufacturing resources
- Database emerges to support an overall organization and management of manufacturing resources

**1913 – Mass Production**
- Pursue high production efficiency
- Documentation participates in the scientific management of human and machine abilities

**1990s – Global Manufacturing Network**
- Allow a world-wide work-division and collaboration
- Internet enables regionalized production and supply-chain management

**2020s – Cognitive Mass Personalization**
**(*Self-X Cognitive Manufacturing Network*)**
- Able to Self-configuration, Self-optimization & Self-adjust with human-level cognitions
- Cognitive computing, such as IKG and graph embedding bridge the final 'semantic gap' lied between human intelligence and machine intelligence, while DRL enables the machine's self-X capabilities.

**2010s – Mass Personalization/Individualization**
- Advanced manufacturing techniques (e.g. AM and industrial robots) bring high flexible manufacturing capacities
- IIoT, Big Data Analytics, Cloud Services, and Deep Learning empower ubiquitous connections, online/offline smartness, and manufacturing digital transformation.
- Enable a value co-creation paradigm for all stakeholders in the manufacturing system

**2000s – Personalization**
**(*Reconfigurable Manufacturing System*)**
- Authorize customers to co-design personal functions and derive novel specifications
- Sensing techniques and Web 2.0/Mobile Internet are able to collect and transmit data with efficiency, and discover and extract valuable tacit knowledge

**1850 – Craft Production**
- Exactly fulfil customer's needs with a high cost and low efficiency
- Totally dominated by individual's intelligence

Figure 1. Roadmap towards Self-X cognitive manufacturing network

Accordingly, the digital capabilities (i.e., networking, analytics, intelligence) enabled by advanced information and communication technology [19] can empower modern manufacturing systems one step forward in a cognitive manner [20] (i.e., the dashed line). Motivated by the self-organizing networking concept [21], a cognitive mass personalization paradigm is foreseeable based on the so-called Self-X cognitive manufacturing network. The 'Self-X' features, also known as self-organizing features [22,23] are envisioned mainly in three aspects: (1) *Self-configuration* indicates the automation in re-/configuring manufacturing resources (manufacturing “things”) to perform on-demand manufacturing via “plug-and-play” the standard hardware and/or software interfaces [24]; (2) *Self-optimization* aims to achieve an optimal manufacturing performance through reorganizing several nodes of manufacturing resources and their networking [25]; and eventually (3) *Self-adjust/adaptive/healing* is responsible for actively monitoring manufacturing processes to identify any variations and autonomously dealing with potential disruptions without human interventions [26,27].

## 2.2 Industrial knowledge graph and graph embedding

Industrial knowledge graph (IKG) mainly specializes in the manufacturing and production areas, and it organizes the knowledge of manufacturing from multi-source heterogeneous data in the graph manner systematically and semantically [28]. Based on the well-structured knowledge graph, numerous industrial tasks improve their performance. In the product design area, the designers desire the requirements to be integrated and specific. To achieve this goal, the concept graph enables the knowledge graph to fuse the similar entities [29] and considers the contextual information for requirement elicitation [30]. Besides, a multi-level and multi-factor process knowledge graph to illustrate the CAD models and numerical control processes in a standard way to control the data mapping and understand the implicit semantics [31]. Furthermore, industrial scenarios involve numerous domain experts’ experience, of which the formalized IKGs enable the integrated multi-expert knowledge management for collaborative decision making [32]. Apart from problem-solving, more attention has been paid on the causality of problems occurred. The most straightforward manner is to transform the working process into a knowledge graph [33] or disassembling the components as nodes in the knowledge graph [34]. Similarly, an event graph is generated to simulate the manufacturing process, and represent the event logic by setting events as entities in a graph form [35].

Meanwhile, to enable the efficient knowledge querying for solution recommendation, graph embedding has been widely adopted across sectors (e.g., healthcare, social media, etc.), owing to its capability of preserving their attributes and graphical structure correctly in vector space [36]. It transforms nodes and edges in IKG into lower-dimension vectors whilst maximally preserving the information of the graphical structure, which presents strong abilities to processing complex semantic meanings and enables a series of efficient manners for the representation, query, and reasoning on the stored industrial knowledge [36]. Among various graph embedding techniques, graph neural network (GNN) is a prevailing methodology utilized to reflect the impact of interactions of graph-based structural data [37]. In industrial applications, GNNs have been utilized in improving the scheduling ability of manufacturing systems by reinforcement learning and graph convolutional networks [38]. Besides, it determines the edges in the graph by dependencies of sense data [39] or the Pearson Correlation Coefficient among their features [40] and leverages GNNs model in the well-established graph for performing their corresponding tasks. Furthermore, GNNs have been implemented to the tasks that contain graph knowledge, such as treating the skeleton of humans as graph in action recognition in industrial packing processes in computer vision tasks [41] and setting the geometric structures as the graph for the acoustic-based fault diagnosis [42].

It can be found that previous studies have gradually recognized the expandability and explainability of IKG and attempted to transplant it into several manufacturing scenarios to represent and organize massive heterogeneous knowledge resources. However, there still lies a long way to reach the high feasibility and efficacy, as the manufacturing scenario usually requires more solid logic in problem-solving but offers smaller labelled datasets for constructing knowledge graphs and graph neural networks. Moreover, practitioners usually regard IKG as an effective medium for querying and retrieving essential manufacturing knowledge, such as empirical rules and historical cases, but they stop before further utilizing them [43]. In fact, in the manufacturing scenario, the stored heterogeneous manufacturing "things" (e.g., cognitive machines, manufacturing services, materials, stakeholders) and all their complex in-between connections, should be thoroughly cognized with a semantic basis. Therefore, the fetched knowledge can directly drive, optimize, and evolve the highly automated manufacturing processes in a real 'Self-X' manner.

## 2.3 Deep reinforcement learning in manufacturing

With the rapid development of deep learning, DRL has achieved many remarkable successes in applications such as unmanned vehicles, robot learning control, and human-machine gaming [44]. Not until recently, a few researchers began to adopt its capabilities in the manufacturing field, among which mainly lie in the following two aspects.

*Assembly process*. Industrial robots are an important platform for exploiting DRL algorithms. The old-fashion human-involved programming method for assembly tasks lacks precision, adaptability, and flexibility towards the changeable productions. To address these challenges, Inoue et al. [45] depicted how to robustly perform peg-in-hole assembly tasks among tightly spaced holes by training industrial robots with recurrent neural network based DRL. Meanwhile, Schoettler et al. [46] combined DRL with a meta-learning approach to implement practical manufacturing spliced activities flexibly and adaptively. Except for dealing with assembling rigid objects, Luo et al.'s work [47] demonstrated that DRL can also be applied to mixed deformable and rigid objects with the support of sensing devices.

*Manufacturing process scheduling and resource allocation*. The generalization of DRL algorithms empowers the manufacturing systems by performing these tasks more quickly and precisely. In the cloud manufacturing system, Liang et al. [48] adopted Deep Q-Network with the standard of service quality to learn optimal service composition solutions in logistics. For the entity-based task scheduling scheme, Leng et al. [49] provided a decision-making solution, which is generated by Q-Learning, to reduce carbon consumption and improve material utilization during the whole manufacturing process. In addition to task scheduling, Huang et al. [50] used Double-DQN together with the system production loss, to design a preventive maintenance strategy for serial production lines as well.

However, with the ever-increasing manufacturing complexity and flexibility, applications of those single-agent DRL with centralized settings have encountered serious challenges on managing its scalability and flexibility with time efficiency. In this context, multi-agent reinforcement learning (MARL) can be a promising solution. For example, Gabel [51] interpreted the classic job-shop scheduling problems as distributed sequential decision-making problems by adopting the MARL method. Also, Roesch et al. [52] leveraged MARL to control the complex smart grid system of resources, battery storage, electricity self-supply, and short-term market trading. Nevertheless, despite limited contributions, the MARL has been seldom explored in the manufacturing process, let alone in a cognitive manner.

## 3. Methodology

Motivated by those aspects, this research introduces an IKG-based MARL approach for automatic manufacturing task fulfilment with self-configuration and self-optimization capabilities, towards the proposed Self-X cognitive manufacturing network. The flow chart of the proposed approach is shown in Figure 2, denoting the two main parts (i.e., IKG-based self-configurable manufacturing network, and MARL-enabled self-optimized manufacturing process) with its core procedures, to realize the self-configuration and self-optimization of multi-agent-based manufacturing task fulfilment, which are elaborated below.

