Process Bottlenecks Identification and Its Root Cause Analysis using Fusion-based Clustering and Knowledge Graph

Junya Tang[a], Ying Liu[b], Kuo-yi Lin[a], Li Li[a*]

[a] *School of Electronics and Information Engineering, Tongji University, 201804, Shanghai, China*

[b] *Department of Mechanical Engineering, School of Engineering, Cardiff University, Cardiff CF24 3AA, UK*

**corresponding author: lili@tongji.edu.cn*

Abstract

Drawing the strengths of data science and machine learning, process mining has recently emerged as an effective research approach for process management and its decision support. Bottleneck identification and analysis is a key problem in process mining which is considered a critical component for process improvement. While previous studies focusing on bottlenecks have been reported, visible gaps remain. Most of these studies considered bottleneck identification from local perspectives by quantitative metrics, such as machine operation and resource requirement, which can not be applied to information and knowledge-intensive processes. Moreover, the root cause of such bottlenecks has not been given enough attention, which limits the impact of process optimisation. This paper proposes an approach that utilises fusion-based clustering and hyperbolic neural network-based knowledge graph embedding for bottleneck identification and root cause analysis. Firstly, a fusion-based clustering is proposed to identify bottlenecks automatically from a global perspective, where the execution frequency of each stage at different periods is calculated to reveal the abnormal stage. Secondly, a process knowledge graph representing tasks, organisations, workforce and relation features as hierarchical and logical patterns is established. Finally, a hyperbolic cluster-based community detection mechanism is researched, based on the process knowledge graph embedding trained by a hyperbolic neural network, to analyse the root cause from a process perspective. Experimental studies using real-world data collected from a multidisciplinary design project revealed the merits of the proposed approach. The execution of the proposed approach is not limited to event logs; it can automatically identify bottlenecks without local quantitative metrics and analyse the causes from a process perspective.

Keywords: Process Mining; Bottleneck Identification; Root cause Analysis; Knowledge Graph; Hyperbolic Neural Network.

1. Introduction

Process management is the science of observing how work is executed and taking advantage of improvement experience learned from previous processes. Due to higher requirements for manufacturers and service providers, including reducing costs and execution times, enhancing quality and efficiency, and improving the productivity of processes, process management has been utilised for defining, analysing, optimising, monitoring and controlling processes. In process management, bottlenecks have been regarded as crucial and valuable information [1]. For example, bottlenecks such as machine failures and supply chain delays in the manufacturing process will cause task congestion and case delay. Therefore, identifying bottlenecks is the key to accelerating the optimisation process management, which aims to improve work quality and efficiency [2]. For example, identifying production bottlenecks in the automotive assembly process can provide manufacturers with essential decision input [3], identifying bottleneck manufacturing resources in the intelligent workshop can provide a decision-making basis for subsequent production management [4], and identifying bottleneck machines can reduce the influence on the maximum completion time [5].

In recent years, the wide use of information systems facilitated the storage of process data, and a means to bridge the gap between Data Science and Process Science [6] called process mining exists. Process mining plays an essential role in many process management issues, including identifying bottlenecks. Researchers have conducted in-depth research on bottlenecks and proposed various bottleneck identification and analysis methods, especially in the manufacturing process [7]. The universal approach is to define or select a quantitative metric, such as throughput, machine load, and the working time of the machine. When the metric exceeds the constraint, it is considered that a bottleneck has occurred. This approach has been used to reference specific business processes, such as medical and surgical processes. However, this approach does not always work, especially for knowledge-intensive processes, such as some business and design processes, due to the nature of both data and the process.

The first is the diversity of data structures. Besides event logs, more process data can be collected, providing valuable information for different processes, such as the knowledge-intensive process. Contrary to the structured datasets in the manufacturing process, 80% of data in the knowledge-intensive process is unstructured texts [8], for example, design reports, emails, and meeting transcripts. The data does not contain explicit information about the relevant process, which leads to process information lacking visibility and increases the difficulty of selecting a quantitative metric. Therefore, the universal bottleneck identification and analysis approach can not be extended to processes that primarily contain unstructured data. Converting this gap requires combining domain knowledge to fully understand the process information from these data.

