Robotics and Computer–Integrated Manufacturing 86 (2024) 102625

Contents lists available at ScienceDirect

Robotics and Computer-Integrated Manufacturing

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

Full length article

Semantic models and knowledge graphs as manufacturing system reconfiguration enablers

Fan Mo *, Jack C. Chaplin, David Sanderson, Giovanna Martínez-Arellano, Svetan Ratchev

Institute for Advanced Manufacturing, University of Nottingham, Nottingham, Nottinghamshire, NG8 1BB, United Kingdom

ARTICLE INFO

Keywords:
Semantic models
Knowledge graphs
Reconfigurable manufacturing systems
Capability matching

ABSTRACT

Reconfigurable Manufacturing System (RMS) provides a cost-effective approach for manufacturers to adapt to fluctuating market demands by reconfiguring assets through automated analysis of asset utilization and resource allocation. Achieving this automation necessitates a clear understanding, formalization, and documentation of asset capabilities and capacity utilization. This paper introduces a unified model employing semantic modeling to delineate the manufacturing sector's capabilities, capacity, and reconfiguration potential. The model illustrates the integration of these three components to facilitate efficient system reconfiguration. Additionally, semantic modeling allows for the capture of historical experiences, thus enhancing long-term system reconfiguration through a knowledge graph. Two use cases are presented: capability matching and reconfiguration solution recommendation based on the proposed model. A thorough explication of the methodology and outcomes is provided, underscoring the advantages of this approach in terms of heightened efficiency, diminished costs, and augmented productivity.

1. Introduction

The future success of developed economies will depend on maintaining a strong and sustainable manufacturing sector built on cutting-edge technologies, skills, and industrial infrastructure with the ability to create a variety of complex items faster, more effectively, and at a lower cost [1]. The challenges posed by grand societal concerns, such as food security, aging populations, and climate change, make it difficult to develop the necessary industrial infrastructure to provide future commodities and services. Thus, a new, comprehensive strategy in manufacturing research that encompasses complexity, motion processes, high data density, and changing environments is required. Today's manufacturers must deal with volatile market demand and strive to meet the different requirements of customers by developing customized products [2,3]. Based on this, the University of Nottingham, the University of Cambridge, and Imperial College London have proposed a new concept known as "Elastic Manufacturing Systems", investigated in the Elastic Manufacturing project funded by the UK Engineering and Physical Science Research Council (EP/T024429/1) [4], allowing future manufacturing processes to be supplied as a service based on dynamic resource needs and provision across value chains, allowing for entirely new business models and holistic responses in the manufacturing industry.

In the concept of the Elastic Manufacturing System, the reconfiguration ability of the manufacturing system is part of the solution to effectively respond to continuously changing marketing demands. An RMS is a production system intended to accommodate the need for changeable quantities of high-quality goods at a competitive price [5]. RMS features an adaptable hardware and software architecture that permits the modification of production capacity and functionality by physically changing the assets and layout of the system to achieve the required throughput rate and flexibility [6].

The reconfiguration of production processes in response to external changes is a difficult challenge because system reconfiguration requires varied and disparate information on product information, capability information, system layout, process parameters, operation times of multiple assets, the sequence of the operations, material handling systems, and other relevant aspects in combination to correctly determine the required asset selections and configurations [7]. The characteristics of the data make this challenge bigger [8], as manufacturing data typically exists in multiple locations and formats, including structured and semi-structured data (such as data in proprietary vendor-specific formats, as relational databases, data in XML format, JSON format, HTML format, and more), and also unstructured database context reports [9]. Traditional data storage and retrieval systems categorize heterogeneous data based on their forms or modalities and analyze them individually [10]. Effective knowledge exchange and reuse have become challenging, with the conventional building of a knowledge

* Corresponding author.
E-mail address: fan.mo@nottingham.ac.uk (F. Mo).

https://doi.org/10.1016/j.rcim.2023.102625
Received 17 July 2023; Accepted 19 July 2023
Available online 8 September 2023

base requiring manual processing, which is time-consuming and labor-intensive, making it challenging to satisfy the criteria for the automated construction of a knowledge base [11] for the manufacturing system.

Knowledge graphs are proposed as a solution to overcome the limitations mentioned above. These structured, semantic knowledge bases effectively represent diverse information, aptly meeting knowledge representation requirements. Knowledge graphs structure information in an entity-relationship-entity or entity-attribute-value format, making them adept at describing the heterogeneity typical in the manufacturing domain [12]. Beyond simply representing information, machines can process and analyze knowledge graphs efficiently. This attribute is crucial for automation tasks like automatically responding to queries or making system recommendations. Furthermore, the compatibility of knowledge graphs with advanced technologies, such as natural language processing and machine learning, allows for creating sophisticated automated systems [12].

Within a knowledge graph, ontology plays a vital role. It formally defines a set of concepts within a domain and their relationships. This provides a shared understanding of the data's structure and meaning, including the types of entities and relationships that can exist and the rules for combining them [13]. Semantics, another integral part of a knowledge graph, is the interpretation of relationships and connections between entities and concepts. This includes defining relationships and their properties and understanding the entities' meaning. The semantics of a knowledge graph provides a structured way to represent complex information meaningfully, simplifying navigation and information querying.

Besides the knowledge representation problem, which can be addressed by the knowledge graph, to allow for fast, accurate, and automated reconfiguration, two essential elements need to be captured and considered: the capability of the assets and the capacity of those assets. The capability is used to represent the function of the resource in a manufacturing configuration [14]. However, many capability models in the manufacturing domain are proprietary and not vendor-neutral, which reduces the interoperability of different information. Although Järvenpää et al. [15] developed a formal unified description of the capability model, the number of capability classes is limited, and the industry lacks implementation. Furthermore, the model does not include reconfiguration information or dynamic parameters. A formal unified, and comprehensive description of the capability model will allow for a rapid decision-making process. It should be highlighted that incorporating Semantic Web Rule Language (SWRL) rules [16], ontology reasoning [17], and related techniques such as rule-based reasoning systems [18], SPARQL Inferencing Notation (SPIN) [19], and SHAPE Constraint Language (SHACL) [20] into the capability model within the knowledge graph can significantly improve its adaptability and precision. By employing these reasoning processes, a more rigorous and comprehensive understanding of the manufacturing system can be achieved, ultimately facilitating superior decision-making and more effective system reconfiguration.

On the other hand, the capacity of assets is measured and utilized to monitor the real-time production state on the shop floor. In manufacturing firms, capacity is often modeled using a mathematical index called capacity utilization rate [21], which aims to reduce the average cost of production by evaluating economic performance. The challenge is that firms or assets might use different key performance indicators (KPIs), which makes it important to model the capacity models semantically to automate the process. Hence, a capacity model provides the necessary KPIs to monitor the firm's performance in various aspects, such as machines and equipment efficiency and effectiveness and supply chain performance [22,23].

In light of these challenges, our paper proposes a methodology for realizing an automated and precise decision-making process in the manufacturing domain, focusing on the robot manufacturing cell for reconfigurable and adaptive systems. This approach involves the development of a unified formal ontology model encompassing capability, capacity, and reconfiguration information. The methodology employs knowledge graphs to capture and represent knowledge semantically. By utilizing this proposed model to represent manufacturing information, the system gains a deeper understanding, making more efficient, accurate, and prompt decisions. In essence, the critical contributions of the paper are as follows:

1. Semantic modeling is used to define resource capability, capacity, and reconfiguration formally.
2. An automatic data pipeline is used to generate the knowledge graph for a reconfigurable system focusing on the robotic manufacturing cell.
3. Two demonstration cases of how these semantic models work together to assist the efficient decision-making and the successful execution of system reconfiguration with the help of the knowledge graph.

The remainder of the paper is organized as follows. Section 2 reviews the existing information models in the manufacturing domain and the knowledge graph representation of information models. Section 3 describes the detailed information of the ontology model representing capability, capacity and reconfiguration information. Section 3.4 describes the process of building a knowledge graph based on the proposed ontology model. Section 4 describes two validation use cases based on the proposed methodology. Section 5 describes the conclusions and outlines of future work.

2. Related work

This section presents an overview of the relevant work from three perspectives: research on manufacturing system ontology models, knowledge graphs and their applications, and the challenges of creating reconfigurable manufacturing systems. To support the reconfiguration of RMS with automated tools, all three topics must be considered. There has been significant research on these individual topics but limited examples of all three being considered together.

2.1. Ontology models in the manufacturing domain

In the manufacturing domain, there has been an increasing interest in using emerging knowledge representation technologies – such as ontologies, semantics, and semantic web technologies – to support collaboration, interoperability, and adaptation needs. One of the earliest manufacturing ontologies was the Process Specification Language (PSL), developed to provide a neutral language for representing process-related knowledge and supporting application integration [24]. The XML-based approach may fulfill interoperability requirements across diverse systems, but it only describes the manufacturing process's structure, making it difficult to express its implicit semantic content [25]. Most existing resource description approaches are domain-specific and offer only partial solutions for specific applications, hence lacking a comprehensive view. Lu et al. [26] present an ontology-based approach to enable semantic interoperability throughout the whole process of service provision in the cloud. However, this work does not talk much about the application of this ontology model and how to enable complex decision-making processes with this model. Wang et al. [27] present an ontology model to model task semantics and description in cloud manufacturing systems. However, this paper lacks a vendor-neutral description of the resource in this model, which hinders interoperability. Järvenpää et al. [15] presented a developed information model MaRCO, which provides resource vendors with a standard, vendor-independent way to describe the capabilities of the resource offerings. The authors describe how to use this model to do capability matching. The authors outline the utilization of the model for capability matching in the manufacturing sector. Capability matching involves determining the suitability of a manufacturing process or equipment for a specific product or component by evaluating its accuracy, speed, and capacity compared to the requirements of the product or component being produced. However, the limited number of capability classes is due to a lack of industry-wide implementation, and the model does not account for reconfiguration information or dynamic parameters. Based on the literature review, there appears to be a scarcity of systematic research pertaining to capacity modeling in manufacturing reconfiguration. While some papers address components of capacity models in manufacturing [28–30], they do not provide a comprehensive and systematic framework.

2.2. Knowledge graphs in the manufacturing domain

Knowledge graphs are composed of structured information about the real world, describing entities and relations among them [17]. In a knowledge graph, each fact is represented as a triple (h, r, t), which indicates that there exists a relationship named r between the head entity h and the tail entity t, e,g., *(Milling machine, hasTools, Milling cutter)*. Compared with traditional data storage and computation, knowledge graph technology focuses on collecting, managing, and processing unstructured, heterogeneous data. It is better at representing and computing relationships, which can handle complex and diverse association analysis and infer new knowledge [31]. One advantage of the knowledge graph is that it can be easily extended to model new relationships and entities without changing the original schema. Besides, knowledge graphs are better suited for answering complex queries that involve multiple entities and relationships [32], for example, "how to produce the hinged product of the airplane?" This kind of query would be difficult or impossible to answer using traditional relational databases or flat file structures.

A knowledge graph comprises a schema layer and an entity layer [33]. The schema layer contains concepts, properties, and relationships between concepts. The entity layer contains the specific entities that are instantiated from these ontological concepts. Taking the milling machine once again as an example, *(Milling machine, hasTools, Milling cutter)* is the entity representation from the entity layer. Both the milling machine and milling cutter are instantiated from the resource class of the schema layers. The semantic relationship of these entities in the schema layer is *(Asset, hasTools, Asset)*. The data structure of the knowledge graph is compatible with the data structures on which many technical tasks in artificial intelligence are based (e.g., big data with a heterogeneous structure and multiple associations), which can provide strong support for subsequent machine learning and inference tasks, helping enterprises to improve performance in intelligent search, intelligent Q&A, intelligent recommendation, and big data analysis applications [34].

Knowledge graphs have been applied in the manufacturing domain recently. Zhou et al. [35] present a unified knowledge graph-driven production resource allocation approach, allowing fast resource allocation decision-making for given order inserting tasks, subject to the resource machining information and the device evaluation strategy. Xia et al. [36] introduce an industrial knowledge graph (IKG)-based multi-agent reinforcement learning (MARL) method for achieving the Self-X cognitive manufacturing network. Knowledge graphs make it straightforward to express the connections within entities clearly and effectively. It allows people to analyze problems based on the connections between knowledge. Despite the numerous benefits of knowledge graphs in manufacturing, some limitations must be considered, which include the following:

1. **Data integration**: Integrating data from multiple sources can be a challenging and time-consuming process. The quality and format of the data may also vary, making it difficult to standardize and integrate it into a knowledge graph [37].
2. **Expertise**: Building and maintaining a knowledge graph requires specialized skills and knowledge, making it challenging for organizations to implement it themselves [38].
3. **Lack of Standardization**: The lack of standardization in knowledge graph technology and data representation can limit interoperability and integration with other systems [39].

