Combining Chronicle Mining and Semantics for Predictive Maintenance in Manufacturing Processes

Qiushi Cao [a,*], Ahmed Samet [b], Cecilia Zanni-Merk [a], François de Bertrand de Beuvron [b] and Christoph Reich [c]

[a] *Normandie Université, INSA Rouen, LITIS, 76000 Saint-Étienne-du-Rouvray, France*
E-mails: qiushi.cao@insa-rouen.fr, cecilia.zanni-merk@insa-rouen.fr
[b] *ICUBE/SDC Team (UMR CNRS 7357)-Pole API BP 10413, 67412 Illkirch, France*
E-mails: ahmed.samet@insa-strasbourg.fr, francois.debertranddebeuvron@insa-strasbourg.fr
[c] *Hochschule Furtwangen University, 78120 Furtwangen, Germany*
E-mail: rch@hs-furtwangen.de

Editors: First Editor, University or Company name, Country; Second Editor, University or Company name, Country
Solicited reviews: First Solicited Reviewer, University or Company name, Country; Second Solicited Reviewer, University or Company name, Country
Open reviews: First Open Reviewer, University or Company name, Country; Second Open Reviewer, University or Company name, Country

Abstract. Within manufacturing processes, faults and failures may cause severe economic loss. With the vision of Industry 4.0, artificial intelligence techniques such as data mining play a crucial role in automatic fault and failure prediction. However, data mining especially pattern mining results normally lack both machine and human understandable representation and interpretation of knowledge, bringing obstacles to novice users to interpret the prediction results.

To tackle this issue, in this paper we introduce a novel hybrid approach to facilitate predictive maintenance tasks in manufacturing processes. The proposed approach is a combination of data mining and semantics, within which chronicle mining is used to predict the future failures of the monitored industrial machinery, and a Manufacturing Predictive Maintenance Ontology (MPMO) with its rule-based extension is used to predict temporal constraints of failures and to represent the predictive results formally. As a result, Semantic Web Rule Language (SWRL) rules are constructed for predicting occurrence time of machinery failures in the future. The proposed rules provide explicit knowledge representation and semantic enrichment of failure prediction results, thus easing the understanding of the inferred knowledge. A case study on a semi-conductor manufacturing process is used to demonstrate our approach in detail.

Keywords: Semantics, Chronicle Mining, Predictive Maintenance, Manufacturing Process, Industry 4.0

1. Introduction

Manufacturing processes are sets of structured operations to transform raw material or semi-finished product parts into further completed products. To ensure high productivity, availability and efficiency of manufacturing processes, the detection of harmful tendencies and conditions of production lines is a crucial issue for manufacturers. In general, anomaly detection on production lines is performed by analyzing data collected by sensors, which are located on machine components and also in production environments. The collected data record real-time situations and reflect the correctness of mechanical system conditions. When

*Corresponding author. E-mail: qiushi.cao@insa-rouen.fr.

the tendency of a mechanical failure emerges, experienced operators in factories are able to take appropriate operations to prevent the outage situations of production systems. However, as the collected data become more heterogeneous and complex, it is conceivable that the operators may fail to respond to mechanical failures timely and accurately. In the context of Industry 4.0, advanced techniques such as the Industry Internet of Things (IIoT) and Cloud Computing enable machines and production systems in smart factories to be interconnected to exchange data continuously. This trend has brought opportunities to manufactures to effectively manage and use the collected big data and has triggered the demand of methodologies to detect anomalies on production lines automatically.

In the manufacturing domain, the detection of anomalies such as mechanical faults and failures enables the launching of *predictive maintenance* tasks, which aim to predict future faults, errors, and failures and also enable maintenance actions. Normally, a predictive maintenance task relies on the monitoring of a measurable system diagnostic parameter, which identifies the state of a system [2]. In this way, maintenance decisions, such as calling the intervention of a machine operator, are proposed based on the severity of anomalies, to prevent the halt of the production lines and to minimize economic loss. Several techniques have been used to detect wear and tear in mechanical units and to predict future machinery conditions, such as machine learning, data mining, statistics, and information theory [39]. Among these techniques, data mining has shown notable competence for automatic anomaly detection in industry [29]. In smart factories, data mining is normally performed by obtaining and processing sensor data, which contain measurements of physical signals of machinery, such as temperature, voltage, and vibration. By identifying events and patterns that are not consistent with the expected behavior, potential hazards in production systems, such as a mechanical system deterioration tendency, could be detected.

However, to interpret the data mining results, domain knowledge is required. This brings obstacles to novice users for having a deep understanding of the results. Furthermore, sometimes the extracted knowledge is presented in a complex structure, therefore needs formal knowledge representation methods to facilitate the understanding and exploitation of it [31]. To overcome this issue, semantic technologies have been utilized in several research efforts to promote the interpretation and management of knowledge [3, 7, 31, 32]. Several stages of data mining can benefit from the involvement of formal semantics, such as data transformation, algorithm selection, and post-processing [7]. Moreover, the use of semantic technologies can also integrate the capitalization of domain experts' experience. For example, in a predictive maintenance task of machine cutting tool, when data mining algorithms fail to identify the occurrence time of a future cutter failure, logic-based expert rules which capitalize experience of domain experts can be applied to propose predictive decisions.

In the context of predictive maintenance in smart factories, pattern mining has been widely used to discover frequently occurring temporally-constrained patterns, through which warning signals can be sent to humans for a timely intervention [4]. Among pattern mining techniques, chronicle mining has been applied to industrial data sets for extracting temporal information of events and to predict potential machinery failures [6]. However, even though chronicle mining results are expressive and interpretable representations of complex temporal information, domain knowledge is required for users to have a comprehensive understanding of the mined chronicles [16]. As the predictive maintenance domain is becoming more knowledge-intensive, tasks performed in this domain can often benefit from incorporating domain and contextual knowledge, by which the semantics of the chronicle mining results can be explicitly represented and clearly interpreted. This helps to reduce the semantic gap issue, which stands for the incoherence between the knowledge extracted from industrial data and the interpretation of the knowledge from a user [7]. However, to the best of our knowledge, no work has been proposed to combine chronicle mining, and semantics to facilitate the predictive maintenance of manufacturing processes. Also, most of the existing research works about predictive maintenance in the manufacturing domain merely focus on the classification of operating conditions of machines (e.g., normal operating condition, breakdown condition...), while lacking the extraction of specific temporal information of failure occurrence. This brings obstacles for users to perform maintenance actions with the consideration of temporal constraints. To this end, in this paper, we use an ontology-based approach to represent chronicle mining results in a semantic rich format, which enhances the sharing and reuse of knowledge. By specifying domain semantics and annotating industrial data with rich and formal semantics, ontologies with their rule-based extensions help to reduce the aforementioned issues. In more detail, the contributions of this paper are as follows:

- We present a domain ontology named Manufacturing Predictive Maintenance Ontology (MPMO), which is a Web Ontology Language (OWL) [8] based ontology designed to model the knowledge related to condition-based maintenance. The MPMO ontology provides the foundation to formally represent chronicles with their numerical time constraints, for the purpose of predictive maintenance.
- We propose an algorithm to transform chronicles into Semantic Web Rule Language (SWRL) based logic rules, by which the predictive results are formalized, thus interpretable for both human and machines. The proposed transformation enables the automatic generation of SWRL rules from chronicle mining results, thus allowing an automatic semantic approach for machinery failure prediction.
- We evaluate the feasibility and effectiveness of our approach by conducting experimentation on a real industrial data set. The performance of SWRL rule construction and the quality of failure prediction is evaluated against the aforementioned data set.

The rest of this paper is structured as follows. Section 2 provides a review of existing ontology-based models and systems developed for predictive maintenance. Section 3 introduces the foundations and basic notions of chronicle mining and semantics that are necessary for describing our approach. It contains formal definitions of chronicles and the Semantic Web Rule Language (SWRL). Sections 4 presents a hybrid semantic approach for automatic failure prediction. The approach includes the use of the MPMO ontology, which models necessary and principle knowledge related to chronicles. Also, the algorithm for transforming chronicles to SWRL-based predictive rules is introduced in detail. Section 5 evaluates our approach through a real industrial data set. Section 6 gives concluding remarks and outlines future research directions.

2. Related Work

In recent years, several efforts have been proposed to facilitate knowledge representation and interpretation in predictive maintenance tasks. Among them, the ontology-based approach is an effective and notable method that has drawn considerable attention from researchers. Ontologies are explicit specifications of conceptualizations, and they are comprehensive and reusable knowledge repositories in various domains [37]. In general, this type of approach uses ontologies to formally define the semantics of knowledge and data, and utilizes sets of logic rules to enable ontological reasoning, for inferring new knowledge. The available research works related to this approach can be categorized into two major fields, according to different purposes: i) using ontology-based approach to represent data mining results in a formal and structured way, to further enrich knowledge bases; ii) using ontology-based approach to facilitate knowledge formalization,sharing and reuse in the predictive maintenance domain.

To formalize the data mining results and to facilitate the interpretation of them, many researchers tried to incorporate explicit domain knowledge with using ontologies. The DAMON ontology [20] is developed as a data mining ontology to simplify the development of distributed knowledge discovery systems. The ontology is used as a knowledge reference model to help domain experts solve tasks. Also, the ontology enables users to search for data mining resources and software when they want to find solutions for a specific problem. The EXPO ontology [18] formalizes concepts about experimental design, methodologies and results representation in a general way. The ontology promotes the sharing of experimental results within and among different subjects, and it can reduce the information duplication and loss in the sharing process. The OntoDM-core ontology [30] is developed to formally describe core data mining entities. The ontology provides a framework to represent essential and basic data mining concepts, such as data sets, data mining tasks, algorithms, and constraints. The advantage of this ontology is its powerful representation of constraint-based data mining activities.