Another reason is the difference between the manufacturing process and the knowledge-intensive process. The former is formal and repetitive, while the latter is flexible and unpredictable [9]. In the manufacturing process, the bottleneck can be identified from a local perspective that observes a quantitative metric of a single object, usually the machine [10]. And these metrics can intuitively explain the cause of the bottleneck [11]. For example, the throughput and load of machines are two standard metrics. However, in the information and knowledge-intensive process, the factors causing the bottleneck are more uncertain, which leads to the difficulty of using a quantitative metric to measure these factors. Usually, researchers can only identify the bottleneck from the global perspective, such as identifying a bottleneck in a sub-process by its delay. However, this approach has a substantial limitation. The metric, from a global perspective, can not explain the cause of the bottleneck. Solving this problem calls for an in-depth analysis of the bottleneck.

To meet the requirements of improving various processes, this paper presents a data-driven methodology for identifying bottlenecks and analyzing the root causes in the knowledge-intensive process. In detail, the proposed method contains two main stages: identifying the bottleneck first and then analyzing its root causes. The first stage is identifying bottlenecks from unstructured input documents using a fusion-based clustering algorithm. The second stage focuses on analyzing the root causes of bottlenecks from a process perspective to enable empirical knowledge learning

considering multiple elements, e.g., task decomposition, human interaction and resource allocation. To analyse the root cause of bottlenecks that many previous works do not consider, a knowledge graph-based root cause analysis algorithm is studied. A knowledge graph is built to represent tasks, organisations and people as hierarchical and logical patterns. Based on this knowledge graph, we extract the element features and the relation features using the knowledge graph embedding. Considering the hierarchical and logical structure of the knowledge graph, the hyperbolic neural network is used to do knowledge graph embedding because of its high-fidelity and parsimonious representation compared with more Euclidean space approaches. Based on the knowledge graph embedding obtained, hyperbolic clustering-based community detection is applied to analyse the root cause of bottlenecks. The main contribution of this work includes:

- Focusing on unstructured data, we proposed an automatic process-oriented bottleneck identification method. Through knowledge extraction, it no longer relies on human intervention.
- Promoting process-oriented multi-dimensional root-cause analysis by combining knowledge graph-related technologies with universal process mining technologies.
- Based on the process knowledge graph embedding trained by a hyperbolic neural network, a hyperbolic cluster-based community detection mechanism is researched.

The rest of this paper is structured as follows. Section 2 reviews relevant studies of process mining, bottlenecks analysis in process management and knowledge graph embedding. Section 3 outlines the proposed methodology for bottleneck identification and its root cause analysis. Section 4 reports an experimental study using real-world business process data to demonstrate the effectiveness of the method. Section 5 gives the results and discussions of the experimental study. Section 6 concludes the paper.

2. Literature Review

2.1 Process Mining

Process mining is a technique providing insight into real processes, discovering, monitoring and improving them by analysing the process data. Three main tasks in process mining are model discovery, conformance checking and process enhancement. Process model discovery aims to support process improvements by discovering models from historical event logs, and it is the basis of various downstream tasks [6]. Traditionally, process mining focuses on automatically discovering a model that describes the causal relations or execution patterns of activities. In recent years, process mining has been applied successfully in many real cases, such as cases in the manufacturing industry [12], financial industry [13,14], and healthcare processes [15,16]. However, traditional process mining approaches cannot be applied directly to the knowledge-intensive processes that contain semi-structured and unstructured data, such as reports, emails, meetings and conversation records.

As a remedy, approaches mining from natural language texts were proposed. For example, Aa H et al. [17] presented a tailored natural language processing approach to identify activities and their relations. Friedrich et al. [18] augmented natural language processing techniques with an anaphora resolution mechanism to generate a process model. These process mining approaches follow a similar scheme. First, tokenize and tag texts by syntactic analysis, then detect activities and operators using semantic analysis and domain knowledge, and generate a model by discovering sequence flows using predefined signal words. These approaches have a significant limitation: the input text is required to describe a model sequentially, and the statements must be related to the process model. To solve this limitation, advanced process information extraction approaches were conducted. For example, Lijun Lan et al. focused on design process knowledge extraction and design process discovery from email data without explicit process information [9]. She proposed a deep belief net (DBN) that can automatically extract hidden process information and then model the design process. Elleuch M et al. achieved frequent activity discovery via a pattern discovery-

based approach. These process mining approaches can perform well in extracting hidden information from unstructured data and discovering process models. Still, the limitation that these approaches rely on the help of expert knowledge remains to be resolved.