2.3. Reconfigurable manufacturing systems

An RMS is a manufacturing system that can be easily adapted and reconfigured to accommodate changes in production requirements, product design, and production volume [40]. The key characteristic of an RMS is its ability to quickly switch between different production processes and product types without extensive retooling or major investments in new equipment. This versatility enables manufacturers to respond quickly to changes in market demand and product requirements, reducing production downtime and costs while increasing production efficiency and competitiveness [6,41,42]. There are several types of reconfiguration in manufacturing systems, including:

1. **Layout reconfiguration**: When the position of the equipment is considered as one of the configurations of the manufacturing system. Then the layout optimization problem can be regarded as a problem of selecting the best configuration. This includes the physical layout of the manufacturing system, such as the location of machines and equipment, as well as the spatial relationships between them [43,44]. Layout reconfiguration involves considering the optimal positioning of all machines in the production line, with the decision variable being the potential range of each machine's position. However, in some cases, certain machines may not be able to change their positions due to constraints such as fixed installation or specialized functions. Thus, the position information for these machines is considered to be fixed at 0 within the range of potential positions. Additionally, changes to the production environment, such as adding, removing, or updating machines, can impact the overall layout configuration and are therefore considered part of the layout optimization process. When a layout reconfiguration is carried out, the primary goal is to optimize the manufacturing system for maximum efficiency, productivity, or any other objective. However, it is crucial to consider the impact of these changes on machines that might be left out of the reconfigured layout. While unused machines in a reconfigured layout can be considered a liability, several strategies are available to manage and minimize their impacts on the overall manufacturing system, such as redeployment, leasing, recycling, and maintenance. These options can help ensure that the reconfiguration process leads to a more efficient, productive, and cost-effective manufacturing operation.
2. **Resource selection**: Resource selection refers to the process of choosing the appropriate resources (e.g., machines, tools, and personnel) to complete a task or production process. Resource selection aims to ensure that the necessary resources are available, efficient, and cost-effective for the task at hand [45].
3. **Job scheduling**: Although job scheduling is not specific to RMSs, it is still a critical aspect of manufacturing that can impact the overall system performance, including reconfigurable manufacturing systems. Job scheduling involves determining the optimal sequence of jobs to be processed, taking into account various constraints such as machine availability, production capacity, and due dates for jobs. Therefore, job scheduling is an important consideration in the design and operation of reconfigurable manufacturing systems to ensure efficient and effective production. Various techniques, such as mathematical optimization and heuristic algorithms, have been developed to address job scheduling problems in manufacturing [46–48].

3. Methodology

Motivated by current limitations of existing manufacturing semantic modeling methods [24–27], such as difficulties in achieving effective reconfiguration, few unified ontology models that represent the capability, capacity, and reconfiguration information, the difficulties in processing heterogeneous data from the manufacturing domain [7], the need for rapid response in the reconfiguration of the elastic manufacturing system [1], and low levels of automation in the decision-making process [11], this paper provides a methodology for achieving fast, optimized reconfiguration by utilizing an ontology model and a knowledge graph. We have proposed an ontology model for capability, capacity, and reconfiguration to formally describe these manufacturing features in a machine-interpretable way. The knowledge graph-based method is applied in this methodology because of its capability to deal with heterogeneous data, and it can be dynamically updated.

To address knowledge graph limitations as described in Section 2.2, the methodology proposes data transformation, mapping techniques, and machine learning algorithms for data integration. Automated tools and frameworks are recommended to tackle skill acquisition challenges, while the development and adoption of standardized knowledge representation languages are encouraged.

3.1. Knowledge graph building approach

As mentioned in Section 2, the knowledge graph consists of the schema layer and the entity layer. To build the knowledge graph focusing on reconfiguration, a methodology to build both the schema layer and the entity layer of the knowledge graph should be defined.

The construction of knowledge graphs can be achieved through either a top-down or bottom-up approach. The top-down methodology entails pre-defining the schema layers by means of data sets and expert knowledge and subsequently updating the entity layer in accordance with the predefined schema [49]. This strategy is widely adopted in the creation of domain-specific or application-specific knowledge graphs and is heavily contingent upon the input of domain specialists. Conversely, the bottom-up approach commences at the entity layer with the extraction of entities and relationships from structured and unstructured data sources, followed by the establishment and ongoing refinement of the schema layer based on the aggregated data derived from the entity layer [50].

The exclusive reliance upon either a top-down or bottom-up strategy bears certain limitations. The top-down approach necessitates substantial participation from domain specialists, thus ensuring data precision but at the expense of incurring considerable costs and consuming copious amounts of time. Meanwhile, while more frugal compared to the top-down methodology, the bottom-up strategy requires an abundance of data, which can pose a daunting challenge to procure within the realm of manufacturing. The proposed methodology in this paper aspires to automate the decision-making process and attain an optimal level of reconfiguration efficiency. Due to the inherent deficiencies associated with exclusively utilizing either the top-down or bottom-up approaches, a combination of both methods was opted for to fabricate the knowledge graph in our methodology, as described in [51].

3.2. Constructing the schema layer of the knowledge graph

An ontology model named OCCR (Ontology model of Capability, Capacity, and Reconfiguration) was developed to construct the schema layer of the knowledge graph. This ontology model has been developed based on Järvenpää's model [15], incorporating additional semantic models and refining existing ones, such as incorporating a more detailed capability model including metrology capability, as well as incorporating information related to reconfiguration, capacity, and tasks. The purview of the OCCR model is to formally model the capabilities, capacities, and reconfiguration information, along with their associated models, of reconfigurable manufacturing systems, thus enabling a more cost-effective response mechanism for manufacturing firms. This is achieved by enabling the automated analysis of the utilization of available assets, and the autonomous allocation of capacity to optimize the utilization of assets in response to fluctuating market demands. The OCCR model consists of seven semantic models, namely the task, product, process, capability, capacity, assets and reconfiguration models, each of which comprises the relevant ontology classes. A comprehensive representation of these semantic models is presented in Table 1, with a more in-depth description of each model provided in the following texts.

Table 1
Seven semantic models and their symbolic representations.

Models	Symbolic representation
Task	TAS
Product	PRT
Process	PRS
Capability	CAB
Capacity	CAP
Assets	ASS
Reconfiguration	REC

3.2.1. Task model

The task model describes the information about how customer orders and requests that are initiated internally within the factory are processed. It represents the tasks or activities needed to complete a customer order or fulfill a request. It provides a high-level overview of the steps involved in the production process. The task model is divided into two submodels; non-reconfiguration-related tasks and reconfiguration-related tasks.

1. ***Non-reconfiguration-related task model (NRT)***
   A non-reconfiguration-related task is defined in the model, which typically originates from the customer and is intended to meet specific requirements for producing the product, such as quantity and timeline. This task can also be interpreted as a production task.
2. ***Reconfiguration-related task model (RT)***
   The reconfiguration-related task is defined in the model, typically as an internal task of the factory. This model provides information about reconfiguration, which can involve different types of reconfiguration, such as layout reconfiguration, resource selection, and job scheduling.

3.2.2. Product model

In a reconfigurable robotic manufacturing system, the product being created plays a pivotal role. The selection of the most efficient production procedures is heavily influenced by the geometric attributes of the product, such as its size and shape. For instance, if the product is large or has a complex shape, robotic manufacturing cells equipped with industrial robots that have a large workspace and high flexibility might be suitable. These robots can manipulate large objects or accurately navigate around complex shapes, making them ideal for these types of tasks. On the other hand, if the product has intricate geometric features or requires precise assembly, robotic cells with robots that possess high precision and advanced control features might be a better choice. These robots can handle delicate assembly tasks and accurately follow complex trajectories, thereby ensuring the quality of the finished product. In addition, the product's requirements for assembly and handling can also influence the choice of robotic cells. For example, if the product requires specific positioning or orientation during assembly, robots with advanced vision systems or force-sensing capabilities might be necessary.

In summary, the specific characteristics and requirements of the product significantly influence the selection and configuration of the robotic cells in a reconfigurable manufacturing system

3.2.3. Process model

This model provides a structured representation of the necessary operations and requirements to complete a manufacturing or reconfiguration task step. A process model comprehensively explains the various steps involved in a manufacturing task or reconfiguration process, defining the inputs, outputs, and dependencies of each step. Furthermore, it is linked with the capability model to suggest the required capabilities to execute the process.

3.2.4. Capability model

A capability model in the manufacturing domain represents the capabilities and constraints of the factory in terms of the manufacturing processes and technologies it has, the resources and skills which are available, and the regulations and standards it must comply with. It provides a comprehensive view of the factory's capabilities and helps to make informed decisions about the products and processes that can be manufactured within the factory. It consists of two subclasses: simple capability and combined capability. In the capability model, the simple and combined capabilities are linked by "hasInputCapability" relations.

1. ***Simple capability (SC)***
   A simple capability is the capability that a single asset has. For example, the fixture has the single capability of "fixturing".
2. ***Combined capability (CC)***
   Combined capabilities are combinations of two or more (simple or combined) capabilities. It could be divided by functional decomposition into simple, lower-level capabilities.

3.2.5. Capacity model

This model is a structured representation of manufacturing KPIs that monitor the performance of the shop floor, focusing on reconfiguration. Our capacity model in the manufacturing domain refers to a mathematical representation of the production capacity of a factory or production line. It considers various factors such as available resources (e.g., machines, labor), production processes, and constraints to determine the maximum output that can be achieved under certain conditions. The capacity model adapts to changes in production conditions to achieve reconfiguration, such as the introduction of new products, changes in demand, or the introduction of new technologies.

In the OCCR model, the capacity model can include various key performance indicators (KPIs) to measure the efficiency and effectiveness of a system's capacity. Some examples of KPI formulations that can be used in the capacity model, along with a description of the cost model used in the manufacturing context, are:

1. **Cost:**
   In the capacity model, a cost model is a method used to estimate and allocate costs associated with a system's capacity. A cost model typically includes factors such as fixed costs, variable costs, and overhead costs. The specific cost model used will depend on the nature of the system being measured and the goals of the analysis. Some examples of cost models that can be used in the capacity model include:

   (a) *Fixed cost model:* This model assumes that fixed costs are spread evenly across all units produced. The formula for this model is:

   $$FixedCostPerUnit = TotalFixedCosts/NumberOfUnitsProduced \tag{1}$$

   (b) *Variable cost model:* This model assumes that costs vary based on the number of units produced. The formula for this model is:

   $$VariableCostPerUnit = TotalVariableCosts/NumberOfUnitsProduced \tag{2}$$

   By incorporating cost models into KPI calculations, the capacity model allows for a more comprehensive analysis of the efficiency and effectiveness of a system's capacity. This can help companies identify areas where they can reduce costs and improve their overall performance.

2. **Capacity utilization rate:**
   This KPI measures the percentage of a manufacturing system's capacity that is currently being used. The fixed cost model is commonly used with this KPI in manufacturing. The formula for this KPI is:

   $$CapacityUtilizationRate = (ActualOutput/DesignCapacity) \tag{3}$$

3. **Cycle time:**
   This KPI measures the time it takes to complete a single cycle of a manufacturing process. The process cost model is commonly used with this KPI in manufacturing, as costs may be associated with specific process steps. The formula for this KPI is:

   $$CycleTime = TotalTime/NumberOfCycles \tag{4}$$

3.2.6. Assets model

In the robotic manufacturing domain, assets are referred to as physical objects, machinery, equipment, and also software utilized in production. Asset models represent these assets commonly used by organizations to manage, maintain, and optimize their utilization. The semantic model classifies assets into four categories: hardware (equipment and tooling), software, human workforce, and the reconfiguration solver. Asset models include information about the asset's location, age, condition, maintenance history, and other pertinent information to aid in informed decisions regarding their utilization and replacement.

3.2.7. Reconfiguration model

This model describes the features that make an RMS dynamic with the capacity and functionality to adapt to the customer request changes. The features can be decision variables, optimization variables, constraints in the reconfiguration scenarios. We defined three types of reconfiguration in our current OCCR model: layout optimization, resource selection in terms of reconfiguration, and job scheduling.

3.2.8. Relationships between the seven semantic models

Fig. 1 depicts the relationships between the seven semantic models. These semantic models work together to achieve the decision-making process and the cost-effectiveness of the reconfiguration of the robotic manufacturing system. For the reconfiguration-related task and the non-reconfiguration-related task, they have different processes for utilizing the semantic models.

When a new task comes to the robotic manufacturing system, the task model figures out if the task is about reconfiguration or not. Depending on the type of task, we use our semantic model in different ways to help make decisions and manage the reconfiguration process.

For a non-reconfiguration task (NRT), the task model displays the necessary product information and connects it with the product model through the relationship "required product". The non-reconfiguration task model displays details such as product type, quantity, and delivery timeline. Meanwhile, the product model showcases the product's features linked to the process model through the relationship "required process". For instance, if the product requires a hole, then the drilling process must be executed to meet this requirement.

The process model selects the relevant production process since the task is non-reconfigurable. The process model also specifies the requirements for each process, which vary among processes. Once the necessary process for producing the product has been determined, the capability-matching process begins. This process entails identifying the required capabilities to execute the process using information from the process model, such as the accuracy and force required for the process "Inserting".

