A Method for Identifying the Timeliness of Manufacturing Data Based on Weighted Timeliness Graph

Zehua Liu[1,2], Xuefeng Ding[1,2], Yuming Jiang[1,2], and Dasha Hu[1,2(✉)]

[1] College of Computer Science, Sichuan University, Chengdu 610065, China
hudasha@scu.edu.cn

[2] Big Data Analysis and Fusion Application Technology Engineering Laboratory of Sichuan Province, Chengdu 610065, China

Abstract. Timeliness is one of the important indicators of data quality. In industrial production processes, a large amount of dependent data is generated, often resulting in unclear timestamps. Therefore, this article combines the conclusion dependency graph into a process dependency graph to determine the identification order of the timeliness of each process data; By constructing a weighted timeliness graph (WTG) and path single flux, a data timeliness identification method that does not completely rely on timestamps is proposed. Finally, a time-effectiveness identification method based on weighted time-effectiveness graph was discussed through an example and 9 dependency rules, and the effectiveness of the method was verified through a set of experiments.

Keywords: Manufacturing Big Data · Quality Appraisal · Dependency Rules · Data Timeliness

1 Introduction

A large number of sensors will collect various types of data in industrial production processes. Before using data, it is necessary to evaluate its quality, and the timeliness of data is an important indicator of data quality [1]. Therefore, a time dependent method for identifying timeliness is proposed. This method can effectively identify the timeliness of data when the timestamp is incomplete. The prerequisite for assessing the timeliness of data is that the data is correct and can meet work needs. Compared to the correctness of data, the timeliness of data does not necessarily need to be tested in the real world [2]. Therefore, the measurement of data timeliness should be an estimate, not a validation statement under certainty, to determine the probability of data validity [3]. For a large amount of data, it is reasonable to quantify timeliness through this estimation when the validity of the data is unclear [4].

Reference [5] proposes a probability base metric for calculating the timeliness of Wiki articles related to timeliness events. Reference [6] defines a recurrent timeliness rules (RTR) to evaluate the timeliness of periodic data generated during the manufacturing process. Reference [7] define a measure as a function that depends on the age of the

X. Yang et al. (Eds.): ADMA 2023, LNAI 14176, pp. 64–72, 2023.
https://doi.org/10.1007/978-3-031-46661-8_5

attribute value at the time the currency is evaluated and the sensitivity parameters that make the measure suitable for the application environment. Reference [8] assumes that timestamps are available and mentions using the current time variable to represent the latest time. Reference [9] developed an uncertain constraint database scheme based on the constraint database scheme, and abstracted the important example of the complexity of query identification in uncertain time constraint databases. Reference [10] was used to investigate the work on time constraint satisfaction problem (TCSP) by finding estimates that satisfy a set of time variables with time constraints. Reference [11] first studied the use of rules $\forall t_1, \cdots, t_j : R(t_1[EID] = t_j[EID] \wedge \varphi \rightarrow t_u \prec_A t_v), j \in [1, k]$ to help identify the timeliness of data when there is no clear timestamp in the database. Document [12] proposed a dynamic functional dependency relationship, which stipulates that a copy function can require some attributes that cannot be independently changed to be copied together. It is necessary to judge the time series relationship describing different attribute values of the same entity based on a small amount of time series rules obtained from domain knowledge, so as to identify which values are outdated. The disadvantage is that the current method cannot identify whether a value is outdated or invalid at a given time point.

In summary, the current research on data timeliness is not suitable for direct application to the identification of industrial data timeliness. There are two main reasons: firstly, it is unable to effectively provide the timeliness of the current data without clear timestamps. Secondly, it is impossible to quantify the timeliness of data at a given time point. Therefore, this article proposes a method for evaluating the timeliness of manufacturing data based on weighted timeliness maps, which is used to identify the timeliness of industrial data through time dependency relationships without a clear timestamp.

2 Data Timeliness Identification Method

This article presents the dependency relationships between various conclusions by constructing a conclusion dependency sequence diagram. Then, merge the conclusion dependency graph into a process dependency sequence graph to determine the calculation order of the timeliness of each process data. Finally, the timeliness of data in a certain process is determined through a weighted timeliness graph.

2.1 Limited Timeliness Dependency Rules

Time dependent rules refer to the identification of the timeliness of production data through the dependency relationships between processes.