2.2 Bottlenecks Analysis in Process Management

A process bottleneck is a crucial factor restricting the efficiency of the process, which is also the essential factor process analysts need to consider for improving or redesigning processes. Usually, a bottleneck is defined as the congestion points that slow down the process [19,20]. Identifying the bottleneck accurately dramatically improves the process, including time efficiency, cost and resource allocation. Some scholars have researched specific processes, such as medical processes [13,14], surgical processes [21] and especially manufacturing processes [3].

Some in-depth research on bottleneck identification has been widely used in the manufacturing process. For example, Lei Q et al. proposed a constraint and sensitivity analysis-based identification approach for a bottleneck in a job shop [2]. Caesarita Y et al. proposed a machine operational bottleneck identification and ranking model in mine operations [5]. Wang S et al. focused on energy storage investment requirements and proposed an operational bottleneck identification approach to analyse them [22]. These bottleneck identification approaches follow a similar scheme containing three steps: define or select a quantitative metric, calculate the normal value range of the metric, and detect objects outside the normal range. One major limitation of these approaches is that a quantitative metric reflecting processes is difficult to find or define.

It is feasible to reflect process performance by observing single objects, such as machine states in manufacturing and some specific processes, reducing the difficulty of defining quantitative metrics. However, it is difficult for a single object to reflect the entire process performance in information and knowledge-intensive processes due to its complexity and flexibility. To solve this problem, researchers can only identify the bottleneck from the global perspective. For example, identifying a bottleneck in a sub-process by its delay and identifying bottlenecks in mixed multiple-

concurrency short-loop structures using Petri nets and unbounded Petri nets [23, 4]. However, this approach has a substantial limitation compared with quantitative metrics of single objects. Quantitative metrics of single objects such as throughput, machine load, and the machine's working time can explain the bottleneck's causes, while the metric from a global perspective can not.

2.3 Knowledge Graph Embedding

To analyse the root cause of bottlenecks in information and knowledge-intensive processes, the individual workflow, tasks, organisation, time information, and the relation between them should be paid attention to. It is difficult to explain the root causes of bottlenecks by process metrics such as control flow in information and knowledge-intensive processes. However, the knowledge-intensive process is complex, which means the additional information, such as task decomposition, social network and organization, will significantly impact the process. These factors cannot be ignored when analyzing the causes of bottlenecks. But in the current flat and linear model, the information of each part is independent. The graph structure characteristics of the knowledge graph can well represent the feature of elements and the relation between elements. The knowledge graph can effectively connect and integrate each part of the information.

The knowledge graph has been applied in the industrial field due to its excellent representation ability [24,25]. For example, Mingfei Liu et al. used a knowledge graph to represent data in IIoT-enable cognitive manufacturing [26]. Xinyu Li et al. used ontology to represent process knowledge for decision support [27]. Akshay G. Bharadwaj constructed a product design knowledge graph from large CAD model repositories [28]. Pouya Zangeneh used ontology-based knowledge representation to support industrial megaprojects [29]. These studies focused on using knowledge to represent complex knowledge or large data. The knowledge graph embedding methods make it possible to use machine learning and deep learning approaches to analyse knowledge graph data, which can be utilized to analyze the correlation between each part of process information. For example, traditional graph analysis approaches can also be developed based on knowledge graph

embedding technology, such as community detection. Traditional community detection approaches are always based on graph features, such as nodes, edges and paths [30]. The combination of graph embedding and clustering provides a new approach to community detection [31, 32].

Traditional knowledge graph embedding methods rely on geometric properties that have few parameters and fail to encode logical properties [33]. Recent complex embedding approaches such as ComplEx, RotatE and QuatE models can effectively capture logical properties [34-36]. However, these embedding models require high dimensional space. A deep neural network is another approach [37]. However, pre-trained knowledge graph embedding requirements limit its application. Process knowledge graphs exhibit hierarchical and logical patterns, which means the above approaches are not effective enough. For these data, hyperbolic embedding methods can achieve high-fidelity and parsimonious representations. MuRP is the first method that studies knowledge graph embedding in hyperbolic space [38]. However, MuRP also fails to encode some logical properties. Ines Chami combined hyperbolic embedding with an attention mechanism to solve this problem, and an improved model was proposed for complex relational patterns [39].