The assets semantic model links with the capability model, so once the required capabilities are clear, the candidate assets that meet the specifications can be identified. Additionally, dynamic parameters from the capacity model, such as utilization rate, cost, and working

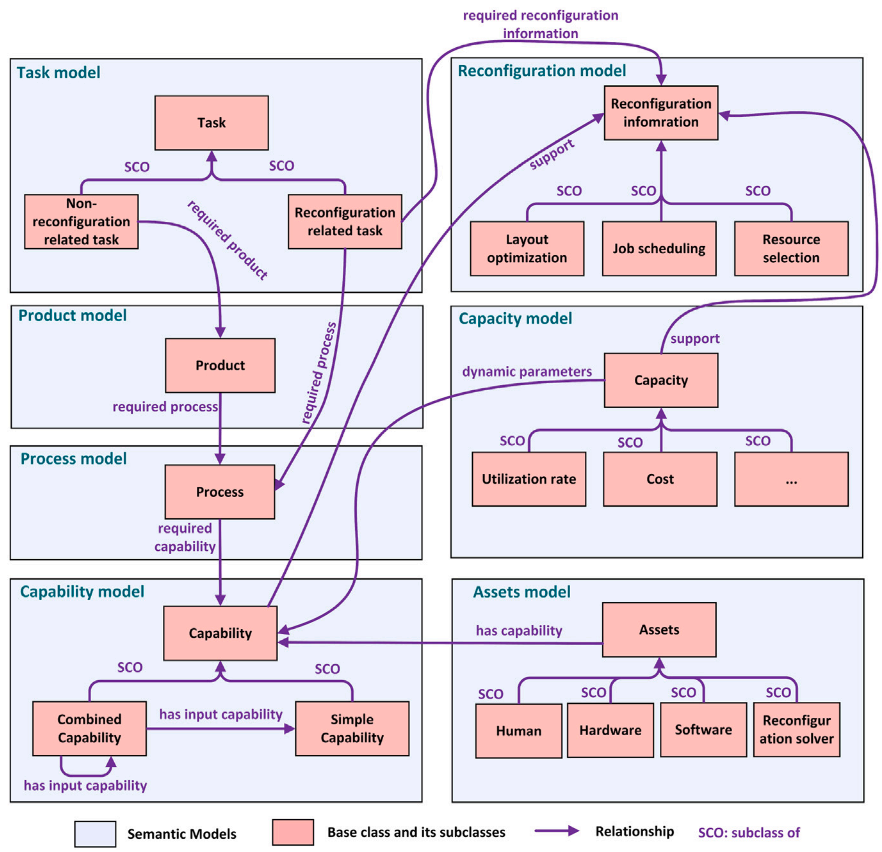

Fig. 1. Linking semantic models in the optimization and the decision-making process for the reconfiguration.

status, can be considered to optimize the capability-matching process. Ultimately, the most suitable assets, possessing the required capabilities and specifications, are selected.

In the context of reconfiguration-related tasks, the task semantic model serves as a repository for the type of reconfiguration involved. The OCCR model defines three types of reconfiguration, which can be either a single type or a combination thereof. The task model is linked directly to the reconfiguration model, enabling the retrieval of information necessary for reconfiguration based on the type indicated in the task model. The reconfiguration model provides information about the decision variables, optimization objectives, and constraints that should be considered during the reconfiguration process. The task model is also linked to the process semantic model through the relationship “required process”.

For reconfiguration-related tasks (RT), there are two types of required processes. One type is the reconfiguration solution, such as the “layout reconfiguration process”, “resource selection process”, and “job scheduling process”. The other type is the current process that needs reconfiguration. For instance, as depicted in Fig. 2, consider a work cell consisting of a robot, a profile board storage rack, profile boards and a frame on the automated guided vehicle. The robot picks the profile board from the storage rack and places it on the frame. If the customer requests to optimize the layout of the current work cell, the semantic model identifies two types of processes. The reconfiguration solution process, in this case, is the “layout reconfiguration process,” while the current processes that will be subject to the reconfiguration are “pick profile board” and “place profile board.”

With the clear identification of the two types, the capability-matching process commences. If resource selection is part of the reconfiguration types of the RT task, then the capability matching process is employed to determine the feasibility and viability of alternative assets in replacing the existing ones in the production line. For the capability

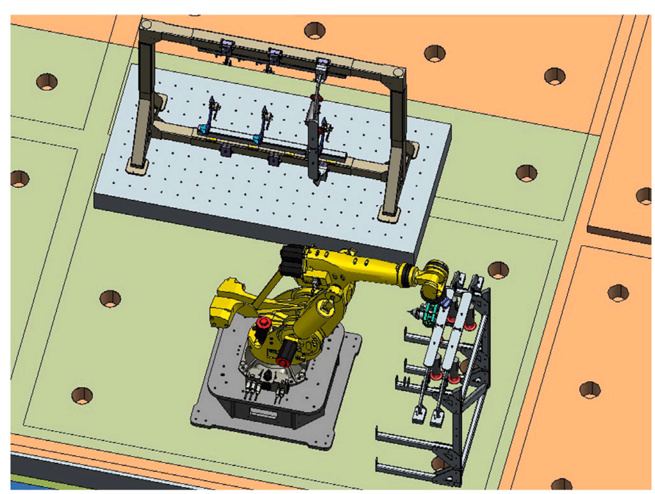

Fig. 2. Example to show the relationship between the two types of the process in the reconfiguration-related task.

matching of the reconfiguration solution, potential assets that are capable of executing the reconfiguration are sought. For instance, the appropriate algorithm and simulation platform can be identified.

In the OCCR model, all the semantic models can be described in Eq. (5) [52]. In this equation, ASS, TAS, PRT, REC, PRS, CAB, CAP are the seven semantic model which we proposed before. P represents the property information of the semantic information. R represents the relationship information, which describes the relationship between the two models. For example, robot “hasCapability” force applying. “hasCapability” is the relationship information, which is used to describe the relationship between the ontology class “robot” and the ontology class “force applying”. L describes the constraint rules for the formation of the classes. For example, the subclasses should inherit all the properties from the parent class.

$$OCCR = \{ASS \cup TAS \cup PRT \cup REC \cup PRS \cup CAB \cup CAP \cup P \cup R \cup L\} \tag{5}$$

3.2.9. Special requirements of the OCCR model

To summarize, the specific requirements addressed by the OCCR model are:

1. The *Capability Model* must outline the capabilities that can be allocated to assets, along with their designations and their associated capability characteristics. The model should strive for comprehensive and adaptable properties to augment its scalability.
2. The *Assets Model* should have a unified definition and general properties for assets, which would facilitate further optimization, such as cost, running status, and size.
3. The *Capacity Model* should provide a performance monitoring report on the current production state, calculated by evaluating the utilization rate through statistical assessment of KPIs related to the output and efficiency of the shop floor's assets.
4. The *Reconfiguration Model* should provide explicit information on the type of reconfiguration required for a particular activity, utilizing a vendor-neutral approach.
5. The suggested *Knowledge Representation* must support reasoning and enable actions, such as matching asset capabilities to product needs, searching for and selecting appropriate assets, and offering practical reconfiguration recommendations.

3.3. Reconfiguration information in the OCCR model

As previously mentioned in Section 3.2, a new semantic model has been defined to capture the necessary information for supporting reconfiguration in manufacturing systems. As described in Section 2.3, various types of reconfiguration exist. In the current model, the subclasses of layout reconfiguration, resource selection, and job scheduling have been selected to represent the different types of reconfiguration. These subclasses have been included under the reconfiguration semantic model, as shown in Fig. 1. Decision variables, optimization objectives, constraints, and reconfiguration results have been used as subclasses of each of the three subclasses to describe the reconfiguration information.

In manufacturing reconfiguration optimization, *decision variables* are values that can be chosen or adjusted to optimize the objective, such as cost, efficiency, or production rate. *Optimization objectives* are the objectives that can be optimized depending on the specific application and goals of the reconfiguration optimization problem in the manufacturing domain. *Constraints* in the manufacturing reconfiguration domain refer to limitations or restrictions on the decision variables that must be met to ensure a feasible and practical solution. The *reconfiguration result* records the results of the optimization problem, which can be used as an experience for further reconfiguration problems in the future. A subclass in the asset semantic model is also considered: *Reconfiguration solver*. This subclass represents the enabling technology for achieving the reconfiguration, such as the algorithm, the simulation platform, virtual reality (VR), and augmented reality (AR).

In the following subsections, the three reconfiguration types in the reconfiguration semantic model will be explained in detail.

3.3.1. Layout reconfiguration

In the field of RMSs, the performance of operations on products (or product families) according to their operational requirements is crucial. Layout design and optimization play a key role in RMSs since these systems require different layout configurations when switching from one product family to another. The following information has been identified as essential and has been classified as subclasses in the proposed semantic model:

- **Decision-Variables:** The position information of assets (e.g., coordinates and rotation angles) is considered as subclasses of the decision variables in the manufacturing reconfiguration optimization. The position information is classified into coordinates (such as Cartesian coordinates, cylindrical coordinates, polar coordinates, and spherical coordinates) and orientation (such as Euler angles and rotation matrices).
- **Optimization Objectives:** The optimization objectives in the semantic model include cost, quality, cycle time, space utilization, and robot maneuverability.
- **Constraints:** Constraints include inequality constraints and equality constraints. They are mathematical relationships used in optimization problems to ensure that the solutions satisfy certain requirements or limitations. In the context of layout optimization in manufacturing, these constraints can represent various physical or operational restrictions.

  1. Inequality Constraints: These are constraints that establish upper or lower bounds on the values of the decision variables or on functions of the decision variables. In the layout optimization problem, inequality constraints may include:

     (a) Non-collision constraints (NC): Ensuring that machines do not overlap or collide with each other in the layout. This can be formulated by setting a minimum distance between the edges of any pair of machines.
     (b) Reachability constraints (RC): Ensuring that machines or workstations are accessible to workers or material handling equipment, such as robots or conveyor systems. This can be formulated by setting a maximum distance between machines or specifying a minimum clearance for pathways or aisles.
     (c) Area constraints (AC): Ensuring that the total area occupied by the machines does not exceed a predefined maximum area. This constraint can be formulated as the sum of the areas of individual machines being less than or equal to the maximum allowed area.

  2. Equality Constraints: These are constraints that require an exact relationship between the decision variables or functions of the decision variables. In the layout optimization problem, some of the equality constraints may include:

     (a) Resource Allocation Constraints (RAC): These constraints ensure that the total number of certain types of machines or resources within the entire layout is fixed. For example, if there are a limited number of robotic assembly cells available, the layout optimization problem should include a constraint that ensures the exact number of robotic assembly cells is used in the layout.
     (b) Grouping Constraints (GC): These constraints ensure that a specified number of certain types of machines or resources are grouped together within a specific section of the layout. For instance, if a certain manufacturing process requires three specific machines to be located close together for efficiency, the layout optimization problem should include a constraint that ensures exactly these three machines are grouped together in the layout.

Both inequality and equality constraints help to model the physical and operational limitations of the manufacturing environment and ensure that the resulting layout is practical, feasible, and efficient.

One example of optimizing the layout of a manufacturing system is depicted below. The optimization objectives are:

1. Minimize Space Utilization (SU)
2. Minimize Cycle Time (CT)
3. Minimize Total Distance between Machines (TD)

The decision variables in this problem are the coordinates of each machine in the manufacturing system. The optimization problem can be formulated as:

$$\begin{aligned}
\text{minimize} \quad & f(x) = \{SU(x), CT(x), TD(x)\} \\
\text{subject to} \quad & NC_i(x) \le 0, \quad i = 1, \dots, m_1 \\
& RC_j(x) \le 0, \quad j = 1, \dots, m_2 \\
& AC_k(x) \le A_{\max}, \quad k = 1, \dots, n_1 \\
& RAC_l(x) = R_l, \quad l = 1, \dots, n_2 \\
& GC_m(x) = G_m, \quad m = 1, \dots, n_3
\end{aligned} \tag{6}$$

where:

- x: A vector representing the coordinates of each machine in the manufacturing system.
- $f(x)$: A vector containing the objective functions to be minimized, which include SU, CT, and TD.
- $NC_i(x)$: The ith non-collision constraint function that must be satisfied by the coordinates x.
- $RC_j(x)$: The jth reachability constraint function that must be satisfied by the coordinates x.
- $AC_k(x)$: The kth area constraint function that must be satisfied by the coordinates x, with $A_{\max}$ being the maximal allowed maximal area.
- $RAC_l(x)$: The lth resource allocation constraint function that must be satisfied by the coordinates x, with R_l being the exact number of a certain type of resource required.
- $GC_m(x)$: The mth grouping constraint function that must be satisfied by the coordinates x, with G_m being the exact number of a certain type of resource required in a specific group.
- m_1: The number of non-collision constraints.
- m_2: The number of reachability constraints.
- n_1: The number of area constraints.
- n_2: The number of resource allocation constraints.
- n_3: The number of grouping constraints.

By solving this optimization problem, the optimal layout for the manufacturing system can be determined, considering the objectives and constraints. The layout reconfiguration can be achieved by adjusting the decision variables (i.e., the coordinates and rotation angles) of the assets within the manufacturing system. This process allows the system to be reconfigured efficiently for different product families or operational requirements.

3.3.2. Resource selection in reconfiguration

Resource selection for reconfiguration refers to the process of choosing the right assets (e.g., hardware, software, personnel) to implement changes in a system or to apply to processes to achieve a desired outcome. This involves evaluating various options based on factors such as cost, compatibility, performance, and availability and selecting the ones that best meet the needs of the reconfiguration effort. Resource selection aims to ensure that the reconfiguration is carried out efficiently and effectively, with minimal disruption to existing operations. It aims to find if the current assets in the production line meet the requirement and if they are needed to be replaced. The following aspects are important in this section, and thus this information is stored as the ontology model classes in our model:

- **Decision-Variables:** The resource information, the number of product types, product requirements, and job information are considered the decision variables in our model. These factors are implemented as the subclasses of the decision variables in the proposed semantic model.
- **Optimization Objectives:** Resource utilization, cost (investment cost, capital cost), workload, running status of the machines, energy consumption, and remaining useful life are considered to be the optimization objectives.
- **Constraints:** In the context of resource selection, constraints can be categorized into equality and inequality constraints.

    1. Inequality Constraints

        (a) Demand Constraints (DC): Ensuring that the selected resources are sufficient to meet the demand of the reconfiguration task without exceeding the available resources.
        (b) Investment Constraints (IC): Ensuring that the total investment for the selected resources does not exceed the budget allocated for the reconfiguration task.
        (c) Space Constraints (SC): Ensuring that the selected resources can be accommodated within the available space in the production line or facility.

    2. Equality Constraints

        (a) Total Resource Allocation (TRA): Ensuring that the total number of required resources is equal to the available resources in the system.
        (b) Specific Resource Requirements (SRR): Ensuring that the selected resources meet the exact specifications or requirements for the reconfiguration task.