The use of an ontology-based approach can also facilitate knowledge formalization, sharing and reuse in the predictive maintenance domain. In the context of predictive maintenance, several ontologies and ontology-based intelligent systems are developed to achieve this goal. To enhance the expressiveness of these ontologies, several rule-based extensions were proposed to perform ontological reasoning, in order to facilitate maintenance decisions of users. We review existing ontologies according to two aspects: ontologies that model manufacturing processes and ontologies that model preventive maintenance tasks.

As indicated in the introduction, manufacturing processes are structured sets of operations that transform raw materials or semi-finished product segments into further completed product parts. Over the last decades, several ontologies have been developed to represent knowledge about manufacturing processes. The Process Specification Language (PSL) ontology [21] is one of the early-stage contributions. This ontology axiomatizes a set of semantic primitives that are essential for describing a wide range of manufacturing processes. The axioms defined in this ontology model the key elements of manufacturing processes, such as process scheduling, process modeling, production planning, and project management [21]. Another contribution in this subdomain is the Manufacturing Service Description Language (MSDL) ontology, which defines a well-defined framework for formal representation of manufacturing services [12]. This ontology formalizes manufacturing capabilities of manufacturing resources in different levels of abstraction, based on which a rule-based extension of the ontology is proposed to enable automatic supplier discovery. At last we mention the Manufacturing Reference Ontology (MRO) [42] that is developed to formalize a set of core concepts about the manufacturing in a high abstraction level. The ontology categorizes the manufacturing domain into eight general concepts: *Realized Part*, *Part Version*, *Manufacturing Facility*, *Manufacturing Resource*, *Manufacturing Method*, *Manufacturing Process*, *Feature* and *Part Family*. This categorization enables further development of more specific ontologies in the production domain.

Compared to ontologies that model manufacturing processes, ontologies for predictive maintenance are much less numerous. These type of ontologies normally focus on the issues of fault or failure prognostics and machine health monitoring. Among these ontologies, the OntoProg Ontology [9] addresses the failure prediction of machines in smart factories. The ontology is developed based on a set of international standards, and a classification for severity criteria, detection, diagnostics and prognostics of failure modes is provided. The ontology standardizes the concepts that are necessary for tackling machinery failure analysis tasks. As another most recent contribution, the Sensing System Ontology [10] is proposed to define the embedded sensing systems for industrial Product-Service Systems (PSSs). This ontology is used as the backbone of the PSS knowledge-based framework and it describes the sensors that are embedded on PSSs, for the aim of providing customized services for users.

After reviewing the ontologies mentioned above, we recognize that none of them provides satisfactory knowledge representation of both manufacturing and predictive maintenance domains. Some of these ontologies focus on a narrow field, such as the manufacturing resource planning domain, and they do not formalize predictive maintenance-related concepts, e.g., machinery *Failure* and *Fault*. Also, none of the existing ontologies standardize the concepts related to chronicle mining. To jointly use chronicle mining with semantic technologies for a predictive maintenance task on a piece of machinery, the knowledge base should incorporate not only the machine-interpretable knowledge of manufacturing entities such as product and process, but also the knowledge about chronicles within which the machinery failures are described in a structured way. In this context, in this paper, we present the MPMO ontolgoy with its rule-based extension, for the goal of proposing an automatic semantic approach for machinery predictive maintenance. The proposed automatic semantic approach aims to bridge the semantic gap issue mentioned in Section 1.

3. Foundations and Basic Notions

In this section, we introduce foundations and basic notions of chronicle mining and semantics that are necessary for describing our approach. The foundations include a formal description of Sequential Pattern Mining (SPM) and chronicles, as well as an introduction to Semantic Web Rule Language (SWRL).

3.1. Foundations of Sequential Pattern Mining

In industry, data collected for preventive maintenance tasks are normally represented as sets of sequences with time stamps [5]. To cope with this type of data sets, SPM is one important technique to extract frequently occurring patterns. SPM was first studied by [33], to analyze customer purchase behavior sequences. One SPM task could be described as follows: Given a data set containing a number of sequences, the goal of SPM is to find sequential patterns whose support exceed a predefined numeric support threshold.

This support threshold indicates the minimal number of occurrences of the sequential patterns, and the found patterns are called frequent sequential patterns. For the output of SPM algorithms, each frequent sequential pattern is a sequence which consists of a set of items in a certain order.

To give a formal description of sequential patterns, in this subsection we review the definitions of key concepts. A sequence S is a set of ordered itemsets, denoted by $S =< SID, < I_1 I_2 I_3 ... I_n >>$, with SID standing for the index of the sequence with I_j representing a non-empty set of items. Given two sequences $S_a =< SID, < a_1 a_2 a_3 ...a_m >>$ and $S_b =< SID, < b_1 b_2 b_3 ...b_n >>$, the sequence S_a is considered to be the subsequence of S_b, denoted as $S_a \subseteq S_b$, if there exists integers $1 \leqslant k_1 < k_2 < ... < k_m \leqslant n$ such that $a_1 \subseteq b_{k1}$, $a_2 \subseteq b_{k2}$, ..., $a_m \subseteq b_{km}$ [38]. One example of sequence data set is shown in Table 1. In the table, each row is a sequence of elements. The elements are presented with a certain order, showing the precedence relationships among them. For example, regarding the definitions we recalled before, the sequence $< ce(ac) >$ is the subsequence of $< \underline{\mathbf{c}}(ab\underline{\mathbf{e}})(\underline{\mathbf{ac}}f) >$. If we set the minimum support to 3, we can validate that $< (ab)c >$ is a sequential pattern with the support of 3.

Over the last decades, considerable contributions have been settled in the research field of SPM [28]. As a result, various SPM algorithms have been proposed to mine frequent sequential patterns. Based on these proposed SPM algorithms, a variety of approaches and experiments have been launched to improve the performance and efficiency of SPM tasks.

3.2. Sequential Pattern Mining with Time Intervals

Even though sequential patterns contain information about the orders of items, the algorithms introduced in the previous section can not specify the time intervals between elements and items. In real-world situations, the occurrences of events are often recorded with temporal information, such as time points and time intervals between events. Thus, several contributions have been proposed to obtain the time intervals between successive items in sequences. The notion of the time-interval sequential pattern is first presented by Yoshida et al. [27]. The authors name this kind of patterns as *"delta patterns"*. A delta pattern is an ordered list of itemsets with the time intervals between two neighboring itemsets. It can be represented as $A \xrightarrow{[0,3]} B \xrightarrow{[2,5]} C$, where $A \to B \to C$ is a frequent sequential pattern. The time intervals $[0, 3]$ and $[2, 5]$ are bounding intervals, which means the transition time of $A \to B$ is contained in the time interval $[0, 3]$, and the transition time of $B \to C$ is placed in the time interval $[2, 5]$.

With the introduction of delta patterns, a group of algorithms were proposed to facilitate the mining process in temporal sequence data sets. One significant contribution is the work by Hirate et al. [41]. In this work, the authors propose the Hirate-Yamana algorithm to mine all frequent time-extended sequences. To do this, the authors generalize SPM with item intervals. In the generalization, they define a set of time-extended sequences , denoted as $S_t =< SID, (t_{1,1}, i_1), (t_{1,2}, i_2), (t_{1,3}, i_3), ..., (t_{1,n}, i_n) >>$, where i_j means an item, and $t_{\alpha,\beta}$ is the item interval between items i_α and i_β, $t_{\alpha,\beta}$ can be interpreted according to two aspects of conditions [41]:

- If the data sets contain time stamps, which indicate the transaction occurrences of items, then $t_{\alpha,\beta}$ becomes the time interval and can be computed by the equation $t_{\alpha,\beta} = i_\beta.time - i_\alpha.time$, where $i_\beta.time$ and $i_\alpha.time$ are time stamps of items i_α and i_β respectively. For example, one time-extended sequence could be $< (0, c), (1, abe), (3, ac), (5, f) >$, which means item c occurs at time point 0, followed by itemset *abe* occurring at 1 time unit later. Itemset *ac* occurs 2 time unites after *abe*, and the last itemset f occurs 2 time unites after *ac*.
- If the data sets do not contain time stamps, then $t_{\alpha,\beta}$ may become the item gap and defined by the equation $t_{\alpha,\beta} = \beta - \alpha$. In this case, the item gap is defined as the number of items that occur between two items. This type of representation is suitable to be applied to data sets which contain fixed item intervals, but it is not applicable to data sets which contain various length of time intervals.

The study on existing notions and algorithms help to capture the core concepts in the domain of time-interval SPM. These core concepts form the foundations of chronicle mining.

3.3. Foundations of Chronicle Mining

As introduced in the previous section, the temporal patterns we consider in this paper are chronicles. To give formal definition of chronicles, we start by introducing the concept of *Event*, given by [6].

Definition 1 (Event)**.** *Let $\mathbb{E}$ be a set of event types, and $\mathbb{T}$ a time domain such that $\mathbb{T} \subseteq \mathbb{R}$. $\mathbb{E}$ is assumed totally ordered and is denoted $\leqslant_\mathbb{E}$. According to [6], an event is a couple (e, t) where $e \in \mathbb{E}$ is the type of the event and $t \in \mathbb{T}$ is its time. In SPM, events represent itemsets of a single sequence.*

Table 1

An example sequence data set.