2.4 A Brief Summary

According to the literature review, existing research into identifying bottlenecks has drawn much attention and applied to some areas. Unfortunately, current works on bottleneck identification have significant limitations facing unstructured data due to the difficulty of selecting a quantitative metric for process-oriented bottlenecks. Although root cause analysis has been studied for decades, its focus has always been in the engineering domain and has not been extended to multiple aspects of the process. This paper proposes an approach for process-oriented bottleneck identification and its root cause analysis, which aims to extend the state-of-the-art to processes of various domains.

3. Methodology

This paper proposes a machine learning-based bottleneck identification and root cause analysis approach. The framework of the proposed approach is shown in Figure 1, which contains three stages. We start with a data preprocessing step, extracting information from unstructured process documents. In stage 2, we combine the topic model and clustering to identify bottlenecks. In stage 3, a process knowledge graph is built based on the information extracted. Then, a hyperbolic neural network is applied to do knowledge graph embedding, and a clustering-based community detection approach is combined to analyse the bottlenecks' root causes.

3.1 Bottleneck Identification: BTMDW-Fusion K-means

Considering that the distance function significantly affects the clustering accuracy in K-means, the BTMDW (Biterm topic model with dynamic window)-Fusion K-means clustering-based bottleneck identification algorithm was proposed [40]. Process documents are collected and preprocessed and then represented as vectors. Due to the advantages of the topic model approach in distinguishing meaning and Doc2vec in dealing with polysemy, process documents are modelled with the topic model and Doc2vec. Since the document length affects the performance of traditional topic model approaches, such as LDA and BTM [41], BTMDW (Biterm topic model with dynamic window) was proposed by introducing a sliding window to BTM, which can adapt to different lengths of the process documents. After BTMDW topic modelling and Doc2vec modelling, documents are represented as two kinds of vectors. JS divergence and Euclidean are adopted to calculate the document similarity, respectively. Finally, the fusion distance is applied to K-means to achieve high-quality clusters to improve the quality of discovered topics from document clusters.

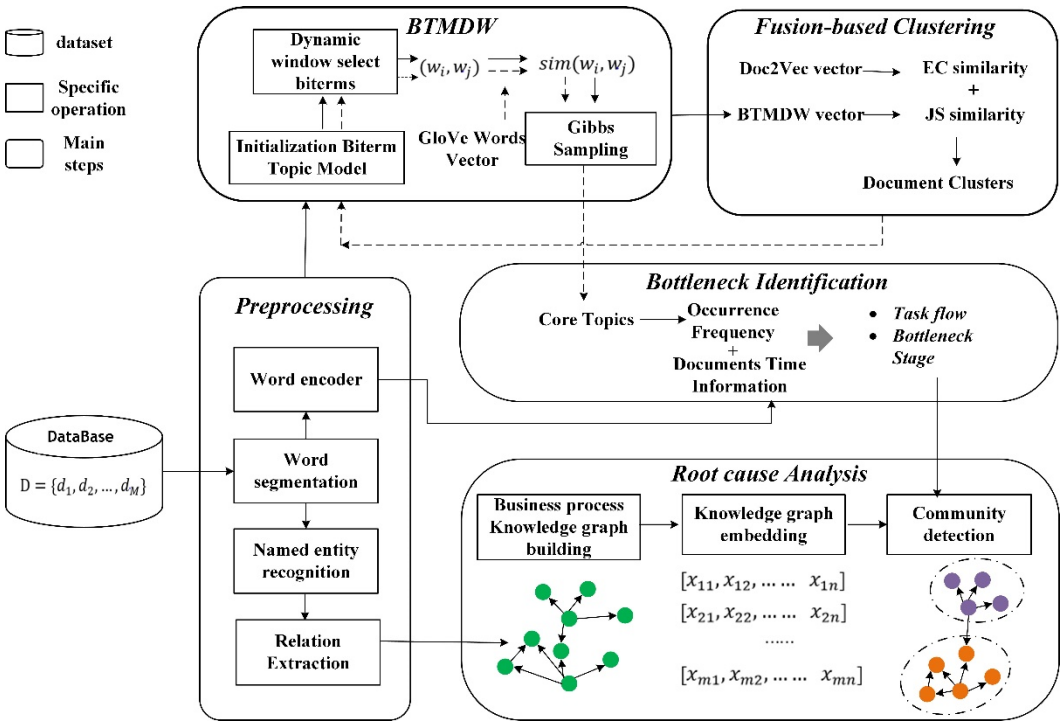


Figure 1. The framework of bottlenecks identification and its root cause analysis (the dotted line is the process of extracting the core topic from the obtained document clusters)

