KSPMI: A Knowledge-based System for Predictive Maintenance in Industry 4.0

Qiushi Cao[a,*], Cecilia Zanni-Merk[b], Ahmed Samet[c], Christoph Reich[d], François de Bertrand de Beuvron[c], Arnold Beckmann[a] and Cinzia Giannetti[e]

[a]Department of Computer Science, Swansea University, SA1 8EN Swansea, United Kingdom
[b]LITIS, Normandie Université/INSA Rouen, 76000 Saint-Étienne-du-Rouvray, France
[c]ICUBE/CSIP, INSA de Strasbourg, 24 Boulevard de la Victoire, 67084 Strasbourg, France
[d]IDACUS Institute of Data Science, Cloud Computing and IT Security, Hochschule Furtwangen University, 78120 Furtwangen, Germany
[e]The Future Manufacturing Research Institute (FMRI), Faculty of Science and Engineering, Swansea University, SA1 8EN Swansea, United Kingdom

ARTICLE INFO

Keywords:
Industry 4.0
Predictive maintenance
Knowledge-based system
Chronicle mining
Ontology reasoning

ABSTRACT

In the context of Industry 4.0, smart factories use advanced sensing and data analytic technologies to understand and monitor the manufacturing processes. To enhance production efficiency and reliability, statistical Artificial Intelligence (AI) technologies such as machine learning and data mining are used to detect and predict potential anomalies within manufacturing processes. However, due to the heterogeneous nature of industrial data, sometimes the knowledge extracted from industrial data is presented in a complex structure. This brings the semantic gap issue which stands for the lack of interoperability among different manufacturing systems. Furthermore, as the Cyber-Physical Systems (CPS) are becoming more knowledge-intensive, uniform knowledge representation of physical resources and real-time reasoning capabilities for analytic tasks are needed to automate the decision-making processes for these systems. These requirements highlight the potential of using symbolic AI for predictive maintenance.

To automate and facilitate predictive analytics in Industry 4.0, in this paper, we present a novel Knowledge-based System for Predictive Maintenance in Industry 4.0 (KSPMI). KSPMI is developed based on a novel hybrid approach that leverages both statistical and symbolic AI technologies. The hybrid approach involves using statistical AI technologies such as machine learning and chronicle mining (a special type of sequential pattern mining approach) to extract machine degradation models from industrial data. On the other hand, symbolic AI technologies, especially domain ontologies and logic rules, will use the extracted chronicle patterns to query and reason on system input data with rich domain and contextual knowledge. This hybrid approach uses Semantic Web Rule Language (SWRL) rules generated from chronicle patterns together with domain ontologies to perform ontology reasoning, which enables the automatic detection of machinery anomalies and the prediction of future events' occurrence. KSPMI is evaluated and tested on both real-world and synthetic data sets.

1. Introduction

In today's manufacturing, increasing global competition, fast technology evolution and customers' perceptions of product quality have triggered the demand for future strategical plans and advanced manufacturing techniques. To meet these challenges, traditional manufacturing companies are leveraging smart and digital technologies to achieve higher productivity and increased automation. This leads to a new industrial stage where vertical and horizontal manufacturing process integration and product connectivity are helping companies to achieve higher industrial performance [19]. This digitalisation and automation process is called *Industry 4.0*. Industry 4.0 is considered as a new industrial age where several emerging technologies are converged to provide digital solutions [25]. Benefiting from the rapid advance in data analytics and networking technologies, traditional manufacturing companies are transforming into the so-called *Smart Factories*, where production systems are autonomously controlled by computer programs.

With the development of internet technologies, smart factories are benefiting from ubiquitous connectivity and seamless data exchange among manufacturing systems. This trend is a key enabler for Industry 4.0, where manufacturing entities are equipped with smart technologies such as Artificial Intelligence (AI), Internet of Things (IoT), Cyber-Physical Systems (CPS), and Cloud Computing. These advanced technologies are integrated to ensure more reliable, increased automation, and self-monitoring manufacturing systems.

In industry, a *predictive maintenance* task relies on the monitoring of a measurable system diagnostic parameter, which identifies the state of a system [1, 3, 29]. Based on the current status of a system, early signs of machine anomalies (e.g. machine faults or failures) could be detected. Machine anomalies may be harmful to machine operation thus leading to the breakdown of manufacturing systems. The aim of predictive maintenance is to act before faults or failures happen, in order to ensure the smooth operation of production lines. Once possible machine anomalies are detected or predicted, diagnostic tasks will be activated to discover the root cause of anomalies. In this way, maintenance actions such as the intervention of machine operators can be proposed to avoid the breakdown of production lines. Predictive maintenance aims to improve reliability, availability,

*Corresponding author. E-mail address: qiushi.cao@swansea.ac.uk
ORCID(s):

and efficiency of manufacturing systems. In the manufacturing industry, data used for predictive analytics are often heterogeneous and big in volume [52].

Within manufacturing processes, big industrial data is collected from sensors installed on machines as well as manufacturing environments. To process the collected big data, statistical AI technologies are used to extract valuable information from heterogeneous data sources (e.g. product data, operational data, environment data). These technologies aim to achieve in-depth understanding and gain insight from data for accurate and timely decision making [26, 64]. The strong computational capability of statistical AI technologies is significantly improving the efficiency and accuracy of decision making. This enables manufacturing systems to be self-aware and self-maintained. This allows the systems to capture dynamic aspects of manufacturing environments and to be maintained automatically during system operation.

However, due to the heterogeneous nature of industrial data, sometimes the knowledge extracted from industrial data is presented in a complex structure. The data is often not machine-understandable, and are stored in data silos that are often not interconnected yet contain data that are semantically related. This brings the problem of semantic interoperability, which stands for the inability of different systems to exchange information with unambiguous meaning of data [**?**]. Therefore formal knowledge representation methods are required to facilitate the understanding and exploitation of the knowledge. Also, the task and context-specific nature of statistical AI methods have limited their reusability and implementation in knowledge-intensive manufacturing environments. This hampers the data-driven models to be reused under different contexts for automatic decision making. Furthermore, as the manufacturing domain is becoming more knowledge-intensive, uniform knowledge representation of physical resources and real-time reasoning capabilities for analytic tasks are needed to automate the decision-making processes for these systems. These limitations have highlighted the potential of using symbolic AI technologies such as domain ontologies and rule-based systems for predictive maintenance.

To automate and facilitate predictive maintenance tasks, in this paper, we present a novel Knowledge-based System for Predictive Maintenance in Industry 4.0, named KSPMI. KSPMI is developed based on a novel hybrid approach that leverages both statistical and symbolic AI technologies. The execution of the software starts from the chronicle mining (a special type of sequential pattern mining approach) process on industrial data sets, after which machine degradation models are extracted from the data indicating the time of occurrence of future machinery failures. After that, symbolic AI technologies, especially domain ontologies and Semantic Web Rule Language (SWRL) rule-based reasoning, use the extracted chronicle patterns to query and reason on system input data with rich domain and contextual knowledge. In this way, KSPMI enables the detection of future machinery failures as well as the prediction of their time of occurrence. The prediction of failures is associated with a pair of numeric values, which give the upper and lower numeric bounds of their time of occurrence. To improve the performance of failure prediction, KSPMI implements a rule pruning process for selecting a subset of best-quality SWRL rules for ontology reasoning. The rule pruning process is followed by an expert rule integration process, which enables the rule base to improve its quality progressively. During the integration process, KSPMI is programmed able to detect and avoid potential rule quality issues such as rule conflict, subsumption, and redundancy. The performance of KSPMI is evaluated and tested on both real-world and synthetic data sets. To the best of our knowledge, KSPMI is the first software that combines chronicle mining and ontology reasoning for Industry 4.0 predictive maintenance.

The remainder of this paper is structured as follows. Section 2 gives a comprehensive literature review on the existing approaches for Industry 4.0 predictive maintenance. Section 3 introduces the theoretical foundations of this paper. Within this section, formal definitions of relevant concepts and notions are given. Section 4 demonstrates the proposed novel hybrid approach that leverages both statistical and symbolic AI technologies for predictive maintenance. The proposed approach gives the foundation for the development work of KSPMI. Section 5 shows the implementation of KSPMI on two types of data sets. The first experimentation is performed on a real-world data set collected from a semi-conductor manufacturing process. The second experimentation is conducted on synthetic data sets. Section 6 concludes the paper and gives future perspectives.

2. Related work

Over the last decades, a great number of research efforts have been proposed to automate and facilitate smart manufacturing in Industry 4.0. For an Industry 4.0 predictive maintenance task, appropriate method selection is vital to manufactures. Among these research works, AI-based approaches have shown promising performance in anomaly prediction and classification. In this subsection, we categorise the existing AI-based smart manufacturing and predictive maintenance approaches into four groups: i) data-driven approaches; ii) physical model-based approaches; iii) knowledge-based approaches; iv) hybrid model-based approaches. Fig. 1 gives our categorisation results. As KSPMI is developed based on both knowledge-based and data-driven methods, the literature review focuses mainly on these two types of technologies. Following the hierarchy presented in Fig. 1, we describe the state of the art approaches for Industry 4.0 predictive maintenance in the following subsections.

2.1. Data-driven approaches

Recently, data-driven approaches have become a notable solution for smart manufacturing and predictive maintenance. Together used with industrial wireless sensor networks (IWSNs), CPS, and IoT technologies, big industrial data is collected and processed in an intelligent manner to facilitate decision

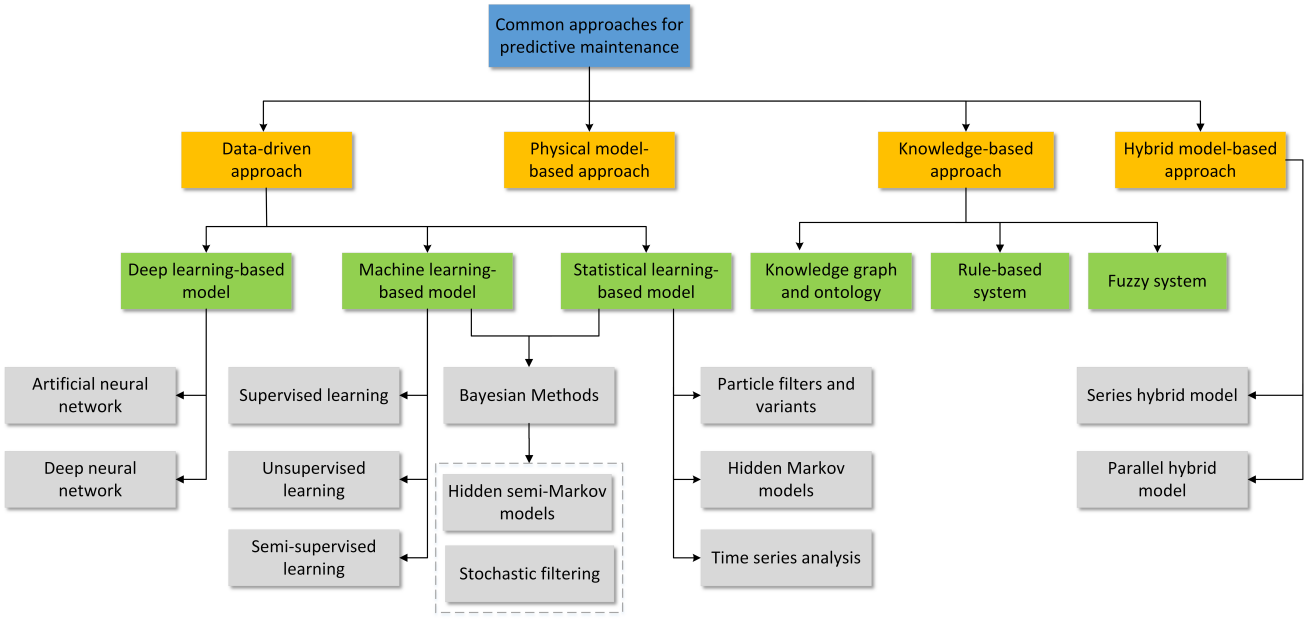


Figure 1: Classification of the common predictive maintenance approaches.

making. Because of the exponential growth of data volume and the rapid development of data acquisition technologies, data-driven methods have attracted wide attention regarding the predictive maintenance of industrial equipment [63, 2]. We further classify data-driven approaches into machine learning and deep learning methods.

Machine learning is known as the core technique in AI. Following the exponential increase of data generated within manufacturing processes, a great number of machine learning algorithm have been developed to solve real-world problems. These high-performance algorithms have significantly contributed to the digitalisation and automation of the manufacturing industry. The first set of algorithms are logistic regression (LR) [50, 38]. LR is a classification method with the lowest algorithm complexity. It is a supervised learning algorithm that applies to labelled data. Thus, when a large number of labelled features can be acquired, and model complexity is a crucial concern, LR can be considered to solve the problem [63].

The second set of machine learning algorithms are based on support vector machine (SVM) models. Normally, SVM models are used for binary classification. These type of models could achieve high classification accuracy for both linear and nonlinear problems [58]. Furthermore, a wide range of optimised algorithms is available in the literature to improve the computation performance of SVM [16]. To solve multi-class tasks, SVM models map low dimension features to hyperplanes for describing a diverse range of machinery faults and failures.

The third set of machine learning algorithms are decision tree (DT) and random forest (RF). DT models treat a complex decision making process by decomposing it into a set of simpler decisions. This decomposition process is done by recursively partitioning the covariate space into subspaces [63]. In this way, DT models facilitate the interpretation of classification results. RF is another tree-based classification method that consists of multiple DT classifiers. Within a RF classifier, each DT classifier is generated using a random vector sampled independently from the input vector, and each tree casts a unit vote for the most popular class to classify an input vector [48]. The main advantage of RF models is their ability to handle high dimensional data without feature selection. Because of the independent nature of the trained trees, the training speed is usually fast thus facilitating the implementation of RF models.