SID	Sequences
10	$< c(\underline{\mathbf{ab}}e)(a\underline{\mathbf{c}}f) >$
20	$< (bcd)(ac)(bd)(adf)f >$
30	$< (cd)(\underline{\mathbf{ab}})(b\underline{\mathbf{c}}f)e >$
40	$< b(df)(bdf)c(ab) >$
50	$< (ab)(bef)de >$
60	$< (\underline{\mathbf{ab}}e)(\underline{\mathbf{c}}d)(ce) >$

A sequence contains a set of ordered events, which are timestamped. The events contained in a sequence appear according to their time of occurrences.

Definition 2 (Sequence)**.** *Let $\mathbb{E}$ be a set of event types, and $\mathbb{T}$ a time domain such that $\mathbb{T} \subseteq \mathbb{R}$. $\mathbb{E}$ is assumed totally ordered and is denoted $\leqslant_{\mathbb{E}}$. According to the definition in [6], a sequence is a couple $\langle SID, \langle (e_1, t_1), (e_2, t_2), ..., (e_n, t_n) \rangle \rangle$ such that $\langle (e_1, t_1), (e_2, t_2), ..., (e_n, t_n) \rangle$ is a sequence of events. For all $i, j \in [1, n], i < j \Rightarrow t_i \leqslant t_j$. If $t_i = t_j$ then $e_i <_{\mathbb{E}} e_j$.*

When the events are time-stamped, how to describe the quantitative time intervals among different events is vital important for the prediction of possible future events. To achieve this goal, we introduce the notion *temporal constraints* in the following definition. The definition of *temporal constraints* is adopted from the one introduced in [6].

Definition 3 (Temporal constraint)**.** *A temporal constraint is a quadruplet (e_1, e_2, t^-, t^+), denoted $e_1[t^-, t^+]e_2$, where $e_1, e_2 \in \mathbb{E}$, $e_1 \leqslant_{\mathbb{E}} e_2$ and $t^-, t^+ \in \mathbb{T}$.*

t^- and t^+ are two integers which are called lower bound and upper bound of the time interval, such that $t^- \leqslant t^+$. A couple of events (e_1, t_1) and (e_2, t_2) are said to satisfy the temporal constraint $e_1[t^-, t^+]e_2$ iff $t_2 - t_1 \in [t^-, t^+]$.

We say 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 $e_2 = e'_2$

With obtaining introducing the *events* and *temporal constraints* among different events within a sequence, we are able to to define the concept of chronicles [6].

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

1. *$\mathcal{E} = \{e_1 ... e_n\}$, where $\forall i, e_i \in \mathcal{E}$ and $e_i \leqslant_{\mathbb{E}} e_{i+1}$,*
2. *$\mathcal{T} = \{t_{ij}\}_{1 \leqslant i < j \leqslant |\mathcal{E}|}$ is a set of temporal constraints on $\mathcal{E}$ such that for all pairs (i, j) satisfying $i < j$, t_{ij} is denoted by $e_i[t^-_{ij}, t^+_{ij}]e_j$.*

$\mathcal{E}$ is called the episode of $\mathcal{C}$, according to the definition of episode's discovery in sequences [6].

In the chronicle discovery process, *support* is used as a measure to compute the frequency of a pattern inside a sequence. It can therefore be formalized by the definition below.

Definition 5 (Chronicle support)**.** *An occurrence of a chronicle $\mathcal{C}$ in a sequence S is a set $(e_1, t_1) ... (e_n, t_n)$ of events of the sequence S that satisfies all temporal constraints defined in $\mathcal{C}$. The support of a chronicle $\mathcal{C}$ in the sequence S is the number of its occurrences in S, or the percentage of its occurrences in the sequence S [5].*

The relevance of a chronicle is essentially based on the value of its support.

To illustrate these basic definitions, we give an example including a sequence and a chronicle extracted from it. Assuming a sequence S contains three events $< A, B, C >$, represented as follows:

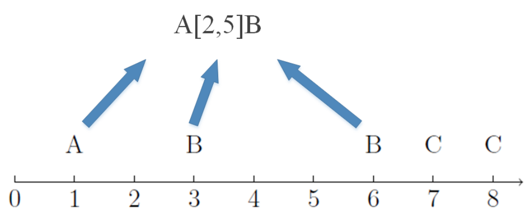

Fig. 1. A sequence representing three events.

In Fig. 1, time constraints that describe the pattern {A, B, C} are noted by A[2,5]B, B[1,5]C and A[6,7]C. Here [2,5], [1,4] and [6,7] lower and upper bounds of the time intervals among events.

After the generation of temporal constraints, these events can be represented as a graphical way, as shown in Fig. 2. In the figure, events are represented by the circles, and temporal constraints are displayed through arrows among events. The values above each arrow are quantitative numerical bounds of temproal constraints.

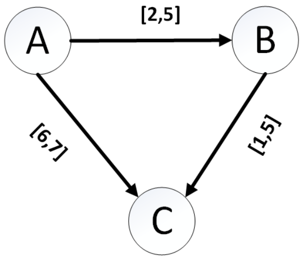

Fig. 2. Example of a chronicle

In the domain of predictive maintenance, frequent chronicle mining has been used to detect machine anomalies in advance. To combine frequent chronicle mining and semantics for facilitating predictive maintenance tasks, a special type of chronicles, called *failure chronicles* is introduced [5].

Definition 6 (Failure chronicle)**.** *For a chronicle $\mathcal{C}_F = (\mathcal{E}, \mathcal{T})$, we say that $\mathcal{C}_F$ is a failure chronicle if and only if the events that describe it are set according to their order of occurrence in the sequence, and that the end of the chronicle is the event that represents the failure, i.e. for $\mathcal{E} = \{e_1 \cdots e_n | e_i \leqslant_{\mathbb{E}} e_{i+1}, i \in [1, n]\}$, e_n is the failure event.*

In [5], a new algorithm called CPM has been introduced to mine frequent failure chronicles. Based on their work, in this paper, we propose a novel algorithm to automatically generate SWRL rules from frequent failure chronicles. The generated SWRL rules aim to provide decision making for predictive maintenance in industry. The algorithm is introduced in Section 4.

3.4. Semantic Web Rule Language

Semantic Web Rule Language (SWRL) is based on a combination of its sublanguages OWL DL and OWL Lite with the RuleMarkup Language. A SWRL rule is in the form of an implication between an antecedent (body) and consequent (head), which can be interpreted in a way that whenever the conditions specified in the antecedent hold, then the conditions specified in the consequent must also hold [14]. In SWRL, a rule has the syntax: *Antecedent* $\rightarrow$ *Consequent*, where both the antecedent (body) and consequent (head) contains zero or more atoms. Atoms in SWRL rules can be the form of C(x), P(x,y), where C(x) is an OWL class, P is an OWL property, and x,y are either variables, OWL individuals or OWL data values [14].

In this work, the reason we choose SWRL rules is two-fold. Firstly, SWRL provides model-theoretic semantics and has the advantage of its close association with OWL ontologies, which enables the definition of complex rules for reasoning about individuals in ontologies. Secondly, the use of SWRL to write rules is independent of rule implementation languages within rule engines, which has the advantage of the flexible selection of rule engines and inference platform.

To represent data mining results, especially chronicles, in a formal and structured way, we use ontologies as well as SWRL rules to propose predictive rules. The proposed rules describe events and temporal constraints within chronicles, and predict a special type of event (a machinery failure), with corresponding to temporal information.

4. A Novel Hybrid Semantic Approach For Predictive Maintenance

To propose the novel hybrid semantic approach for predictive maintenance, we jointly use data mining and semantic technologies, within which chronicle mining is used to predict the future failures of the monitored industrial machinery, and domain ontologies with their rule-based extension is used to predict temporal constraints of failures and to represent the predictive results formally. The procedure of the semantic approach is shown in Fig. 3. Firstly, data preprocessing is implemented on raw industry data sets to obtain sequences in the form of pairs (event, time stamp), where each sequence finishes with the failure event. Secondly, frequent chronicle mining algorithms mine the pre-processed data to discover frequent patterns that indicate machinery failures. Thirdly, based on the mined frequent patterns, semantic technologies are used to automate the generation of SWRL-based predictive rules. These rules enable ontological reasoning over individuals in ontologies, thus facilitating decision making.