3.1.1 BTMDW-based document similarity

The process of BTMDW-based document similarity measurement is divided into two parts. The first part obtains the document vector representations by BTMDW modelling. The second part calculates the document similarity using JS divergence.

The probability $P(z|d)$ was used to represent the document, in which d donates the document and z donates the topic. The number of documents is N_d and the number of topics is N_z. Then, document d_i can be represented by a vector of the posterior distribution of topics that are trained by the BTMDW model [42]:

$$d_{i\_BTMDW} = \{p(z_1|d_i), p(z_2|d_i), \ldots, p(z_{N_z}|d_i)\} \quad (1)$$

JS divergence is often utilised to measure the distance between two distributions and is used to calculate the distance between two document vectors d_{i_BTMDW} and d_{j_BTMDW} trained by BTMDW, which reflects the similarity between the two documents d_i and d_j. The distance DIS_{JS} is as function (2):

$$DIS_{JS}\left(d_{i\_BTMDW}, d_{j\_BTMDW}\right) = \frac{DIS_{KL}(d_i||\frac{d_i+d_j}{2})+DIS_{KL}(d_j||\frac{d_i+d_j}{2})}{2} \quad (2)$$

3.1.2 Doc2vec-based document similarity

The process of Doc2vec-based document similarity measurement also has two steps. The first step uses Doc2vec to model the process documents. The second step uses Euclidean distance to calculate document similarity. The vector size is donated as M, the maximum epoch is donated as max_epochs, and the hyperparameter $alpha$. The output of the Doc2vec is the M dimension document vector as function (3).

$$d_{i\_Doc2Vec} = \{v_1(d_i), v_2(d_i), \dots, v_M(d_i)\} \quad (3)$$

Doc2vec solves the high dimensionality of document vectors so that Euclidean distance can work well. Euclidean distance was used to calculate the distance between two document vectors $d_{i\_Doc2Vec}$ and $d_{j\_Doc2Vec}$ trained by Dov2Vec. The distance DIS_E is as function (4), in which $||\cdot||_2$ donates the L_2 norm.

$$DIS_E\left(d_{i\_Doc2Vec}, d_{j\_Doc2Vec}\right) = ||d_{i\_Doc2Vec} - d_{j\_Doc2Vec}||_2 \quad (4)$$

3.1.3 Fusion distance K-means

After obtaining the statistical document vector by BTMDW and the semantic document vector by Doc2vec, the fusion distance DIS based on these two document vectors was applied to K-means. The fusion distance between two documents $d_i$and d_j is defined as a linear combination of DIS_{JS} and DIS_E, with a combination coefficient λ $(0 < \lambda < 1)$ shown as formula (5). Firstly, randomly select K initial centres of clusters from document collection D. After that, use the fusion distance to calculate the similarity between the cluster centre and other documents. Each document will be assigned to the most similar cluster. Then, the cluster centre is updated according to the calculation.

$$DIS\left(d_i, d_J\right) = \lambda \cdot DIS_{JS}\left(d_{i\_BTMDW}, d_{j\_BTMDW}\right) + (1-\lambda) \cdot DIS_E\left(d_{i\_Doc2Vec}, d_{j\_Doc2Vec}\right) \quad (5)$$

The overall procedure of BTMDW-Fusion K-means shows in Algorithm 1.