Definition 1. (Limited timeliness dependency rule): In Rule r, there is a dependency relationship between each process, and the data generated in Process A will be limited by its related processes. The degree of limitation is quantified by the strength of the dependency. The rules that have limited dependencies between these process data are defined as limited time dependency rules, and their dependencies can be represented as:

$$r : \forall t_i, t_j(\psi_B \rightarrow t_i \prec_A t_j, \beta) \tag{1}$$

Among them, $t_i \prec_A t_j$ represents the generation of date t_i before the generation of date t_j, β It is the dependency strength of the rule, ψ Represents the temporal relationship or constraint conditions in process B. The dependency strength of rule represents the likelihood of the rule represented by r. Because in actual industrial production processes, each process or employee is relatively independent, probability can be used to represent the relationship between different process data.

Let lhs(r) represents the left part of the rule, and rhs(r) represents the right part of the rule, that is, in formula (1), $lhs(r) = \psi$, $rhs(r) = t_i \prec_A t_j$. For the right part of the rule, $rhs(r) = tml(lhs(r)) \times tml(r)$, where the dependency strength calculation of the left part $lhs(r) = \psi$ follows the following rule:(a). If ψ is the condition $t_i[A]opt_j[A]$ or $t_k[A]opa$ is determined, then if ψ is true, then $tml(\psi) = 1$ is satisfied, otherwise $tml(\psi) = 0$, where $op \in \{=, \neq, <, >, \leq, \geq\}$ and a are constants; (b). If ψ is $t_i \prec_A t_j$ or $t_k \prec_A \tau$ and ψ is not the right part of any rule r, then the value of $tml(\psi)$ is obtained statistically; (c). If ψ is the right part of other rules $r\prime$, the value of $tml(\psi)$ is obtained by the calculation method of the right part of rules $r\prime$ dependence strength; (d). If $\psi = \psi_1 \wedge \psi_2$, then ψ represents the conjunction of ψ_1 and ψ_2, and the dependence strength $tml(\psi) = min\{tml(\psi_1), tml(\psi_2)\}$ after the conjunction.

2.2 Process Dependency Sequence Graph

To determine the calculation order of the timeliness of each process data, it is necessary to construct a process dependency graph. To construct a process dependency graph, it is first necessary to construct a conclusion dependency graph based on dependency rules, and then merge them. The specific steps for constructing a dependency sequence diagram are as follows. Firstly, based on the type of conclusion, determine whether the conclusion needs to be included in the dependency sequence diagram. The conclusion of $t_i[A]opt_j[A]$ or $t_k[A]opa$ is a deterministic conclusion that satisfies the condition of 1, otherwise it is 0 and does not need to be added to the conclusion dependency diagram; The conclusions of $t_i \prec_A t_j$ and $t_i \prec_A \tau$ are non-deterministic and need to be added to the conclusion dependency sequence diagram. Then, when the conclusion is determined to be non-deterministic, two types of nodes, $(\prec, A, *, *)$ and $(\prec, A, \tau)$, are constructed for each process A to represent the conclusions of type $t_i \prec_A t_j$ and type $t_i \prec_A \tau$. Finally, scan the rule set $\Sigma(r)$, is there a rule r_k that makes $\psi_1 = lhs(r_k)$, $\psi_2 = rhs(r_k)$? If so, add directed edges from ψ_1 to ψ_2 to the dependency order graph.

The overall principle for merging conclusion dependency graphs into process dependency graphs is: (a). Merge conclusion nodes containing multiple identical processes in the diagram into one process node; (b). The directed edge no longer represents the dependency relationship between conclusions, but rather represents the dependency relationship between various processes.

2.3 Weighted Timeliness Graph

To evaluate the timeliness of all data items in process A, it is necessary to determine the values of $tml(t_i \prec_A t_j)$ and $tml(t_i \prec_A \tau)$ separately. Therefore, weighted timelines graph (WTG) has been defined.