4.1. Domain Knowledge

Within an intelligent system, ontologies contain the domain knowledge to operate. In this work, the MPMO ontology is developed to describe the concepts and relationships within chronicles. The definitions of key concepts and relationships in the MPMO ontology is based on the basic notions introduced in Section 3. To illustrate the global architecture of the ontology, we use a UML notation where boxes stand for ontology classes, and arrows are object properties. Data prop-

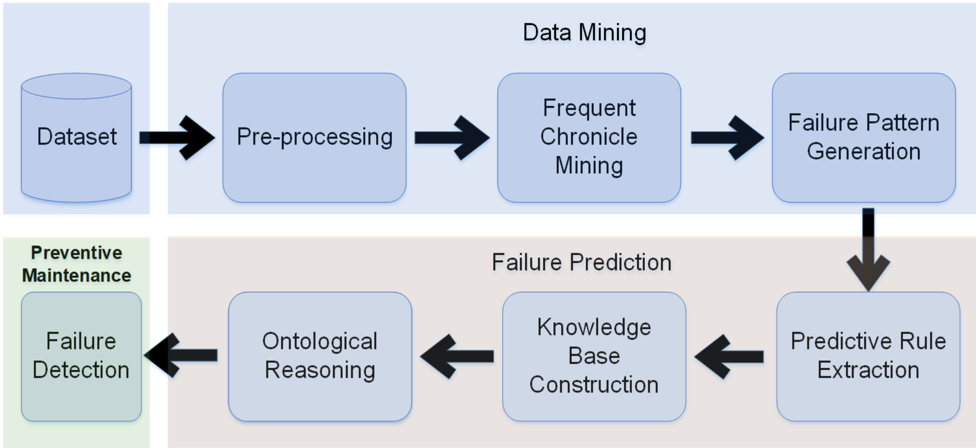

Fig. 3. The procedure of the semantic approach for predictive maintenance.

erties are indicated by class attributes. The global architecture of this ontology is shown in Fig. 4. For the purpose of clarity, only a subset of the whole classes and relationships are presented.

To introduce the MPMO ontology, we give the axioms of the key classes. The axioms defining the main classes of the MPMO ontology are presented below using the description logic (DL) syntax [11].

- *ManufacturingResource*: This class describes the resources that are used within manufacturing processes. It consists three subclasses: *FinancialResource*, *HumanResource*, and *PhysicalResource*. Among the three subclasses, *PhysicalResource* stands for a set of physical entities that the predictive maintenance tasks are performed upon, such as machine tools, workpieces, and final products. The definition of this class is extended from the class *MASON: Resource*, in the MASON ontology [35]. The DL axioms for defining this class and the *PhysicalResource* class are

$$ManufacturingResource \equiv HumanResource \sqcup PhysicalResource \sqcup FinancialResource,$$

and

$$ManufacturingResource \sqsubseteq \forall MakesUse-Of^{-1}.ManufacturingProcess.$$

- *ManufacturingProcess*: It describes different types of structured sets of operations that transform raw materials or semi-finished product segments into further completed product parts [32]. The DL axioms for defining this class are

$$ManufacturingProcess \equiv AssemblyProcess \sqcup FinishingProcess \sqcup FormingProces \sqcup MachiningProcess \sqcup MouldingProcess,$$

and

$$ManufacturingProcess \equiv \exists MakesUseOf.Ma-nufacturingResource \sqcap \exists hasProcessInput.W-orkpiece \sqcap \exists Produces.RealizedPart.$$

- *Chronicle*: Chronicles are a special type of temporal patterns, in which temporal orders of events are quantified with numerical bounds [6]. To introduce this concept in the MPMO ontology, we use the following axiom.

$$Chronicle \equiv \forall hasEvent.Event \sqcap (\geqslant 1\ hasEvent.Event) \sqcap \forall hasTimeInterval.Ti-meInterval \sqcap (\geqslant 1\ hasTimeInterval.TimeInte-rval) \sqcap\ \exists isLearnedFrom.ManufacturingPro-cess.$$

- *Event*: . In predictive maintenance tasks, an *Event* is generally associated with a set of *ObservedProperties* which indicate the correctness of the operation of a piece of machinery. In this context,

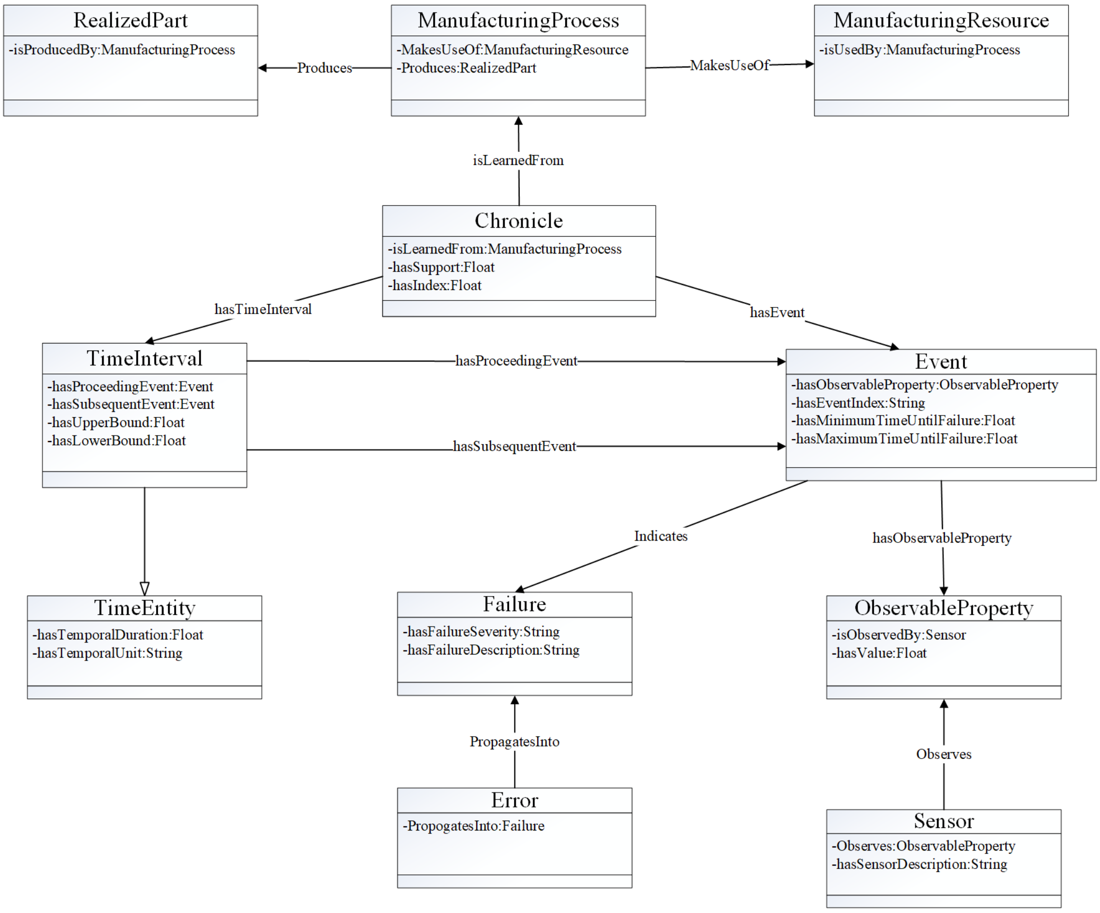

Fig. 4. The global architecture of the MPMO ontology.

the DL axioms for defining this class is

$$Event \equiv \forall hasObservedProperty.Observed- Property \sqcap (\geqslant 1\ hasObserved.Property).$$

– *ObservedProperty*: This is an attribute which represents some significant measurable characteristic of a monitored *ManufacturingProcess*, *ManufacturingResource* or *RealizedPart*. The value of an *ObservedProperty* is measured by sensors which are located at different components of the monitored entity. This class is also called *Attribute*. The DL axioms for defining this class are

$$ObservedProperty \sqsubseteq \exists hasObserved- Property^{-1}.Event \sqcap \exists Observes^{-1}.Sensor.$$

– *Failure*: This class represents the *Failures* that are indicated by *Events*. A *Failure* is the inability of an entity to perform one required function, and it can be the result of a propagation of a machinery error [1]. The following axiom is used to define this class:

$$Failure \sqsubseteq \forall PropagatesInto^{-1}.Error.$$

– *TimeInterval*: A temporal entity with an extent or duration. The definition of this class is adopted from the Time Ontology [17]. The axiom for describing this class is

$$TemporalInterval \sqsubseteq \exists hasProceeding- Event.Event \sqcap \exists hasSubsequentEvent.Event \sqcap \exists hasTimeInterval^{-1}.Chronicle.$$

4.2. Rules

In the proposed semantic approach, different SWRL rules are used for predicting machinery failures. The launching of these rules allows reasoning over individuals contained in the MPMO ontology. In this subsection, we first introduce SWRL rules which are used to predict the time interval between a certain event and a future failure, and then introduce the algorithm developed for transforming chronicles into SWRL rules. The proposed rules and algorithm enable the semantic approach for automatic failure prediction in the predictive maintenance domain.

4.2.1. Failure Time Prediction Rules

Chronicles provide not only the order of occurrence of events, but also the intervals of time they occur in. As the mining of sequential data sets can generate frequent failure chronicles, SWRL rules can be proposed to reason about temporal information of machinery failures. Therefore, when a new sequence of timestamped events arrive, SWRL rules can be launched to predict the time intervals among different events and future failures.

As stated in Section 4.1, an event within a chronicle is determined by a set of observed properties (with their associated values). Based on this definition, we construct the antecedent of such a rule by describing quantitative values of observed properties (attributes) and the temporal constraints inside a chronicle. The consequent of such a rule comprises the lower and upper bounds of the time intervals among certain events and the failure. Fig. 5 gives an example failure chronicle within which the last event is a failure, which is elicited from [5]. Inside the chronicle, A, B and C are different events. The three events are identified by their associated observed properties and quantitative values. The observed properties and quantitative values are obtained by a feature selection method, that determines the most relevant attributes in predicting the future failures. The last event C indicates a failure, and the time intervals among events A, B with event C gives the temporal information of a future failure (event C).