### 3.1 IKG-based self-configurable manufacturing network

To establish the cognitive manufacturing network, an IKG is firstly established based on the extracted empirical knowledge and recognized patterns in the manufacturing process by leveraging the massive human-generated and machine-sensed multimodal data. Then, a graph neural network-based embedding algorithm is proposed based on the established IKG, to achieve task decomposition and configuration searching in a stepwise manner.

#### *3.1.1 IKG establishment*

IKG aims to describe the synergistic mechanism among different tasks, and further to decompose the manufacturing tasks and provides configuration space by considering their corresponding physical objects, functions, and constraints. Therefore, it should include, but not limited to the following type of nodes and edges, as listed in Table 1.

Table 1. The node and edge in a typical IKG

| **Node** | **Description** | **Edge** | **Description** |
|---|---|---|---|
| Task | The specific tasks and sub-tasks in the manufacturing | Has_Property | The edge between components or objects entities and their task |
| Object entity | The operable objects in the specific task | Attribute | The features or attributes belonging to the objects |
| Environment factor | Tasks' required environment, like temperature, location | Requirement | The task capabilities |
| Object template | The characteristics and attributes of the tasks' objects, like size, shape | Environment | The templates of the objects |
| Task function | Usage of each task, like assembly, disassembly | Has_function | The capabilities of different tasks |
| Task space | The restriction of the specific tasks | | |

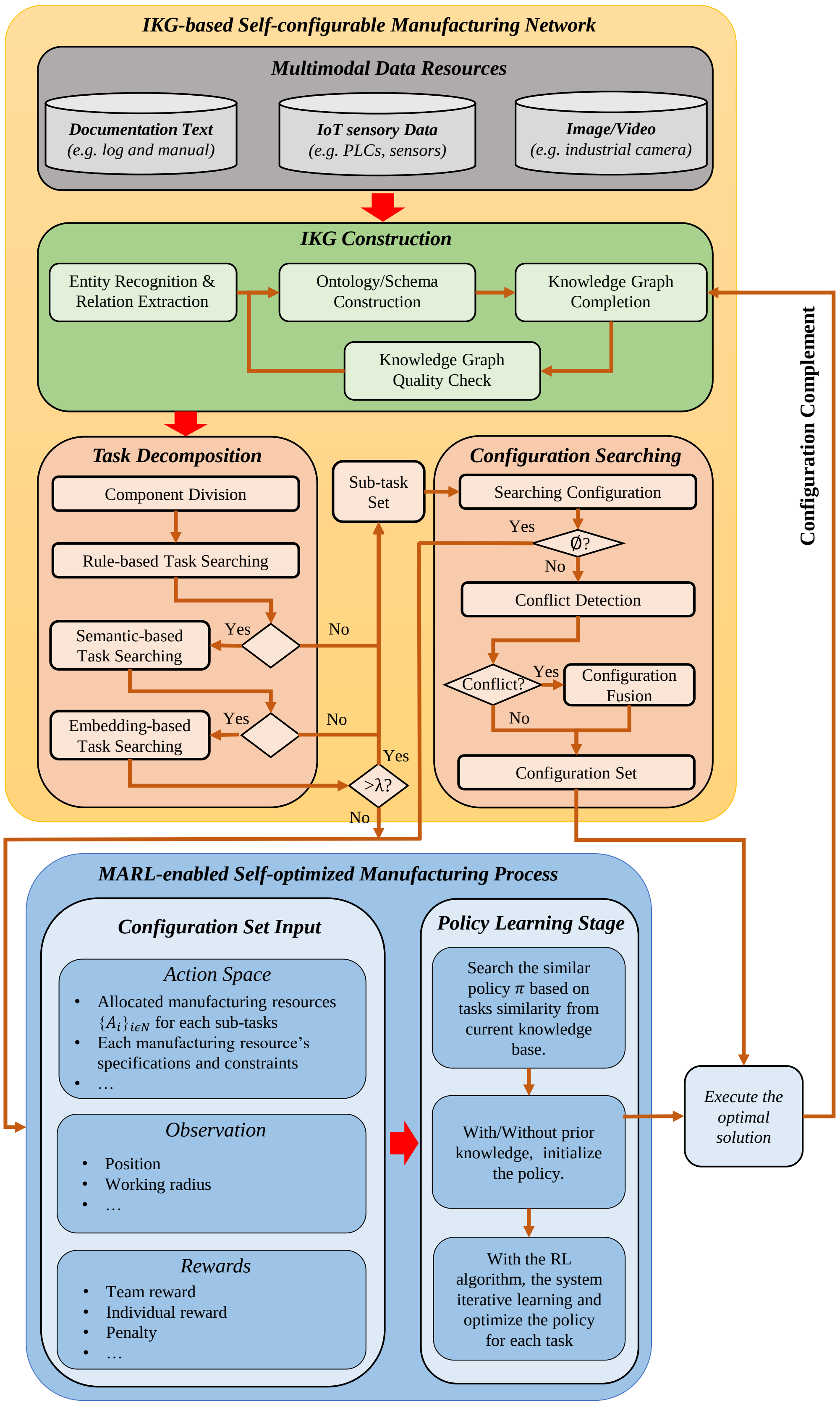

IKG-based Self-configurable Manufacturing Network
Multimodal Data Resources
Documentation Text (e.g. log and manual)
IoT sensory Data (e.g. PLCs, sensors)
Image/Video (e.g. industrial camera)
IKG Construction
Entity Recognition & Relation Extraction
Ontology/Schema Construction
Knowledge Graph Completion
Knowledge Graph Quality Check
Task Decomposition
Component Division
Rule-based Task Searching
Semantic-based Task Searching
Embedding-based Task Searching
Yes
No
Yes
No
Sub-task Set
>λ?
Yes
No
Configuration Searching
Searching Configuration
Yes
∅?
No
Conflict Detection
Conflict?
Yes
Configuration Fusion
No
Configuration Set
Configuration Complement
MARL-enabled Self-optimized Manufacturing Process
Configuration Set Input
Action Space
Allocated manufacturing resources $\{A_i\}_{i\epsilon N}$ for each sub-tasks
Each manufacturing resource's specifications and constraints
…
Observation
Position
Working radius
…
Rewards
Team reward
Individual reward
Penalty
…
Policy Learning Stage
Search the similar policy $\pi$ based on tasks similarity from current knowledge base.
With/Without prior knowledge, initialize the policy.
With the RL algorithm, the system iterative learning and optimize the policy for each task
Execute the optimal solution

Figure 2. The overall flowchart of the proposed IKG-based MARL approach.

Figure 3 depicts a generic flowchart for IKG construction. In manufacturing scenarios, the knowledge resource usually includes heterogeneous data sources, including IoT sensors, image, human-generated text, and domain-specific/open datasets. These resources are multimodal with different forms, and hence require separate processing and comprehensive fusion.

To tackle this problem, the knowledge extraction process combines with the manufacturing domain knowledge and terms as the keyword corpus. Based on the external knowledge, named entity recognition [53], relation extraction [54], and attribute extraction [55] are conducted effectively and targeted. Besides, the multisensory data is extracted to the knowledge in the same space through the manufacturing-based mapping framework. Additionally, the image data also requires customized processes, like transfer learning [56] and few-shot learning [57], due to the limited dataset. With the pre-defined nodes and edges in Table 1, the ontology/schema can be established. Meanwhile, the tabular knowledge should be integrated and compressed to improve the quality of knowledge. Apart from the grammatical refinement, entity disambiguation distinguishes the entities which have the same name but different meaning, differentiating them to their referent entities in a knowledge base [58]. On the contrary, entity resolution applies to the entities having different names but the same meaning, and they should be mapped to the corresponding correct entity [59].