Definition 2. (weighted timeliness graph, WTG): Let D represent a set of data, Σ Represents a collection of time dependent rules, T representative Σ All sets containing time in the time rule, where A represents a certain process, θ Is the effective time threshold of D, and the weighted time graph of process A is WTG_A, which is defined as:

1. WTG_A contains data items t and time τ in two types of conclusions $t_i \prec_A t_j$ and $t_i \prec_A \tau$, and both take a certain time τ as the initial node of the WTG_A graph, where τ includes the original time in the dataset and the threshold we set θ;
2. aggregate Σ Each rule r in is defined as the initial node of the WTG_A graph if $\tau \in T \cup \{\theta\}$ causes $\tau \subset rhs(r)$;
3. aggregate Σ For each rule r in, if $t_1, t_2 \in D$ causes $rhs(r) = t_1 \prec_A t_2$, $tml(lhs(r)) > 0$ or $tml(r) > 0$, update the weight of the directed edge from t_2 to t_1;
4. The weight of the directed edge t_1, t_2 in the WTG_A graph is denoted as $weight(t_1, t_2) = tml(r) \times tml(lhs(r))$.

The implementation details of constructing a weighted timeliness graph are shown in Algorithm 1.

Algorithm 1: Weighted timeliness graph construction algorithm

Input: Dataset D, Rule Set Σ, Process A, designated time θ

Output: Weighted timeliness graph

01. $V = new\ ArrayList(); E = new\ ArrayList(); visited(r) = 0$// Initialization.
02. for τ in T: // Scan time set, whether there is a specified time.
03. if $\tau_k = \theta$:
04. $createNode(\tau_k)$
05. for r in Σ:
06. if $visited(r) = 0$:
07. if $rhs(r_x) = t \prec_A \tau_k$: // Scan Rule Collection Σ. Is it a conclusion containing time τ.
08. for t in D:
09. if $tml(lhs(r_x(t_k))) \neq 0$:
10. $V.add(createNode(t_k)); \omega_y = weight(\tau_k, t_k); E.add(createEdge(\tau_k, t_k, \omega))$
11. $visited(r_x) = 1$ //Set the access location of rule r to 1.
12. if $rhs(r_x) = t_i \prec_A t_j$:
13. for t_i, t_j in D:
14. if $tml\ (lhs(r_x(t_i, t_j))) \neq 0$:
15. $V.add(createNode(t_i)); V.add(createNode(t_j))$
16. $\omega_y = weight(t_j, t_i); E.add(createEdge(t_j, t_i, \omega_y))$
17. $visited(r_x) = 1$
18. else: // If the rule r has already been accessed.
19. $\alpha, \beta = ExtractVertex(rhs(r_x)); \omega' = weight(\alpha, \beta)$
20. $\omega = Max\{\omega', \omega_y\}; E.updateEdge(\beta, \alpha, \omega)$//

2.4 Path Single Flux of Weighted Timeliness Graph

The timeliness evaluation of data item $t_i[A]$ requires determining the possibility of conclusion $t_i \prec_A \tau$ being established, that is, the value of $tml(t_i \prec_A \tau)$. The possibility of conclusion $t_i \prec_A \tau$ can be obtained by immediate inference of a rule r_k or there is a directed path $Path(\tau, t_i)$ from τ to t_i in the weighted time effect graph.

Definition 3. (Path single flux of weighted timeliness graph): The single flux of path $Path(\tau, t_i)$ in the WTG_A graph is denoted as $sFlux(Path(\tau, t_i))$, which is defined as the weight of the maximum directed edge that can flow in a single path from τ to t_i.

The weighted time effect graph path single flux can be mainly divided into two meanings:

1. (a). In the weighted efficiency graph of process A, when there is only one path from τ to t_i in path $Path(\tau, t_i)$, we take the directed edge with the highest weight in the path as the flux of this path. Namely, $Flux(Path(\tau, t_i)) = min\{weight(t_i \prec_A v_1), weight(v_1 \prec_A v_2), \cdots, weight(v_k \prec_A \tau)\}$.
2. (b). In the weighted efficiency chart of process A, when there are multiple paths from τ to t_i in path $Path(\tau, t_i)$, we take the maximum flux value of the multiple paths as the single flux of path $Path(\tau, t_i)$ in the graph WTG_A. Namely, $sFlux(Path) = Max\{Flux(Path_1), Flux(Path_2), \cdots, Flux(Path_k)\}$.

3 Example Discussion and Experimental Verification

In order to further explain the defined method for assessing timeliness based on weighted timeliness graph, this article selects an example for discussion. Finally, the effectiveness of the method was verified by testing its recall and accuracy on real datasets.