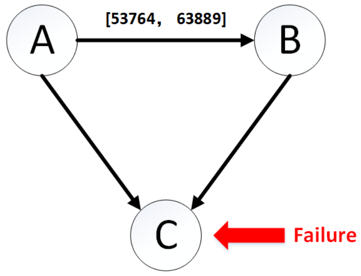

Fig. 5. Example of a chronicle.

Based on this chronicle, a SWRL rule can be elicited. Fig. 6 demonstrates how the rule that describes different events and temporal constraints can be constructed from the chronicle in Fig. 5. Within the rule, *Chronicle* stands for the root class of all the chronicle individuals in the ontology. *hasEvent* is the object property that links individuals of the class *Chronicle* and those under the class *Event*. *hasA1V*, *hasA2V*, *hasA3V*, and *hasA4V* are data properties that assign quantitative values of attributes to the two individuals A and B under the *Event* class. *TimeInterval* corresponds to the root class of all individuals of time intervals. There are two object properties that link *TimeInterval* with *Event*: *hasSubEvent* and *hasProEvent*, among which *hasSubEvent* corresponds to the subsequent event of a time interval, and *hasProEvent* indicates the proceeding event of a time interval. In this case, event A is the proceeding event of the time interval between A and B, and event B is the subsequent event of this time interval. By describing the numerical values of different attributes and the time interval with its proceeding and subsequent events, temporal constraints among events A, B with the failure C are indicated. The temporal constraints comprise the minimum time duration between an event with the failure, described by the data property *hasMinF*, and the maximum time duration between an event with the failure, described by another data property *hasMaxF*.

4.2.2. Automatic Rule Generation Based on Chronicles

To enable the automatic generation of a SWRL rule, in this work we propose an algorithm to transform chronicles into predictive SWRL rules. Algorithm 1 demonstrates the general idea of our rule transformation method. It runs in four major steps.

1. The function *LastNonfailureEvent* extracts the last non-failure event within a chronicle.
2. For each temporal constraint in a chronicle, the two functions *ProceedingEvent* and *SubsequentEvent* extract the proceeding and subsequent events of the time interval that is defined in this temporal constraint. Then the two events and this time interval forms different atoms in the antecedent of the rule, and they are treated as conjunctions.

Chronicle(?c) ^ hasEvent(?c, ?e1) ^ hasEvent(?c, ?e2) ^ hasA1V(?e1, ?v1) ^ swrlb:lessThan(?v1, 5) ^ hasA2V(?e1, ?v2) ^ swrlb:lessThan(?v2, 4) ^ hasA3V(?e2, ?v3) ^ swrlb:greaterThan(?v3, 2) ^ hasA4V(?e2, ?v4) ^ swrlb:lessThan(?v4, 5) ^ TimeInterval(?t) ^ hasSubEvent(?t, ?e2) ^ hasProEvent(?t, ?e1) ^ hasLowerBound(?t, ?lb) ^ swrlb:equal(?lb, 53764) ^ hasUpperBound(?t, ?ub) ^ swrlb:equal(?ub, 63889)
-> hasMinF(?e1, 12117) ^ hasMaxF(?e1, 63881) ^ hasMinF(?e2, 33075) ^ hasMaxF(?e2, 56921)

Fig. 6. Example of a SWRL-based predictive rule, based on the chronicle introduced in Fig. 5.

3. For each last non-failure event before the failure (there could be multiple last events before the failure), extract the temporal constraint between this event and the failure. The extracted temporal constraint is treated as a conjunction with the last event, to form the consequent of the rule.
4. At last, a rule is constructed as an implication between the antecedent and the consequent.

A sequence can be described by one or multiple chronicles. To improve the quality of failure prediction, we only keep the most relevant chronicles for the rule transformation. In this context, we take features of chronicles such as *Chronicle Support* as a reference measure, to select the most relevant chronicles.

5. Experiments

We validate our approach by conducting experimentation on the SECOM data set [24], which contains measurements of features of semi-conductor productions within a semi-conductor manufacturing process. To evaluate the effectiveness of our approach, a software prototype is developed based on Java 10.0.2, Protégé 5.5.0 [15], OWL API [22] and SWRL API [23]. Among them, the OWL API is used to build and manipulate the MPMO ontology. Different types of chronicles are created as individuals within the MPMO ontology, and SWRL-based predictive rules are proposed using the transformation algorithm introduced in Section 4.2.2. To enable ontology reasoning, the SWRL API, which includes a SWRL Rule Engine API, is used to create the transformed rules and then execute them. Within this process, the Drools rule engine [19] is used for rule execution. At last, the inferred knowledge is returned to the OWL API, and stored in the new ontology. The running environment of the software prototype is Microsoft Windows 10.

Algorithm 1 Algorithm to transform a chronicle into a predictive SWRL rule.

Require: $\mathcal{C}_F$: A chonicle model within which the last event is a failure event, $\mathcal{E}$: the episode of $\mathcal{C}_F$ which contains different types of events in a chronicle.
Ensure: R

```
EL ← LastNonfailureEvent(C_F, E)
    ▷ Extract the last non-failure event before the failure within a chronicle.
R ← ∅, A ← ∅, C ← ∅, Atom_a ← ∅, Atom_c ← ∅.
for each e_i[t_ij^-, t_ij^+]e_j ∈ T do
    pe ← ProceedingEvent(e_i[t_ij^-, t_ij^+]e_j)
        ▷ Extract the proceeding event of this time interval
    se ← SubsequentEvent(e_i[t_ij^-, t_ij^+]e_j)
        ▷ Extract the subsequent event of this time interval
    Atom_a ← [t_ij^-, t_ij^+] ∧ pe ∧ se
    A ← Atom_a ∧ ([t_ij^-, t_ij^+] ∧ pe ∧ se)
end for each
for each el ∈ EL do
    ftc ← FailureTimeConstraint(el, TI)
        ▷ Extract the time constraint between the last event before the failure and the failure event.
    Atom_c ← el ∧ ftc
    C ← Atom_c ∧ (el ∧ ftc)
end for each
R ← (A → C)
return R
```

5.1. The SECOM Data Set

In the SECOM data set, 1567 recordings and 590 attributes are collected, with each recording being characterized by a time stamp referring to the time that the data is recorded. Each recording is also associated with a label, which is either 1 or -1. The label of every recording explains the correctness of the event, with -1 corresponding to a non-failure event, and 1 refers to a failure. However, the data contained in SECOM data set do not have the same types of attributes and values, that some of the information contained in the data is irrelevant to the failure prediction task thus is considered as noise. Moreover, due to the inter-dependency among individual features and the complex behavior of combined features, it is difficult to extract frequent patterns and rules based on analysis of all the 590 attributes. Thus, in this context, instead of going through the entire data set and use all 590 attributes for failure prediction, we use feature selection methods [13] to identify and select the most relevant attributes in predicting the failures. The selected attributes are subsequently used to extract the key factors and patterns that lead to machine failures. This reduces the data processing time and memory consumption.

To obtain frequent failure chronicles, we use the frequent chronicle mining approach introduced in [5]. In [5], an industrial data pre-processing method is introduced, including data discretization and sequentialization. Fig. 7 shows different steps within the data mining, especially the frequent chronicle mining approach. The steps presented in Fig. 7 elaborates the data mining procedure which is described in Fig. 3. The approach starts with the aforementioned feature selection, after which a feature subset of the SECOM data set is obtained while retaining a suitably high accuracy in representing the original data set. As a result, 10 most relevant attributes are selected as the optimal subset of all 590 attributes. After the feature selection, data discretization [36] is employed to discretize continuous values for obtaining nominal ones. Thereafter, data sequentialization is used to transform the data into the form of pairs (event, time stamp), where each sequence finishes with a failure. With obtaining sequences that contain failures, CloSpan algorithm [40] is applied to the pre-processed data set, to extract frequent sequential patterns. Also, the frequent chronicle mining algorithm introduced in [5] is used to extract the temporal constraints among these sequential patterns. Up to this step, we are able to obtain frequent failure chronicles that will be transformed into predictive rules.

As introduced in Section 4, to improve the quality of failure prediction, we take *Chronicle Support* as a reference measure, to select the most relevant failure chronicles for failure prediction. As a result, only a subset of all frequent chronicles are used for predictive rule transformation. Table 2 shows the failure chronicles that have the 10 highest chronicle support. We use these chronicles as examples to demonstrate the predictive rule generation approach. In Table 2, each failure chronicle is described by the number of events that it contains, the number of time intervals among events, all the observed properties (attributes) that characterize the failure chronicle, and the chronicle support. For the ease of demonstration, we label the 590 attributes as $A_1, A_2, A_3...A_{590}$.

For an event within a failure chronicle, it is not only identified by a set of attributes, but also the quantitative values of them. To obtain the corresponding quantitative attribute values for describing each event, data discretization has been applied to the SECOM data set. After data discretization, the quantitative data has been translated into qualitative data. Also, an association between each numerical value and a certain interval has been created. Taking the failure chronicle $\mathcal{C}_{F5}$ in Table 2 as an example, Fig. 8 shows the graphical view of it, and the numerical intervals for describing the events within this failure chronicle are shown in Table 3. The temporal constraints in Fig. 8 are in the unit in millisecond.

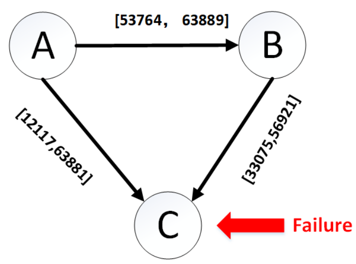

Fig. 8. The graphical view of failure chronicle $\mathcal{C}_{F5}$.