Algorithm 1: BTMDW-Fusion K-means

Input: Design process document set $D = \{d_1, d_2, \dots, d_{N_d}\}$, hyperparameter α and β, window size win, similarity threshold μ, sampling parameters S, vocabulary V, fusion coefficient λ, cluster number (task number) K, vector size M, document length threshold τ
Output: Document clusters
1. Select unordered word pair co-occurring in a dynamic window as biterms, and get the biterm set $\boldsymbol{B}$.
2. Calculate word similarity matrix in vocabulary V using Glove and Cosine distance.
3. Randomly initialise the topic assignments for all the biterms.
4. **for** *iter*=1 to N_{iter} **do:**
    for each biterm $b_i = (w_{i,1}, w_{i,2}) \in \boldsymbol{B}$
        Draw topic k from $P(z_i = k|\boldsymbol{z}_{-i}, \boldsymbol{B})$ according to [6]
        Update n_k and $n_{w|k}$
    Compute $\boldsymbol{\phi}$ and $\boldsymbol{\theta}$
5. Calculate $P(z|d)$ and get a representation of documents: d_{i_BTMDW}.
6. Train Doc2Vec and get documents representation $d_{i\_Doc2Vec}$.
7. Randomly select K design documents from data set D as the initial cluster centres $c_e, e = 1, \dots, K$.
    Repeat:
        Calculate the distance between documents and c_e according to the distance function (5).
        Update cluster centres c_e.
    Until:
        Cluster centre c_e is no longer changed.

* $w_{i,\cdot}$ donates the word in b_i
* n_k donates the number of biterms assigned to topic k
* $n_{w|k}$ donates the number of times that word w assigned to topic k
* $\boldsymbol{\phi}, \boldsymbol{\theta}$ are the multinomial distribution parameters [27]

3.2 Root Cause Analysis of Bottlenecks

The second part proposed a root cause analysis algorithm for bottlenecks. The root cause refers to the factor that influences the bottlenecks, for example, the people executing the task, the rationality of the task, and the efficiency of the organisation. The algorithm includes three steps. Process analysis has several perspectives. However, these perspectives are independent and difficult to correlate. The knowledge graph is an effective way to connect information from different perspectives. Considering that, a process knowledge graph was built from process documents first. Secondly, community detection was used to analyse the correlation between each factor and bottlenecks before which Knowledge graph embedding was conducted. Then,

knowledge graph embedding was trained from the process knowledge graph. Last, hyperbolic clustering-based community detection was used to analyse the root cause of bottlenecks based on the knowledge graph embedding.

3.2.1 Process Knowledge Graph Building

For knowledge graph building, the problem statement is to formulate raw texts into a knowledge graph using natural language processing, including sentence segmentation, entity extraction and relation extraction. Sentence segmentation is splitting the text documents into shortlisted sentences only where there are precisely a few subjects and objects. Nodes and edges are the two main elements in a knowledge graph. In the process knowledge graph, nodes are the entities presented in the sentences, which entity extraction approaches can extract with the help of parts of speech tags. Usually, the nouns and the proper nouns are more likely to be entities. However, in the process document, the sentence is more complicated because an entity can span multiple words, and some entities are composed of compound words. To address these problems, rule matching-based dependency analysis of the sentence was also used to help entity extraction. For example, $X\ such\ as\ Y$ is a pattern where the type of Y is found out from X. Edges in the knowledge graph are the relations connecting these entities. These elements are extracted in an unsupervised manner using the grammar of the sentences. The extracted entities and relations are stored as triples [source entity, relation, object entity]. The package called networkx in Spacy can build the process knowledge graph from these extracted triples.

3.2.2 Hyperbolic Convolution Neural Network-based Knowledge Graph Embedding

Based on the process knowledge graph, the knowledge graph embedding is used to extract features and express them as vectors. In the knowledge graph embedding problem, we got a set of triples $(h, r, t) \in G$ in which $h, t \in E$ and $r \in R$. E and, respectively, represent entity and relation sets. The knowledge graph embedding is to map entities $e \in E$ to a vector named entity

embeddings $v_e \in V^{d_e}$ and map relations $r \in R$ to a vector named relation embeddings $v_r \in V^{d_r}$ of some vector space V (traditionally Real Space) to preserve the knowledge graph structure. These embeddings are trained by optimising a score function $S(\cdot,\cdot,\cdot)$: $E \times R \times E \to \mathbb{R}$ (Real Space) that measures similarity between triples. Embedding models aim to optimise the score function for each triple to ensure valid triples will receive lower scores than invalid triples. This work aims to learn hyperbolic embeddings that can preserve the latent hierarchies of knowledge graphs, which can better encode complex logical patterns such as symmetry, anti-symmetry and inversion.