Table 1. Process data example.

Number	PID	Load test	Hot test	High Voltage test	Vibration test	Warranty
t_1	SC18091	2000 W	280°C	8.08×10^5pa	20000 Hz	2022
t_2	SC18092	2000 W	200°C	6.06×10^5pa	10000 Hz	2021
t_3	SC18093	2000 W	160°C	6.05×10^5pa	8000 Hz	2020

As shown in Table 1, an example D of a process flow with three tuples is given. The process flow example D includes four process data examples of load testing, high pressure test, high temperature test and vibration test, as well as the identification code PID and Warranty of each product.

In addition, its corresponding set of restricted failure dependency rules has been defined $\Sigma(r)$, $\Sigma(r)$ contains 5 dependency rules, denoted as r_1 to r_5, as shown in Table 2.

There are 5 rules in Table 2 that represent the dependency relationships between different process data in Table 1. It can be seen that in practice, there may be situations where multiple dependency rules derive the same conclusion. Due to the stronger dependency strength of rules, they often have stronger persuasiveness. Therefore, the

Table 2. Formulation rule representation of dependency intensity uncertainty.

Number	Formulaic representation
r_1	$\forall t_i, t_j (t_i[Loadtest] < t_j[Loadtest] \bigwedge t_i[Warranty] < t_j[Warranty] \rightarrow t_i \prec_{Loadtest} t_j, 0.9)$
r_2	$\forall t_i, t_j (t_i \prec_{Loadtest} t_j \rightarrow t_i \prec_{Highvoltagetest} t_j, 0.8)$
r_3	$\forall t(t[Loadtest] = 2000W \bigwedge t[Highvoltagetest] = 280°C \bigwedge t[BID] = SC18091 \rightarrow t \prec_{Loadtest} 2023, 1)$
r_4	$\forall t(t \prec_{Loadtest} 2023 \rightarrow t \prec_{Highvoltagetest} 2023, 0.8)$
r_5	$\forall t_i, t_j (tml(t_j \prec_{Loadtest} 2023) = min\{tml(t_j \prec_{Loadtest} t_i), tml(t_i \prec_{Loadtest} 2023)\})$

value with higher dependency strength is chosen as the final dependency strength of this conclusion. The formula is:

$$rhs(r) \Rightarrow tml(Q) = max\{tml(rhs(r_1)), tml(rhs(r_2)), \cdots, tml(rhs(r_k))\}, i \in [1, k] \quad (2)$$

Among them, $tml(rhs(r_i)) = tml(r_i) \times tml(lhs(r_i))$.

On the basis of the original r_2 and r_4, three rules r_6 : $\forall t_i, t_j (t_i \prec_{Vibrationtest} t_j \rightarrow t_i \prec_{Loadtest} t_j, 0.8)$, r_7 : $\forall t_i, t_j (t_i \prec_{Vibrationtest} t_j \rightarrow t_i \prec_{Highvoltagetest} t_j, 0.8)$, r_8 : $\forall t_i, t_j (t_i \prec_{Loadtest} 2023 \rightarrow t_i \prec_{Vibrationtest} t_j)$ have been added. Since the left part of rules r_1 and r_3 belong to deterministic conclusions, they do not need to appear in the dependency order diagram. Based on these five rules, a conclusion dependency order diagram was constructed, as shown in Fig. 1.

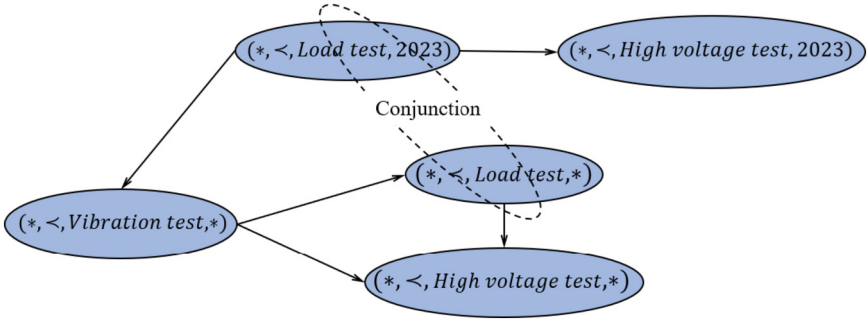

Fig. 1. Dependency sequence diagram.