The traditional machine learning algorithms suffer from the lack of expressiveness and defect of dimensionality [28]. To solve this problem, deep learning techniques have been developed to extract structured information from data sets using layered machine learning algorithms. Over the last decades, the manufacturing industry has benefited from the dramatic advance and growth of deep learning methods. These deep learning approaches have significantly improved the productivity, efficiency, and reliability of manufacturing systems. In this subsection, we introduce the most classic and widely-used deep learning models: artificial neural network (ANN) and deep neural network (DNN).

ANN models are originally inspired by the biological neural networks that constitute animal brains. An ANN is essentially a computing system that consists of a large number of connected units or nodes. These nodes simple processors that are linked with many interconnections. Unlike the knowledge-based systems where a set of rules are prescribed by human experts or by machine documents, an ANN is able to learn underlying rules from a given collection of representative examples [63]. Because of this merit, ANN is extensively applied to deal with nonlinear and random data. This technique is suitable for large scale and complex systems.

DNN refers to a neural network with many hidden layers, and all the layers are connected to each other. When the number of units within each layer and the number of layers is significantly increased, DNN can better represent complex functions that classic ANN [41]. When enough labelled training data is given, DNN can help humans to establish mapping functions to ease operation. Compared to traditional data-driven methods, DNN is able to extract and represent interesting features in a more intelligent way. This characteristic of DNN has made it exceptionally useful in minimising defects in manual design features [35].

2.2. Physical model-based approaches

Physical models (also known as physics of failure or behavioural models) quantitatively characterise the behaviour of a failure mode using physical laws (e.g. from first principles) [56]. Normally, this type of models uses a mathematical representation of the physical behaviour of a machine degradation process to compute the Remaining Useful Life (RUL) of machinery. The mathematical representation reflects how the monitored system responds to stress from both macroscopic and microscopic levels. To obtain an accurate description of the system, it is of vital importance to identify one or several system diagnostic parameters that are specific to the predictive maintenance task. Compared to other types of models, physical models provide the most accurate and precise estimation of the RUL [56].

However, physical models require the behaviour of a system to be derivable from first principles. This may bring obstacles to its implementation when the understanding of failure mechanisms can only be partially obtained. Hence, physical model-based approaches are more likely to be used in isolated cases where failure/faults mechanisms are well understood and predictive maintenance systems are well developed.

2.3. Knowledge-based approaches

A *knowledge-based system* maintains a knowledge base that stores the symbols of a computational model in the form of statements about the domain and performs reasoning by manipulating these symbols [30]. This type of systems measure the similarity between a new observation and a databank of previously described situations and deduce appropriate decisions [56]. Knowledge-based approaches can be further classified into knowledge graphs and ontologies, rule-based systems, and fuzzy systems.

The term *knowledge graph* is often used as a synonym for *ontology* [23]. The concept of *ontology* originated in philosophy. It studies concepts that directly relate to being, in particular becoming, existence, reality, as well as the basic categories of being and their relations [57]. In computer science, an ontology is considered as "*an explicit specification of a conceptualization for a domain of interest* [32]." Within this definition, *specification* refers to an act of describing or identifying something precisely. This requires the concepts and relationships in ontologies to be clearly defined by using formal logic. *Conceptualisation* stands for an intentional semantic structure that encodes implicit knowledge constraining the structure of a piece of a domain [7, 14]. Normally, the conceptualisation within an ontology is formalised by a logic theory that is written in a certain language. Also, ontologies provide reasoning capabilities by which new knowledge can be inferred.

To facilitate predictive maintenance, an ontology called OntoProg is described in [45]. OntoProg is a generic ontology that incorporates international standards for rigorous conceptualisation. This ontology is designed for Prognostics Health Management (PHM) mechanical items of manufacturing systems. The main goal is for predicting the Remaining Useful Life (RUL) of mechanical components. To enable timely interventions of maintenance, SPARQL queries [49] are used to retrieve the stored knowledge about real-world activities. In [11], a domain ontology for smart condition monitoring is introduced. This ontology formalises domain knowledge relevant to condition monitoring for manufacturing processes. It is developed into three ontology modules: the *Manufacturing Module*, the *Context Module*, and the *Condition Monitoring Module*. The effectiveness and usability of the ontology are tested on a conditional maintenance task of bearings in rotating machinery. The domain ontology is further extended in the literature [12], where a domain ontology named Manufacturing Predictive Maintenance Ontology (MPMO) is developed. In their work, MPMO is used together with sequential pattern mining techniques to enable anomaly detection and prediction on production lines. The proposed ontology is tested on a real-world data set collected from a semi-conductor manufacturing process.

Rule-based systems are another type of knowledge-based approach that has been widely used in Industry 4.0. Normally, this type of systems encodes problem-solving knowledge of domain experts in terms of a set of situation-action (IF-THEN) rules [34]. In the literature, rule-based systems have been used for intelligent product design [61, 60, 59]. To facilitate the use of the knowledge sources scattered within different engineering domains, rule-based reasoning are used together with domain ontologies to enable intelligent retrieval of abstract solutions for product design. These rule-based systems also allow users to eliminate technical contradictions using inventive principles. Rule-based systems are also used for diagnostic tasks in railway systems [37, 24]. In [37], a rule-based diagnostic system is deployed in railway systems as well as power-generating turbines at Siemens. In their work, Semantic Diagnostic Rule Language (SDRL) rules are used to process signals from sensors installed in equipment by filtering, aggregating, and combining sequences of time-stamped measurements recorded by the sensors [37]. As another relevant work, a rule-based approach is proposed to predict impending failures and alarms of critical rail car components [40]. To derive useful and interesting rules for anomaly prediction, the authors use machine learning techniques to learn rules from historical sensor measurements automatically. This approach has been proved able to drive proactive inspections and repairs, reducing operational equipment failures [40].

Fuzzy systems are often used when a real-world situation

requires flexible decision making. In classic predicate logic that is used by classic knowledge-based systems, a statement is either true or false. Thus a data entity can only belong to one set and excluded from the remaining sets. However, it is not always feasible to follow this principle when real-world situations involve "vague" knowledge for decision making [56]. To overcome this challenge, fuzzy set theory [65] is used to enable the partition of data based on a variable's "degree of truth". In this way, traditional IF-THEN rules are fuzzified to fuzzy rules. The obtained fuzzy rules use membership functions to define how input data is mapped to particular fuzzy variables. Typical use cases of fuzzy systems include anomaly detection [20], failure classification [13], fault diagnosis [17], and supply chain management [10].

2.4. Hybrid model-based approaches

A hybrid model-based approach is a combination of physics-based and data-driven prognostics approaches that attempts to leverage the strengths from both categories [36]. In the literature, a hybrid model-based predictive maintenance task can be classified into series and parallel approaches.

For a series approach, a physical model is first used to establish prior knowledge about the monitored manufacturing process. On the other hand, data-driven methods behave like a state estimator to capture unmeasured process parameters. Within this process, data-driven methods serve as an online parameter estimation technique to continuously update model parameters when new data is available [36].

A parallel approach takes advantage of the strong computational capability of data-driven models to predict residuals that are not explained by first principle models [43]. Most of the literature work uses a fusion process to integrate the outputs of physical model-based and data-driven approaches.

Hybrid approaches have been used in the literature for predictive maintenance. In [62], a digital twin-driven hybrid approach is proposed to predict the performance degradation in CNC machine tools. However, this approach only applies data-driven methods for predict degradation, which lacks the incorporation of domain knowledge about system's failure mechanism (e.g. friction, tear, wear and crack growth). In [39], a semantic-enriched information model is developed to assess the state of shop floor by semantic reasoning. However, in the semantic information model, there lacks a mechanism for evaluating the quality of rules. In [42], a hybrid predictive maintenance approach is introduced. In their approach, model-based and digital twin data-driven methods are jointly used. Nevertheless, the hybrid approach does not involve a knowledge module which captures domain knowledge coming from experts, machine documents, and international standards. In [22], a hybrid intelligent model is developed to facilitate degradation process prediction of rotational machinery. In the model, only a selective neural network ensemble model is used. Since the neural network works as a black box and lacks semantic interoperability, a uniform knowledge representation framework is needed. For our KSPMI system, we aim to address the aforementioned challenges and limitations by combining data-driven and knowledge-driven approaches.

3. Theoretical foundations

In this section, we introduce the theoretical foundations and basic notions that are necessary for describing the functionalities of KSPMI. We start with the concepts relevant to chronicle mining. The goal of chronicle mining is to extract frequent sequential patterns to predict machinery failures and their time of occurrence.

3.1. Chronicle mining

Chronicle mining is a special type of sequential pattern mining approach that can derive the order and temporal information of events. The aim of this mining process is to extract sequential patterns that contain rich temporal information from data. We first give the foundations of chronicle mining. The following definitions and notions are aligned to [18, 54]. Let $\mathbb{R}$ denote the set of real numbers.

Definition 1 (Event). *Let $\mathbb{E}$ be a set of* event types*, and $\mathbb{T} \subseteq \mathbb{R}$ a* time domain*. $\mathbb{E}$ is assumed to be totally ordered, which is denoted by $\leq_{\mathbb{E}}$. An* event *is a pair (e, t) where $e \in \mathbb{E}$ is the type of the event and $t \in \mathbb{T}$ its time.*

A set of events may appear together in a certain order to form a *sequence*.

Definition 2 (Sequence). *A* sequence *is a pair $\langle SID, \langle (e_1, t_1), (e_2, t_2), \ldots, (e_n, t_n)\rangle\rangle$ where SID is the index of the sequence, and $\langle (e_1, t_1), (e_2, t_2), \ldots, (e_n, t_n)\rangle$ a finite sequence of events, such that for all $i, j \in [1, n]$ with $i < j$, $t_i < t_j$ or $(t_i = t_j \wedge e_i <_{\mathbb{E}} e_j)$.*

As mentioned before, a sequence may contain a set of events that appear in a certain order. When these events are time-stamped, it allows us to calculate the quantitative time intervals among different events. The computation of these quantitative time intervals is of paramount importance for predicting the time of occurrence of future failures. The time intervals together with their quantitative temporal values form the definition of *temporal constraints*.

Definition 3 (Temporal constraint). *A* temporal constraint *is a quadruplet (e_1, e_2, t^-, t^+), denoted as $e_1[t^-, t^+]e_2$, such that $e_1, e_2 \in \mathbb{E}$, $e_1 \leq_{\mathbb{E}} e_2$ and $t^-, t^+ \in \mathbb{T}$, $t^- \leq t^+$. A couple of events (e, t) and (e', t')* satisfies *a temporal constraint $e_1[t^-, t^+]e_2$ iff $e = e_1$, $e' = e_2$, and $t' - t \in [t^-, t^+]$.*

Within this definition, t^- and t^+ are two reals that stand for lower and upper boundaries of the associated time interval. Observe that $[t^-, t^+] = [-\infty, \infty]$ is allowed, which is equivalent to no temporal constraint on the associated events. We also define that $e_1[a, b]e_2 \subseteq e_1'[a', b']e_2'$ iff $[a, b] \subseteq [a', b']$, $e_1 = e_1'$, and $e_2 = e_2'$.

After introducing the definitions of *events* and *temporal constraints* within a data sequence, the concept of *chronicle* is given.

Definition 4 (Chronicle). *A* chronicle *is a pair* $\mathcal{C} = (\mathcal{E}, \mathcal{T})$ *such that:*

1. $\mathcal{E} = \langle e_1, \ldots, e_n \rangle$, *where* $e_i \in \mathbb{E}$ *and* $e_i \leq_{\mathbb{E}} e_{i+1}$, *for all* i;
2. $\mathcal{T} = \langle t_{ij} \rangle_{1 \leq i < j \leq n}$ *is a sequence of temporal constraints on* $\mathcal{E}$ *such that for all pairs (i, j) satisfying* $i < j$, t_{ij} *is denoting a temporal constraint of the form* $e_i[t_{ij}^-, t_{ij}^+]e_j$.

$\mathcal{E}$ *is called the* episode *of* $\mathcal{C}$.

Within a chronicle mining process, *chronicle support* is an important measure indicating the frequency of an extracted chronicle. We formalise the definition of *chronicle support* by the following definition [53].

Definition 5 (Chronicle support). *An* occurrence *of a chronicle* C *in a sequence* S *is a subsequence of* S *that satisfies all temporal constraints in* C. *The* support *of a chronicle* C *in a set* SD *of sequences is the number of sequences in* SD *in which* C *occurs.*

To illustrate these aforementioned definitions, we give an example of a chronicle. Assume we have a data sequence shown in Fig. 2. Within this data sequence, A, B, C are different types of events. Integers stand for the timestamps of these events. Temporal constraints are quadruplets that associated with events. In this example, the time interval between event A and C has lower bound of 3 and upper bound of 8. Hence, the temporal constraint is represented as $A[3, 8]C$.

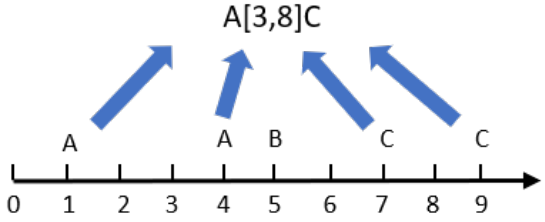


Figure 2: Example of a data sequence.

After applying the chronicle mining approach on this data sequence, the generation of temporal constraints is enabled. As a result, these events are represented as a graphical way, as shown in the lower part of Fig. 3. In the figure, events are represented by the circle-shaped nodes, and temporal constraints are displayed as the edges among these three events. The values associated to each edge are quantitative numerical boundaries of temporal constraints. For each temporal constraint, a lower and upper numerical boundary are extracted.