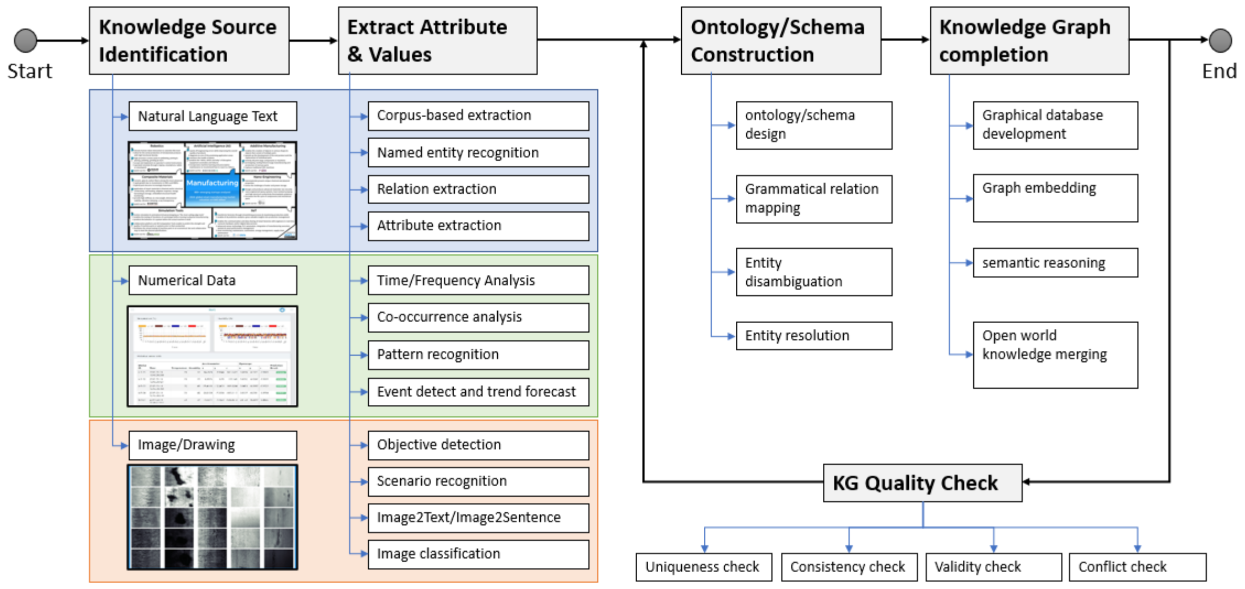


Figure 3. The proposed IKG construction process

Based on the refined knowledge, a GCN model is trained to transfer the knowledge into the same vector space. It enables an efficient storage structure and largely reduce the computational complexity in the subsequent reasoning process, which is a fundamental process for IKG establishment, denoted as [60]:

$$H^{l+1} = \sigma\left(\widetilde{D}^{-\frac{1}{2}}\tilde{A}\widetilde{D}^{-\frac{1}{2}}H^{l}W^{l}\right) \tag{1}$$

, where $\widetilde{D}$ is the degree matrix, $\tilde{A}$ is the adjacent matrix, $H^{l}$ is the node embedding matrix in layer $l$, $W^{l}$ is the trainable weight matrix in layer $l$. Besides, semantic reasoning establishes new edges in the knowledge graph. Additionally, merging knowledge updates the knowledge timely. As the manufacturing scenarios concern more about the knowledge quality rather than the quantity, IKG requires a quality check, which guarantees the uniqueness, consistency, validity, and free-of-conflict of the stored knowledge.

### *3.1.2 GCN-based task decomposition and configuration searching*

According to the IKG established above, the potential configuration space for different tasks can be searched. Task decomposition divides the overall manufacturing task into sub-tasks to discover any existing solutions to fulfil it. The core procedures of task decomposition are depicted in Figure 2, including rule-based methods, semantic-based methods, and embedding similarity-based methods for performing different searching tasks stepwise. If no sub-task set is found, it will go directly to Section 3.2. Otherwise, further configuration searching should be performed.

Configuration searching aims to find a suitable initial configuration space, including actions, rewards and observations for undertaking the manufacturing process. If there is one or more existing configurations performed before, conflict detection and configuration fusion will be performed, before the execution of the optimal solution. Otherwise, the manufacturing task will go directly to Section 3.2 as well.

However, it is often difficult to obtain the undisputed and unique configuration through querying the knowledge graph in reality. Therefore, a tailored graph-based algorithm is provided to calculate the best configuration among numerous configuration candidates and coordinate the ambiguous configuration space in different sub-tasks, as shown in **Algorithm 1**.

**Algorithm 1.** Pseudo code of task decomposition and configuration searching

```
Input: IKG, Manufacturing task
Output: Sub-tasks set (STS), Configuration set (CfgS)
Methods: Task decomposition and configuration searching
1    Load initial IKG, Manufacturing task (MT)
2    //Divide the MT into different sub-component as component set (CompS)
3    CompS = Divide_component(MT)
4    Initialize STS
5    For comp ∈ CompS do
6        // Task_search_rule: find the sub-task by rule-based method in IKG
7        Task = Task_search_rule(comp)
8        If Task != Null then
9            STS.add(Task)
10           Continue
11       Else
12           // Task_search_semantic: find the sub-task by semantic-based method
13           Task = Task_search_semantic(comp)
14       End if
```

```
    If Task != Null then
        STS.add(Task)
        Continue
    Else
        // Find_maxprob_task: find the most similar task and the corresponding probability
        Task, Task_prob = Find_maxprob_task(comp, Task_embedding set)
        If Task_prob larger than threshold $\phi$ then
            STS.add(Task)
        End if
    End if
End for
Initialize CfgS
For sub-task ∈ STS do
    // Find_configuration: search the configuration in the IKG
    cfg = Find_configuration(sub-task)
    // Conflict: search the configuration in the IKG
    If Conflict(cfg, CfgS) != Null then
        Conflicted_cfg = Conflict(cfg, CfgS)
        Fused_cfg = Fusion(cfg, Conflicted_cfg)
        If Fused_cfg != Null then
            CfgS.add(Fused_cfg)
        End if
    Else
        CfgS.add(cfg)
    End if
End for
Return STS, CfgS
```

According to **Algorithm 1**, the core terms are further explained as follows:

*Divide_component* is a function that divides the complex manufacturing tasks into corresponding components set by its ontology, schema, and systematic structure.

*Task_search_rule* is a fundamental function of task searching. Rule-based searching is a traditional approach to query the node by setting the condition, including the structural condition and feature condition. Such as treating the *Object entity* node as the component, and then searching the corresponding tasks based on the *Has_property* edge.

*Task_search_semantic* is a method that calculates the similarity of the candidate component with other *task* node by semantic understanding based on the graph embedding obtained from *H* matrix (1). The semantic similarity equation is as follow:

$$\vec{t} = argmin\left(\left\|\vec{h} + \vec{e} - \vec{t}\right\|^2\right) \tag{2}$$

, where $\vec{h}$ is the vector of object entity node, $\vec{e}$ is the vector of *has_property* edge, $\vec{t}$ is the vector of the possible *task* node.

*Find_maxprob_task* is a function that searches the most similar task by calculating the cosine similarity among the candidate component embedding with the embedding obtained from graph embedding algorithm Eq. (1). After the calculation, selecting the *task* node with the highest probability as the output. The cosine similarity, which serves as semantic distance, is as follow:

$$\mathrm{SEM\_D}(\overrightarrow{t_1}, \overrightarrow{t_2}) = \frac{\overrightarrow{t_1} \cdot \overrightarrow{t_2}}{\|\overrightarrow{t_1}\|\|\overrightarrow{t_2}\|} = \frac{\sum_{i=1}^{n} \overrightarrow{t_{1\imath}} \times \overrightarrow{t_{2\imath}}}{\sqrt{\sum_{i=1}^{n} (\overrightarrow{t_{1\imath}})^2} \times \sqrt{\sum_{i=1}^{n} (\overrightarrow{t_{2\imath}})^2}} \tag{3}$$

, where $\overrightarrow{t_{1i}}$ and $\overrightarrow{t_{2i}}$ represents the component vector of vector $t_1$ and $t_2$ respectively. However, if the highest similarity is still smaller than the threshold $\lambda$, the obtained task will not add into the sub-tasks set.

*Conflict* is a function to find whether the candidate configuration conflicts with the configurations the proposed algorithm had found before. Based on the semantic understanding and the knowledge graph structure, the function determines whether it is a conflict or not by:

$$\varphi = STR\_D(cfg, cfgs) - SEM\_D(cfg, cfgs) \tag{4}$$

, where STR_D calculates the structural distance based on the structure and their weight in the IKG by Dijkstra [61]. If the $\varphi$ is larger than the specific threshold $\rho$, it represents conflict exist, and otherwise not.