In the conclusion dependency sequence diagram shown in Fig. 1, $t_i \prec_{Loadtest} t_j$ and $t_j \prec_{Loadtest} 2023$ can be combined to obtain a new conclusion $t_i \prec_{Loadtest} 2023$. At this point, the three conclusions $(*, \prec, Loadtest, \tau)$, $(*, \prec, Loadtest, *)$, and $(*, \prec, Vibrationtest, *)$ interact with each other (there is a hidden loop), so the unique identification order for the timeliness of the conclusion cannot be determined. At this point, it is necessary to merge the conclusion dependency sequence diagram, as shown in Fig. 2.

As shown in Fig. 2, the calculation order for the timeliness of each process data can be determined based on the merged process dependency sequence diagram of the conclusion dependency sequence diagram as follows: *Vibrationtest*, *Loadtest*, *Highvoltagetest*.

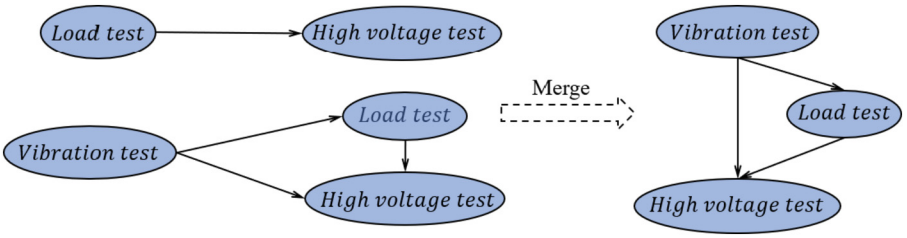

Fig. 2. Merging conclusion dependency sequence graphs.

In order to make the case more convincing, a new rule $r_9 = \forall t(t[Vibrationtest] = 800\,\text{Hz} \rightarrow t \prec_{Loadtest} 2023, 0.8)$ is added to Σ based on rule sets r_1 and r_3. By using these three rules, the weighted efficiency graph $WTG_{Loadtest}$ of process *Loadtest* at time 2023 can be obtained, as shown in Fig. 3 (a). The timeliness relationship between t_1, t_2, and t_3 data items in the WTG diagram of the *Loadtest* process at time $\tau = 2023$ in Fig. 3 (a) is shown in Fig. 3(a). In Fig. 3(b), there is $weight(t_1, t_2) = tml(t_1 \prec_{Highvoltagetest} t_2) \times tml(r_2) = 0.9 \times 0.8 = 0.72$.

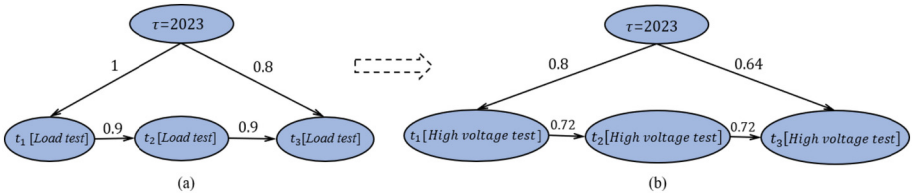

Fig. 3. Weighted timeliness graph of process *Loadtest* and process *Highvoltagetest* at 2023.

Using $t_3 \prec_{Loadtest} 2023$ in Fig. 3 as an example to illustrate how to calculate the single flux of a path, if there are two paths from $\tau = 2023$ to t_3. Therefore, $Flux(Path_1(2023, t_3)) = \min\{1, 0.9, 0.9\} = 0.9$, $Flux(Path_2(2023, t_3)) = \min\{0.8\} = 0.8$ and $tml(t_3[Loadtest]) = sFlux(Path(2023, t_3)) = Max\{0.9, 0.8\} = 0.9$.

In order to verify the effectiveness of the time effective identification method based on the weighted timeliness graph, this paper conducts simulation experiments on real datasets from the industrial big data innovation platform(https://www.industrial-bigdata.com/Data). The experimental results are shown in Fig. 4.

As shown in Fig. 4, it can be seen that the recall and precision both increase with the increase of the number of rules. However, when the number of rules reaches 4, the recall rate and accuracy remain stable at a certain value and do not change.