Within a predictive maintenance task, to enable the detection of machine anomalies, a special type of chronicles needs to be introduced. Within this type of chronicles, the last event stands for a machinery failure, e.g. event C in Fig. 3. In this context, event A and B are normal events that lead to a machinery failure. Based on this description, we introduce the concept of *failure chronicle*.

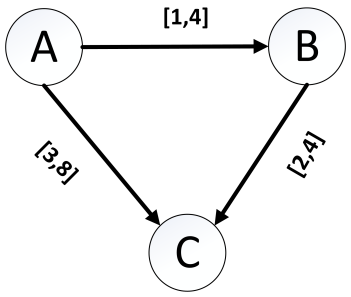


Figure 3: Example of a chronicle.

Definition 6 (Failure chronicle). *Let* $C_F = (\mathcal{E}, \mathcal{T})$ *be a chronicle. We say that* C_F *is a* failure chronicle *iff the last event in its episode is the failure event, that is, for* $\mathcal{E} = \langle e_1, \ldots, e_n \rangle$, e_n *is the failure event.*

For KSPMI, the output of the chronicle mining process is a set of frequent failure chronicles. These frequent failure chronicles contain information about the normal and abnormal events (failures) that are useful for predicting future events' occurrence.

3.2. Ontologies and SWRL rules

As KSPMI is a knowledge-based predictive maintenance system, it maintains a knowledge base that incorporates domain knowledge to operate. Normally, the domain knowledge is structured and represented by formal conceptual models, such as ontologies [31]. In computer science, an ontology is considered as "*an explicit specification of a conceptualization for a domain of interest*" [32]. According to [8], the formal definition of an ontology is given as follows:

Definition 7 (Ontology). *An ontology is a quadruplet* $O = \langle C, P, Sub, Applic \rangle$, *where:*

- C *is the set of classes used to describe the concepts of a domain of interest.*
- P *is the set of properties used to describe the individuals of the classes in* C.
- Sub *is the subsumption relation function defined as* $\text{Sub} : C \rightarrow 2^C$, *where* 2^C *denotes the power set of* C. *For a class* c *in* C, $\text{Sub}(c)$ *is the set of* c*'s directly subsumed classes. Class* c_1 *subsumes class* c_2 *iff* $\forall x \in c_2, x \in c_1$.
- Applic *is a function defined as* $\text{Applic} : C \rightarrow 2^P$. *This function associates each ontology class to those properties (object properties and data properties) that are applicable to each instance of this class.*

Apart from domain ontologies, KSPMI also uses SWRL rules to perform ontology reasoning. SWRL is an expressive OWL-based rule language that combines OWL DL and OWL Lite with Unary/Binary Datalog RuleML sublanguages [27]. SWRL rule-based reasoning can increase the amount

of knowledge encoded within an ontology. Amendable to World Wide Web Consortium (W3C) standards, SWRL is considered as the standard rule language of the Semantic Web [47]. Normally, SWRL rules are horn like IF-THEN-based logic statements expressed in terms of OWL concepts and reason on OWL individuals. For example, one rule that reasons on *hasParent* and *hasBrother* properties to imply *hasUncle* property can be written as

$$hasParent(?x1, ?x2) \wedge hasBrother(?x2, ?x3) \rightarrow hasUncle(?x1, ?x3),$$

where $?x1$, $?x2$, $?x3$ are named OWL classes.

In this work, the prediction of machine failures is enabled by combining domain ontologies and SWRL rules for ontology reasoning. The used SWRL rules are generated from the frequent failure chronicles that are obtained from the chronicle mining process. We name them as *chronicle rules* and propose the formal definition of it:

Definition 8 (Chronicle rule). *A chronicle rule is an implication $R: EC \wedge TEC \rightarrow TF$ such that:*

- *$EC = \langle ec_1, ..., ec_n \rangle$ is a sequence of non-failure events, where $n = |EC|$. The events in EC are ordered according to $\leq_{EC}$, that is, $ec_i \leq_{EC} ec_{i+1}$ for all i.*
- *$TEC = \langle t_{ij} \rangle_{1 \leq i < j \leq n}$ is a sequence of temporal constraints on EC. For all pairs (i, j) satisfying $i < j$, t_{ij} denotes a temporal constraint of the form $ec_i[t_{ij}^-, t_{ij}^+]ec_j$.*
- *Let $k = n+1$. $TF = \langle t_{ik} \rangle_{1 \leq i < k}$ is a sequence of temporal constraints between the non-failure events EC and a failure event F For all i satisfying $i < k$, t_{ik} denotes a temporal constraint of the form $ec_i[t_{ik}^-, t_{ik}^+]F$.*

During the prediction process, to improve the performance of the system, we enable the system to prune a chronicle rule base for selecting a subset of best-quality rules. In Section 4, we introduce the two quality measures for rule pruning. Furthermore, KSPMI is programmed with capability to detect rule quality issues such as *conflict*, *subsumption*, and *redundancy*. In Section 3.3, we give formal definitions to these three rule quality issues.

3.3. Rule quality issues

As chronicles rules are extracted from industrial data sets, the quality of these rules is highly dependant on the quality of data sets. Since the data sets are collected from a certain period of machine operation, they may not cover all real-world conditions. As a result, it is almost inevitable that the extracted chronicle rules may fail to predict failures that actually will happen. When faced with this situation, normally an intervention of domain experts is required to examine the current situation and propose appropriate decision. During the intervention process, a (human) experience is identified, adapted and improved for better system performance. In this work, we enable KSPMI to capitalise the experience of domain experts in the form of expert rules. The expert rules have a similar format with chronicle rules and able to predict failure occurrence when chronicle rules fail to propose an appropriate decision. In this way, the expert rules will complement the mined chronicle rule base. For future prediction, the expert rules will be integrated into the chronicle rule base and launched together to achieve better prediction performance.

For KSPMI, the expert rules are kept as a separate rule base against the mined chronicle rule base. Nevertheless, when expert rules are integrated into the mined chronicle rule base, a set of rule quality issues such as rule *redundancy*, *conflict* and *subsumption* may occur. Rule quality issues may cause logic inconsistency, thus reducing the reliability and performance of the rule-based reasoning process. To avoid possible rule quality issues and obtain a valid rule base, we propose a rule base verification step to examine potential issues during the integration process.

To address the above challenges, KSPMI implements a rule base integration module where the integrated rule base is verified and refined. During this process, *Redundancy*, *Conflict*, and *Subsumption* are considered as three important rule quality issues. Following Definition 8, the definition of rule *Redundancy*, *Conflict*, and *Subsumption* are formalised as follows [15]:

Definition 9 (Chronicle rule redundancy). *Two chronicle rules $R: EC \wedge TEC \rightarrow TF$ and $R': EC' \wedge TEC' \rightarrow TF'$ are* redundant *iff the two rules have the same sets of events, and the events have the same temporal constraints:*

$$\begin{aligned} &\text{ChroRedundancy}(R, R') \Longleftrightarrow \\ &(EC = EC') \wedge (TEC = TEC') \wedge (TF = TF') \ . \end{aligned} \tag{1}$$

Definition 10 (Chronicle rule conflict). *Two chronicle rules $R: EC \wedge TEC \rightarrow TF$ and $R': EC' \wedge TEC' \rightarrow TF'$ have a* conflict *iff the two rules have the same sets of non-failure events and the same temporal constraints for these non-failure events, but conflicting temporal constraints of a failure:*

$$\begin{aligned} &\text{ChroConflict}(R, R') \Longleftrightarrow \\ &(EC = EC') \wedge (TEC = TEC') \wedge (TF \neq TF') \ . \end{aligned} \tag{2}$$

Definition 11 (Chronicle rule subsumption). *A chronicle rule $R: EC \wedge TEC \rightarrow TF$* subsumes *another chronicle rule $R': EC' \wedge TEC' \rightarrow TF'$ iff they have the same temporal constraints of a failure, but rule R contains additional restrictions on the set of non-failure events or on the temporal constraints of these non-failure events:*

$$\begin{aligned} &\text{ChroSubsums}(R, R') \Longleftrightarrow \\ &(TF = TF') \wedge ((EC' \wedge TEC') \models (EC \wedge TEC)) \ . \end{aligned} \tag{3}$$

These definitions are used to detect issues with regard to rule base verification. When different rule bases are integrated, the aforementioned three rule quality issues are examined to refine the integrated rule base.

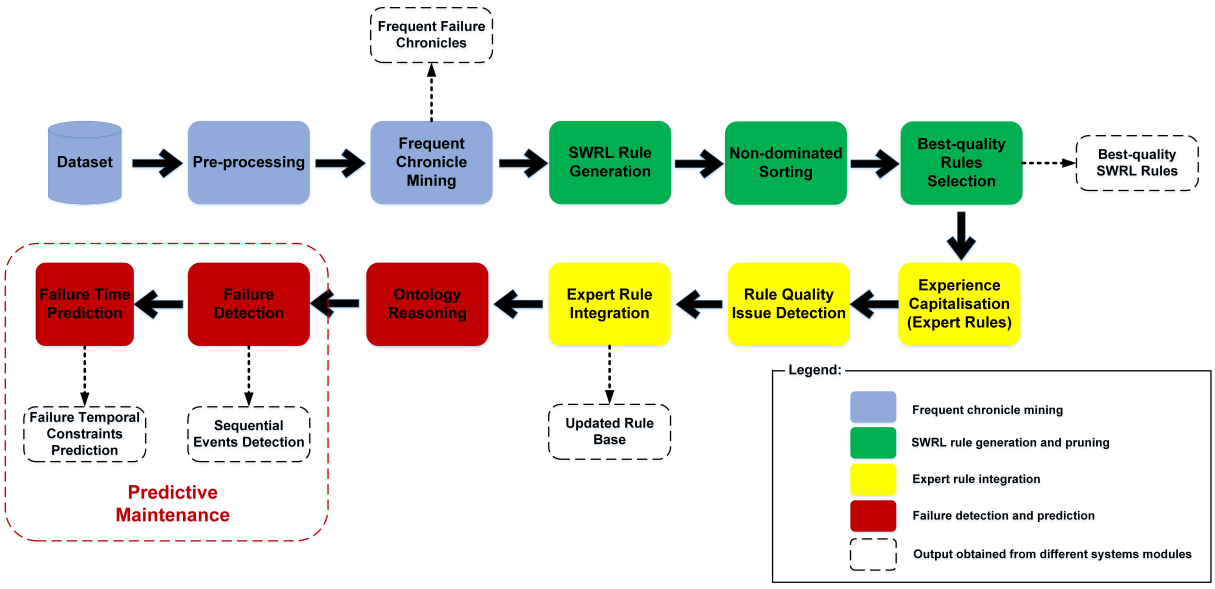


Figure 4: An overview of the proposed approach.

4. Proposed approach

This section gives the core approach based on which the KSPMI software is developed and programmed. As a hybrid predictive maintenance method that combines chronicle mining and semantic technologies, the proposed approach consists of two main phases. These two phases are further divided into four main processes: i) the frequent chronicle mining process; ii) the SWRL rule generation and pruning process; iii) the expert rule integration process; and iv) the failure detection and prediction process. Fig. 4 gives an overview of our approach. We describe these four processes as well as their main steps:

1. The frequent chronicle mining process. Chronicles are a type of sequential patterns that incorporates rich temporal information of different events for decision making. As introduced in Section 3.1, chronicles are essentially *event-time* pairs that are represented in a graph format where vertices are events and edges are labelled with intervals representing the time between the two linked events [55]. Frequent chronicle mining is a special type of sequential pattern mining approach that extracts all frequent chronicles from data. The extracted chronicles contain information about the events' occurrence order as well as their time of occurrence. In this paper, we use the Clasp-CPM algorithm introduced in [54] to extract a set of frequent failure chronicles from industrial data.
2. The SWRL rule generation and pruning process. Within this step, a set of SWRL predictive rules are generated from the extracted frequent failure chronicles. These rules are IF-THEN-based logic statements that enable rule-based reasoning. To enable rule generation, we use the SWRL rule generation algorithm that was developed in our previous work [12]. The SWRL rule generation algorithm allows automatic transformation of the extracted chronicles into a set of SWRL predictive rules. The pseudo codes of this algorithm is shown in Algorithm 1. The algorithm first extracts different types of events from a failure chronicle, and then discover the order of these events as well as their temporal constraints. At last, a rule is generated as an implication between the antecedent and the consequent. After rule generation, a rule pruning process is evoked to remove low-quality rules. The rule pruning process is enabled by a multi-objective optimisation process that applied to rule *accuracy* and *coverage* measures. The definition of rule accuracy and coverage can be found in [15]. Rule accuracy and coverage measures are computed based on the analysis of the relationship between a decision rule $\boldsymbol{R}$ and target class $\boldsymbol{F}$. The relationship is represented by a 2×2 contingency table which contains a cross-tabulation of categories of data and the frequency data appearance for cross-classification [6]. The contingency table is shown in Table. 1. In Table 1, n_{af} denotes the number of training examples for which both the antecedent and the consequent of the rule are true. $n_{a\bar{f}}$ denotes the number of training examples for which the antecedent of the rule is true, and the consequent of the rule is not true. On the other hand, $n_{\bar{a}f}$ denotes the number of training examples for which the antecedent is not true, and the consequent of the rule is true. $n_{\bar{a}\bar{f}}$ denotes the number of training examples for which neither the antecedent and the consequent of the rule is true. n_a, $n_{\bar{a}}$, n_f, $n_{\bar{f}}$ are marginal totals, and N is the total number of training examples.
Based on the information provided by the contingency table, accuracy and coverage can be derived. Among

Table 1
The contingency table for computing rule quality measures.