*Fusion* is a function that fuses the conflicting configuration. This function calculates the significance of different tasks, deciding the significant ranking of different corresponding configurations [62]. The node importance equation as follows:

$$\Pr(t_i) = (1 - d) + d * \sum_{i=1}^{m} \frac{\Pr(t_j)}{C(t_j)} \tag{5}$$

, where $Pr(t_j)$ represents the importance of node A, $C(t_j)$ is the edge number that node $t_i$ has, d presents the damping coefficient. This equation stops until it is convergent. Meanwhile, the community detection algorithm divides the graph into subgraphs [63], the integrated configuration fusion is combined with the importance ranking of configuration and the obtained subgraphs. The community detection algorithm is denoted as:

$$\Delta Q = \left[\frac{\sum_{in} + k_{i,in}}{2m} - \left(\frac{\sum_{tot} + k_i}{2m}\right)^2\right] - \left[\frac{\sum_{in}}{2m} - \left(\frac{\sum_{tot}}{2m}\right)^2 - \left(\frac{k_i}{2m}\right)^2\right] \tag{6}$$

, where in the community, $\sum_{in}$ is the summing of edges' weight, $\sum_{tot}$ is the summing of edges' weight of the nodes, $k_i$ is the edge's weight of node *i*, $k_{i,in}$ is the summing of edges' weight of the node *i* to the node, and *m* is the summing of edges' weight in the network, respectively.

*Find_configuration* is a function to search the configuration of the specific sub-task on the IKG, based on the semantic pathway established by the nodes of *Object entity* and edges of *Environment* and *Requirement*.

Following this manner, the self-configurable manufacturing network can be established in a semantic-based manner to allocate on-demand manufacturing resources for performing the task.

### 3.2 MARL-enabled self-optimized manufacturing process

In a multi-agent manufacturing system, physical manufacturing resources like robots, CNC machine tools, 3D-printers are represented as individual agents, and should be controlled in a decentralized manner. Hence, how to perform the collaboratively manufacturing tasks optimally remains a challenge. To overcome it, in this work, MARL is adopted to self-optimize the manufacturing process in a decentralized manner.

#### *3.2.1 Preliminary*

Compared to Markov Decision Processes (MDPs) of a single-agent reinforcement learning (SARL), MDPs are extended to the Partially Observable Markov Games (POMG) regarding the interaction among decentralized agents in MARL [64], of which the preliminaries are as follows:

A multi-agent POMG consisting of $N$ agents, is defined as a 4-tuple ($\{S_i\}_{i\epsilon N}$, $\{O_i\}_{i\epsilon N}$, $\{A_i\}_{i\epsilon N}$, $\{R_i\}_{i\epsilon N}$):

- $\{S_i\}_{i\epsilon N}$ : $S$ is a set of states describing the state of the environment and the current configuration of agents.
- $\{O_i\}_{i\epsilon N}$: $O$ is a set of observation spaces, and each agent gains their local observation from the global states $S$.
- $\{A_i\}_{i\epsilon N}$: $A$ is a set of action spaces describing the potential individual action space of each agent.
- $\{R_i\}_{i\epsilon N}$: $R$ is a set of immediate rewards gained by each agent after interacting with the environment.

In practice, the action of agent $i$ is sampled from a stochastic policy $\pi_{\theta \mathrm{i}}$ parameterized by $\theta_\mathrm{i}$ in the current state. The policy of each agent $\pi_{\theta \mathrm{i}}$: $O_i \times A_i$ chooses the action from corresponding agents' action space with the local observation of the agent. Meanwhile, all the agents execute their actions in the same state, and the next state is produced by the transition function $T$: $S_t \times A_1 \times \cdots \times A_N \rightarrow S_{t+1}$. The agents could gain their next partial observation $O_\mathrm{i}$ : $S \rightarrow O_\mathrm{i}$. In addition to state, the immediate reward for each agent can also be gained from the reward function $R_\mathrm{i}$ : $S_t \times A_1 \times \cdots \times A_N \times S_{t+1} \rightarrow \mathbb{R}$. In general, all the agents aim to maximize their cumulative rewards: $R_\mathrm{i} = \sum_{t=0}^{t=H} \gamma^t r_i^t$, where $\gamma$ is the discount factor and $H$ is the time horizon of the task.

#### *3.2.2 MARL approach*

To achieve the self-optimization capability, an ideal strategy generated by the MARL approach should have the following properties to accomplish the tasks [65]:

- *Cooperation*, which enables each agent to not only accomplish its own tasks but also not to prevent others from accomplishing their tasks.
- *Flexibility*, which can be efficiently adapted to dynamic layout (i.e., different numbers or positions of agents) of manufacturing scenarios.

To achieve it, in this work, an integration of Independent Learning (IL) [66] and Soft Actor-Critic (SAC) [67] is chosen, which is depicted in **Algorithm 2**. Essentially, IL is to apply SARL method to each agent in the multi-agent manufacturing environment. Specifically, from lines 9–12, the code represents centralized training and decentralized execution learning mechanism of IL, which is shown in Figure 3. In other words, not only the model structure and parameters of integrated SARL are the same and shared, but also the data from each agent is collected to update the model together [68], which increases the extensibility and generalization of the IL.

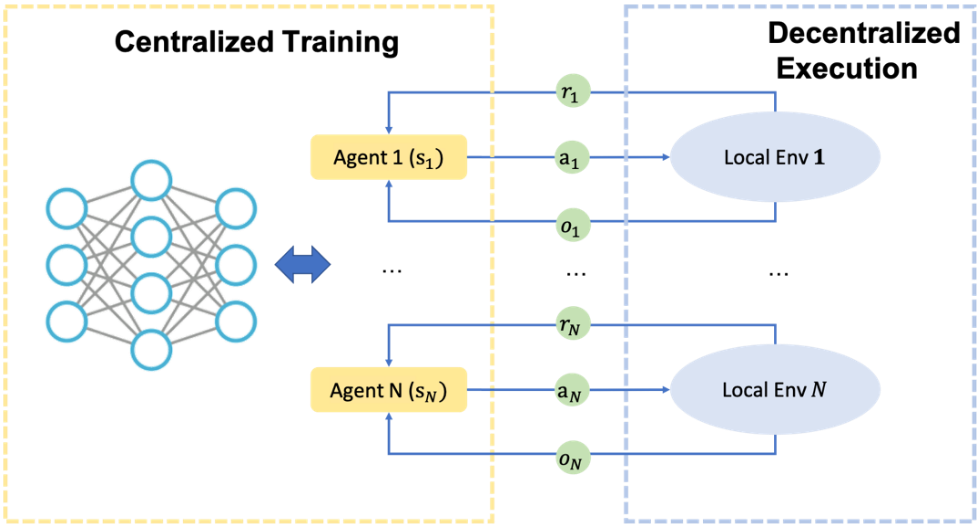


Figure 4. The centralized training and decentralized execution mechanism.

Furthermore, lines 14 to 19 of **Algorithm 2** depicts the MARL learning phase with SARL. The SAC RL algorithm has been widely adopted in a variety of benchmarks and real robot control tasks [69], which not only can optimize policy to obtain higher cumulative reward but also maximize the entropy of the policy to lead the policy as diverge as possible. The SAC policy function together with the consideration of entropy, is defined as follows:

$$\pi^*_{\text{MaxEnt}} = \arg\max_{\pi} \sum_t E_{(s_t,a_t)\sim\rho_\pi}\left[r(s_t,a_t) + \alpha H\big(\pi(\cdot|\ s_t\ )\big)\right] \tag{7}$$

, where the H refers to the entropy $H(P) = \underset{x\sim p}{E}[-\log P\,(x)]$, and $\alpha$ is the term to control the importance of entropy. $\alpha$ has a significant impact on performance for different tasks. Therefore, the authors construct setting $\alpha$ as a constrained optimization problem, while maximizing the expected cumulative reward and maintaining the policy entropy is greater than a threshold $\mathcal{H}_0$.

$$\underset{\pi_0,\dots,\pi_T}{max}\ E\left[\sum_{t=0}^{T} r(s_t,a_t)\right] s.t.\ \forall t,\ \mathcal{H}(\pi_t) \geq \mathcal{H}_0 \tag{8}$$