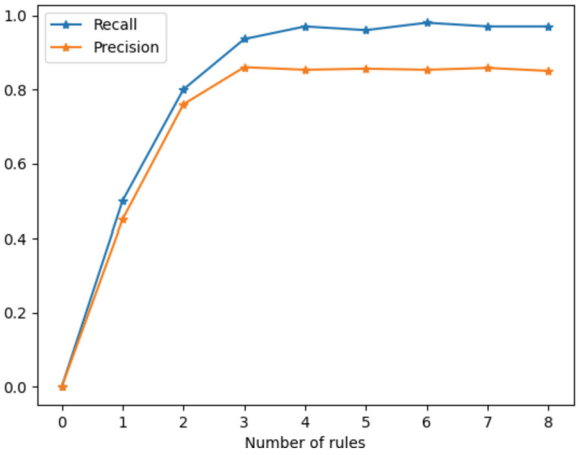

Fig. 4. The trend of recall and precision with the number of rules.

4 Conclusion

The quality appraisal method of manufacturing big data based on timeliness-dependent rules effectively evaluates the quality of manufacturing big data from the perspective of data timeliness. This model can effectively identify the timeliness of manufacturing big data through the dependency relationship between production process data without a clear timestamp. The quantification of rule dependency intensity was achieved by limiting the time dependent rules. The calculation order of the timeliness of each process data was determined through the process order dependency graph. The timeliness of the data was evaluated through a weighted timeliness graph (WTG) and its path single flux. Finally, the identification method was discussed through three process data instances and nine dependency rules, and its effectiveness was verified on real datasets (The recall rate can reach around 0.97 and the precision rate can reach around 0.82).

In future research on timeliness identification, the weight values of edges in the weighted timeliness graph can be set to dynamically change to adapt to different industrial application scenarios.

Acknowledgement. This work was supported in part by the National Key R&D Program of China under Grant No. 2020YFB1707900 and 2020YFB1711800; the National Natural Science Foundation of China under Grant No. 62262074, U2268204 and 62172061; the Science and Technology Project of Sichuan Province under Grant No. 2022YFG0159, 2022YFG0155, 2022YFG0157.

References

1. Li, M., Li, J., Cheng, S., Sun, Y.: Uncertain rule based method for determining data currency. IEICE Transactions on Information and Syst. E101.D (10), 2447–2457(2018)
2. Batini, C., Scannapieco, M.: Data and Information Quality: Dimensions, Principles and Techniques. Springer Publishing Company, Incorporated (2016)

3. Even, A., Shankaranarayanan, G., Berger, P.D.: Evaluating a model for cost-effective data quality management in a real-world CRM setting. Decis. Support. Syst. **50**(1), 152–163 (2010)
4. Firmani, D., Mecella, M., Scannapieco, M., Batini, C.: On the meaningfulness of "big data quality" (invited paper). Data Science Eng. **1**(1), 6–20 (2015)
5. Klier, M., Moestue, L., Obermeier, A.A., Widmann, T.: Event-driven assessment of currency of wiki articles: a novel probability-based metric. In: International Conference on Interaction Sciences (2021)
6. Liu, Z., Ding, X., Tang, J., Jiang, Y., Hu, D.: Anomaly monitoring of process based on recurrent timeliness rules (AMP-RTR). Applied Sciences **12**(24), 12917 (2022)
7. Ballou, D., Wang, R., Pazer, H., Tayi, G.K.: Modeling information manufacturing systems to determine information product quality. Manage. Sci. **44**(4), 462–484 (1998)
8. Dyreson, C.E., Jensen, C.S., Snodgrass, R.T.: Now in temporal databases. In: Encyclopedia of Database Systems. Springer, New York (2018)
9. Koubarakis, M.: The complexity of query evaluation in indefinite temporal constraint databases. Theoret. Comput. Sci. **171**(1), 25–60 (1997)
10. Bodirsky, M., Kára, J.J.J.A.: The complexity of temporal constraint satisfaction problems. Association for Computing Machinery **57**(9), 1–41 (2010)
11. Fan, W., Geerts, F., Wijsen, J.: Determining the currency of data. ACM Trans. Database Systems (TODS) **37**(4), 25–29 (2012)
12. Vianu, V.J.J.A.: Dynamic functional dependencies and database aging. Association for Computing Machinery **34**(1), 28–59 (1987)