	F is true	F is not true	
A is true	n_{af}	$n_{a\bar{f}}$	n_a
A is not true	$n_{\bar{a}f}$	$n_{\bar{a}\bar{f}}$	$n_{\bar{a}}$
	n_f	$n_{\bar{f}}$	N

them, the rule accuracy is defined as

$$Accuracy(R) = n_{af}/n_a, \tag{4}$$

and rule coverage is defined as

$$Coverage(R) = n_{af}/n_f. \tag{5}$$

This multi-objective optimisation step uses the fast non-dominated sorting algorithm introduced in [46] to select a subset of best quality rules based on the two quality measures. The best quality rules have the highest accuracy and coverage measures among the whole rule base thus are sorted within the first Pareto Front. After rule pruning, the best-quality rules construct a new chronicle rule base.

3. The expert rule integration process. Within this process, the experience of domain experts is capitalised in the form of expert rules. We aim to complement the chronicle rule base (rules generated from the previous step) with a set of expert rules. By this, we address the incompleteness issue of the chronicle rule base. During this integration step, a set of rule quality issues such as *redundancy*, *conflict*, and *subsumption* are examined when heterogeneous rules are combined to achieve a satisfactory level of reasoning performance. During the rule integration process, the automatic detection of rule quality measures is enabled by a novel algorithm. This novel algorithm is adapted from our previous research [15]. The pseudo codes of it is shown in Algorithm 2. Within this algorithm, three functions extract different sets of atoms from an expert rule. After that, three steps are proposed to detect rule redundancy, subsumption, and conflict issues among the expert rule and the whole chronicle rule base. As a result, a refined rule based is obtained after integrating expert rules with the chronicle rule base.
4. The failure detection and prediction process. The detection and prediction of machine anomalies are achieved by ontology reasoning. KSPMI uses the Manufacturing Predictive Maintenance Ontology (MPMO) as the domain ontology together with generated SWRL rules to perform reasoning. The formalism and description of MPMO can be found in our recently published paper [12]. As a result, the occurrence of machinery failures is detected. Also, the temporal information of the failures is predicted. By this, KSPMI is able to detect and predict possible machinery failures that may happen in the future.

Algorithm 1 Algorithm to transform a chronicle into a predictive SWRL rule [12].

Require: $\mathcal{C}_F = (\mathcal{E}, \mathcal{T})$: A failure chronicle model within which the last event type is a failure event, $\mathcal{E}$ its episode, and $\mathcal{T}$ its temporal contraints.
Ensure: R: a predictive SWRL rule
1: $EL \leftarrow$ LastNonfailureEvent($\mathcal{C}_F$)
2: ▷ Extract the last non-failure events before the failure within a chronicle.
3: $R \leftarrow \emptyset$, $A \leftarrow \emptyset$, $C \leftarrow \emptyset$
4: **for each** $e_i[t_{ij}^-, t_{ij}^+]e_j \in \mathcal{T}$ **do**
5: $pe \leftarrow$ PrecedingEvent($e_i[t_{ij}^-, t_{ij}^+]e_j$)
6: ▷ Extract the preceding event of this time interval.
7: $se \leftarrow$ SubsequentEvent($e_i[t_{ij}^-, t_{ij}^+]e_j$)
8: ▷ Extract the subsequent event of this interval.
9: $Atom_a \leftarrow [t_{ij}^-, t_{ij}^+] \wedge pe \wedge se$
10: $A \leftarrow Atom_a \wedge A$
11: **end for each**
12: **for each** $el \in EL$ **do**
13: $ftc \leftarrow$ FailureTimeConstraint($el, \mathcal{C}_F$)
14: ▷ Extract the time constraint between the last event before the failure and the failure event.
15: $Atom_c \leftarrow el \wedge ftc$
16: $C \leftarrow Atom_c \wedge C$
17: **end for each**
18: $R \leftarrow (A \rightarrow C)$
19: ▷ A rule R is generated as an implication between the antecedent and consequent.
20: **return** R

KSPMI is developed into four system modules that correspond to these four processes. In the next section, we demonstrate the implementation of KSPMI on several real-world and synthetic data sets.

5. KSPMI: the knowledge-based predictive maintenance system

In Section 4, we introduced the four core modules of KSPMI. In this section, we first describe the development environment and programming tools for KSPMI[1][2]. Then we introduce the core functionalities of KSPMI by following the key steps introduced in Section 4: frequent chronicle mining, SWRL rule generation and pruning, expert rule integration, failure detection and prediction.

5.1. Software development environment

Several software and tools are used for the development of KSPMI. They are listed as follows:

[1]The source code and all used data sets for KSPMI can be found at: https://github.com/caoppg/KSPMI.git

[2]A demonstration video for KSPMI can be found at: https://sites.google.com/view/kspmiknowledge-basedsystem/home

Algorithm 2 Algorithm to detect the three issues when an expert rule is integrated into the chronicle rule base, adapted from [15].

Require: A chronicle rule base $\mathcal{R}$ which contains a set of failure chronicles, and an expert rule R_e which is in the form of a failure chronicle.
Ensure: $\mathcal{R}'$: the integrated rule base.
1: $\mathcal{R}' \leftarrow \mathcal{R}$.
2: **for each** $R \in \mathcal{R}$ **do**
3: **if** ChroRedundancy(R_e, R) **then**
4: ▷ Rule redundancy issue detected. A *ChroRedundancy* message is printed, no change in the rule base.
5: **end if**
6: **if** ChroSubsumes(R_e, R) **then**
7: ▷ Rule subsumption issue detected.
8: Remove(R, $\mathcal{R}'$)
9: ▷ Remove the subsumed chronicle rule from the rule base.
10: Integrate(R_e, $\mathcal{R}'$)
11: ▷ Integrate the expert rule into the rule base.
12: **end if**
13: **if** ChroConflict(R_e, R) **then**
14: ▷ Rule conflict issue detected.
15: Remove(R, $\mathcal{R}'$)
16: ▷ Remove the conflict chronicle rule from the rule base.
17: Integrate(R_e, $\mathcal{R}'$)
18: ▷ Integrate the expert rule into the rule base.
19: **end if**
20: **end for each**
21: **return** $\mathcal{R}'$

- Java Platform, Standard Edition (Java SE) [3]. Java SE is a computing platform for developing and deploying portable codes for desktop and server applications. This platform uses Java as programming language and belongs to the Java software-platform family.

- Eclipse[4]. It is an open source and integrated development environment (IDE) for computer programming. Eclipse maintains essential tools for Java developer, which includes a Java IDE, a Concurrent Version System (CVS) client, Git client, Maven integration and WindowBuilder. It also contains a base workspace and an extensible plug-in system for developers to customise the programming environment.

- Protégé[5]. Developed by Stanford University, Protégé is a free and open-source software for developing knowledge-based systems and editing ontologies. Since its release, Protégé has become the most popular software for developing and maintaining ontologies. When developing knowledge-based systems, Protégé can be transferred into a Java-based Application Programming Interface (API). During our development phase, three Protégé-based APIs and libraries have been used: OWL API[6], SWRL API[7], and SQWRL API[8].

- Drools[9]. It is a popular Java-based open source rule engine. Drools has various important features such as rules, rule flows, temporal modelling, decision tables, and complex event processing. These features support a wide range of reasoning tasks such as ontology reasoning, temporal reasoning, and imperfect reasoning.

- Sequential Pattern Mining Framework (SPMF)[10]. It is a Java-based open source data mining library, with specialisation in pattern mining. With current version v2.46, SPMF supports 202 data mining algorithms. SPMF can be used via a simple user-freindly interface, command line, or source codes. For KSPMI, we integrate the source codes of pattern mining algorithms into KSPMI.

In the next sections, we demonstrate the key features and functionalities of KSPMI. The main Graphical User Interfaces (GUI) of KSMPI will be presented alongside experimental results obtained from both real-world and synthetic data sets.

5.2. Experimentation on a real-world industrial data set

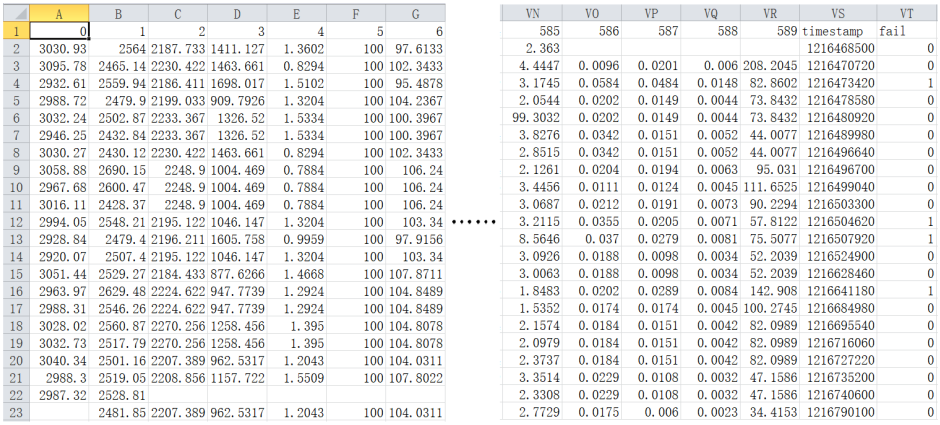
KSPMI takes sequential data sets as input. Our first experimentation is conducted on the UCI SECOM data set [44], which contains measurements of features of semi-conductor production within a semi-conductor manufacturing process.

5.2.1. The UCI SECOM data set

In the UCI SECOM data set, 1567 data records and 590 attributes are collected. The rows in the data set stand for test points. Every test point consists of 589 attributes, where each attribute is associated with its numerical value. The values in the second last column are the timestamps when these test points are recorded. Since the timestamps are in ascending order, the test points are considered as a sequence of events that are timely ordered. A binary label is given in the last column for each row, which indicates a pass (0) or fail (1) of the recorded test point. In total, 104 number of records represent the failures of production. The data is stored in a raw text file, within which each line represents an individual example of recording with its timestamp. The features are separated by spaces. In the input data set, events are associated with timestamps, and each test point is given a binary label (e.g. −1 corresponds to normal status, and 1 corresponds to a failed status). Fig. 5 shows an excerpt of the UCI SECOM data set as system input.

However, not all the data contained in the UCI SECOM data set is relevant to the failure prediction task. Indeed,

[3] https://www.oracle.com/java/technologies/javase-downloads.html
[4] https://www.eclipse.org/
[5] https://protege.stanford.edu/
[6] http://owlapi.sourceforge.net/
[7] https://github.com/protegeproject/swrlapi
[8] https://github.com/protegeproject/swrlapi/wiki/SQWRL
[9] https://www.drools.org/
[10] https://www.philippe-fournier-viger.com/spmf/



	A	B	C	D	E	F	G	VN	VO	VP	VQ	VR	VS	VT
1	0	1	2	3	4	5	6	585	586	587	588	589	timestamp	fail
2	3030.93	2564	2187.733	1411.127	1.3602	100	97.6133	2.363					1216468500	0
3	3095.78	2465.14	2230.422	1463.661	0.8294	100	102.3433	4.4447	0.0096	0.0201	0.006	208.2045	1216470720	0
4	2932.61	2559.94	2186.411	1698.017	1.5102	100	95.4878	3.1745	0.0584	0.0484	0.0148	82.8602	1216473420	1
5	2988.72	2479.9	2199.033	909.7926	1.3204	100	104.2367	2.0544	0.0202	0.0149	0.0044	73.8432	1216478580	0
6	3032.24	2502.87	2233.367	1326.52	1.5334	100	100.3967	99.3032	0.0202	0.0149	0.0044	73.8432	1216480920	0
7	2946.25	2432.84	2233.367	1326.52	1.5334	100	100.3967	3.8276	0.0342	0.0151	0.0052	44.0077	1216489980	0
8	3030.27	2430.12	2230.422	1463.661	0.8294	100	102.3433	2.8515	0.0342	0.0151	0.0052	44.0077	1216496640	0
9	3058.88	2690.15	2248.9	1004.469	0.7884	100	106.24	2.1261	0.0204	0.0194	0.0063	95.031	1216496700	0
10	2967.68	2600.47	2248.9	1004.469	0.7884	100	106.24	3.4456	0.0111	0.0124	0.0045	111.6525	1216499040	0
11	3016.11	2428.37	2248.9	1004.469	0.7884	100	106.24	3.0687	0.0212	0.0191	0.0073	90.2294	1216503300	0
12	2994.05	2548.21	2195.122	1046.147	1.3204	100	103.34	3.2115	0.0355	0.0205	0.0071	57.8122	1216504620	1
13	2928.84	2479.4	2196.211	1605.758	0.9959	100	97.9156	8.5646	0.037	0.0279	0.0081	75.5077	1216507920	1
14	2920.07	2507.4	2195.122	1046.147	1.3204	100	103.34	3.0926	0.0188	0.0098	0.0034	52.2039	1216524900	0
15	3051.44	2529.27	2184.433	877.6266	1.4668	100	107.8711	3.0063	0.0188	0.0098	0.0034	52.2039	1216628460	0
16	2963.97	2629.48	2224.622	947.7739	1.2924	100	104.8489	1.8483	0.0202	0.0289	0.0084	142.908	1216641180	1
17	2988.31	2546.26	2224.622	947.7739	1.2924	100	104.8489	1.5352	0.0174	0.0174	0.0045	100.2745	1216684980	0
18	3028.02	2560.87	2270.256	1258.456	1.395	100	104.8078	2.1574	0.0184	0.0151	0.0042	82.0989	1216695540	0
19	3032.73	2517.79	2270.256	1258.456	1.395	100	104.8078	2.0979	0.0184	0.0151	0.0042	82.0989	1216716060	0
20	3040.34	2501.16	2207.389	962.5317	1.2043	100	104.0311	2.3737	0.0184	0.0151	0.0042	82.0989	1216727220	0
21	2988.3	2519.05	2208.856	1157.722	1.5509	100	107.8022	3.3514	0.0229	0.0108	0.0032	47.1586	1216735200	0
22	2987.32	2528.81						2.3308	0.0229	0.0108	0.0032	47.1586	1216740600	0
23		2481.85	2207.389	962.5317	1.2043	100	104.0311	2.7729	0.0175	0.006	0.0023	34.4153	1216790100	0

Figure 5: An excerpt of an input data set for KSPMI.

some of the data is irrelevant and is considered noise. To deal with this challenge in high-dimensional data, feature selection is a feasible way to remove irrelevant and redundant data. This can improve learning accuracy, facilitate a better understanding of data, reduce data processing time and memory consumption [9]. In this work, the feature selection method introduced in [33] is used to select a relevant subset of the feature set for failure prediction.