According to Eq. (8), the loss function of optimizing $\alpha$ is as follows:

$$J(\alpha) = E_{a_t \sim \pi_t}[-\alpha \log \pi_t \,(a_t \mid \pi_t) - \alpha \mathcal{H}_0] \tag{9}$$

The SAC implementation mainly consists of two functions, i.e., critic (value) function $Q_\theta(\mathrm{s}, \mathrm{a})$ and actor (policy) function $\pi_\phi(\mathrm{a}|\mathrm{s})$. The value function is used for estimating the action value of manufacturing agent in a state. The policy network outputs an action distribution used for sampling deterministic action when it receives state representation inputs. Both functions are approximated by a multi-layer neural network individually. The value function and the loss (objective) function used for optimizing value functions are defined as follows:

$$Q^{\pi}_{\text{soft}}(s,\, a) = \underset{s_t, a_t \sim\, \rho_\pi}{E}\left[\textstyle\sum_{t=0}^{\infty} \gamma^t\, r(s_t,\, a_t) + \alpha \sum_{t=1}^{\infty} \gamma^t\, H\big(\pi(\cdot \mid s_t)\big) \mid s_0 = s,\, a_0 = a\right] \tag{10}$$

$$J_Q(\theta) = \underset{\substack{(s_t, a_t, s_{t+1}) \sim \mathcal{D}, \\ a_{t-1} \sim \pi_\phi}}{E}\left[\frac{1}{2}\left(Q_\theta(s_t, a_t) - \left(r(s_t, a_t) + \gamma\left(Q_\theta(s_{t+1}, a_{t+1}) - \alpha \log\left(\pi_\phi(a_{t+1} \mid s_{t+1})\right)\right)\right)\right)^2\right] \tag{11}$$

Furthermore, the representation of policy $\pi$ has been discussed in Eq. (7). The loss function of policy, measured by Kullback–Leibler divergence and used for updating actor network, is stated in Eq. (11). In the KL-divergence term, since the $\mathcal{Z}$ is not affected by the parameter $\phi$ when deriving the parameters of the policy, it is ignored when requiring gradients.

$$J_\pi(\phi) = D_{\text{KL}}\left(\pi_\phi(\cdot \mid s_t) \,|\, \exp\left(\frac{1}{\alpha} Q_\theta(s_t, \cdot) - \log \mathcal{Z}(s_t)\right)\right) \tag{12}$$

With the above two loss functions (Eq. (11) and Eq. (12)), the manufacturing agent could collect data via interaction with environment, and optimize these loss functions to tune the parameters by multi-layer neural network in advance. In addition to these, there still exists tricks to improve performance borrowed from SAC, such as double Q-network and target network [70].

**Algorithm 2.** Pseudo code of the proposed MARL approach

```
1    Initialize parameters of policy network $\phi$
2    Initialize manufacturing agent target network parameters $\theta_1, \theta_2$
3    Initialize an experience replay buffer D ⟵ $\emptyset$
4    Initialize learning rates $\lambda_Q$, $\lambda_\pi$, $\lambda$
5    Initialize weighting factor $\tau$ for exponential moving average
6    For episode = 1 to $N$ do
7        Observe initial state $s$
8        For each environment step from h = 1 to $H$ do
9            For each manufacturing agent $i$, it observes an initial observation $o_{i,h}$ and samples
     an action $a_{i,h}$ according to current policy. Then agent $i$ gains the reward
     $r_{i,h}$ and next observation $o_{i,h+1}$.
10       Store the transition ($o_{i,h}$, $a_{i,h}$, $r_{i,h}$, $o_{i,h+1}$) in replay buffer D.
```

```
11          $o_{i,h} \leftarrow o_{i,h+1}$
12          End for
13          For each gradient step do
14              Update Q-value network $\theta_j \leftarrow \theta_j - \lambda_Q \widehat{\nabla_{\theta_j}} J_Q(\theta_j)$ for $j \in \{1,2\}$
15              Update policy network weights $\phi \leftarrow \phi - \lambda_\pi \widehat{\nabla_\phi} J_\pi(\phi)$
16              Adjust temperature $\alpha \leftarrow \alpha - \lambda \widehat{\nabla_\alpha} J(\alpha)$
17              Update target network $\overline{\theta_i} \leftarrow \tau\theta_i + (1-\tau)\,\overline{\theta_i}$ for $i \in \{1,2\}$
18          End for
19      End for
20      Return $\phi, \theta_1, \theta_2$
```

3.3 Complement of cognitive manufacturing network

An optimal policy obtained from the MARL contains information about environmental factors, task spaces, etc., which are different from the existing cases in the IKG-based configuration set. Hence, knowledge complement should be further conducted to store the newly generated configurations as its nodes and edges. Meanwhile, optimized solutions should be updated in the existing IKG dynamically. Based on the performance in MARL of the IKG-based configuration, the entity resolution and alignment technologies enable to fuse the similar configurations to the most suitable one with their corresponding condition. The fusion is based on the concatenated vector of the node embedding and its first order nodes' embedding as the representative vector.

$$V_i = concat(c_i \| CS_i) \tag{13}$$

, where $c_i$ is the embedding of node *i*, $CS_i$ is the average of first-order nodes' embedding, $V_i$ is the integrated vector to represent node *i*. Comparing the semantic distance obtained from (3) with the standard threshold to determine whether to fuse these two nodes or not. If it needs to be fused, the configuration with better performance in multi-agent RL will be reserved.

## 4. An illustrative example

To depict the feasibility of the proposed method to stepwise achieve the Self-X cognitive manufacturing network, an illustrative example of simulated multi-robot reaching task is carried out. The multi-agent manufacturing system setting is simplified to the unified multiple UR5 robots as the agents, accomplishing the same underlying task, i.e., reaching their respective target poses adaptively without any collision. It is worth noting that this task can be generalized to perform various advanced-level manipulation tasks (e.g., painting, welding, assembly) as well. According to the aforementioned core procedures, it mainly includes four steps, as shown in Figure 5.

*Step 1: IKG for cognitive manufacturing network establishment.* In this example, the holistic IKG of multi-robot manufacturing resources is firstly established in the Neo4j environment as described in Table I and depicted in Figure 5, including the empirical knowledge (e.g., successful cases), constraints (e.g., model conflict) and technical specifications (e.g., working radius), as the prerequisite, to enable the multi-robot task decomposition and configuration searching.

*Step 2: Prior rules offered by IKG querying.* Based on the IKG, the querying process based on **Algorithm 1** can be conducted. In the multi-robot reaching task, according to the multiple UR5 robots' schema and architecture, the *Divide_component function* decomposes it into different kinds of sub-components as a component set, including shoulder, wrist, elbow, gripper, etc. With the obtained component set, searching their corresponding tasks of each component as subtasks set. In the sub-tasks set, the wrist has its rotation task, elbow has its supporting task, etc. Part of these tasks can be achieved through precise querying or fuzzy querying in the IKG (*Task_search_rule*), while other potential and undirected tasks can be obtained by semantic searching with the graph embedding (*Task_search_semantic*). Nevertheless, in some extreme cases, previous methods fail to find their task, then *Find_maxprob_task* can be leveraged to find the most similar task as the potential task. For instance, the multiple UR5 robots encounter a new object to pick, which requires the IKG to seek similar tasks in other relevant nodes with historical records. Based on the sub-tasks set, their corresponding configuration of each task can be achieved through querying (*Find_configuration*) in IKG as a configuration set. However, those obtained configurations set may have inner redundancy or restrictions. For example, the gripper has two tasks to place the object in similar coordinates, or two different grippers have the same pick and place task in the same place and same time. To avoid any conflict and to generate an integrated and systematic configuration set, the configuration set should be disambiguation (*Conflict*) and alignment (*Fusion*). With these configuration stilling processes, the obtained feasible configuration of "*4 UR5 robots to perform their corresponding target pose without collision in a compact workspace*" is retrieved, and formatted as the input of MARL, as follows:

• **Reward:** The reward function consists of three components, i.e., group reward, individual reward, and penalty term.

*Group Reward***.** The manufacturing system is rewarded only when all manufacturing agents achieve their corresponding goals.

*Individual Reward.* Each manufacturing agent receives a reward when it individually reaches the goal.

*Penalty Term.* As the manufacturing system has collisions, it gains a penalty.