Based on the descriptions of the failure chronicle $\mathcal{C}_{F5}$, we use the algorithm introduced in Section 4.2.2 to generate a SWRL-based predictive rule automatically. The result of this rule generation is shown in Fig. 9. In this rule, *hasA58V*, *hasA63V*, *hasA64V*, *hasA102V*, *hasA204V*, *hasA209V*, *hasA347V*, *hasA476V* are data properties in the MPMO ontology that link individuals of the *Event* class with XML Schema Datatype values. They correspond to the quantitative values of the attributes $A_{58}, A_{63}, A_{64}, A_{102}, A_{204}, A_{209}$, A_{347}, and A_{476} in the SECOM data set. To describe the numerical intervals which are obtained by discretization, SWRL Built-Ins are used to specify the upper and lower numerical boundaries. The consequent of this rule comprises the temporal constraints among *Events* *A*, *B* and *C*. The minimum time duration between an event with the failure is described by the data property *hasMinF*, and the maximum time duration between an event with the failure is described by another data property *hasMaxF*. By this way, the temporal con-

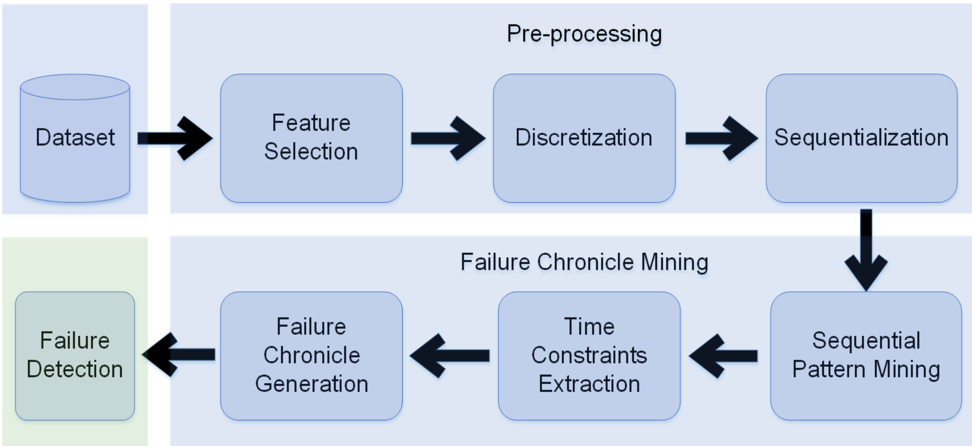

Fig. 7. Different steps used in the frequent failure chronicle mining approach, adapted from [5].

Table 2
Extracted failure chronicles that have the highest 10 chronicle support.

Failure Chronicle	Number of Events	Number of Time Intervals	Attributes	Chronicle Support
$\mathcal{C}_{F1}$	3	3	$A_{63}, A_{64}, A_{102}, A_{204}, A_{209}, A_{476}$	83.65%
$\mathcal{C}_{F2}$	3	3	$A_{63}, A_{64}, A_{102}, A_{204}, A_{209}, A_{347}, A_{476}$	82.69%
$\mathcal{C}_{F3}$	3	3	$A_{58}, A_{64}, A_{102}, A_{204}, A_{209}, A_{476}$	82.69%
$\mathcal{C}_{F4}$	3	3	$A_{58}, A_{63}, A_{102}, A_{204}, A_{209}, A_{347}$	81.73%
$\mathcal{C}_{F5}$	3	3	$A_{58}, A_{63}, A_{64}, A_{102}, A_{204}, A_{209}, A_{347}, A_{476}$	81.73%
$\mathcal{C}_{F6}$	3	3	$A_{58}, A_{102}, A_{204}, A_{209}, A_{347}, A_{476}$	80.77%
$\mathcal{C}_{F7}$	3	3	$A_{58}, A_{204}, A_{209}, A_{347}, A_{476}$	80.77%
$\mathcal{C}_{F8}$	4	4	$A_{63}, A_{64}, A_{102}, A_{204}, A_{209}, A_{347}, A_{476}$	78.84%
$\mathcal{C}_{F9}$	4	4	$A_{58}, A_{63}, A_{102}, A_{204}, A_{209}, A_{347}$	78.84%
$\mathcal{C}_{F10}$	4	4	$A_{58}, A_{204}, A_{209}, A_{347}, A_{476}$	78.84%

Table 3
Attributes with their numerical intervals within the failure chronicle $\mathcal{C}_{F5}$.

Event	Attribute	Numerical Value Interval
A	63	[89.2564, 94.8757)
A	204	[4925.1678, 4999.2456)
A	209	[20.1884, 23.0750)
A	347	[6.4877, 6.9573)
A	476	[125.1988, 137.4435)
B	58	[4.5537, 4.8994)
B	63	[89.3158, 94.8757)
B	64	[90.0196, 94.3934)
B	102	[-0.1188, 0.5231)
B	347	[6.2446, 6.9574)

straints of a future failure is inferred by the launching of such a predictive SWRL rule. This rule is an instantiation of the generic rule introduced in Fig. 6.

5.2. Results Evaluation

To evaluate the usefulness and effectiveness of our approach, we conduct results evaluation from two perspectives: i) the evaluation of the MPMO ontology;

```
Semi-conductorManufacturingProcess(?s) ^ Chronicle(?c) ^ isLearnedFrom(?c, ?s) ^
hasEvent(?c, ?e1) ^ hasEvent(?c, ?e2) ^ hasA63V(?e1, ?v1) ^ swrlb:lessThan (?v1, 94.8757) ^
swrlb:greaterThanOrEqual(?v1, 89.2564) ^ hasA204V(?e1, ?v2) ^ swrlb:lessThan (?v2, 4999.2456)
^ swrlb:greaterThanOrEqual(?v2, 4925.1678) ^ hasA209V(?e1, ?v3) ^ swrlb:lessThan (?v3,
23.0750) ^ swrlb:greaterThanOrEqual(?v3, 20.1884) ^ hasA347V(?e1, ?v4) ^ swrlb:lessThan (?v4,
6.9573) ^ swrlb:greaterThanOrEqual(?v4, 6.4877) ^ hasA476V(?e1, ?v5) ^ swrlb:lessThan (?v5,
137.4435) ^ swrlb:greaterThanOrEqual(?v5, 125.1988) ^ hasA58V(?e2, ?v6) ^
swrlb:greaterThanOrEqual(?v6, 4.5337) ^ swrlb:lessThan (?v6, 4.8994) ^ hasA63V(?e2, ?v7) ^
swrlb:greaterThanOrEqual(?v7, 89.3158) ^ swrlb:lessThan (?v7, 94.8757) ^ hasA64V(?e2, ?v8) ^
swrlb:lessThan(?v8, 94.3934) ^ swrlb:greaterThanOrEqual(?v8, 90.0196) ^ hasA102V(?e2, ?v9) ^
swrlb:lessThan(?v9, 0.5231) ^ swrlb:greaterThanOrEqual(?v9, -0.1188) ^ hasA347V(?e2, ?v10) ^
swrlb:lessThan (?v10, 6.9574) ^ swrlb:greaterThanOrEqual(?v10, 6.2446) ^ TimeInterval(?t) ^
hasSubEvent(?t, ?e2) ^ hasProEvent(?t, ?e1) ^ hasLowerBound(?t, ?lb) ^ swrlb:equal(?lb, 53764) ^
hasUpperBound(?t, ?ub) ^ swrlb:equal(?ub, 63889)
-> hasMinF(?e1, 12117) ^ hasMaxF(?e1, 63881) ^ hasMinF(?e2, 33075) ^ hasMaxF(?e2, 56921)
```

Fig. 9. The SWRL-based predictive rule transformed from the failure Chronicle C_{F5}.

and ii) the evaluation of the SWRL rule-based failure prediction results. It should be noted that for evaluation we focus on the quality of semantic enrichment to the chronicle mining results, and the evaluation of the performance of the chronicle mining phase is out of the scope of this paper.

5.2.1. *Evaluation of the MPMO Ontology*

Ontology evaluation enables users to assess the quality of ontologies. It is essential for the wide adoption of ontologies, since ontologies can be shared and reused by different users, and the quality of ontologies such as the consistency, completeness, and conciseness of taxonomies are key considerations when different users reuse ontologies in specific contexts. In this paper, to evaluate the quality of the proposed MPMO ontology, we use OOPS!, which is an online ontology evaluation tool [25]. The reason we choose this tool for ontology evaluation is two-fold. Firstly, OOPS! allows automatic detection of common pitfalls in ontologies, and the detection of pitfalls can be executed independently of the ontology development software and platforms. Secondly, it enlarges the list of errors that can be detected by most recent ontology evaluation tools, thus providing a broader scope of anomaly detection in ontologies [25].