As one example, the goal is to optimize resource selection for reconfiguration, considering the following objectives:

1. Minimize Resource Utilization (RU)
2. Minimize Cost (C)
3. Minimize Time (T)

The decision variables in this problem are the resources chosen for the reconfiguration effort. The optimization problem can be formulated as:

$$\begin{aligned}
\text{minimize} \quad & f(x) = \{RU(x), C(x), T(x)\} \\
\text{subject to} \quad & DC_i(x) \le 0, \quad i = 1, \dots, m_1 \\
& IC_j(x) \le 0, \quad j = 1, \dots, m_2 \\
& SC_k(x) \le 0, \quad k = 1, \dots, m_3 \\
& TRA_l(x) = 0, \quad l = 1, \dots, n_1 \\
& SRR_m(x) = 0, \quad m = 1, \dots, n_2
\end{aligned} \tag{7}$$

where:

- x: A vector representing the resources chosen for the reconfiguration effort.
- $f(x)$: A vector containing the objective functions to be minimized, which include Resource Utilization (RU), Cost (C), and Time (T).
- $DC_i(x)$: The ith demand constraint function that must be satisfied by the resources x.
- $IC_j(x)$: The jth investment constraint function that must be satisfied by the resources x.
- $SC_k(x)$: The kth space constraint function that must be satisfied by the resources x.
- $TRA_l(x)$: The lth total resource allocation constraint function that must be satisfied by the resources x.
- $SRR_m(x)$: The mth specific resource requirements constraint function that must be satisfied by the resources x.

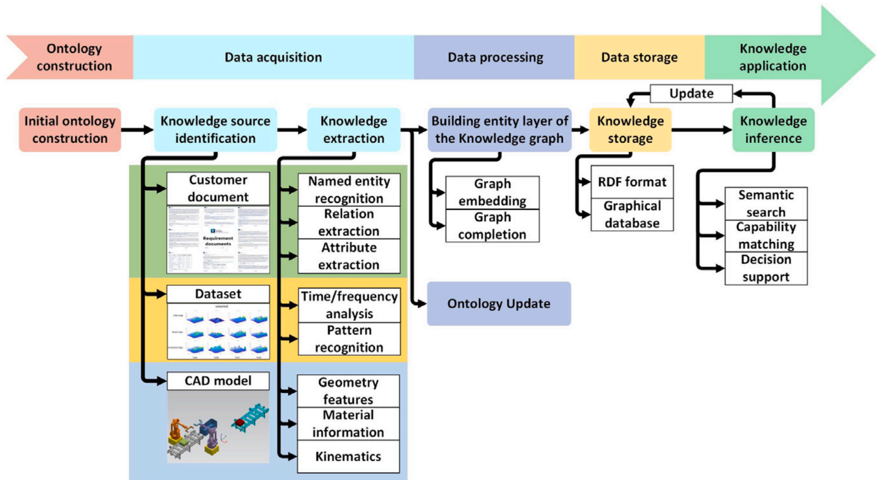

Fig. 3. Construction of the entity layer of the reconfiguration knowledge graph.

- m_1: The number of demand constraints.
- m_2: The number of investment constraints.
- m_3: The number of space constraints.
- n_1: The number of total resource allocation constraints.
- n_2: The number of specific resource requirements constraints.

By solving this optimization problem, the optimal resource selection for the manufacturing system reconfiguration can be determined, considering the objectives and constraints. Resource reconfiguration can be achieved by adjusting the decision variables (i.e., the types and quantities of resources) of the assets within the manufacturing system. This process allows the system to efficiently adapt to different product families or operational requirements while minimizing resource utilization, cost, and time.

3.3.3. Job scheduling

In the proposed ontology model, the scheduling problem is described as a set of decisions concerning the sequence of parts to be released into the system, the selection of the operation/resource pair, and the sequence of parts assigned to each resource in the production process. This model includes the following information as subclasses of the decision variables.

- **Decision-Variables:** Available assets to perform the manufacturing jobs, jobs that need to be performed, and a set of operations for all the jobs, which must be performed in a given order based on the constraints.
- **Optimization Objectives:** Makespan, the workload of the most loaded resource, production rate, flow time, tardiness, and resource utilization.
- **Constraints:** In the context of job scheduling, constraints can be categorized into equality and inequality constraints.

1. Inequality Constraints
    (a) Shortest Processing Time (SPT): The job with the shortest processing time should be processed first.
    (b) First In, First Out (FIFO): The job that entered the system first should be processed first.
    (c) Most Work Remaining (MWKR): The job with the most work remaining should be processed first.
    (d) Earliest Due Date (EDD): The job with the earliest due date should be processed first.
2. Equality Constraints
    (a) Machine Constraints (MC): A job can only be processed on one machine at a time.
    (b) Job Constraints (JC): A job must be completed before the next job can be started.
    (c) Asset Constraints (AC): An asset can only be used by one job at a time.
    (d) Precedence Constraints (PC): Certain jobs may have a specific order in which they must be processed.

As one example, we aim to optimize the job scheduling for reconfiguration considering the following objectives:

1. Minimize Makespan (M)
2. Minimize Workload of the Most Loaded Asset (WL)
3. Minimize Tardiness (T)

The decision variables in this problem are the sequence of parts to be released into the system, the selection of the operation/asset pair, and the sequence of processes assigned to each asset in the production process. The optimization problem can be formulated as follows:

$$\begin{aligned}
\text{minimize}\quad & f(x) = \{M(x), WL(x), T(x)\} \\
\text{subject to}\quad & EDD_i(x) \le 0, \quad i = 1, \ldots, m_1 \\
& MC_j(x) = 0, \quad j = 1, \ldots, n_1 \\
& JC_k(x) = 0, \quad k = 1, \ldots, n_2 \\
& AC_l(x) = 0, \quad l = 1, \ldots, n_3 \\
& PC_m(x) = 0, \quad m = 1, \ldots, n_4
\end{aligned} \tag{8}$$

where:

- x: A vector representing the decision variables.
- $f(x)$: A vector containing the objective functions to be minimized, which include Makespan ($M(x)$), Workload Balance ($WL(x)$), and Total Time ($T(x)$).
- $\mathrm{EDD}_i(x)$: The ith earliest due date constraint function that must be satisfied by the decision variables x.
- $\mathrm{MC}_j(x)$: The jth machine constraint function that must be satisfied by the decision variables x.
- $\mathrm{JC}_k(x)$: The kth job constraint function that must be satisfied by the decision variables x.
- $\mathrm{AC}_l(x)$: The lth asset constraint function that must be satisfied by the decision variables x.
- $\mathrm{PC}_m(x)$: The mth precedence constraint function that must be satisfied by the decision variables x.
- m_1: The number of earliest due date constraints.
- n_1: The number of machine constraints.
- n_2: The number of job constraints.
- n_3: The number of asset constraints.

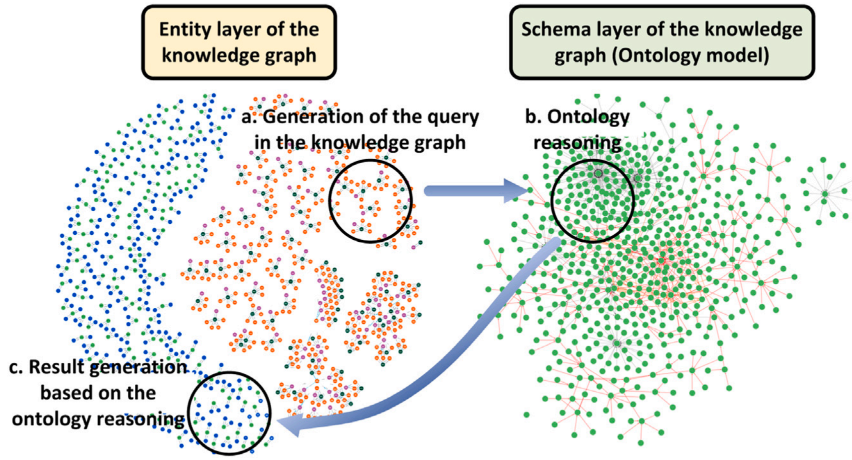

Fig. 4. Knowledge inference process in the knowledge graph: a. Generation of the query in the knowledge graph. b. Ontology reasoning. c. Generation of the inference results based on the ontology reasoning.

- n_4: The number of precedence constraints.

By solving this optimization problem, the optimal solution for the job scheduling problem can be found, considering the objectives and constraints. The decision variables x represent the allocation of jobs to machines, and the constraints ensure that the schedule satisfies the earliest due dates, machine capacities, job requirements, asset availability, and job dependencies. The optimal solution helps to minimize the makespan, balance the workload, and reduce the total time needed to complete the job schedule.

3.4. Construction and utilization of the entity layer of the knowledge graph

The creation of the entity layer of the knowledge graph is based on the OCCR model and the synthesized knowledge derived from the manufacturing data. The schema layer is then continually refined through the integration of bottom-up and top-down methods in the proposed framework by incorporating valuable information and insights obtained from the entity layer.

Fig. 3 shows the detailed steps of building the entity layer of the knowledge graph. The implementation of each step will depend on the application domain and organization. Detailed information about each step is depicted below:

1. **Initial ontology construction**
   Our proposed methodology uses the OCCR model as the initial ontology.
2. **Knowledge source identification**
   The next step is to identify the different sources of the manufacturing data, which will be used to tailor the model to the application domain. In manufacturing scenarios, the knowledge resource usually includes heterogeneous data sources, including customer requirement documents, datasets, and CAD models. These resources are multi-modal with different forms and hence require separate processing methods.
3. **Knowledge extraction**
   The knowledge extraction method is applied to extract the source data. For the customer requirement document, natural language processing techniques such as named entity recognition [53], relation extraction [54], and attribute extraction [55] are utilized. The knowledge extraction process combines the manufacturing domain knowledge and terms as the keyword corpus. For the dataset, the time/frequency analysis and pattern recognition process is applied to extract the data. For the CAD model data, a data extraction tool such as API is applied to extract the important geometry features, material information, and kinematics [56].

Fig. 5. Physical layout of one of the test plants of the OMNIFACTORY.

4. **Building the entity layer of the knowledge graph**
   The entity layer of the knowledge graph is established using the extracted data and the ontology model provided by Section 3.2 to construct the schema layer. Despite being constructed from multiple sources, the generated knowledge graph may still have incomplete information, missing certain triples. To compensate for this, knowledge graph completion is performed through a combination of manual completion by engineers, completion based on established rules, and automatic completion utilizing either graph structure or embedding-based algorithms [57].
5. **Knowledge storage**
   Once the entity layer of the knowledge graph has been established and updated, it can be stored in either graph databases [58] or Resource Description Format (RDF) format [50]. These

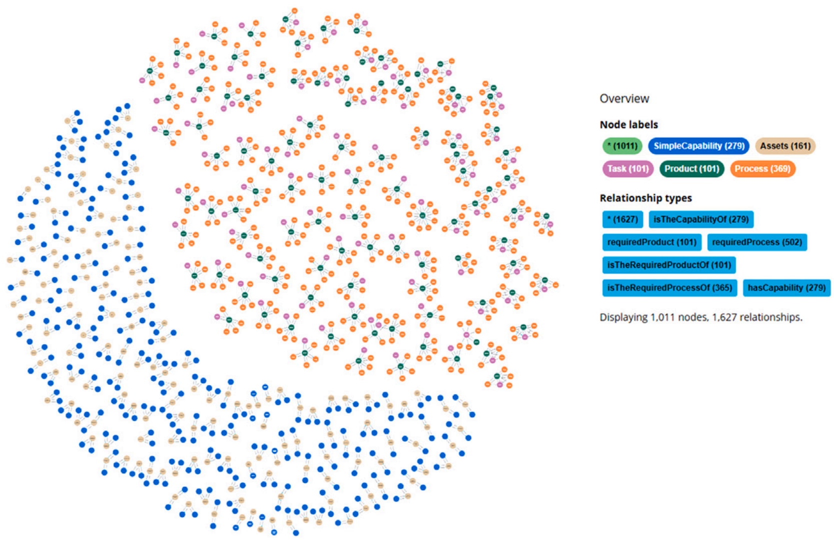

Fig. 6. Construction of the industry reconfiguration knowledge graph.

storage solutions provide efficient querying capabilities and effective management of substantial volumes of knowledge graph data.

6. **Knowledge inference**
After the creation of the schema layer and entity layer of the knowledge graph, the resulting knowledge graph can be applied to various aspects of RMS and beyond. The proposed ontology model enables reasoning for capability matching and recommendations for reconfiguration solutions, as illustrated in Fig. 4. Capability matching should consider not only the static requirements, such as the required payload and reachability but also the capacity information, such as the cost and utilization rate. If there are multiple candidate assets after the capability matching process, Mo et al. developed a resource selection process in [59] to select the most appropriate assets. Once the reconfiguration model recommends a solution, the engineer would decide on the criteria for optimizing the reconfiguration. Layout optimization, resource selection, and job scheduling are typically multi-objective optimization problems. If the engineers can decide the weights of the objectives in advance, then the multi-objective optimization problem can be converted into a single-objective optimization problem [60]. Otherwise, a posteriori method can be used, aiming to produce all Pareto optimal solutions or a representative subset of the Pareto optimal solutions [61].