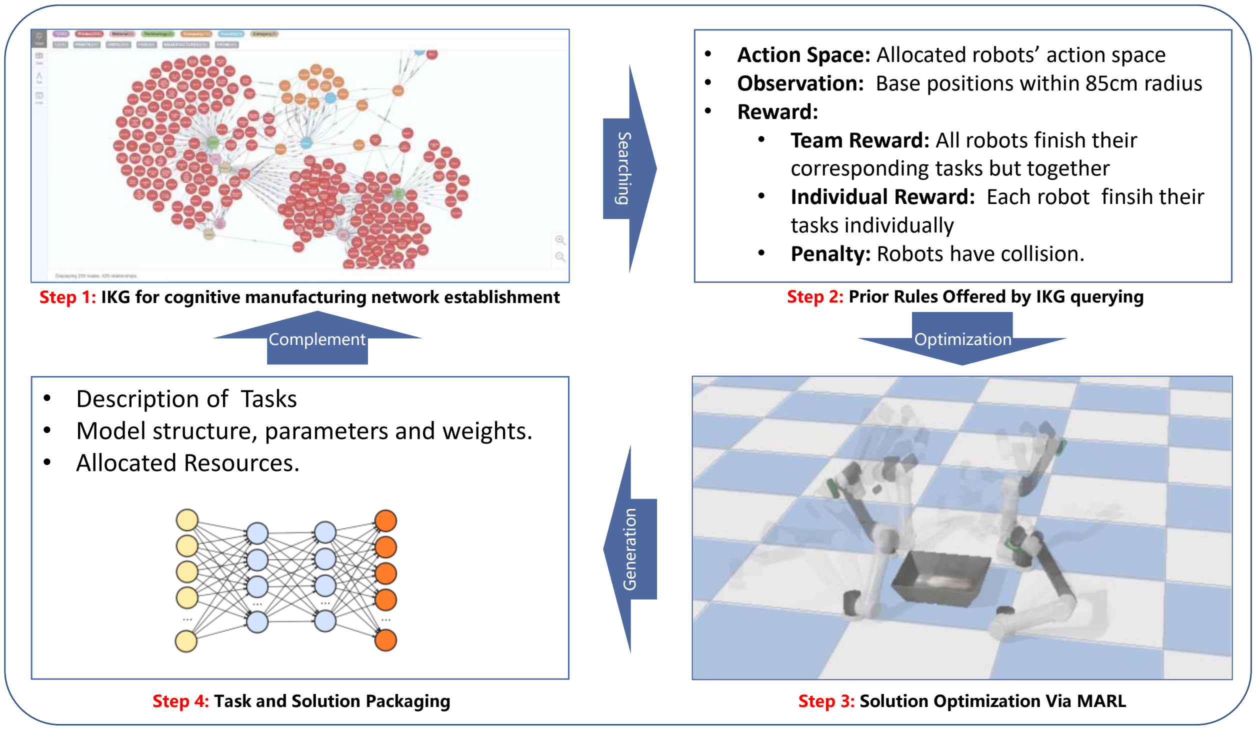
• **Observation:** Individual position of each UR robot within 85 cm radius.

• **Action Space**: 1) Degrees of Freedom (6 rotating joints); 2) Payload (5 kg); 3) Repeatability (±0.1 mm / ±0.0039 in); 4) Weight with cable (18.4 kg / 40.6 lbs); 5) Reach (850 mm); 6) Motion Range: ±360°; 7) Maximum speed: ± 180°/Sec

*Step 3: Solution optimization via MARL.* With above settings as input, the MARL can be applied based on **Algorithm 2** to search the optimal solution for the reaching task. In this case, the manufacturing environment is carried out by PyBullet [68] and the simulations are performed on a laptop with 2.3 GHz 8-core i9-9980H CPU, 32 GB RAM. The motion trajectory of the four UR5 robots in Step 3 is depicted in Figure 5, which is screen captured by combining multiple sampled frames during the robot's movement.

*Step 4: Task and solution packaging.* After acquiring the optimal solution via MARL, the characteristics of the task and the feasible solution, including documentation of tasks, model structure, parameters and weights, and manufacturing resources, are packaged and further manually backpropagated to IKG, which improves the completeness and diversity of IKG.

*Discussion.* Following the above-mentioned steps, a self-configurable and self-optimizing multi-robot manufacturing network based on different tasks can be realized, which can dynamically adapt its solution space to generate new solutions to accomplish new tasks. It is worth noting that, although the collision-free completion of the reaching task by robots is a basic motion planning task, application of this task can be widely applied to perform a variety of other advanced, complex manufacturing tasks. For example, in the disassembly task of an ageing energy vehicle battery, multiple robots are employed to collaborate to disassemble the entire battery component and different robots take the duty of disassembling and operating different parts of the battery. In this series of tasks, the types of batteries are sundry, the robot disassembly system may employ different numbers of robots to complete the disassembly task. Therefore, the scalability of the MARL acquired in the above robot reaching experiment to meet such requirements. Secondly, the machine vision system can provide each robot the pose and position information of the target part. However, when multiple robots complete their tasks in parallel, the high integration of automotive batteries makes the operating environment crowded. Hence, the flexible collision-free motion planning policy derived from MARL can effectively solve this problem. Finally, since different external indexes may make the target indicators of different disassembly tasks different (time, cost, and so on), the IKG approach can automatically adjust the proportion and composition of the shaped reward of MARL to better meet the expectations.



Step 1: IKG for cognitive manufacturing network establishment
Searching
Action Space: Allocated robots' action space
Observation: Base positions within 85cm radius
Reward:
Team Reward: All robots finish their corresponding tasks but together
Individual Reward: Each robot finsih their tasks individually
Penalty: Robots have collision.
Step 2: Prior Rules Offered by IKG querying
Complement
Optimization
Description of Tasks
Model structure, parameters and weights.
Allocated Resources.
Generation
Step 4: Task and Solution Packaging
Step 3: Solution Optimization Via MARL

Figure 5. Core steps of the illustrative example

## 5. Conclusion

Enabled by the prevailing implementation of human-level information processing of cognitive computing, IIoT and big data analytics, today's manufacturing systems are rapidly shifting towards a Self-X cognitive manufacturing network for achieving cognitive mass personalization. Nevertheless, it still lies a long way to readily achieve its Self-X capabilities. To address the issue, as an explorative study, this research proposes a systematic IKG-based MARL-enabled approach to support the self-configuration and self-optimization of multi-agent-based manufacturing systems in a network-based manner. An illustrative example of multi-robot collaboration is also provided to validate the feasibility of the proposed approach. The main scientific contributions of this work can be summarized into two aspects, namely:

(1) *IKG and graph-embedding techniques for self-configurable cognitive manufacturing network establishment*. The re-/configuration of manufacturing "things" can be performed by the well-defined knowledge extraction from multimodal data, IKG establishment, task decomposition and configuration searching procedures in a semantic-based manner. Meanwhile, with the interoperability of standardized M2M communication protocol (e.g., OPC UA) in a IIoT environment, on-demand manufacturing resources can be allocated efficiently.

(2) *MARL-enabled self-optimized decentralized manufacturing process.* The MARL is formed as the integration of IL and SAC, where decentralized manufacturing tasks can be fulfilled by self-optimizing the manufacturing process, and further to complement the IKG towards Self-X cognitive manufacturing network eventually.

Despite these achievements, there still lies some limitations in this research. For instance, the automated establishment and dynamic evolvement of the IKG (e.g., knowledge complement) remains a challenge. Meanwhile, in this research, the mass personalization issue is mainly concerned by the self-organizing capabilities of available manufacturing resources on-site, while distributed manufacturing scenarios are not involved. Moreover, the self-healing/adaptive capabilities to actively identify any variations and autonomously dealing with potential disruptions without human interventions, are still far to achieve with the current level of manufacturing/robot learning intelligence. Therefore, it is envisioned that future works of Self-X cognitive manufacturing network can be done in the following aspects: 1) *Ubiquitous semantical connections*, with an IKG-based organization on massive heterogeneous entities, all stakeholders and all manufacturing resources can be interlinked and interacted in the manufacturing network with semantic-rich and logic-solid relations even in the geographically distributed manufacturing environment for mass personalization; (2) *Human-machine co-evolvement*, as semantic gap bridged by IKG, human operators can break their thinking-sets with creative insights discovered by the cognitive machines; and (3) *Scalable MARL for decentralized multi-agent manufacturing systems.* Compared with the centralized training and decentralized execution mechanism, robust strategies should be brought up to retrain the policy with varying scale of multiple agents.