After the feature selection step, data discretisation [51] is performed on the data set. The goal of data discretisation is to transform continuous variables, functions or data attribute values into a finite set of (discrete) intervals. Each interval is associated with a range of data values. For each variable, 20 bins are created to discretize continuous variables into discrete numeric intervals. We then use integers from 1 to 20 to represent these discrete numeric intervals. After that, data sequentialisation is implemented to transform the UCI SECOM data set into a sequential format. This will enable us to perform frequent chronicle mining on the data. To do this, the sequentialised data is structured in the form of pairs (event, timestamp). Within each data sequence, the last event represents a failure. This allows us to extract a set of frequent failure chronicles out of data. For a failure chronicle, all the events happen before the last one are considered "normal" events. Up to this step, we have finished the pre-processing of the UCI SECOM data set.

5.2.2. The frequent chronicle mining GUI

The execution of KSPMI starts with the frequent chronicle mining phase on input data. To show the functionalities of the software, we take the pre-processed UCI SECOM data set as input. After frequent chronicle mining, we obtain a set of frequent sequential patterns which contain the order information of events. The CloSpan algorithm also allows to filter less informative patterns and retain closed frequent sequential patterns that best describe the data set. Thereafter, we execute the Clasp-CPM algorithm to obtain a set of frequent failure chronicles that describe failures events and their temporal constraints. The frequent failure chronicles are mined from similar data sequences to the one shown in Fig. 2, where the last events are failure. For a failure chronicle, the last node also represents a failure (e.g. if the pattern shown in Fig. 3 is a failure chronicle, then the last event C represents a failure event). These frequent failure chronicles will be transformed into SWRL rules for the goal of failure prediction.

Fig. 6 shows the output of the frequent chronicle mining phase. On the top of the interface, a table containing frequent failure chronicles is displayed. Each row of the table corresponds to a frequent failure chronicle, and each chronicle is associated with three important measures: chronicle support, accuracy, and coverage. Among these three measures, chronicle support is calculated during the pattern mining process. By clicking on each table entry, the accuracy and coverage measures are displayed on the top right side. Also, a graph-based chronicle is shown on the lower part of the interface. Within each graph, nodes represent different events (non-failure events and failure events), and edges stand for time intervals among events. The integers associated with each node are the nominal attributes we obtain from discretisation. A set of integers (e.g. 4 5 7 11 13 19) together form the description for an event. An event with integer 0 indicates a failure, which is normally the last event within a chronicle.

A "Transform the extracted chronicles into SWRL rules" button is designed at the bottom of this interface to allow users to proceed. After clicking, the SWRL rule generation and pruning phase is enabled. The generation of SWRL rules is enabled by Algorithm 2, where each frequent failure chronicle is transformed into an IF-THEN-based logic rule. As a result, all the extracted chronicles are transformed, and SWRL rules are generated to form a rule base. We name it as a "chronicle rule base". After that, the generated rules will be pruned according to rule accuracy and coverage mea-

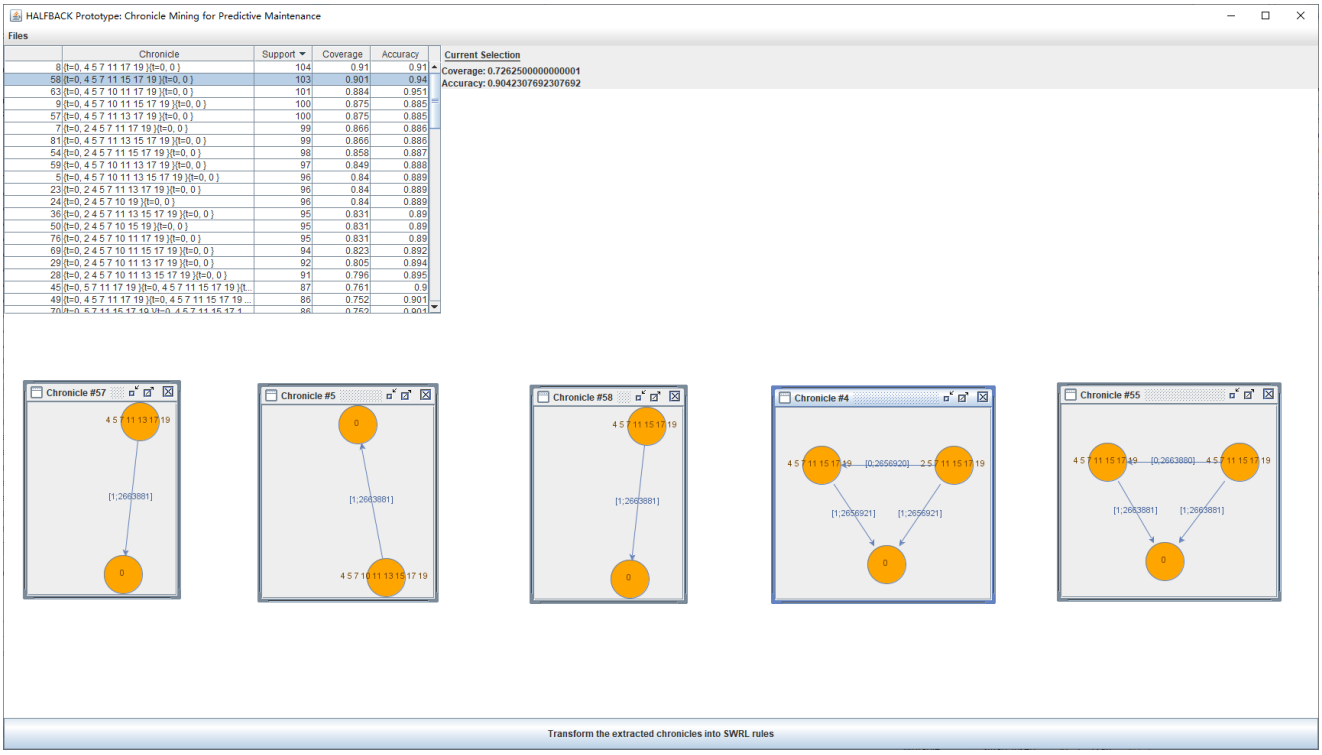


Figure 6: The chronicle mining GUI of KSPMI.

sures. We will introduce the rule pruning process in Section. 5.2.3.

5.2.3. The SWRL rules generation and rule pruning GUI

With obtaining a set of frequent failure chronicles, we aim to generate SWRL predictive rules for ontology reasoning. In the SWRL rules generation and rule pruning GUI, the extracted chronicles are firstly transformed into rules using Algorithm 2. Syntax of the generated rules are shown on the left side of the GUI, as shown in Fig. 7. Each generated rule is a horn-like logic statement where antecedent (body) implies consequent (head). As the rules are transformed from chronicles, the antecedent of a rule describes normal events as well as their temporal constraints. The consequent of a rule contains information about the temporal constraints between normal events and the failure event.

Normally, the number of chronicle rules transformed from chronicles is large. However, due to a certain degree of imprecision in real-world data, some of the extracted chronicle rules may suffer from low quality. Also, overfitting is a problem that often arises in rule-based classification and prediction problems [4]. In this context, these low-quality and overfitting rules may reduce the accuracy and efficiency of KSPMI. Therefore, we design a rule pruning software module to prune the chronicle rule base and select a subset of high-quality rules for failure prediction.

In this context, KSPMI implements a rule pruning process applied to the rule base obtained from frequent chronicle mining. As introduced in Section 4, two measures are

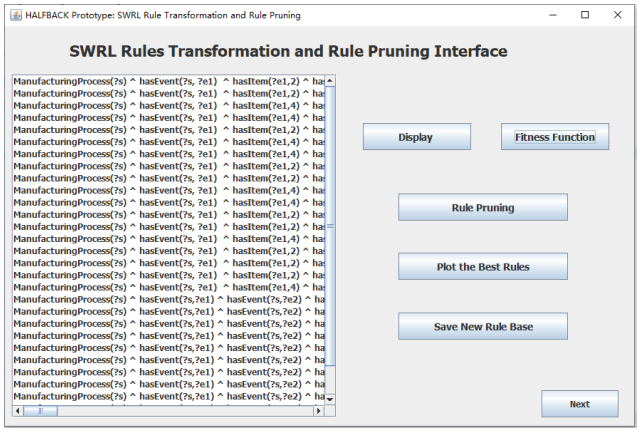


Figure 7: The SWRL rules generation and rule pruning GUI.

computed to evaluate rule quality. We identify accuracy and coverage as the two rule measures for evaluating the quality of SWRL rules, as these two measures are two basic characteristics of a rule and indicate a rule's reliability [5].

We then compute the average values of these two quality measures under different chronicle support. Table 2 shows the results. From the results, we see that as chronicle support increases, the accuracy of rules also increases. This is because as the support of extracted chronicles increases, more relevant chronicles are extracted. However, the number of extracted chronicles decreases. As a result, there are fewer SWRL rules generated. As the number of SWRL rules decreases, fewer data sequences are covered by the rules. This

is the reason why the average values of rule coverage show a downtrend in the table.

Table 2
Two rule quality measures under different chronicle support.

Chronicle support	Accuracy	Coverage
0.5	76.52%	74.26%
0.6	74.14%	75.71%
0.7	76.98%	74.35%
0.8	79.33%	70.49%
0.9	82.56%	68.10%
1.0	84.45%	67.71%

The SWRL rules transformation and rule pruning GUI allows users to visualise the transformed rules and to select the best-quality rules by using a multi-objective optimisation approach. We use the fast non-dominated sorting approach introduced in [46] to obtain the ranking of Pareto dominance for chronicle rules. After that, the rules within the first front are selected to perform rule-based reasoning. In this way, we only keep a subset of chronicle rules that have the best quality. These best-quality rules are afterwards launched for failure prediction.

The fitness function of the multi-objective optimisation approach can be shown by clicking the "Fitness Function" button. For KSPMI, we use the product of rule accuracy and coverage as fitness function. It is introduced by Equation 6:

$$Fitness(R) = Accuracy(R) \times Coverage(R). \quad (6)$$

The fitness value reflects the quality of a rule. The higher this value is, the better quality of the SWRL rule is.

The button "Rule Pruning" enables the pruning of the rule base by using the fast non-dominated sorting algorithm on the two rule quality measures. To visualise the results, a subset of best-quality rules are displayed on the left side, and the objective function values of them can be plotted by clicking the button "Plot the Best Rules". For UCI SECOM dataset, the rule pruning results are shown on the right side of Fig. 8. We select those rules within the first Pareto Front and consider them as best quality. After rule pruning, we use the best-quality rules to construct a new chronicle rule base.

5.2.4. The expert rule integration GUI

After obtaining a set of best-quality rules, users can proceed to the expert rule integration phase, where expert rules are used to complement the chronicle rule base. The goal of this step is to improve the overall fitness of the whole rule base, for achieving a better performance regarding failure prediction.

Fig. 9 shows the expert rule integration GUI of KSPMI. The used expert rules are retained as a separate rule base against the chronicle rule base. Users can open the expert rule file and push the stored expert rules into the system. At the lower part of this GUI, potential rule quality issues are detected and then printed. Fig. 10 shows an example of an expert rule file. This rule file can be opened by clicking the button "Open Expert Rules". The expert rules are the same format as the chronicle rules, with differences in rule atoms within either antecedent or consequent part.

The expert rules can be integrated into the chronicle rule base by clicking the "Input Expert Rule" button. After clicking this button, the system tries to push the input expert rule into the chronicle rule base. If there is no rule quality issue detected, the expert rule is directed integrated into the chronicle rule base. A "No rule quality issue detected" message is printed in the text area of GUI. If any quality issue detected, appropriate actions are automatically performed according to the decision making process in Algorithm 2. Within this example shown in Fig. 9, a conflict issue is detected among the integrated expert rule and chronicle rule base. The system then automatically removes the conflict chronicle rule and add the expert rule into the rule base. As we assign fitness values of expert rules as 1, the quality of expert rules is considered higher than chronicle rules. This is the reason that KSPMI always keeps the expert rule instead of the chronicle rule when a conflict issue is detected.

5.2.5. The failure detection and prediction GUI

After the expert rule integration process, we proceed to the failure prediction step. To enable failure prediction, we use Drools rule engine to perform ontology reasoning on the individuals that are populated in the MPMO ontology (this ontology is introduced in our previous paper [12]). Within this step, ontology reasoning is performed on the object and data properties of the individuals in MPMO.

After ontology reasoning, a SQWRL query is created to retrieve the prediction results. SQWRL is built on the SWRL rule language that treats a SWRL rule antecedent as a pattern specification for a query. In the consequent part, a SQWRL query replaces a rule consequent with a retrieval specification. By matching the individuals within the ontology, the proposed SQWRL aims to retrieve the temporal information of failures.

Fig. 11 shows the first round of prediction results. After frequent chronicle mining, SWRL rule generation and pruning, 5 expert rules are taken as input for integration. After detecting potential quality issues, the updated rule base consists of 6 rules. Then the proposed SQWRL query is executed. As a result, 13 rows of prediction results are returned. They are shown in the table at the bottom part of the failure detection and prediction GUI. In the bottom table, the left two columns are the individuals populated in the MPMO ontology. The right two columns gives the minimum and maximum numeric time values from an event to a future failure. The computation of these two numeric time values are performed by the Clasp-CPM algorithm. These two columns together with the event pair gives the temporal constraints of failures.