4. Implementation and validation

To validate the proposed methodology, a dataset was created based on information from the OMNIFACTORY demonstrator at the University of Nottingham [62] as shown in Fig. 5. The OMNIFACTORY serves as a national demonstrator and testbed for smart manufacturing systems in the United Kingdom, with the aim of enabling fast and accurate reconfiguration on the shop floor based on customers' customized requirements. The OMNIFACTORY is a new facility that cost £3.8 million and is designed to revolutionize manufacturing, making it more efficient and cost-effective. It is located on the University's Jubilee Campus and features a bespoke flooring system that provides a unique reconfigurable environment. Despite this reconfiguration potential, understanding what configuration is required to produce a new or changed product is an unsolved problem. The dataset used for validation was created based on technical documents, equipment information, and product design documents from OMNIFACTORY and partner companies. The dataset is comprised of 101 distinct tasks,

```
MATCH  (a:Task {Name:"Task 100"})-[:requiredProduct]→
(b:Product)-[:requiredProcess]→(c:Process)
RETURN a,b,c
```

Fig. 7. Query command to find the required process for "Task 100" in Neo4j.

each of which is categorized as either reconfiguration-related or non-reconfiguration-related. For each task, the dataset specifies the requirements that need to be fulfilled, as well as the processes necessary to complete the task. Besides, the dataset includes 161 candidate assets with information supporting capability matching and the reconfiguration process. These assets include the production line assets and assets from the asset pool, such as hardware, software, human workforce, and reconfiguration solvers to support the production and reconfiguration process. To maintain confidentiality, an anonymization process was applied, including changing the task names to "Task 1", "Task 2" etc., and the customer names to "Company A", "Company B" etc. Similarly, product features were replaced with generic names such as "Feature 1" and "Feature 2".

4.1. Building the schema layer and the entity layer of the knowledge graph for OMNIFACTORY

Neo4j was utilized as the implementation platform for visualizing the knowledge graph and developing knowledge graph applications [63]. To establish a connection between the Neo4j graph database and the programming interface, Py2neo was used as it allows for the manipulation of the Neo4j database within a Python environment [64]. The decision to use Py2neo was motivated by its ability to automatically construct the knowledge graph in Neo4j through Python programming, as well as its ease of updating the knowledge graph in Neo4j. The advantages of Py2neo align with the objective of achieving a high degree of automation in the reconfigurable manufacturing system.

As mentioned in Section 3, in the manufacturing domain, the semantic model can never be exhaustive enough to cover all concepts. For example, to achieve reconfigurability, technologies are always developing. New software or algorithms to achieve reconfigurability will be updated to the semantic model continuously. As previously stated, the combination of the top-down and the bottom-up approaches was applied in our validation case. The schema layer was created based on the OCCR model and the experience of the engineers according to the top-down approach at first. Then the entity layer was updated based on the data we utilized and the generated schema layer. Due to

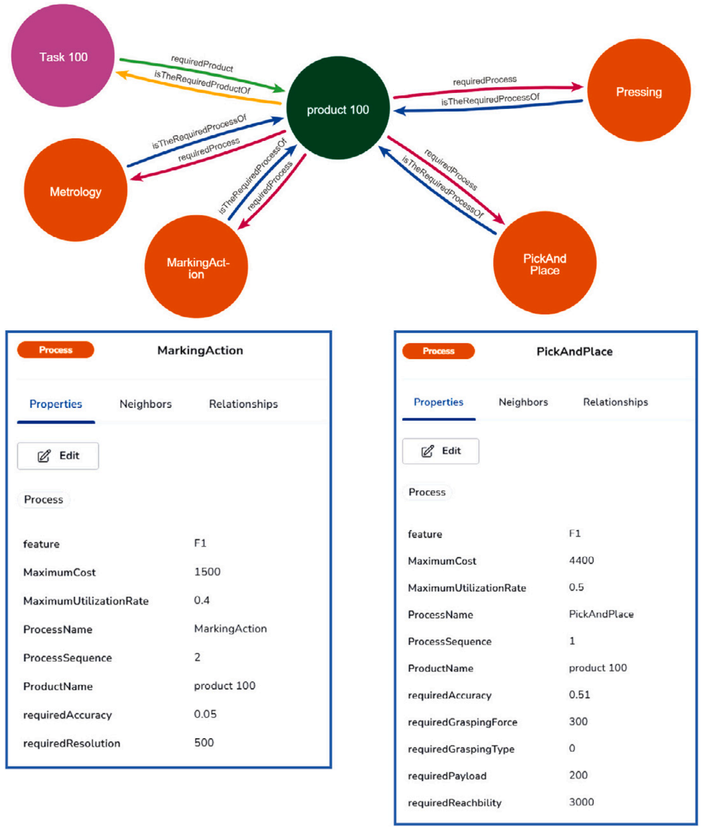

Fig. 8. Information about task 100 and its related nodes in the knowledge graph.

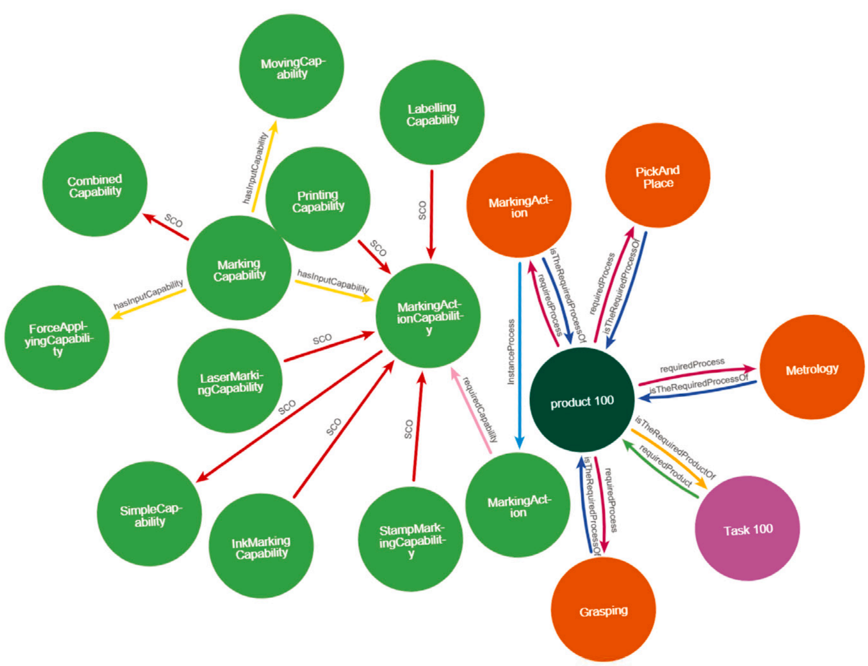

Fig. 9. Capability matching for "MarkingActionCapability" with the help of ontology reasoning.

the characteristics of the bottom-up approach, the schema layer was updated based on the entity layer's data. Taking one of the subclasses of the assets semantic model as an example, at first, we got the information from the OCCR model to create the schema layer of the robots in our knowledge graph. Then we utilized a Python crawler package with the name of "Scrapy" to extract the robot information from the internet, communicated with the engineers, and referred to the technical documents to get the entity information of the robot data. Then the entity layer was updated. At last, the subclass of the schema layer was updated [65,66].

Fig. 6 is the generated knowledge graph with explicit relationships. To enhance and complete the knowledge graph, an ontology-rule-based

Ontology reasoning command in Neo4j without considering the capacity information

```
MATCH (:Task {Name:"Task 100"})-[:requiredProduct]→(:Product)-
[:requiredProcess]→(d:Process {ProcessName:"MarkingAction"})
WITH d
MATCH (x:Concept {name: d.ProcessName + "Capability"})
CALL n10s.inference.nodesInCategory(x,
{inCatRel:"InstanceCapability", subCatRel:"SCO"}) YIELD node AS
capability
WITH capability, d
MATCH (capability)-[:isTheCapabilityOf]→(dev:Assets)
WHERE dev.Resolution > d.requiredResolution AND dev.Accuracy <
d.requiredAccuracy
RETURN DISTINCT dev.Name AS AssetName, dev.Type AS AssetType,
dev.Resolution AS Resolution, dev.Accuracy AS Accuracy, dev.Cost
AS Cost, dev.UtilizationRate AS UtilizationRate
```

Capability matching results without considering the capacity information

AssetName	AssetType	Resolution	Accuracy	Cost	UtilizationRate
InkMarker-3	InkMarker	800	0.03	800	0.1
InkMarker-2	InkMarker	600	0.02	600	0.5
LaserMarker-3	LaserMarker	1800	0.015	1850	0.14
LaserMarker-1	LaserMarker	1200	0.01	1700	0.08
LaserMarker-2	LaserMarker	1500	0.012	1650	0.15
Printer-3	Printer	2000	0.02	2100	0.15

Enhanced results

Ontology reasoning command in Neo4j with considering the capacity information

```
MATCH (:Task {Name:"Task 100"})-[:requiredProduct]→(:Product)-
[:requiredProcess]→(d:Process {ProcessName:"MarkingAction"})
WITH d
MATCH (x:Concept {name: d.ProcessName + "Capability"})
CALL n10s.inference.nodesInCategory(x,
{inCatRel:"InstanceCapability", subCatRel:"SCO"}) YIELD node AS
capability
WITH capability, d
MATCH (capability)-[:isTheCapabilityOf]→(dev:Assets)
WHERE dev.Resolution > d.requiredResolution AND dev.Accuracy <
d.requiredAccuracy AND dev.Cost < d.MaximumCost AND
dev.UtilizationRate < d.MaximumUtilizationRate
RETURN DISTINCT dev.Name AS AssetName, dev.Type AS AssetType,
dev.Resolution AS Resolution, dev.Accuracy AS Accuracy, dev.Cost
AS Cost, dev.UtilizationRate AS UtilizationRate
```

Capability matching results with considering the capacity information

AssetName	AssetType	Resolution	Accuracy	Cost	UtilizationRate
InkMarker-3	InkMarker	800	0.03	800	0.1

Fig. 10. The capacity model enhances the resource selection process.

Command in Neo4j to find the input capability of PickAndPlace

```
MATCH (:Task{Name:"Task 100"})-[:requiredProduct]→
(:Product)-[:requiredProcess]→
(d:Process{ProcessName:"PickAndPlace"})
WITH *
MATCH (x:Concept {name: d.ProcessName +
"Capability"})
CALL n10s.inference.nodesInCategory(x,
{inCatRel:"SCO_RESTRICTION"}) YIELD node AS f
MATCH (x)-
[r:SCO_RESTRICTION{onPropertyName:"hasInputCapabilit
y"}]→(f)
RETURN f,r,x
```

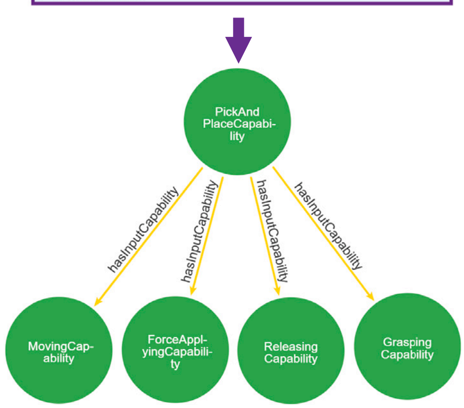

Fig. 11. Capability decomposition process for PickAndPlace capabiltiy.

methodology was employed in our study, as described in the work of Chen et al. [57].

4.2. Use cases of the knowledge graph applications

Once the knowledge graph has been established, its effectiveness and the improvements it brings to the reconfiguration process can be demonstrated through two specific use cases. The first use case addresses the selection of resources for a task characterized as non-reconfiguration-related, while the second use case deals with providing reconfiguration recommendations for a reconfiguration-related task.

4.2.1. Use case 1: Resource selection for NRT

In our first use case, task 100 in the knowledge graph is selected as the demonstration task. "Task 100" is non-reconfiguration-related. We used the knowledge graph to help the task find the most appropriate assets. This process consists of two steps: finding the required process to produce the product from "Task 100" (step 1), capability matching between the required process and the available capability, and finding the candidate assets based on the capability information (step 2). In this example, we will not only show the capability matching based on ontology reasoning but also show how to decompose the combined capability to find the potential assets for the input capability of the combined capability. For the first step, the required process for "Task 100" can be found via the query command as shown in Fig. 7 in Neo4j. Through the implemented methodology, it was discovered that product 100 has two distinct features.

Fig. 8 demonstrates that feature 1 necessitates the "PickAndPlace" and "MarkingAction" processes, while feature 2 requires the "Pressing" and "Metrology" processes.In relation to the capability matching process (step 2), Fig. 8 demonstrates also that the requirements for various processes within the task can be queried utilizing the generated knowledge graph. As an example, for feature 1, "PickAndPlace" and "MarkingAction" are the necessary processes. Given that the "MarkingCapability" is categorized as a simple capability, while the "PickAndPlaceCapability" is a combined capability in the OCCR model, we can employ the capability matching process to illustrate how our model functions with different types of capabilities. Specifically, the matching process for the "MarkingAction" process and the "PickAndPlace" process can be demonstrated.