In OOPS!, ontology pitfalls are classified into three categories: structural, functional, and usability-profiling. Under each category, fine-grained classification criteria is provided to cope with specific types of anomalies. In general, these three categories can be described as follows [25]:

- Structural dimension: It focuses on anomaly detection on syntax and formal semantics. Since the MPMO ontology consists of logical axioms, the syntax and logical consistency can be evaluated and validated through anomaly detection within this category. To be more specific, This category is composed of five criteria: i) modeling decisions, which evaluates whether users use the ontology implementation language in a correct way; ii) real world modeling or common sense, which evaluates the completeness of the domain knowledge formalized by the MPMO ontology; iii) no inference, which checks whether the desired knowledge can be inferred through ontology reasoning; iv) wrong inference, which refers to the detection of inference that lead to erroneous or invalid knowledge; and v) ontology language, which assesses the correctness of the ontology development language of the MPMO ontology.
- Functional dimension: It considers the intended use and functionality of the MPMO ontology. Under this category, two specific criteria are used to evaluate the MPMO ontology: i) requirement completeness, which evaluates coverage of the domain knowledge that is formalized by the MPMO ontology; ii) application context, which

evaluates the adequacy of the MPMO ontology for a given use case or application.

– Usability-profiling dimension: It evaluates the level of ease of communication when different groups of users use the same ontology. Within this category, two specific criteria are applied for ontology evaluation: i) ontology understanding, which evaluates the quality of information or knowledge that is provided to users for easing the understanding of the ontology; ii) ontology clarity, which assesses the quality of ontology elements for being easily recognized and understood by users. These criteria is commonly used to check the quality of ontologies when users do not have sufficient domain knowledge.

To evaluate the MPMO ontology according to the aforementioned categories, we uploaded the ontology code to the OOPS! online tool. After loading the ontology code, the ontology pitfall scanner is used to check the pitfalls that exist in the MPMO ontology. Fig. 10 shows the evaluation result. The result shows that our ontology is free of bad practices in the structural, functional, and usability-profiling dimensions of evaluation. Moreover, the MPMO ontology is developed and formalized using OWL, which is a widely used language for knowledge representation and ontology development. This eases the reuse of the MPMO ontology in other contexts and also simplifies the integration of the MPMO ontology with other knowledge components that are developed with the same language.

5.2.2. *Evaluation of the SWRL Rule-based Failure Prediction Results*

To evaluate the quality of the SWRL rule-based failure prediction results, we apply the SWRL rules on the sequences in the SECOM data set, and three measures are used to assess the quality of these rules: the *True Positive Rate*, the *Precision* of failure prediction, and the *F-measure*. The equations for computing these three measures are shown in Equation 1, 2 and 3.

$$\frac{TP}{TP + FN} \tag{1}$$

$$\frac{TP}{TP + FP} \tag{2}$$

$$\frac{2TP}{2TP + FP + FN} . \tag{3}$$

Among them, the *True Positive Rate* aims to measure the percentage of positive sequences that have been correctly classified. In Equation 1, TP (True Positive) is the true positive results standing for the number of valid sequences that at least one SWRL rule could predict the failures in these sequences, and FN (False Negative) is the false negative results which stand for the number of sequences that no SWRL rule could predict the failures in these sequences.

The *Precision* of failure prediction measures the percentage of sequences based on which the SWRL rules are constructed correctly. For a given sequence, failure chronicles are extracted through chronicle mining and SWRL rules are constructed for failure prediction. After applying the SWRL rules, if the predicted failure temporal constraints are out of the range of the failure occurrence time intervals in the sequence, then it indicates that the SWRL rules could not predict the temporal constraints of the failure in this sequence. Thus, the failure is classified as *False Positive*. In Equation 2, TP (True Positive) is the true positive results standing for the number of valid sequences that at least one SWRL rule could predict the failures in these sequences, and FP (False Positive) is the number of sequences for which the SWRL rules incorrectly predict the temporal constraints of the future failures.

With obtaining the above two measures, we can compute the *F-measure* according to the Equation 3.

Table 4 shows the experimental results of the three measures. The three measures are computed according to different frequency thresholds of sequences in the data set. We use ft_{min} to denote the minimum frequency threshold of a sequence in the data set.

We can see from Table 4 that all computed values for the three measures are above 80%, which shows the results are encouraging. As the minimum frequency threshold ft_{min} values decreases, the values of three measures show an increase tendency. This can be explained as follows: as ft_{min} increases, the number of extracted chronicles decreases, which lead to the decrease of the number of transformed SWRL rules. For this reason, each sequence for testing is less likely to be validated by the transformed SWRL rules.

Since the SWRL rules are generated from chronicle mining results, the quality of their prediction exclusively depend on the mined frequent chronicles. In this context, the 10-fold cross validation principle [26] is used to evaluate the quality of failure prediction. To apply the 10-fold cross validation principle, the SECOM data set is partitioned into two parts: the training set and the test set. Firstly, chronicles are extracted from

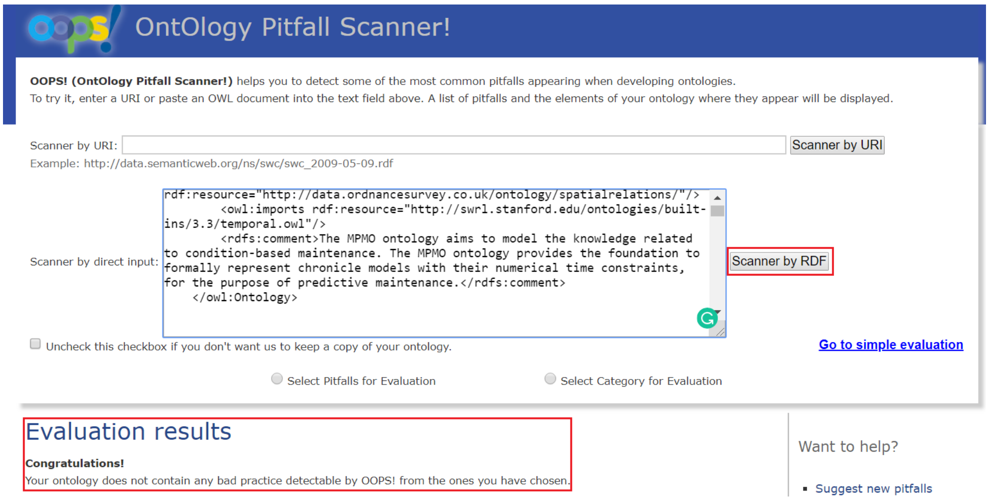

Fig. 10. Screenshot of the ontology evaluation result using OOPS! online tool.

Table 4
True Positive Rate, Precision and F-measure of Failure Prediction Based on SWRL Rules.

ft_{min}	True Positive Rate	Precision	F-measure
1	83.63% ±6.43%	84.62% ±6.55%	86.55% ±4.89%
0.9	85.45% ±4.98%	87.49% ±6.16%	88.54% ±6.06%
0.8	87.27% ±7.50%	84.58% ±6.55%	85.71% ±6.98%
0.7	89.09% ±6.68%	86.22% ±6.43%	87.52% ±6.51%
0.6	90.90% ±7.93%	88.71% ±5.26%	89.21% ±5.43%
0.5	90.90% ±7.93%	86.83% ±4.41%	87.88% ±5.77%

the training sequences in the training set. Then, for the test set, we check for each sequence, its membership in at least one chronicle among those extracted. The number of sequences validated by the chronicles is computed to estimate its percentage with respect to the sequence set. This procedure is repeated 10 times to validate all the sequences of the database.

The launching of such a set of SWRL-based predictive rules enables the prediction of temporal constraints of future machinery failures. This allows users to take further maintenance actions, such as the replacement of the machine tools used on the production line. The performance of failure prediction could be enhanced by considering a new set of rules that reason about the severity levels of failures. We are currently applying machine learning techniques to classify the severity levels of failures, according to the temporal constraints among the failures and other events.

6. Conclusion and Future Perspectives

This paper demonstrates a novel hybrid approach for implementing predictive maintenance in industry. The proposed hybrid approach is a combination of frequent chronicle mining and semantics, within which chronicle mining is used to extract frequent chronicles based on industrial data sets, and a knowledge-based structure is used to automate the SWRL rule generation process and to formalize the predictive maintenance results.

The contributions of this paper are three-fold. Firstly, chronicles are formally represented with the use of ontologies, by which the main concepts and relationships for describing chronicles are formalized, then easing the knowledge representation and interpretation of frequent chronicle mining results. Secondly, a novel algorithm for transforming chronicles into SWRL-based predictive rules is introduced. The novel algorithm allows the automatic generation of SWRL rules based on the mined frequent chronicles, thus enabling an automatic semantic approach for predictive maintenance.

Thirdly, the reasoning about temporal constraints of future machinery failures is enabled by the joint use of data mining and semantics, which allows the implementation of maintenance actions such as alarm launching.

However, there are two major problems that need to be solved. The first problem is the partition method of numerical values. Since the rules we proposed in Section 5 are based on crisp logic, when the numeric values of attributes collected by sensors are considerably close to partition thresholds, the rules proposed in Section 5 may fail to partition these numeric values into correct categories. To deal with such kind of uncertainty situations, the use of fuzzy logic should be considered and a fuzzy semantic approach needs to be implemented. This approach will use machine learning techniques to automatically derive membership functions and fuzzy if-then rules from data sets. The fuzzy rules aim to enhance the representation of imprecise severity level of machinery failures. For example, an identification of failure will be associated with a fuzzy index, indicating the grade of its membership to a "low" or "high" level of failure. The fuzzy approach will be applied to tackle the challenge of symbol anchoring problem [34].