After the first round of prediction, we continue integrating more expert rules into the rule base. A larger rule base is used for the second round prediction. For the second round, the updated rule base consists of 9 rules, as shown in Fig. 12.

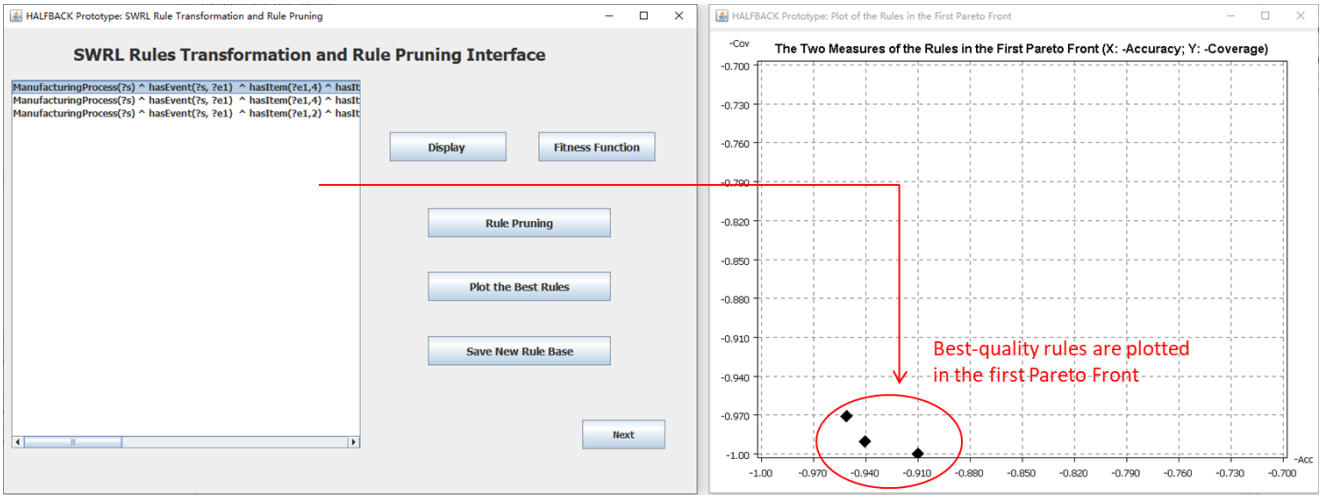


Figure 8: The rule pruning results. The best-quality rules are plotted within the first Pareto Front.

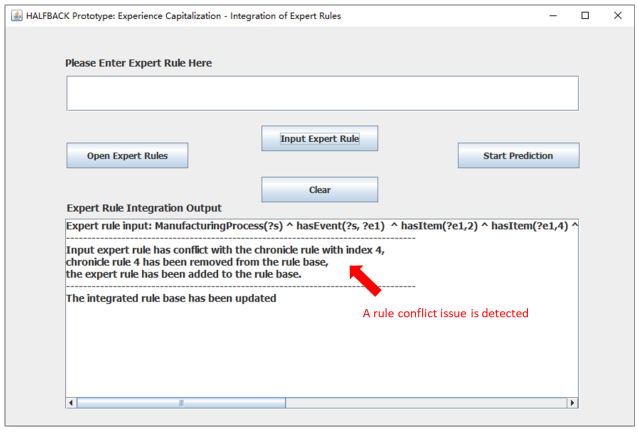


Figure 9: The expert rule integration GUI.

```
Expert rule 1:
ManufacturingProcess(?s) ^ hasEvent(?s, ?e1)  ^ hasItem(?e1,2) ^ hasItem(?e1,4) ^ hasItem(?e1,5) ^ hasItem(?e1,7) ^
hasItem(?e1,11) ^ hasItem(?e1,17) ^ hasItem(?e1,19)-> hasMinF(?e1, 1) ^ hasMaxF(?e1, 2656921)

Expert rule 2:
ManufacturingProcess(?s) ^ hasEvent(?s, ?e1)  ^ hasItem(?e1,2) ^ hasItem(?e1,4) ^ hasItem(?e1,5) ^ hasItem(?e1,7) ^
hasItem(?e1,11) ^ hasItem(?e1,17) ^ hasItem(?e1,19)-> hasMinF(?e1, 1) ^ hasMaxF(?e1, 2656921)

Expert rule 3:
ManufacturingProcess(?s) ^ hasEvent(?s, ?e1)  ^ hasItem(?e1,4) ^ hasItem(?e1,5) ^ hasItem(?e1,7) ^ hasItem(?e1,11) ^
hasItem(?e1,17) ^ hasItem(?e1,19)-> hasMinF(?e1, 3) ^ hasMaxF(?e1, 1567841)

Expert rule 4:
ManufacturingProcess(?s) ^ hasEvent(?s, ?e1)  ^ hasItem(?e1,2) ^ hasItem(?e1,4) ^ hasItem(?e1,5) ^ hasItem(?e1,7)
-> hasMinF(?e1, 3) ^ hasMaxF(?e1, 1567841)

Expert rule 5:
ManufacturingProcess(?s) ^ hasEvent(?s, ?e1)  ^ hasItem(?e1,2) ^ hasItem(?e1,4) ^ hasItem(?e1,5) ^ hasItem(?e1,7) ^
hasItem(?e1,11) ^ hasItem(?e1,17) ^ hasItem(?e1,19)-> hasMinF(?e1, 1) ^ hasMaxF(?e1, 2656921)

Expert rule 6:
ManufacturingProcess(?s) ^ hasEvent(?s, ?e1)  ^ hasItem(?e1,2) ^ hasItem(?e1,8) ^ hasItem(?e1,9) -> hasMinF(?e1, 115) ^
hasMaxF(?e1, 784613)

Expert rule 7:
ManufacturingProcess(?s) ^ hasEvent(?s, ?e1)  ^ hasItem(?e1,2) ^ hasItem(?e1,4) ^ hasItem(?e1,5) ^ hasItem(?e1,7) ^
hasItem(?e1,8) -> hasMinF(?e1, 525) ^ hasMaxF(?e1, 254651)

Expert rule 8:
ManufacturingProcess(?s) ^ hasEvent(?s, ?e1)  ^ hasItem(?e1,2) ^ hasItem(?e1,4) ^ hasItem(?e1,6) ^ hasItem(?e1,7) ^
hasItem(?e1,8) -> hasMinF(?e1, 310) ^ hasMaxF(?e1, 421974)
```

Figure 10: An example expert rule base for the UCI SECOM data set.

Then we use this new rule base to perform ontology reasoning. This time we obtain 39 rows of results indicating the temporal information of 39 failures from the SQWRL query. In this way, the rule base is progressively updated with more expert rules integrated. As a result, the overall coverage of the rule bases increases. This enables KSPMI to detect and predict more potential failures and their time of occurrence.

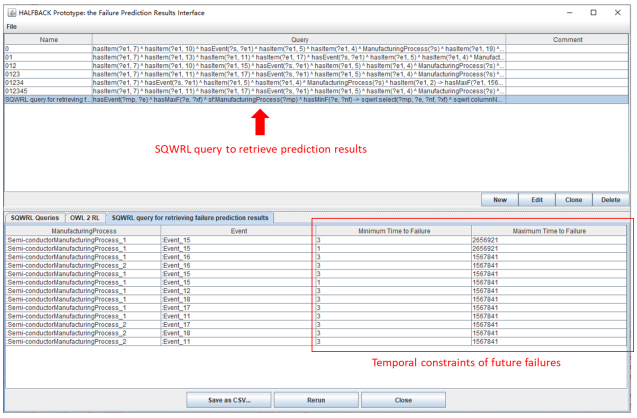


Figure 11: The first round of failure detection and prediction. 6 rules are used for failure prediction, SQWRL query returns 13 rows of prediction results.

5.2.6. Evaluation of rule quality

We then conduct another experimentation to simulate the expert rule integration process and to evaluate the overall quality of the rule base. To do this, the UCI SECOM data set is separated into a training set (80%) and a test set (20%). We apply frequent chronicle mining on the training set to extract chronicles and rules. The rules are then applied to the test set for failure prediction. Within the training set, a subset of chronicle rules that have the highest fitness values are treated as expert rules. The reason we do this is that these high-fitness chronicle rules have a higher chance to validate the data sequences in the test set. Thus they are considered as expert rules. The remaining rules are regarded as chronicle rules.

To obtain expert rules, the fast non-dominated sorting algorithm [46] is used to select a subset of best-quality chronicle rules. The rules that fall into the first Pareto Front are considered as expert rules. Since these expert rules Pareto-dominate all the rest, they are more reliable than other rules.

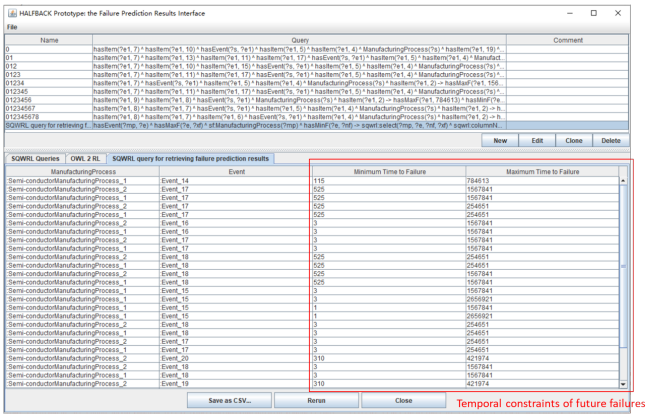


Figure 12: The second round of failure detection and prediction. 9 rules are used for failure prediction, SQWRL query returns 39 rows of prediction results.

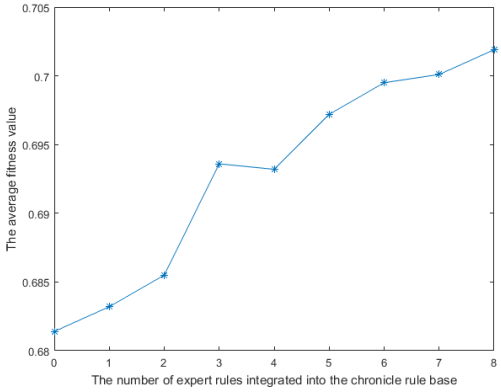


Figure 13: The average fitness values of the rules in the integrated rule base with different number of expert rules.

All the rules in the other Pareto Fronts construct a chronicle rule base.

During the expert rule integration process, Algorithm 2 is used to detect redundancy, conflict, and subsumption issues. If no issue is detected, the expert rule is directly integrated. If any issue is detected, Algorithm 2 is executed to take appropriate actions for rule base update. After that, the average fitness of the updated rule base is computed. With each expert rule integration, the updated rule base is launched for failure prediction against the test set.

Fig. 13 shows the obtained average fitness values with different number of expert rules as input. X axis is the number of expert rules, and Y axis stand for the average fitness value of the whole rule base. For the ease of visualization, we choose chronicle support=0.6 and show the change of fitness values. In general, as more expert rules are integrated into the chronicle rule base, the average fitness increases. The reason is that as more expert rules are integrated, the data sequences in the test set have more chances to be validated by the integrated rule base. This leads to an increase in the fitness value.

However, one exception happens when the number of expert rules = 4. For this exception, the new expert rule fails to validate enough data sequences in the test set. This is because the antecedent part of the expert rule only cover a few data sequence in the test set, or the actual failure occurrence time falls outside of the predicted temporal constraints. These two reasons both indicate unsuccessful failure prediction. As a result, the fitness value of this expert rule lower than the average value of the whole rule base. This leads to a decrease of the average fitness value.

5.2.7. Results comparison

We then validate our approach by comparing the proposed hybrid predictive maintenance method with existing similar methods in the literature. KSPMI is evaluated against the results reported in a similar contribution [12]. In [12], a predictive maintenance system was proposed but without the rule pruning and expert rule integration processes. In this context, we aim to evaluate how the pruning of SWRL rules and the integration of expert rules can improve system performance. To do this, three performance measurements are selected: the *True Positive Rate (TPR)*, the *Precision* of failure prediction, and the *F-measure*. The definitions of these three measurements can be found in [12]. Table 3 shows the comparison results. Within the table, CS stands for Chronicle Support. The average values of the three measures are computed. During the comparison, the number of integrated expert rules is 8, which is the same as the experimentation conducted for Fig. 13. Within this experimentation, the chronicle rules are obtained from the training set (80%), and the performance is evaluated on the test set (20%).

Table 3
Comparison between KSPMI and a hybrid predictive maintenance system introduced in [12].

System	CS	TPR	Precision	F-measure
[12]	0.5	88.28%	88.28%	86.44%
	0.6	90.38%	88.51%	87.26%
	0.7	89.44%	85.28%	86.55%
	0.8	86.29%	84.44%	85.81%
	0.9	84.21%	86.12%	85.11%
	1.0	82.24%	83.79%	85.35%
KSPMI	0.5	89.82%	89.76%	87.36%
	0.6	90.40%	89.23%	87.81%
	0.7	89.94%	86.54%	88.19%
	0.8	87.35%	86.20%	86.68%
	0.9	86.98%	85.38%	84.45%
	1.0	86.98%	85.38%	84.45%

From the results in Table 3, we observe that KSPMI obtains higher TPR and Precision values than our previous work in [12]. This is because the integrated expert rules hold fitness values of 1, which means their quality is higher than the mined chronicle rules. As a result, the expert rule integration process improves the overall quality of the rule base in KSPMI, thus leading to higher performance than the ap-

Figure 14: Frequent chronicle mining on one synthetic data set. Chronicle support = 0.73.

proach in [12], where no expert rule integration step was implemented.