**Acknowledgements**

The authors wish to acknowledge the funding support from the National Natural Research Foundation of China (No. 52005424), Mainland-Hong Kong Joint Funding Scheme (MHX/001/20), Innovation and Technology Commission, HKSAR, China, National Key R&D Programs of Cooperation on Scientific and Technological Innovation in Hong Kong, Macao and Taiwan (SQ2020YFE020182), and Jiangsu Provincial Policy Guidance Program (Hong Kong/Macau/Taiwan Science and Technology Cooperation, BZ2020049).

## Reference

[1] Tseng MM, Jiao RJ, Wang C. Design for mass personalization. CIRP Ann - Manuf Technol 2010;59:175–8. https://doi.org/10.1016/j.cirp.2010.03.097.

[2] Koren Y, Shpitalni M, Gu P, Hu SJ. Product design for mass-individualization. Procedia CIRP 2015, 36: 64–71. https://doi.org/10.1016/j.procir.2015.03.050.

[3] Zheng P, Wang H, Sang Z, Zhong RY, Liu Y, Liu C, et al. Smart manufacturing systems for Industry 4.0: Conceptual framework, scenarios, and future perspectives. Front Mech Eng 2018. https://doi.org/10.1007/s11465-018-0499-5.

[4] Tao F, Qi Q, Liu A, Kusiak A. Data-driven smart manufacturing. J Manuf Syst 2018. https://doi.org/10.1016/j.jmsy.2018.01.006.

[5] Lee J, Bagheri B, Kao HA. A Cyber-Physical Systems architecture for Industry 4.0-based manufacturing systems. Manuf Lett 2015, 3: 18–23. https://doi.org/10.1016/j.mfglet.2014.12.001.

[6] Mills K. What Is Cognitive Manufacturing? MetrologyNews 2019.

[7] Kumar A, Jaiswal A. A Deep Swarm-Optimized Model for Leveraging Industrial Data Analytics in Cognitive Manufacturing. IEEE Trans Ind Informatics 2021, 17: 2938–2946. https://doi.org/10.1109/TII.2020.3005532.

[8] Modha DS, Ananthanarayanan R, Esser SK, Ndirango A, Sherbondy AJ, Singh R. Cognitive computing. Commun ACM 2011. https://doi.org/10.1145/1978542.1978559.

[9] Qu Y, Pokhrel SR, Garg S, Gao L, Xiang Y. A Blockchained Federated Learning Framework for Cognitive Computing in Industry 4.0 Networks. IEEE Trans Ind Informatics 2021, 17: 2964–2973. https://doi.org/10.1109/TII.2020.3007817.

[10] Zheng P, Liu Y, Tao F, Wang Z, Chen C-H. Smart Product-Service Systems Solution Design via Hybrid Crowd Sensing Approach. IEEE Access 2019, 7: 128463–128473. https://doi.org/10.1109/access.2019.2939828.

[11] Chen M, Herrera F, Hwang K. Cognitive Computing: Architecture, Technologies and Intelligent Applications. IEEE Access 2018, 6: 19774–19783. https://doi.org/10.1109/ACCESS.2018.2791469.

[12] Sisinni E, Saifullah A, Han S, Jennehag U, Gidlund M. Industrial internet of things: Challenges, opportunities, and directions. IEEE Trans Ind Informatics 2018, 14: 4724–34. https://doi.org/10.1109/TII.2018.2852491.

[13] Xu X. From cloud computing to cloud manufacturing. Robot Comput Integr Manuf 2012;28:75–86. https://doi.org/10.1016/J.RCIM.2011.07.002.

[14] Zhang H, Liu Q, Chen X, Zhang D, Leng J. A Digital Twin-Based Approach for Designing and Multi-Objective Optimization of Hollow Glass Production Line. IEEE Access 2017, 5: 26901–11. https://doi.org/10.1109/ACCESS.2017.2766453.

[15] Wang J, Ma Y, Zhang L, Gao RX, Wu D. Deep learning for smart manufacturing: Methods and applications. J Manuf Syst 2018:1–13. https://doi.org/10.1016/j.jmsy.2018.01.003.

[16] Hu SJ. Evolving paradigms of manufacturing: From mass production to mass customization and personalization. Procedia CIRP 2013, 7: 3–8.

https://doi.org/10.1016/j.procir.2013.05.002.

[17] Koren Y, Shpitalni M. Design of reconfigurable manufacturing systems. J Manuf Syst 2010, 29: 130–141. https://doi.org/10.1016/j.jmsy.2011.01.001.

[18] Zheng P, Xu X, Yu S, Liu C. Personalized configuration framework in an adaptable open architecture product platform. J Manuf Syst 2017, 43: 422–435. https://doi.org/10.1016/j.jmsy.2017.03.010.

[19] Zheng P, Wang Z, Chen C-H, Pheng Khoo L. A survey of smart product-service systems: Key aspects, challenges and future perspectives. Adv Eng Informatics 2019, 42: 100973. https://doi.org/10.1016/j.aei.2019.100973.

[20] Iarovyi S, Lastra JLM, Haber R, Del Toro R. From artificial cognitive systems and open architectures to cognitive manufacturing systems. Proceeding - 2015 IEEE Int. Conf. Ind. Informatics, INDIN 2015, 2015. https://doi.org/10.1109/INDIN.2015.7281910.

[21] Wang S, Wan J, Zhang D, Li D, Zhang C. Towards smart factory for Industry 4.0: A self-organized multi-agent system with big data based feedback and coordination. Comput Networks 2015, 101: 158–168. https://doi.org/10.1016/j.comnet.2015.12.017.

[22] Lee J, Ardakani HD, Yang S, Bagheri B. Industrial Big Data Analytics and Cyber-physical Systems for Future Maintenance & Service Innovation. Procedia CIRP 2015, 38: 3–7. https://doi.org/10.1016/j.procir.2015.08.026.

[23] Zhang Y, Qian C, Lv J, Liu Y. Agent and cyber-physical system based self-organizing and self-adaptive intelligent shopfloor. IEEE Trans Ind Informatics 2017, 13: 737–747. https://doi.org/10.1109/TII.2016.2618892.

[24] Bortolini M, Galizia FG, Mora C. Reconfigurable manufacturing systems: Literature review and research trend. J Manuf Syst 2018, 49: 93–106. https://doi.org/10.1016/j.jmsy.2018.09.005.

[25] Lim KYH, Zheng P, Chen C, Huang L. A digital twin-enhanced system for engineering product family design and optimization. J Manuf Syst 2020, 57: 82–93. https://doi.org/10.1016/j.jmsy.2020.08.011.

[26] Lu Y, Xu X, Wang L. Smart manufacturing process and system automation – A critical review of the standards and envisioned scenarios. J Manuf Syst 2020, 56: 312–325. https://doi.org/10.1016/j.jmsy.2020.06.010.

[27] Leng J, Jiang P, Liu C, Wang C. Contextual self-organizing of manufacturing process for mass individualization: a cyber-physical-social system approach. Enterp Inf Syst 2020, 14: 1124–1149. https://doi.org/10.1080/17517575.2018.1470259.

[28] He L, Jiang P. Manufacturing knowledge graph: a connectivism to answer production problems query with knowledge reuse. IEEE Access 2019, 7: 101231–101244.

[29] Li X, Chen C-H, Zheng P, Wang Z, Jiang Z, Jiang Z. A Knowledge Graph-Aided Concept–Knowledge Approach for Evolutionary Smart Product–Service System Development. J Mech Des 2020. https://doi.org/10.1115/1.4046807.

[30] Wang Z, Chen CH, Zheng P, Li X, Khoo LP. A graph-based context-aware requirement elicitation approach in smart product-service systems. Int J Prod Res 2019: 1–17.

https://doi.org/10.1080/00207543.2019.1702227.

[31] Li X, Zhang S, Huang R, Huang B, Xu C, Kuang B. Structured modeling of heterogeneous CAM model based on process knowledge graph. Int J Adv Manuf Technol 2018, 96: 4173–93.