The second problem is the evolution of the ontology and the rule base. Since the manufacturing domain is highly-dynamic, the predictive maintenance system should be able to adapt itself to dynamic situations over time, for example, the change of context. Also, when the system fails to provide satisfactory results through launching the rules, it is required to consult domain experts for decisions about failure prediction and maintenance. In this situation, the domain experts use their expertise and experience to assess the current state of the system and provide appropriate decisions. For example, when the temperature measured by a sensor located at a cutting tool exceeds its threshold and no rule in the rule base is able to warn about his abnormal condition, domain experts can use their experience and expertise to identify this abnormal condition and provide possible solutions in order to avoid the production line to produce unqualified products. In this way, new rules which capitalize experts' experience needs to be proposed to update the initial set of rules in the rule base, in order to facilitate the quality of failure prediction. In this context, when the next time a similar situation needs to be addressed, the rule which capitalizes domain experts' experience will be launched together with the initial rules to identify potential failures and to make predictions. This requires the ontology and the rule base to be capable of coping with the dynamic change of knowledge. To deal with this issue, knowledge base evolution solutions should be proposed: The ontology should be able to adapt itself efficiently to the changes with using ontology evolution techniques, and the rule base should be updated according to the change of context, by implementing contextual reasoning.

Acknowledgements

This work has received funding from INTERREG Upper Rhine (European Regional Development Fund) and the Ministries for Research of Baden-Wrttemberg, Rheinland-Pfalz (Germany) and from the Grand Est French Region in the framework of the Science Offensive Upper Rhine HALFBACK project.

References

[1] A. Avizienis, J.C. Laprie, B. Randell, and C. Landwehr, Basic concepts and taxonomy of dependable and secure computing. *IEEE transactions on dependable and secure computing*, 2004, **1**(1), pp. 11-33.

[2] A. Grall, L. Dieulle, C. Bérenguer, and M. Roussignol, Continuous-time predictive-maintenance scheduling for a deteriorating system, *IEEE transactions on reliability*, 2002, **51**(2), pp. 141-150.

[3] B. Vogel-Heuser, and D. Hess, Guest editorial Industry 4.0-prerequisites and visions. *IEEE Transactions on Automation Science and Engineering*, 2016, **13**(2), pp. 411-413.

[4] C. Dousson, and T.V.Duong, Discovering Chronicles with Numerical Time Constraints from Alarm Logs for Monitoring Dynamic Systems. In *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*, 1999, pp. 620-626.

[5] C. Sellami, A. Samet, M. Tobji, Frequent Chronicle Mining: Application on Predictive Maintenance. In *Proceedings of the 17th IEEE International Conference On Machine Learning And Applications*, 2018.

[6] D. Cram, B. Mathern, and A. Mille, A complete chronicle discovery approach: application to activity analysis. *Expert Systems*, 2012, **29**(4), pp. 321-346.

[7] D. Dou, H. Wang, and H. Liu, Semantic data mining: A survey of ontology-based approaches. In *IEEE International Conference on Semantic Computing (ICSC)*, 2015, pp. 244-251.

[8] D.L. McGuinness, and F. Van Harmelen, OWL web ontology language overview. *W3C recommendation*, 2004, **10**(10).

[9] D.L. Nuez, and M. Borsato, An ontology-based model for prognostics and health management of machines, *Journal of Industrial Information Integration*, 2017, **6**, pp. 33-46.

[10] E. Maleki, F. Belkadi, N. Boli, B.J. van der Zwaag, K. Alexopoulos, S. Koukas, M. Marin-Perianu, A. Bernard, and D. Mourtzis, Ontology-based framework enabling smart Product-Service Systems: Application of sensing systems for machine health monitoring, *IEEE Internet of Things Journal*, 2018.

[11] F. Ameri, and C. McArthur, Semantic rule modelling for intelligent supplier discovery. *International Journal of Computer Integrated Manufacturing*, 2014, **27**(6), pp. 570–590.

[12] F. Ameri, and D. Dutta, An Upper Ontology for Manufacturing Service Description, *ASME Conference Proceedings*, 2006, pp. 651-661.

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

[14] I. Horrocks, P.F. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, and M. Dean, SWRL: A semantic web rule language combining OWL and RuleML. *W3C Member submission*, 2004, **21**, pp. 79.

[15] J.H. Gennari, M.A. Musen, R.W. Fergerson, W.E. Grosso, M. Crubézy, H. Eriksson, N.F. Noy, and S.W. Tu, The evolution of Protégé 5.5.0: an environment for knowledge-based systems development. *International Journal of Human-computer studies*, 2003, **58**(1), pp. 89-123.

[16] J. Pei, J. Han, and W. Wang, Mining sequential patterns with constraints in large databases, In: *Proceedings of the eleventh international conference on Information and knowledge management*, 2002, pp. 18-25.

[17] J.R. Hobbs, and F. Pan, Time ontology in OWL, *W3C working draft*, 2006, **27**, pp. 133.

[18] L.N. Soldatova, and R.D. King, An ontology of scientific experiments. *Journal of the Royal Society Interface*, 2006, **3**(11), pp. 795-803.

[19] M. Bali, Drools JBoss Rules 5.0 Developer's Guide, Packt Publishing Ltd 2009.

[20] M. Cannataro, and C. Comito, A data mining ontology for grid programming. In: *Proceedings of (SemPGrid2003)*, 2003, pp. 113-134.

[21] M. Grninger,Ontology of the process specification language, *Handbook on ontologies*, Springer, Berlin, Heidelberg, 2004, pp. 575-592.

[22] M. Horridge, and S. Bechhofer, The owl api: A java api for owl ontologies, *Semantic Web*, 2011, **2**(1), pp. 11-21.

[23] M.J. O'Connor, R.D. Shankar, C. Nyulas, A.K. Das, M.A. Musen, The SWRLAPI: A Development Environment for Working with SWRL Rules, *OWL: Experiences and Directions (OWLED), 4th International Workshop*, 2008.

[24] M. McCann, and A. Johnston, SECOM data set, 2008, *UCI Machine Learning Repository*.

[25] M. Poveda-Villalón, A. Gómez-Pérez, and M.C. Suárez-Figueroa, Oops!(ontology pitfall scanner!): An on-line tool for ontology evaluation. *International Journal on Semantic Web and Information Systems (IJSWIS)*, 2014, **10**(2), pp. 7-34.

[26] M. Stone, Cross-validatory choice and assessment of statistical predictions, *Journal of the royal statistical society, Series B (Methodological)* 2974, pp. 111-147.

[27] M. Yoshida, T. Iizuka, H. Shiohara, and M. Ishiguro, Mining sequential patterns including time intervals, In *Data Mining and Knowledge Discovery: Theory, Tools, and Technology II*, 2000, pp. 213-221.

[28] P. Fournier-Viger, J.C-W. Lin,R.U. Kiran, Y.S. Koh, R. Thomas, A survey of sequential pattern mining, *Data Science and Pattern Recognition*, 2017, **1(1)**, pp. 54-77.

[29] P.N. Tan, Introduction to data mining, *Pearson Education India*, 2007.

[30] P. Panov, L. Soldatova, and S. Džeroski, Ontology of core data mining entities. *Data Mining and Knowledge Discovery*, 2014, **28**(5-6), pp. 1222-1265.

[31] P. Ristoski, and H. Paulheim, Semantic Web in data mining and knowledge discovery: A comprehensive survey, *Web semantics: science, services and agents on the World Wide Web*, 2016, **36**, pp. 1-22.

[32] Q. Cao, F. Giustozzi, C. Zanni-Merk, F. de Bertrand de Beuvron, and C. Reich, Smart Condition Monitoring for Industry 4.0 Manufacturing Processes: An Ontology-Based Approach. *Cybernetics and Systems*, 2019, pp. 1-15.

[33] R. Agarwal, and R. Srikant, Fast algorithms for mining association rules, In: *Proceeding of the 20th VLDB Conference*, 1994, pp. 487-499.

[34] S. Coradeschi, and A. Saffiotti, An introduction to the anchoring problem, *Robotics and Autonomous Systems*, 2003, **43**(2-3), pp. 85-96.

[35] S. Lemaignan, A. Siadat, J.Y.Dantan, and A. Semenenko, MASON: A proposal for an ontology of manufacturing domain, In *IEEE Workshop on Distributed Intelligent Systems*, 2006, pp. 195-200.

[36] S. Ramírez-Gallego, S. García, H. Mouriño-Talín, D. Martínez-Rego, V. Bolón-Cane., S. Alonso-Betanzo, J.M. Benítez and F. Herrera, Data discretization: taxonomy and big data challenge, *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 2016, **6**(1), pp. 5-21.

[37] T.R. Gruber, A translation approach to portable ontology specifications, *Knowledge acquisition*, 1993, **5**(2), pp. 199-220.

[38] T.Slimani, and A. Lazzez, Sequential Mining: Patterns and Algorithm Analysis. *International Journal of Computer and Electronics Research*, 2013.

[39] V. Chandola, A. Banerjee, and V. Kumar, Anomaly detection: A survey, *ACM computing surveys (CSUR)*, 2009, **41**(3), pp. 15.

[40] X. Yan, J. Han, and R. Afshar, Clospan: Mining closed sequential patterns in large datasets, In *Proceedings of the 2003 SIAM international conference on data mining*, 2003, pp. 166-177.

[41] Y. Hirate, and H. Yamana, Generalized SPM with Item Intervals. *Journal of Computers*, 2006, **1**(3), pp. 51-60.

[42] Z. Usman, R.I.M.Young, N. Chungoora, C. Palmer, K.Case, and J.A. Harding, Towards a formal manufacturing reference ontology. *International Journal of Production Research*, 2013, **51**(22), pp. 6553-6572.