For KSPMI, when chronicle support is increased to 0.9 and 1, the TPR and Precision values remain the same. The reason for this is that the number of best-quality rules (rules within the first Pareto front) obtained after rule pruning stay unchanged. Therefore, after expert rule integration, KSPMI maintains the same rule base under chronicle support= 0.9 and chronicle support= 1. This is the reason that the TPR and Precision remain the same under these two chronicle support values.

5.3. Experimentation on synthetic data sets

The performance of KSPMI is also evaluated on several synthetic data sets[11]. These synthetic data sets are generated following the same format as the one shown in Fig. 5. Within these synthetic data sets, each recording (each row in the data sets) is also timestamped and a label is given to indicate the non-failure/failure event. To apply the Clasp-CPM algorithm for chronicle mining, we tune several parameters to generate different size of synthetic data. These parameters include the number of data sequences in the data set, the size of a single data sequence, the maximum number of items within each event, and the maximum number of items within a singe data sequence. Each data sequence in the synthetic data sets is also associated with a generated timestamp.

Same as the experimentation performed on the UCI SECOM data set, the first step in this experimentation is to apply frequent chronicle mining one synthetic data sets. Fig. 14 shows a screenshot of the frequent chronicle mining GUI we obtained from on synthetic data set. Within this data set, the number of data sequences is 1000, and the chronicle support is set to a randomly-selected number as 0.73 for the Clasp-CPM algorithm. As a result, the maximum frequency we obtained from chronicles is 893, which indicates that this chronicle appeared in 893 data sequences within the whole data set. There are in total 20 chronicles extracted from this data set. For this case study, the integer 0 in chronicles also stands for a failure event.

[11] All used data sets can be found at: https://sites.google.com/view/kspmiknowledge-basedsystem/home

After frequent chronicle mining, we proceed to SWRL rule generation and pruning. After clicking the "Transform the extracted chronicles into SWRL rules" button at the bottom, the failure chronicles are transformed into predictive SWRL rules. 20 SWRL rules are generated after this step. Fig. 15 shows a screenshot of the generated rules.

We then activate the SWRL rule pruning process. The fast non-dominated sorting algorithm [46] is triggered to remove low-quality rules and select best-quality ones for ontology reasoning. Fig. 18 shows the rule pruning result. For the tested synthetic data set, there is only one SWRL rule that dominates the rest. Thus only this non-dominated rule is plotted on the right side of Fig. 18.

After that, the simulated expert rules are integrated into the chronicle rule base. Based on the rule quality definitions given in Section 3.3 and the rule quality issue detection Algorithm 2, KSPMI progressively improve the overall quality of the rule base. Fig. 16 presents the expert rule base for this synthetic data set.

In the end, after pushing all 8 expert rules into KSPMI, the integrated rule base is launched for failure prediction. For this experimentation, the integrated rule base consists of 5 SWRL rules. After launching the SQWRL query, we obtain 28 rows of results. Fig. 17 shows the prediction results for this synthetic data set. Similar to the experimentation conducted on the UCI SECOM data set, the two table columns on the right side together with the events pairs (events-failures) give the temporal constraints for failure occurrence. Up to this step, we have accomplished the failure prediction task for this synthetic data set.

6. Conclusions

This paper introduces a knowledge-based predictive maintenance system for Industry 4.0, named KSPMI. KSPMI enables an automatic predictive maintenance process by leveraging both statistical AI and symbolic AI methods. The execution of KSPMI starts with the chronicle mining process on industrial data sets, after which a set of failure chronicles are extracted. Then, a novel hybrid approach is proposed to transform the extracted chronicles into predictive SWRL rules. After that, rule base pruning and integration approaches are proposed to update the rule base automatically and progressively. Simulated expert rules are taken as input to the rule base integration process. At last, the integrated rule base is used together with domain ontologies to perform ontology reasoning for the goal of failure prediction. KSPMI has been tested and validated on both real-world and synthetic data sets.

The contributions proposed in this paper may induce potential future research. We describe them as follows:

- The first future work is the consideration of more rule quality measures. We aim to involve more rule quality measures for rule pruning, such as *Agreement*, *Information*, and *Logical Sufficiency* [6]. Among them, *Agreement* evaluates the association between the elements of a contingency table with its main diagonal.

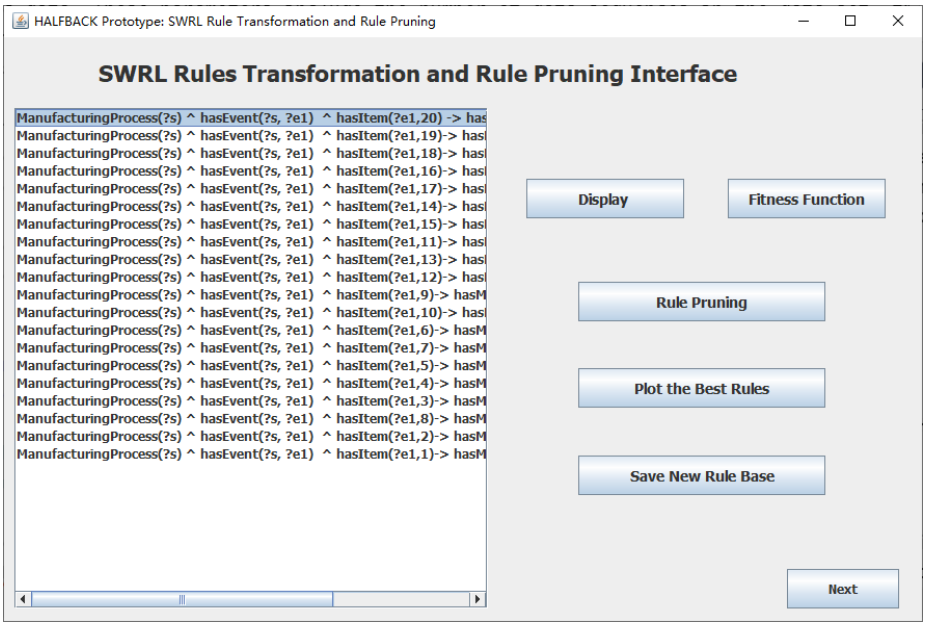


Figure 15: 20 SWRL rules are generated from the chronicles shown in Fig. 14.

```
Expert rule 1:
ManufacturingProcess(?s) ^ hasEvent(?s, ?e1)  ^ hasItem(?e1,18)-> hasMinF(?e1, 1) ^ hasMaxF(?e1, 20)

Expert rule 2:
ManufacturingProcess(?s) ^ hasEvent(?s, ?e1)  ^ hasItem(?e1,18)-> hasMinF(?e1, 5) ^ hasMaxF(?e1, 25)

Expert rule 3:
ManufacturingProcess(?s) ^ hasEvent(?s, ?e1)  ^ hasItem(?e1,18) ^ hasItem(?e1,16)-> hasMinF(?e1, 1) ^ hasMaxF(?e1, 20)

Expert rule 4:
ManufacturingProcess(?s) ^ hasEvent(?s, ?e1)  ^ hasItem(?e1,18)-> hasMinF(?e1, 3) ^ hasMaxF(?e1, 34)

Expert rule 5:
ManufacturingProcess(?s) ^ hasEvent(?s, ?e1)  ^ hasItem(?e1,18)-> hasMinF(?e1, 2) ^ hasMaxF(?e1, 58)

Expert rule 6:
ManufacturingProcess(?s) ^ hasEvent(?s, ?e1)  ^ hasItem(?e1,18)-> hasMinF(?e1, 8) ^ hasMaxF(?e1, 25)

Expert rule 7:
ManufacturingProcess(?s) ^ hasEvent(?s, ?e1)  ^ hasItem(?e1,20)-> hasMinF(?e1, 15) ^ hasMaxF(?e1, 142)

Expert rule 8:
ManufacturingProcess(?s) ^ hasEvent(?s, ?e1)  ^ hasItem(?e1,12)-> hasMinF(?e1, 22) ^ hasMaxF(?e1, 87)
```

Figure 16: An example expert rule base for the synthetic data set.

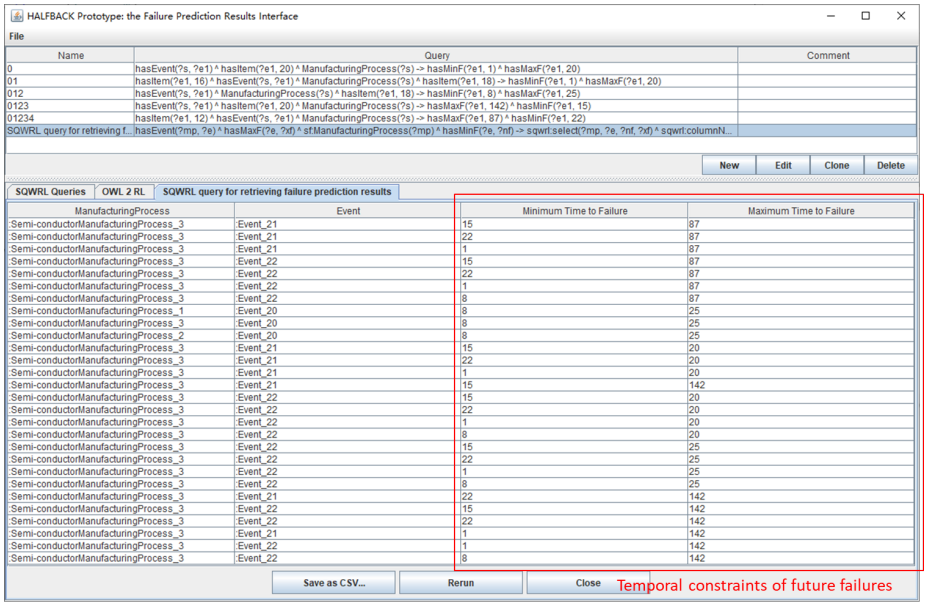


Figure 17: Failure prediction GUI for the synthetic data set. 5 rules are launched for failure prediction, SQWRL query returns 28 rows of prediction results.

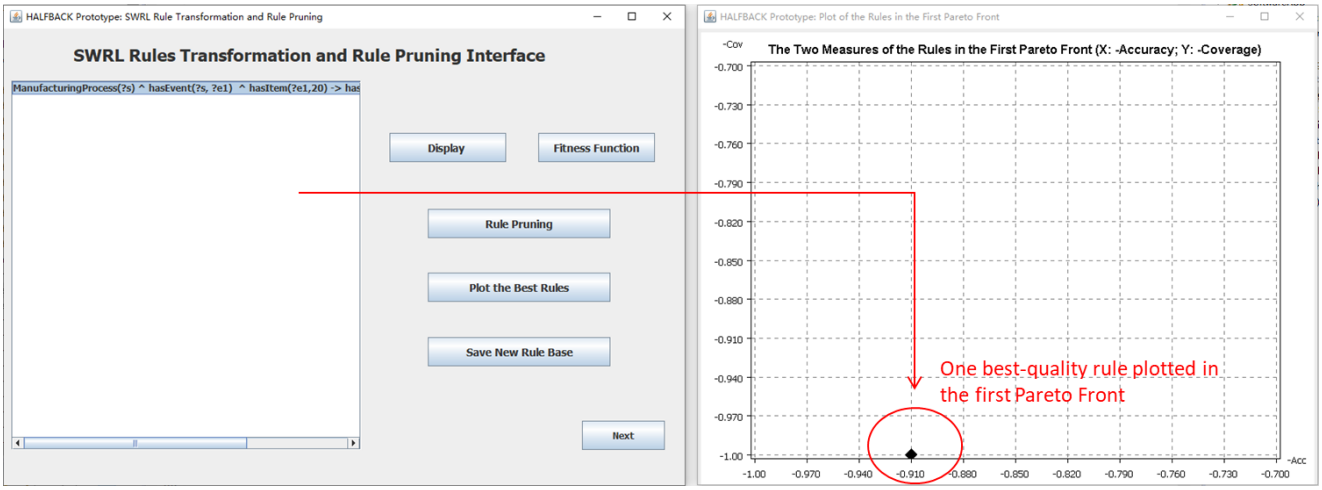


Figure 18: Rule pruning result for one synthetic data set.

Information measures the amount of entities a R needs to encode for correctly classifying an instance into a certain class. An information score can be derived as $Q_{IS} = -log\frac{N_a}{N} + log\frac{N_{af}}{N_a}$, where the values of N_a, N_{af}, N are computed from Table. 1. *Logical Sufficiency* is a standard likelihood ratio statistics that evaluates whether a rule R is encouraging for the classification of class C. The larger this value is, the more feasible R is for classifying C.

- The second future work is to enable the evaluation of rule quality for expert rules. For KSPMI, the proposed approach is based on the hypothesis that the expert rules have a fitness value of 1. Because of this, expert rules are considered more reliable than chronicle rules. However, due to the complex and uncertain characteristics of the manufacturing industry, experts are not always reliable. Once erroneous or fake expert rules are provided, KSPMI may fail to predict failures that are certain to happen in the future. To address this issue, the rule pruning step should be extended to cover the whole integrated rule base. In this way, a subset of best-quality rules are selected from the integrated rule base instead of merely from the chronicle rule base.

- The third future work is the real-time processing of data. Since KSPMI uses classic ontology reasoning for failure prediction, it is not able to make decisions on the fly. Indeed, the manufacturing domain is quite knowledge-intensive and dynamic that requires real-time decision making. To cope with this challenge, stream reasoning [21] will be utilised to replace the current reasoning techniques used in KSPMI. Stream reasoning is a logical reasoning approach that combines Complex Event Processing (CEP) and Semantic Web technologies. In the future, we aim to utilise advanced stream reasoners to continuously query and reason on dynamic industrial data.