The specification requirement of "MarkingAction" is "requiredAccuracy: 0.05" and "requiredResolution: 500". The capacity requirement is the cost and utilization rate. The allowed maximum cost for "MarkingAction" is 1500, and the allowed maximum utilization rate is 0.4. The ontology reasoning approach was used in our use case to find the potential assets based on the requirements and specifications, as well as the capacity information. The knowledge graph made inferences (ontology reasoning) in the capability-matching process. As shown in Fig. 9, with this ontology reasoning approach, not only the related assets for the "MarkingActionCapability" can be found, but also the assets which have the capability of the subclass of "MarkingActionCapability" can be found. In this figure, SCO means "subclass of". The subclasses of the "MarkingActionCapability" are "LaserMarkingCapability", "InkMarkingCapability", "PrintingCapability", "StampMarkingCapability", and "LabellingCapability". The related assets for these subclasses could be automatically found without any extra effort. The

Finding candidate assets for “ForceApplying” and “Moving” capability

```
MATCH (:Task{Name:"Task 100"})-[:requiredProduct]→(:Product)-
[:requiredProcess]→(d:Process{ProcessName:"PickAndPlace"})
WITH *
MATCH (x:Concept {name: d.ProcessName + "Capability"})
CALL n10s.inference.nodesInCategory(x,
{inCatRel:"SCO_RESTRICTION"}) YIELD node AS capability
MATCH (x)←
[r:SCO_RESTRICTION{onPropertyName:"IsTheInputCapabilityOf"}]-
(capabilityConcept)
WITH d, capabilityConcept
CALL n10s.inference.nodesInCategory(capabilityConcept,
{inCatRel:"InstanceCapability"}) YIELD node AS capability
WITH d, capability
MATCH (capability)-[:isTheCapabilityOf]→
(dev:Assets{Type:"Robot"}) WHERE dev.Payload > d.requiredPayload
AND dev.Reachability > d.requiredReachability
RETURN DISTINCT dev.Name as AssetName, dev.Type as AssetType,
dev.Payload as Payload, dev.Reachability as Reachability,
dev.UtilizationRate as UtilizationRate, dev.Cost as Cost
```

AssetName	AssetType	Payload	Reachability	UtilizationRate	Cost
FANUC M-900iB/400L	Robot	400	3704	0.8	5200
FANUC M-2000iA/1700L	Robot	1700	4683	0.3	6000
KUKA KR 240 R3330	Robot	240	3326	0.4	4500
FANUC M-2000iA/900L	Robot	900	4683	0.3	5500
KUKA KR 210 R3300 K ultra	Robot	210	3301	0.4	4500
KUKA KR 210 R3100 ultra	Robot	240	3095	0.4	3200

Finding candidate assets for “Grasping” and “Releasing” capability

```
MATCH (:Task{Name:"Task 100"})-[:requiredProduct]→(:Product)-
[:requiredProcess]→(d:Process{ProcessName:"PickAndPlace"})
WITH *
MATCH (x:Concept {name: d.ProcessName + "Capability"})
CALL n10s.inference.nodesInCategory(x,
{inCatRel:"SCO_RESTRICTION"}) YIELD node AS capability
MATCH (x)←
[r:SCO_RESTRICTION{onPropertyName:"IsTheInputCapabilityOf"}]-
(capabilityConcept)
WITH d, capabilityConcept
CALL n10s.inference.nodesInCategory(capabilityConcept,
{inCatRel:"InstanceCapability"}) YIELD node AS capability
WITH d, capability
MATCH (capability)-[:isTheCapabilityOf]→
(dev:Assets{Type:"Grippers"}) WHERE
dev.GraspingForce_max>d.requiredGraspingForce or
dev.HoldingForce_max>d.requiredGraspingForce
RETURN DISTINCT dev.Name as AssetName, dev.Type as AssetType,
dev.Subtype as Subtype, dev.GraspingForce_max as GraspingForce,
dev.HoldingForce_max as HoldfingForce, dev.Cost as Cost
```

AssetName	AssetType	Subtype	GraspingForce	HoldingForce	Cost
Vacuum Gripper-5	Gripper	Vacuum Gripper		340	1100
Vacuum Gripper-6	Gripper	Vacuum Gripper		400	1200
Finger Gripper-6	Gripper	Finger Gripper	500		1500
Finger Gripper-4	Gripper	Finger Gripper	400		1000
Finger Gripper-5	Gripper	Finger Gripper	450		1200

Fig. 12. Finding candidate assets for “ForceApplying”, “Moving”, “Grasping” and “Releasing” capability.

knowledge inference process (ontology reasoning) in Neo4j is essential because it allows the representation and manipulation of complex relationships between entities in a graph database. Using ontologies, or formal models of a particular domain, the system can automatically deduce new information (in our case, the subclasses of the “MarkingActionCapability”) based on the rules encoded in the ontology. This can improve data accuracy, consistency, and completeness and help users make more informed decisions.

The results of the capability matching process are presented in Fig. 10. For the marking process in task 100, six assets were identified to satisfy the requirement according to the capability matching process, based on the specification requirement (accuracy and resolution), without considering the capacity information. The capacity model was employed to improve the resource selection process. In our study, cost and utilization rate were utilized to represent the capacity information of the candidate assets. It was observed that only “InkMarker-3” fulfilled all the requirements and was thus chosen.

Regarding the execution of the process for “PickAndPlace”, Fig. 8 indicates that the specifications for “PickAndPlace” process include “requiredPayload: 200”, “requiredGraspingForce: 300”, and “requiredReachability: 3000”. The capacity requirements include “MaximumCost: 4000” and “MaximumUtilizationRate: 0.5”. Compared with the capability matching process for a simple capability, this process has an extra process called capability decomposition. Since the “PickAndPlace” capability is a combined capability, the capability decomposition of the combined capability must be performed. The decomposed capability was found based on ontology reasoning as shown in Fig. 11.

It can be observed that “Moving”, “Releasing”, “Grasping”, and “ForceApplying” are the decomposed capabilities (input capabilities). In our OCCR model, “Moving” and “ForceApplying” are two simple capabilities of the robots. Hence, only one robot is required to execute these two capabilities. The same approach applies to “Releasing” and “Grasping”, where only one gripper is necessary to execute these two capabilities. Utilizing ontology reasoning, we obtained the capability-matching results, which are presented in Fig. 12.

The candidate assets for the “ForceApplying” and “Moving” capability, without considering the capacity information, were *Fanuc M-900iB/400L, KUKA KR210 R3300 K ultra, KUKA KR 2100 R3100 ultra, Fanuc M-200iA/900L, KUKA KR 240 R3330, KUKA KR 210–2 3100, Fanuc M-2000iA/1700L*. Similarly, the candidate assets for the “Grasping” and “Releasing” capabilities were *Vacuum Gripper-6, Vacuum Gripper-5, Finger Gripper-5, Finger Gripper-6, and Finger Gripper-4*. To enhance the capability results, the capacity model was utilized, and the cost information considered both the cost of the robot and the gripper. From Fig. 13, it was observed that two combinations satisfied the requirement, namely [*KUKA KR210 R3100 ultra, Finger Gripper-4*], and [*KUKA KR210 R3100 ultra, Vacuum Gripper-5*]. According to the asset selection method proposed by Fan et al. [59], mentioned in Section 3.4, [*KUKA KR210 R3100 ultra, Finger Gripper-4*] was chosen to execute the pick and place process for “Task 100”.

It’s important to note that the calculated costs for these robots and grippers may not align perfectly with their actual market prices. However, these difference doesn’t detract from the effectiveness of our model, as our primary aim is to demonstrate how incorporating capacity information can refine the capability matching results.

4.2.2. Use case 2: Enhancing the reconfiguration task with the semantic reconfiguration model

In this use case, we will demonstrate how we use the reconfiguration model to enhance the reconfiguration process in a use case from the OMNIFACTORY project of the University of Nottingham. This task was stored in the knowledge graph and marked as “Task 50”. As shown in Fig. 14, task 50 is a reconfiguration task in the aerospace domain. There were two types of processes in the reconfiguration task. One type is the current process which needs to be reconfigured. The other type is the reconfiguration solution, such as the layout reconfiguration process, resource selection process, and job scheduling process. With these two types of processes, reconfiguration can be better described. Not only the information about the current production but also the solutions which are needed to do reconfiguration are explained.

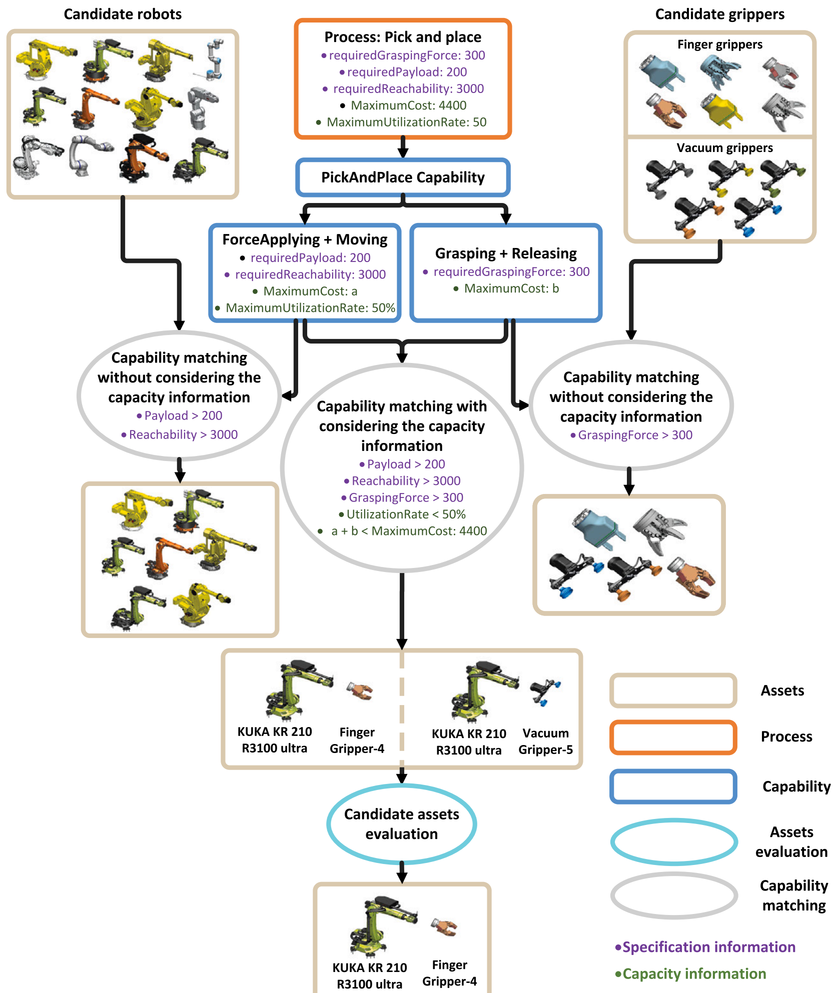

Fig. 13. Detailed process about finding candidate assets for "ForceApplying", "Moving", "Grasping" and "Releasing" capability.

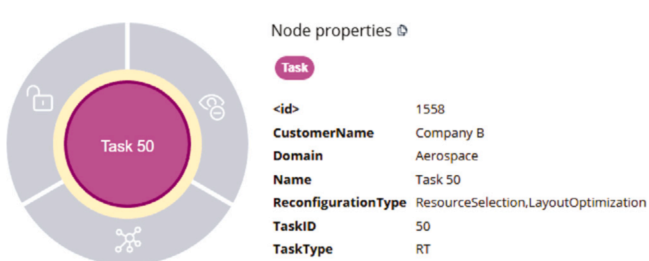

Fig. 14. Information about Task 50.

Regarding the process which needs to be reconfigured, as shown in Fig. 15, there is a frame on the AGV, and the robot needs to pick the front and aft beam to assemble them on the frame so that the frame can be further used for assembling parts. The processes required for the reconfiguration are described below:

1. **Mount the pick-and-place end effector**
   The robot should at first mount the pick-and-place end effector to enable the pick-and-place capability.
2. **Pick-and-place the front beam**
   The robot picks the front beam and places it on the upper side of the frame.
3. **Pick-and-place the aft beam**
   The robot picks the aft beam and places it on the lower side of the frame.
4. **Unmount the pick-and-place end effector**
   The robot unmounts the pick-and-place end effector and places it on the tool stand.
5. **Mount the metrology end effector**
   The robot mounts the metrology end effector to enable the metrology capability
6. **Execute metrology operation**
   The robot utilizes the metrology end effector for metrology on the mounted front beam and the after beam.
7. **Unmount the metrology end effector**
   At last, the robot unmounts the metrology end effector and puts it on the tool stand.

On the other hand, the reconfiguration solution for this task comprises two main components: "resource selection" and "layout optimization". Through ontology reasoning in the knowledge graph, the resource selection model offers guidance on the objectives, decision

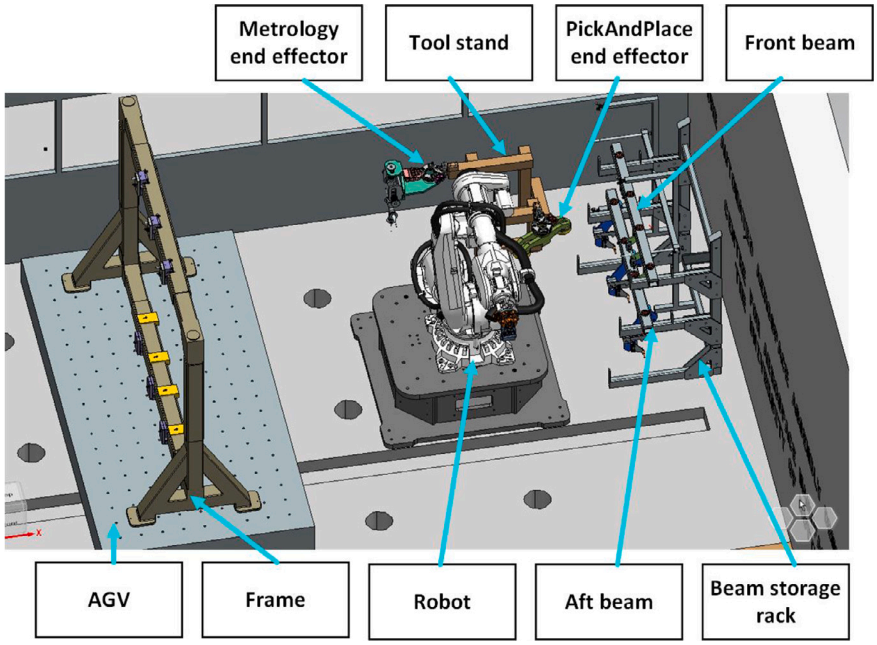

Fig. 15. Use case 2: Reconfiguration task.