[32] Potes Ruiz PA, Kamsu-Foguem B, Noyes D. Knowledge reuse integrating the collaboration from experts in industrial maintenance management. Knowledge-Based Syst 2013, 50:171–86. https://doi.org/10.1016/j.knosys.2013.06.005.

[33] Alsafi Y, Vyatkin V. Ontology-based reconfiguration agent for intelligent mechatronic systems in flexible manufacturing. Robot Comput Integr Manuf 2010, 26: 381–91. https://doi.org/10.1016/j.rcim.2009.12.001.

[34] Hedberg TD, Bajaj M, Camelio JA. Using Graphs to Link Data Across the Product Lifecycle for Enabling Smart Manufacturing Digital Threads. J Comput Inf Sci Eng 2020, 20: 1–15. https://doi.org/10.1115/1.4044921.

[35] Tiacci L. Object-oriented event-graph modeling formalism to simulate manufacturing systems in the Industry 4.0 era. vol. 99. Elsevier B.V.; 2020. https://doi.org/10.1016/j.simpat.2019.102027.

[36] Goyal P, Ferrara E. Graph embedding techniques, applications, and performance: A survey. Knowledge-Based Syst 2018, 151: 78–94. https://doi.org/10.1016/j.knosys.2018.03.022.

[37] Zhou J, Cui G, Zhang Z, Yang C, Liu Z, Wang L, et al. Graph Neural Networks: A Review of Methods and Applications n.d.:1–22.

[38] Hu L, Liu Z, Hu W, Wang Y, Tan J, Wu F. Petri-net-based dynamic scheduling of flexible manufacturing system via deep reinforcement learning with graph convolutional network. J Manuf Syst 2020,55: 1–14. https://doi.org/10.1016/j.jmsy.2020.02.004.

[39] Narwariya J, Malhotra P, TV V, Vig L, Shroff G. Graph Neural Networks for Leveraging Industrial Equipment Structure: An application to Remaining Useful Life Estimation 2018.

[40] Wang M, Li Y, Zhang Y, Jia L. Spatio-temporal graph convolutional neural network for remaining useful life estimation of aircraft engines. Aerosp Syst 2020. https://doi.org/10.1007/s42401-020-00070-x.

[41] Chen Z, Hu H, Li Z, Qi X, Zhang H, Hu H, et al. Skeleton-based Action Recognition for Industrial Packing Process. IoTBDS, 2020, p. 36–45.

[42] Zhang D, Stewart E, Entezami M, Roberts C, Yu D. Intelligent acoustic-based fault diagnosis of roller bearings using a deep graph convolutional network. Measurement 2020, 156: 107585.

[43] Li X, Lyu M, Wang Z, Chen C-H, Zheng P. Exploiting Knowledge Graphs in Industrial Products and Services: A Survey of Key Aspects, Challenges, and Future Perspectives. Comput Ind 2021.

[44] Arulkumaran K, Deisenroth MP, Brundage M, Bharath AA. Deep reinforcement learning: A brief survey. IEEE Signal Process Mag 2017, 34: 26–38. https://doi.org/10.1109/MSP.2017.2743240.

[45] Inoue T, De Magistris G, Munawar A, Yokoya T, Tachibana R. Deep reinforcement

learning for high precision assembly tasks. IEEE Int. Conf. Intell. Robot. Syst., 2017. https://doi.org/10.1109/IROS.2017.8202244.

[46] Schoettler G, Nair A, Ojea JA, Levine S, Solowjow E. Meta-reinforcement learning for robotic industrial insertion tasks. ArXiv 2020. https://doi.org/10.1109/IROS45743.2020.9340848.

[47] Luo J, Solowjow E, Wen C, Ojea JA, Agogino AM. Deep Reinforcement Learning for Robotic Assembly of Mixed Deformable and Rigid Objects. IEEE Int. Conf. Intell. Robot. Syst., 2018. https://doi.org/10.1109/IROS.2018.8594353.

[48] Liang H, Wen X, Liu Y, Zhang H, Zhang L, Wang L. Logistics-involved QoS-aware service composition in cloud manufacturing with deep reinforcement learning. Robot Comput Integr Manuf 2020, 67: 101991. https://doi.org/10.1016/j.rcim.2020.101991.

[49] Leng J, Ruan G, Song Y, Liu Q, Fu Y, Ding K, et al. A loosely-coupled deep reinforcement learning approach for order acceptance decision of mass-individualized printed circuit board manufacturing in industry 4.0. J Clean Prod 2021, 280: 124405. https://doi.org/10.1016/j.jclepro.2020.124405.

[50] Huang J, Chang Q, Arinez J. Deep reinforcement learning based preventive maintenance policy for serial production lines. Expert Syst Appl 2020, 160: 113701. https://doi.org/10.1016/j.eswa.2020.113701.

[51] Gabel T. Multi-Agent Reinforcement Learning Approaches for Distributed Job-Shop Scheduling Problems. Univ Osnabruck, Ger 2009:173.

[52] Roesch M, Linder C, Zimmermann R, Rudolf A, Hohmann A, Reinhart G. Smart grid for industry using multi-agent reinforcement learning. Appl Sci 2020, 10: 1–20. https://doi.org/10.3390/app10196900.

[53] Yadav V, Bethard S. A Survey on Recent Advances in Named Entity Recognition from Deep Learning models. ArXiv 2019.

[54] Peng N, Poon H, Quirk C, Toutanova K, Yih WT. Cross-sentence N-ary relation extraction with graph LSTMs. ArXiv 2017, 5: 101–115. https://doi.org/10.1162/tacl_a_00049.

[55] Sun L. Research on product attribute extraction and classification method for online review. 2017 Int. Conf. Ind. Informatics-Computing Technol. Intell. Technol. Ind. Inf. Integr., 2017, p. 117–21.

[56] Zhuang F, Qi Z, Duan K, Xi D, Zhu Y, Zhu H, et al. A comprehensive survey on transfer learning. Proc IEEE 2020, 109: 43–76.

[57] Sung F, Yang Y, Zhang L, Xiang T, Torr PHS, Hospedales TM. Learning to compare: Relation network for few-shot learning. Proc. IEEE Conf. Comput. Vis. pattern Recognit., 2018, p. 1199–208.

[58] Zhu G, Iglesias CA. Exploiting semantic similarity for named entity disambiguation in knowledge graphs. Expert Syst Appl 2018;101:8–24.

[59] Ebraheem M, Thirumuruganathan S, Joty S, Ouzzani M, Tang N. Distributed representations of tuples for entity resolution. Proc VLDB Endow 2018;11:1454–67.

[60] Zhang S, Tong H, Xu J, Maciejewski R. Graph convolutional networks: a comprehensive

review. Comput Soc Networks 2019, 6: 1–23.

[61] Barbehenn M. A note on the complexity of Dijkstra's algorithm for graphs with weighted vertices. IEEE Trans Comput 1998, 47:263.

[62] Berkhin P. A survey on PageRank computing. Internet Math 2005, 2: 73–120.

[63] Fortunato S. Community detection in graphs. Phys Rep 2010, 486: 75–174.

[64] Omidshafiei S, Pazis J, Amato C, How JP, Vian J. Deep Decentralized Multi-task Multi-Agent Reinforcement Learning under Partial Observability. 34th Int Conf Mach Learn ICML 2017 2017;6: 4108–4122.

[65] Ha H, Xu J, Song S. Learning a decentralized multi-arm motion planner. ArXiv 2020.

[66] Tan M. Multi-Agent Reinforcement Learning: Independent vs. Cooperative Agents. Mach. Learn. Proc. 1993, 1993. https://doi.org/10.1016/b978-1-55860-307-3.50049-6.

[67] Haarnoja T, Zhou A, Abbeel P, Levine S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. 35th Int. Conf. Mach. Learn. ICML 2018, 2018.

[68] Gupta JK, Egorov M, Kochenderfer M. Cooperative Multi-agent Control Using Deep Reinforcement Learning. Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 10642 LNAI, Springer Verlag; 2017, p. 66–83. https://doi.org/10.1007/978-3-319-71682-4_5.

[69] Haarnoja T, Zhou A, Hartikainen K, Tucker G, Ha S, Tan J, et al. Soft Actor-Critic Algorithms and Applications. ArXiv 2018.

[70] Hasselt, H. Double Q-learning. Advances in neural information processing systems, 2010, 23, 2613-2621.