CRediT authorship contribution statement

Qiushi Cao: Conceptualization, Formal analysis, Methodology, Software, Writing - original draft, Writing - review & editing. **Cecilia Zanni-Merk**: Conceptualization, Methodology, Validation, Supervision, Project administration, Funding acquisition. **Ahmed Samet**: Conceptualization, Methodology, Software, Validation, Supervision. **Christoph Reich**: Validation, Supervision, Project administration, Funding acquisition. **François de Bertrand de Beuvron**: Formal analysis, Validation. **Arnold Beckmann**: Project administration, Funding acquisition, Validation, Writing - review & editing. **Cinzia Giannetti**: Validation, Writing - review & editing.

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.

Acknowledgements

This work is mainly funded by INTERREG Upper Rhine (European Regional Development Fund) and the Ministries for Research of Baden-Württemberg, Rheinland-Pfalz (Germany) and the Grand Est French Region in the framework of the Science Offensive Upper Rhine HALFBACK project. Q. Cao and A. Beckmann (in part) are also supported by the Engineering and Physical Sciences Research Council (EPSRC) [grant number EPSRC EP/S018107/1]. C. Giannetti's work is supported by the EPSRC project [EP/S001387/1].

References

[1] Aivaliotis, P., Arkouli, Z., Georgoulias, K., Makris, S., 2021. Degradation curves integration in physics-based models: Towards the predictive maintenance of industrial robots. Robotics and Computer-Integrated Manufacturing 71, 102177.

[2] Aivaliotis, P., Georgoulias, K., Chryssolouris, G., 2017. A rul calculation approach based on physical-based simulation models for predictive maintenance, in: 2017 International Conference on Engineering, Technology and Innovation (ICE/ITMC), IEEE. pp. 1243–1246.

[3] Aivaliotis, P., Georgoulias, K., Chryssolouris, G., 2019. The use of digital twin for predictive maintenance in manufacturing. International Journal of Computer Integrated Manufacturing 32, 1067–1080.

[4] Al-Behadili, H.N.K., Ku-Mahamud, K.R., Sagban, R., 2018. Rule pruning techniques in the ant-miner classification algorithm and its variants: A review, in: 2018 IEEE Symposium on Computer Applications & Industrial Electronics (ISCAIE), IEEE. pp. 78–84.

[5] An, A., Cercone, N., 1999. An empirical study on rule quality measures, in: International Workshop on Rough Sets, Fuzzy Sets, Data Mining, and Granular-Soft Computing, Springer. pp. 482–491.

[6] An, A., Cercone, N., 2001. Rule quality measures for rule induction systems: Description and evaluation. Computational Intelligence 17, 409–424.

[7] Ayadi, A., 2018. Semantic approaches for the meta-optimization of complex biomolecular networks. Ph.D. thesis.

[8] Bellatreche, L., Dung, N.X., Pierra, G., Hondjack, D., 2006. Contribution of ontology-based data modeling to automatic integration of electronic catalogues within engineering databases. Computers in Industry 57, 711–724.

[9] Cai, J., Luo, J., Wang, S., Yang, S., 2018. Feature selection in machine learning: A new perspective. Neurocomputing 300, 70–79.

[10] Caiado, R.G.G., Scavarda, L.F., Gavião, L.O., Ivson, P., de Mattos Nascimento, D.L., Garza-Reyes, J.A., 2021. A fuzzy rule-based industry 4.0 maturity model for operations and supply chain management. International Journal of Production Economics 231, 107883.

[11] Cao, Q., Giustozzi, F., Zanni-Merk, C., de Bertrand de Beuvron, F., Reich, C., 2019a. Smart condition monitoring for industry 4.0 manufacturing processes: An ontology-based approach. Cybernetics and Systems 50, 82–96.

[12] Cao, Q., Samet, A., Zanni-Merk, C., de Bertrand de Beuvron, F., Reich, C., 2020a. Combining chronicle mining and semantics for predictive maintenance in manufacturing processes. Semantic Web , 1–22.

[13] Cao, Q., Samet, A., Zanni-Merk, C., de Beuvron, F.d.B., Reich, C., 2019b. An ontology-based approach for failure classification in predictive maintenance using fuzzy c-means and swrl rules. Procedia Computer Science 159, 630–639.

[14] Cao, Q., Zanni-Merk, C., Reich, C., 2018. Ontologies for manufacturing process modeling: A survey, in: International Conference on Sustainable Design and Manufacturing, Springer. pp. 61–70.

[15] Cao, Q., Zanni-Merk, C., Samet, A., de Beuvron, F.d.B., Reich, C., 2020b. Using rule quality measures for rule base refinement in knowledge-based predictive maintenance systems. Cybernetics and Systems 51, 161–176.

[16] Chang, C.C., Lin, C.J., 2011. Libsvm: a library for support vector machines. ACM transactions on intelligent systems and technology (TIST) 2, 1–27.

[17] Chen, W.H., Liu, C.W., Tsai, M.S., 2000. On-line fault diagnosis of distribution substations using hybrid cause-effect network and fuzzy rule-based method. IEEE Transactions on Power Delivery 15, 710–717.

[18] Cram, D., Mathern, B., Mille, A., 2012. A complete chronicle discovery approach: application to activity analysis. Expert Systems 29, 321–346.

[19] Dalenogare, L.S., Benitez, G.B., Ayala, N.F., Frank, A.G., 2018. The expected contribution of industry 4.0 technologies for industrial performance. International Journal of Production Economics 204, 383–394.

[20] Decker, L., Leite, D., Giommi, L., Bonacorsi, D., 2020. Real-time anomaly detection in data centers for log-based predictive maintenance using an evolving fuzzy-rule-based approach, in: 2020 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), IEEE. pp. 1–8.

[21] Dell'Aglio, D., Della Valle, E., van Harmelen, F., Bernstein, A., 2017. Stream reasoning: A survey and outlook. Data Science 1, 59–83.

[22] Du, S., Lv, J., Xi, L., 2012. Degradation process prediction for rotational machinery based on hybrid intelligent model. Robotics and Computer-Integrated Manufacturing 28, 190–207.

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

[24] Faiz, R., Edirisinghe, E.A., 2009. Decision making for predictive maintenance in asset information management. Interdisciplinary Journal of Information, Knowledge, and Management 4, 23–36.

[25] Frank, A.G., Dalenogare, L.S., Ayala, N.F., 2019. Industry 4.0 technologies: Implementation patterns in manufacturing companies. International Journal of Production Economics 210, 15–26.

[26] Giannetti, C., Ransing, R.S., 2016. Risk based uncertainty quantification to improve robustness of manufacturing operations. Computers & Industrial Engineering 101, 70–80.

[27] Golbreich, C., Imai, A., 2004. Combining swrl rules and owl ontologies with protégé owl plugin, jess, and racer, in: 7th International Protégé Conference, Bethesda, MD.

[28] Goodfellow, I., Bengio, Y., Courville, A., Bengio, Y., 2016. Deep learning. volume 1. MIT press Cambridge.

[29] Grall, A., Dieulle, L., Bérenguer, C., Roussignol, M., 2002. Continuous-time predictive-maintenance scheduling for a deteriorating system. IEEE transactions on reliability 51, 141–150.

[30] Grimm, S., 2009. Semantic matchmaking with nonmonotonic description logics. volume 1. IOS Press.

[31] Gruber, T., 2009. Ontology. Springer.

[32] Gruber, T.R., 1993. A translation approach to portable ontology specifications. Knowledge acquisition 5, 199–220.

[33] Guyon, I., Elisseeff, A., 2003. An introduction to variable and feature selection. Journal of machine learning research 3, 1157–1182.

[34] Hayes-Roth, F., 1985. Rule-based systems. Communications of the ACM 28, 921–932.

[35] Hu, H., Tang, B., Gong, X., Wei, W., Wang, H., 2017. Intelligent fault diagnosis of the high-speed train with big data based on deep neural networks. IEEE Transactions on Industrial Informatics 13, 2106–2116.

[36] Javed, K., Gouriveau, R., Zerhouni, N., 2017. State of the art and taxonomy of prognostics approaches, trends of prognostics applications and open issues towards maturity at different technology readiness levels. Mechanical Systems and Signal Processing 94, 214–236.

[37] Kharlamov, E., Mehdi, G., Savković, O., Xiao, G., Kalaycı, E.G., Roshchin, M., 2019. Semantically-enhanced rule-based diagnostics for industrial internet of things: The sdrl language and case study for siemens trains and turbines. Journal of Web Semantics 56, 11–29.

[38] Kleinbaum, D.G., Dietz, K., Gail, M., Klein, M., Klein, M., 2002. Logistic regression. Springer.

[39] Li, D., Tang, H., 2021. A semantic-level component-based scheduling method for customized manufacturing. Robotics and Computer-Integrated Manufacturing 71, 102144.

[40] Li, H., Parikh, D., He, Q., Qian, B., Li, Z., Fang, D., Hampapur, A., 2014. Improving rail network velocity: A machine learning approach to predictive maintenance. Transportation Research Part C: Emerging Technologies 45, 17–26.

[41] Liu, W., Wang, Z., Liu, X., Zeng, N., Liu, Y., Alsaadi, F.E., 2017. A survey of deep neural network architectures and their applications. Neurocomputing 234, 11–26.

[42] Luo, W., Hu, T., Ye, Y., Zhang, C., Wei, Y., 2020. A hybrid predictive maintenance approach for cnc machine tool driven by digital twin. Robotics and Computer-Integrated Manufacturing 65, 101974.

[43] MANGILI, F., 2013. Development of advanced computational methods for prognostics and health management in energy components and systems .

[44] McCann, M., Johnston, A., 2008. Secom dataset, uci machine learning repository.

[45] Nuñez, D.L., Borsato, M., 2018. Ontoprog: An ontology-based model for implementing prognostics health management in mechanical machines. Advanced Engineering Informatics 38, 746–759.

[46] Ortega, G., Filatovas, E., Garzón, E., Casado, L.G., 2017. Non-dominated sorting procedure for pareto dominance ranking on mul-

ticore cpu and/or gpu. Journal of Global Optimization 69, 607–627.
[47] O'connor, M., Knublauch, H., Tu, S., Musen, M., 2005. Writing rules for the semantic web using swrl and jess. Protégé With Rules WS, Madrid .
[48] Pal, M., 2005. Random forest classifier for remote sensing classification. International journal of remote sensing 26, 217–222.
[49] Pérez, J., Arenas, M., Gutierrez, C., 2009. Semantics and complexity of sparql. ACM Transactions on Database Systems (TODS) 34, 1–45.
[50] Phillips, J., Cripps, E., Lau, J.W., Hodkiewicz, M., 2015. Classifying machinery condition using oil samples and binary logistic regression. Mechanical Systems and Signal Processing 60, 316–325.
[51] Ramírez-Gallego, S., García, S., Mouriño-Talín, H., Martínez-Rego, D., Bolón-Canedo, V., Alonso-Betanzos, A., Benítez, J.M., Herrera, F., 2016. Data discretization: taxonomy and big data challenge. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 6, 5–21.
[52] Romeo, L., Loncarski, J., Paolanti, M., Bocchini, G., Mancini, A., Frontoni, E., 2020. Machine learning-based design support system for the prediction of heterogeneous machine parameters in industry 4.0. Expert Systems with Applications 140, 112869.
[53] Sellami, C., Miranda, C., Samet, A., Tobji, M.A.B., de Beuvron, F., 2019. On mining frequent chronicles for machine failure prediction. Journal of Intelligent Manufacturing , 1–17.
[54] Sellami, C., Miranda, C., Samet, A., Tobji, M.A.B., de Beuvron, F., 2020. On mining frequent chronicles for machine failure prediction. Journal of Intelligent Manufacturing 31, 1019–1035.
[55] Sellami, C., Samet, A., Tobji, M.A.B., 2018. Frequent chronicle mining: Application on predictive maintenance, in: 2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA), IEEE. pp. 1388–1393.
[56] Sikorska, J., Hodkiewicz, M., Ma, L., 2011. Prognostic modelling options for remaining useful life estimation by industry. Mechanical systems and signal processing 25, 1803–1836.
[57] Simons, P., 1999. Parts: A study in ontology .
[58] Susto, G.A., Schirru, A., Pampuri, S., McLoone, S., Beghi, A., 2014. Machine learning for predictive maintenance: A multiple classifier approach. IEEE Transactions on Industrial Informatics 11, 812–820.
[59] Yan, W., Liu, H., Liu, Y., Wang, J., Zanni-Merk, C., Cavallucci, D., Yan, X., Zhang, L., 2018. Latent semantic extraction and analysis for triz-based inventive design. European Journal of Industrial Engineering 12, 661–681.
[60] Yan, W., Liu, H., Zanni-Merk, C., Cavallucci, D., 2015. Ingenioustriz: an automatic ontology-based system for solving inventive problems. Knowledge-Based Systems 75, 52–65.
[61] Yan, W., Zanni-Merk, C., Cavallucci, D., Collet, P., 2014. An ontology-based approach for inventive problem solving. Engineering Applications of Artificial Intelligence 27, 175–190.
[62] Yang, X., Ran, Y., Zhang, G., Wang, H., Mu, Z., Zhi, S., 2022. A digital twin-driven hybrid approach for the prediction of performance degradation in transmission unit of cnc machine tool. Robotics and Computer-Integrated Manufacturing 73, 102230.
[63] Zhang, W., Yang, D., Wang, H., 2019. Data-driven methods for predictive maintenance of industrial equipment: a survey. IEEE Systems Journal 13, 2213–2227.
[64] Zhou, K., Liu, T., Zhou, L., 2015. Industry 4.0: Towards future industrial opportunities and challenges, in: 2015 12th International conference on fuzzy systems and knowledge discovery (FSKD), IEEE. pp. 2147–2152.
[65] Zimmermann, H.J., 2010. Fuzzy set theory. Wiley Interdisciplinary Reviews: Computational Statistics 2, 317–332.