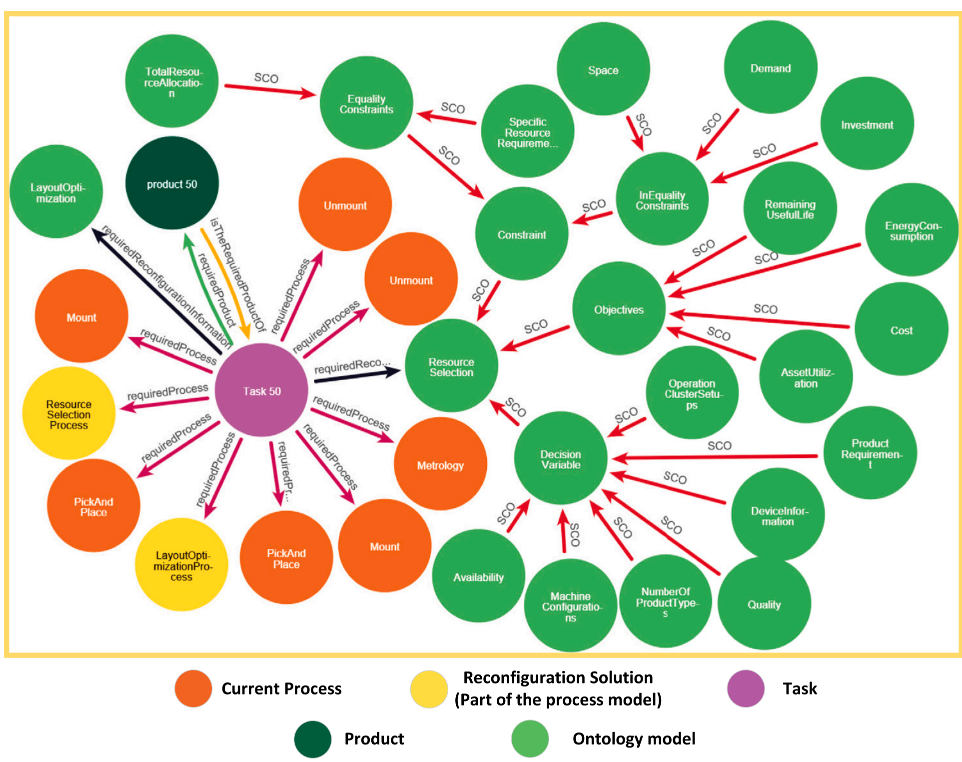

Fig. 16. Reconfiguration model to enhance the resource selection process.

variables, constraints, and reconfiguration solutions that should be considered for this task, as depicted in Fig. 16.

Similarly, the layout optimization semantic model assists in this task by providing recommendations for layout optimization via ontology reasoning in the knowledge graph. These recommendations pertain to the objectives, decision variables, constraints, and reconfiguration solutions in the asset model that should be considered during the reasoning process, as illustrated in Fig. 17.

The reconfiguration model acquires recommendations by querying the process model and capability model. The capability model supplies information about potential assets capable of executing the reconfiguration solution process. For instance, the reconfiguration process

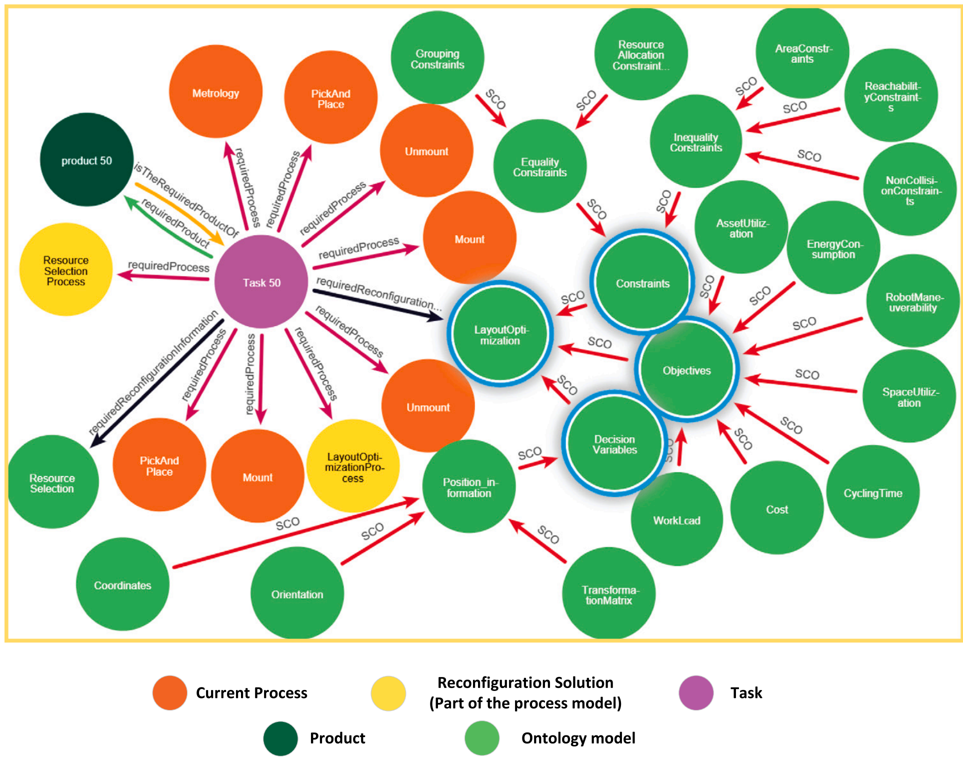

Fig. 17. Reconfiguration model to enhance the layout reconfiguration process.

necessitates resource selection and layout optimization, as discussed in Task 50. The capability model offers potential assets to execute the "ProcessNeedsReconfiguration" and information regarding the reconfiguration solution required for resource selection. Moreover, the capability model determines whether it is essential to change the current assets for reconfiguration. The reconfiguration solution for layout optimization is identified through the connection between the capability semantic model and the asset semantic model. As shown in Fig. 18, various reconfiguration solutions are available to address the layout optimization problem.

The optimization process is executed based on the enhanced information from the reconfiguration model, capability model, capacity model, and process model. A framework designed by Mo et al. [59] is employed for the optimization process. The experimental results are then stored in the semantic model to serve as a reference for future use cases. In summary, the reconfiguration model is capable of performing a series of tasks to optimize the layout and resource selection processes.

1. Provide recommendations about the objects, constraints, and decision variables to be considered in the reconfiguration optimization problem.
2. Help the reconfiguration-related task to find the potential assets to do the reconfiguration task (in our case, resource selection and layout optimization).
3. Store the optimization information in the knowledge graph for future reference.

5. Conclusion

In conclusion, this research paper presents a comprehensive and unified ontological framework that effectively captures capability, capacity, and reconfiguration information in a vendor-neutral manner within the context of robot manufacturing cells. The proposed methodology, which combines top-down and bottom-up strategies, streamlines the construction and updating of the knowledge graph, consequently offering significant advantages for intelligent search, tailored recommendations, and perceptive query resolution.

The capability of the knowledge graph to manage real-time inquiries and dynamic modifications is validated through two distinct use cases. The first use case focuses on a non-reconfiguration task, which employs the knowledge graph to identify the most appropriate assets and requisite processes for product production. Moreover, it demonstrates the process of decomposing combined capabilities to uncover potential assets for the input capability of the combined capability. The second use case examines a reconfiguration task from the OMNIFACTORY project at the University of Nottingham. This case underscores the application of the reconfiguration model in the aerospace domain, illustrating the role of current processes necessitating reconfiguration and reconfiguration solutions in specific reconfiguration types, such as layout, resource selection, and job scheduling processes.

These practical examples substantiate the efficacy of the ontological model and knowledge graph in optimizing the utilization of reconfigurable manufacturing systems. Additionally, the semantic modeling approach contributes to long-lasting improvements in system reconfiguration by documenting past experiences.

Future research endeavors will investigate the relationship between control reconfiguration and the three categories of reconfiguration discussed in the paper. The objective is to incorporate these distinct aspects into a comprehensive reconfiguration model, thereby augmenting the framework's overall comprehension and applicability. Additionally, this methodology will be applied to various industrial use cases to further validate its effectiveness and adaptability. Concurrently, the possibility of enhancing the knowledge graph through semantic embedding techniques will be explored, aiming to improve the system's overall performance.

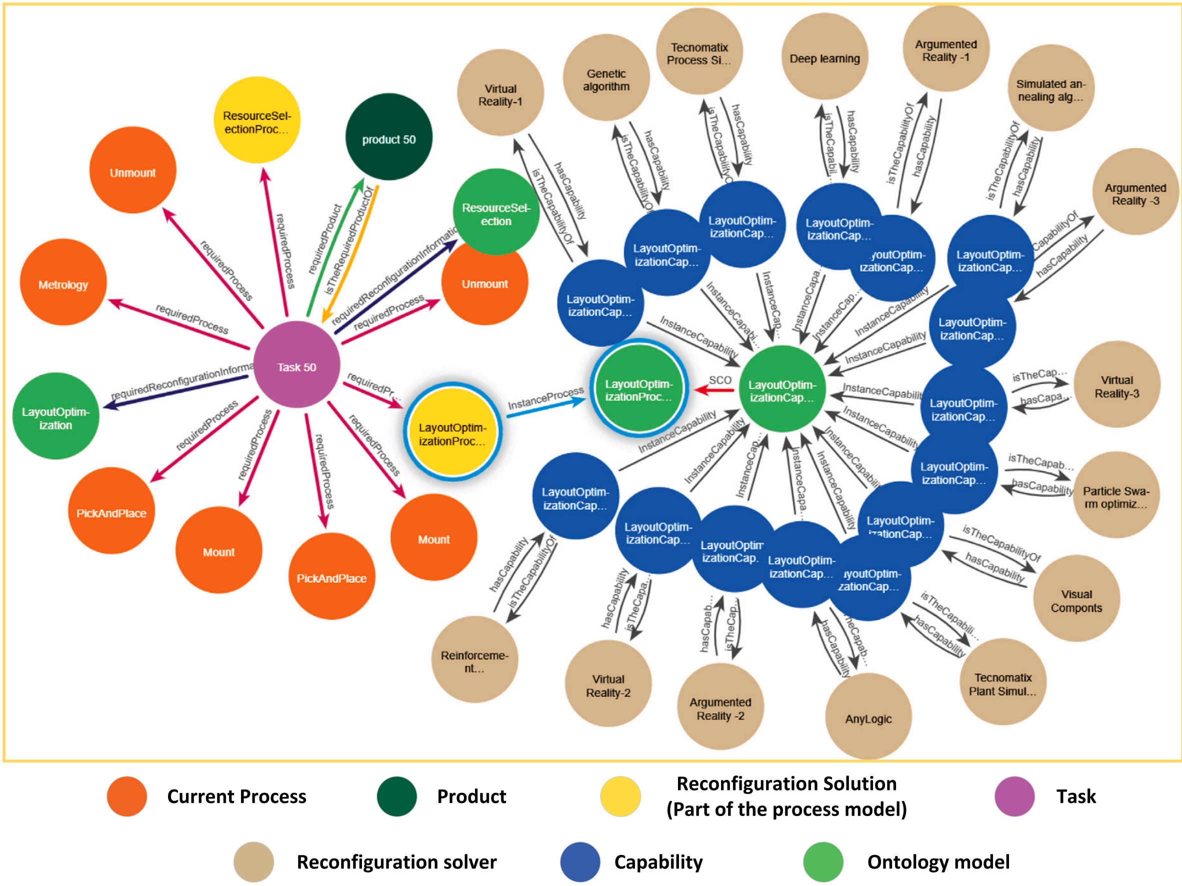

Fig. 18. Recommendations from the knowledge graph to solve the layout optimization problem.

CRediT authorship contribution statement

Fan Mo: Conceptualization, Methodology, Software, Validation, Writing – original draft. **Jack C. Chaplin:** Writing – review & editing, Conceptualization, Supervision. **David Sanderson:** Writing – review & editing, Conceptualization, Supervision. **Giovanna Martínez-Arellano:** Writing – review & editing, Conceptualization, Supervision. **Svetan Ratchev:** Resources, Supervision, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The data that has been used is confidential.

Acknowledgments

This research is supported by DiManD Innovative Training Network (ITN) project funded by the European Union through the Marie Skłodowska-Curie Innovative Training Networks (H2020-MSCA-ITN-2018) under grant agreement no. 814078, and the Elastic Manufacturing Systems project (Project Reference EP/T024429/1) funded by the UK Engineering and Physical Science Research Council. The authors would like to acknowledge the support from Innovate UK project ELCAT (ref 113235) and GKN Aerospace. We want also to express our sincere gratitude to Dr. Peter Kendall and Dr. Basem Elshafei at the University of Nottingham and Dr. Nikolai Kazantsev at the University of Cambridge for their valuable insights and thoughtful discussions that contributed to the development of this paper.

References

[1] Green Deal: New proposals to make sustainable products the norm and boost Europe's resource independence, https://ec.europa.eu/commission/presscorner/detail/en/IP_22_2013.

[2] F. Mo, J.C. Chaplin, D. Sanderson, H.U. Rehman, F.M. Monetti, A. Maffei, S. Ratchev, A Framework for Manufacturing System Reconfiguration Based on Artificial Intelligence and Digital Twin, Springer, 2022.

[3] L.A. Estrada-Jimenez, T. Pulikottil, N.N. Hien, A. Torayev, H.U. Rehman, F. Mo, S.N. Hojjati, J. Barata, Integration of cutting-edge interoperability approaches in cyber-physical production systems and industry 4.0, in: Design, Applications, and Maintenance of Cyber-Physical Systems, IGI Global, 2021, pp. 144–172.

[4] Elastic Manufacturing systems - a platform for dynamic, resilient and cost-effective manufacturing services, https://gow.epsrc.ukri.org/NGBOViewGrant.aspx?GrantRef=EP/T024429/1#:~:text=The%20Elastic%20Manufacturing%20Systems%20concept,a%20high%20degree%20of%20elasticity.

[5] Y. Koren, M. Shpitalni, Design of reconfigurable manufacturing systems, J. Manuf. Syst. 29 (4) (2010) 130–141.

[6] Z.M. Bi, S.Y. Lang, W. Shen, L. Wang, Reconfigurable manufacturing systems: the state of the art, Int. J. Prod. Res. 46 (4) (2008) 967–992.

[7] C. da Cunha, O. Cardin, G. Gallot, J. Viaud, Designing the digital twins of reconfigurable manufacturing systems: application on a smart factory, IFAC-PapersOnLine 54 (1) (2021) 874–879, http://dx.doi.org/10.1016/j.ifacol.2021.08.103.

[8] T. Pulikottil, L.a. Estrada-Jimenez, J.J.P. Abadía, A. Carrera-Rivera, A. Torayev, H.U. Rehman, F. Mo, Big Data Life Cycle in Shop-floor–Trends and Challenges.

[9] L. Ren, Y. Li, X. Wang, J. Cui, L. Zhang, An ABGE-aided manufacturing knowledge graph construction approach for heterogeneous IIoT data integration, Int. J. Prod. Res. (2022) 1–15.

[10] Y. Kong, D. Li, C. Li, D. Chu, Z. Yao, A Multi-source Heterogeneous Data Storage and Retrieval System for Intelligent Manufacturing.

[11] L. Guo, F. Yan, T. Li, T. Yang, Y. Lu, An automatic method for constructing machining process knowledge base from knowledge graph, Robot. Comput.-Integr. Manuf. 73 (2022) 102222.

[12] D. Fensel, U. Şimşek, K. Angele, E. Huaman, E. Kärle, O. Panasiuk, I. Toma, J. Umbrich, A. Wahler, D. Fensel, et al., Introduction: what is a knowledge graph? Knowl. Graphs: Methodol. Tools Sel. Use Cases (2020) 1–10.

[13] L. Ehrlinger, W. Wöß, Towards a definition of knowledge graphs, SEMANTiCS (Posters, Demos, SuCCESS) 48 (1–4) (2016) 2.

[14] G. Martínez-Arellano, K. Niewiadomski, F. Mo, B. Elshafei, J.C. Chaplin, D. Mcfarlane, S. Ratchev, 2023, Enabling coordinated elastic responses of manufacturing systems through semantic modelling.

[15] E. Järvenpää, N. Siltala, O. Hylli, M. Lanz, The development of an ontology for describing the capabilities of manufacturing resources, J. Intell. Manuf. 30 (2) (2019) 959–978.

[16] I. Horrocks, P.F. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, M. Dean, et al., SWRL: A semantic web rule language combining OWL and RuleML, W3C Memb. Submiss. 21 (79) (2004) 1–31.

[17] X. Chen, S. Jia, Y. Xiang, A review: Knowledge reasoning over knowledge graph, Expert Syst. Appl. 141 (2020) 112948.

[18] W. Zhang, B. Paudel, L. Wang, J. Chen, H. Zhu, W. Zhang, A. Bernstein, H. Chen, Iteratively learning embeddings and rules for knowledge graph reasoning, in: The World Wide Web Conference, 2019, pp. 2366–2377.

[19] A. Chadzynski, N. Krdzavac, F. Farazi, M.Q. Lim, S. Li, A. Grisiute, P. Herthogs, A. von Richthofen, S. Cairns, M. Kraft, Semantic 3D city database—An enabler for a dynamic geospatial knowledge graph, Energy AI 6 (2021) 100106.

[20] J. Wright, S.J. Rodríguez Méndez, A. Haller, K. Taylor, P.G. Omran, Schimatos: a SHACL-based web-form generator for knowledge graph editing, in: The Semantic Web–ISWC 2020: 19th International Semantic Web Conference, Athens, Greece, November 2–6, 2020, Proceedings, Part II, Springer, 2020, pp. 65–80.

[21] A.K. Deb, Economic reforms, capacity utilization and productivity growth in Indian manufacturing, Glob. Bus. Rev. 15 (4) (2014) 719–746.

[22] H.U. Rehman, T. Pulikottil, L.A. Estrada-Jimenez, F. Mo, J.C. Chaplin, J. Barata, S. Ratchev, Cloud based decision making for multi-agent production systems, in: EPIA Conference on Artificial Intelligence, Springer, 2021, pp. 673–686.

[23] B. Elshafei, F. Mo, J. Chaplin, G. Arellano, et al., Capacity Modelling and Measurement for Smart Elastic Manufacturing Systems, SAE Technical Paper, 2023, pp. 01–0997.

[24] M. Gruninger, C. Menzel, The process specification language (PSL) theory and applications, AI Mag. 24 (3) (2003) 63.

[25] S.R. Ray, A.T. Jones, Manufacturing interoperability, J. Intell. Manuf. 17 (6) (2006) 681–688.

[26] Y. Lu, Q. Shao, C. Singh, X. Xu, X. Ye, Ontology for manufacturing resources in a cloud environment, Int. J. Manuf. Res. 9 (4) (2014) 448–469.

[27] T. Wang, S. Guo, C.-G. Lee, Manufacturing task semantic modeling and description in cloud manufacturing system, Int. J. Adv. Manuf. Technol. 71 (9) (2014) 2017–2031.

[28] H. Komoto, Y. Furukawa, Modeling environmental performance of manufacturing systems from semantic and computational aspects, Proc. CIRP 107 (2022) 1011–1016.

[29] C. Siedler, P. Langlotz, J.C. Aurich, Modeling and assessing the effects of digital technologies on KPIs in manufacturing systems, Proc. CIRP 93 (2020) 682–687.

[30] D.L. Brandl, D. Brandl, KPI exchanges in smart manufacturing using KPI-ML, IFAC-PapersOnLine 51 (11) (2018) 31–35.

[31] X. Zou, A survey on application of knowledge graph, in: Journal of Physics: Conference Series, Vol. 1487, IOP Publishing, 2020, 012016, (1).

[32] X. Lu, S. Pramanik, R. Saha Roy, A. Abujabal, Y. Wang, G. Weikum, Answering complex questions by joining multi-document evidence with quasi knowledge graphs, in: Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2019, pp. 105–114.

[33] B. Qiao, K. Fang, Y. Chen, X. Zhu, Building thesaurus-based knowledge graph based on schema layer, Cluster Comput. 20 (2017) 81–91.

[34] D. Dessì, F. Osborne, D. Reforgiato Recupero, D. Buscaldi, E. Motta, H. Sack, Ai-kg: an automatically generated knowledge graph of artificial intelligence, in: International Semantic Web Conference, Springer, 2020, pp. 127–143.

[35] B. Zhou, J. Bao, J. Li, Y. Lu, T. Liu, Q. Zhang, A novel knowledge graph-based optimization approach for resource allocation in discrete manufacturing workshops, Robot. Comput.-Integr. Manuf. 71 (2021) 102160.

[36] L. Xia, P. Zheng, X. Li, R.X. Gao, L. Wang, Toward cognitive predictive maintenance: A survey of graph-based approaches, J. Manuf. Syst. 64 (2022) 107–120.

[37] G. Buchgeher, D. Gabauer, J. Martinez-Gil, L. Ehrlinger, Knowledge graphs in manufacturing and production: A systematic literature review, IEEE Access 9 (2021) 55537–55554.

[38] A. Banerjee, R. Dalal, S. Mittal, K.P. Joshi, Generating digital twin models using knowledge graphs for industrial production lines, UMBC Inf. Syst. Dep. (2017).

[39] J. Yuan, H. Li, Research on the standardization model of data semantics in the knowledge graph construction of oil&gas industry, Comput. Stand. Interfaces 84 (2023) 103705.

[40] Y. Koren, Reconfigurable manufacturing and beyond, in: CIRP 3rd International Conference on Reconfigurable Manufacturing, 2005.

[41] Y. Koren, U. Heisel, F. Jovane, T. Moriwaki, G. Pritschow, G. Ulsoy, H. Van Brussel, Reconfigurable manufacturing systems, CIRP Ann. 48 (2) (1999) 527–540.

[42] M. Bortolini, F.G. Galizia, C. Mora, Reconfigurable manufacturing systems: Literature review and research trend, J. Manuf. Syst. 49 (2018) 93–106.

[43] X. Wei, S. Yuan, Y. Ye, Optimizing facility layout planning for reconfigurable manufacturing system based on chaos genetic algorithm, Prod. Manuf. Res. 7 (1) (2019) 109–124.

[44] I. Maganha, C. Silva, L.M.D. Ferreira, The layout design in reconfigurable manufacturing systems: a literature review, Int. J. Adv. Manuf. Technol. 105 (2019) 683–700.

[45] G. Wang, G. Zhang, X. Guo, Y. Zhang, Digital twin-driven service model and optimal allocation of manufacturing resources in shared manufacturing, J. Manuf. Syst. 59 (2021) 165–179.

[46] C. Zhang, W. Song, Z. Cao, J. Zhang, P.S. Tan, X. Chi, Learning to dispatch for job shop scheduling via deep reinforcement learning, Adv. Neural Inf. Process. Syst. 33 (2020) 1621–1632.

[47] R. Chen, B. Yang, S. Li, S. Wang, A self-learning genetic algorithm based on reinforcement learning for flexible job-shop scheduling problem, Comput. Ind. Eng. 149 (2020) 106778.

[48] J. Zhang, G. Ding, Y. Zou, S. Qin, J. Fu, Review of job shop scheduling research and its new perspectives under industry 4.0, J. Intell. Manuf. 30 (2019) 1809–1830.

[49] J. Tan, Q. Qiu, W. Guo, T. Li, Research on the construction of a knowledge graph and knowledge reasoning model in the field of urban traffic, Sustainability 13 (6) (2021) 3191.

[50] X. Hao, Z. Ji, X. Li, L. Yin, L. Liu, M. Sun, Q. Liu, R. Yang, Construction and application of a knowledge graph, Remote Sens. 13 (13) (2021) 2511.

[51] M.Y. Zhang, R.Z. Du, A real-time inference method of graph attention network based on knowledge graph for lung cancer, in: 2021 5th International Conference on Digital Signal Processing, 2021, pp. 326–331.

[52] B. Zhou, J. Bao, Z. Chen, Y. Liu, Kgassembly: Knowledge graph-driven assembly process generation and evaluation for complex components, Int. J. Comput. Integr. Manuf. 35 (10–11) (2022) 1151–1171.

[53] V. Yadav, S. Bethard, A survey on recent advances in named entity recognition from deep learning models, 2019, arXiv preprint arXiv:1910.11470.

[54] N. Peng, H. Poon, C. Quirk, K. Toutanova, W.-t. Yih, Cross-sentence n-ary relation extraction with graph lstms, Trans. Assoc. Comput. Linguist. 5 (2017) 101–115.

[55] L. Sun, Research on product attribute extraction and classification method for online review, in: 2017 International Conference on Industrial Informatics-Computing Technology, Intelligent Technology, Industrial Information Integration (ICIICII), IEEE, 2017, pp. 117–121.

[56] S. Kwon, L.V. Monnier, R. Barbau, W.Z. Bernstein, Enriching standards-based digital thread by fusing as-designed and as-inspected data using knowledge graphs, Adv. Eng. Inform. 46 (2020) 101102.

[57] Z. Chen, Y. Wang, B. Zhao, J. Cheng, X. Zhao, Z. Duan, Knowledge graph completion: A review, Ieee Access 8 (2020) 192435–192456.

[58] H. Liu, G. Jiang, L. Su, Y. Cao, F. Diao, L. Mi, Construction of power projects knowledge graph based on graph database neo4j, in: 2020 International Conference on Computer, Information and Telecommunication Systems (CITS), IEEE, 2020, pp. 1–4.

[59] F. Mo, H.U. Rehman, F.M. Monetti, J.C. Chaplin, D. Sanderson, A. Popov, A. Maffei, S. Ratchev, A framework for manufacturing system reconfiguration and optimisation utilising digital twins and modular artificial intelligence, Robot. Comput.-Integr. Manuf. 82 (2023) 102524.

[60] N. Gunantara, A review of multi-objective optimization: Methods and its applications, Cogent Eng. 5 (1) (2018) 1502242.

[61] S. Petchrompo, A. Wannakrairot, A.K. Parlikad, Pruning Pareto optimal solutions for multi-objective portfolio asset management, European J. Oper. Res. 297 (1) (2022) 203–220.

[62] Introduction of the Omnifactory, https://www.omnifactory.co.uk/.

[63] M. Needham, A.E. Hodler, Graph Algorithms: Practical Examples in Apache Spark and Neo4j, O'Reilly Media, 2019.

[64] F. Zhao, Z. Zhang, D. Wang, KSG: Knowledge and skill graph, in: Proceedings of the 31st ACM International Conference on Information & Knowledge Management, 2022, pp. 4717–4721.

[65] D. Kouzis-Loukas, Learning Scrapy, Packt Publishing Ltd, 2016.

[66] W. Hejing, L. Fang, Z. Long, S. Yabin, C. Ran, Application research of crawler and data analysis based on python, Int. J. Adv. Netw. Monit. Controls 5 (2) (2020) 64–70.