We briefly review hyperbolic geometry. Hyperbolic geometry has constant negative curvature and belongs to non-Euclidean geometry [40]. In this study, we use the Poincare ball model with negative curvature $-c$ ($c > 0$). A d-dimensional Poincare ball is $B^{d,c} = \{\boldsymbol{x} \in \mathbb{R}^d : ||\boldsymbol{x}||^2 < \frac{1}{c}\}$, in which $||\cdot||_2$ denotes the L_2 norm. To build a hyperbolic counterpart, we use the following primitives:

Mobius addition:

$$x \oplus_c y = \frac{\left(1+2c\langle x,y\rangle+c||y||_2^2\right)x+(1-c||x||_2^2)y}{1+2c\langle x,y\rangle+c^2||x||_2^2||y||_2^2} \tag{6}$$

Mobius subtraction:

$$x \ominus_c y = x \oplus_c (-y) \tag{7}$$

Mobius "matrix-vector" multiplication:

$$\boldsymbol{M}\otimes_c x = \frac{1}{\sqrt{c}}\tanh\left(\frac{||\boldsymbol{M}x||_2}{||x||_2} tanh^{-1}(\sqrt{c}||x||_2)\right)\frac{\boldsymbol{M}x}{||\boldsymbol{M}x||_2} \tag{8}$$

The hyperbolic convolution neural network (HCNN) based knowledge graph embedding model has some notions. The valid triple set is donated by G. Corrupt some samples in G to generate an invalid triple set G'. The dimension of embeddings is donated by m and each embedding triple (v_h, v_r, v_t) can be donated by a matrix $\boldsymbol{A} = [\boldsymbol{v_h}, \boldsymbol{v_r}, \boldsymbol{v_t}] \in \mathbb{R}^{m\times 3}$. A filter $\boldsymbol{\varphi} \in \mathbb{R}^{1\times 3}$ is repeatedly operated on the convolution layer over each row of $\boldsymbol{A}$. A feature map $\mathbf{v} = [\boldsymbol{v_1}, \boldsymbol{v_2}, \dots, \boldsymbol{v_k}] \in \mathbb{R}^m$ will be generated finally, in which:

$$\boldsymbol{v_i} = g(\boldsymbol{\varphi}\otimes_c \boldsymbol{A}_{i,:}^{\boldsymbol{T}} \oplus_c b) \tag{9}$$

where $b \in \mathbb{R}$ is a bias term, and g is the activation function, usually select ReLU.

To generate various feature maps, a set of different filters Φ is used, and $\gamma = |\phi|$ denotes the number of filters. These γ feature maps generated by different filters are connected into a single vector $\in \mathbb{R}^{\beta m \times 1}$ which is then inputted into a full connection layer with a weight vector $\boldsymbol{W} \in \mathbb{R}^{\beta m \times 1}$ to calculate a score for the triple (h, r, t). The formula (10) is the definition of score function f:

$$f(h,r,t) = concat(g([\boldsymbol{v_h}, \boldsymbol{v_r}, \boldsymbol{v_t}] * \Phi)) \otimes_c \boldsymbol{W} \tag{10}$$

Where * denotes a convolution operator, and $concat$ denotes the connection operator. In the optimisation stage, to minimise the loss function L, the Adam optimiser is used to train HCNN [43].

$$L = \sum_{(h,r,t)\in\{D \cup D'\}} log\left(1 \oplus_c exp\left(S_{(h,r,t)} \otimes_c f(h,r,t)\right)\right) \oplus_c \frac{\lambda}{2} \otimes_c ||\boldsymbol{W}||_2^2 \tag{11}$$

where $S_{(h,r,t)} = \begin{cases} 1\ for\ (h,r,t) \in G \\ -1\ for\ (h,r,t) \in G' \end{cases}$.

3.2.3. Hyperbolic Clustering-based Community Detection

This work analysed the root cause of process bottlenecks by hyperbolic clustering-based community detection, which can extract subtasks and operator networks closely related to bottlenecks. To preserve the hierarchical nature of process data, knowledge graph embedding was done in the hyperbolic space. Similarly, clustering was also performed in the hyperbolic space. Knowledge graph embeddings obtained through HCNN were used as the input of hyperbolic clustering, and the community centred on each main task can be obtained. Then, analyse its root cause based on the knowledge graph of the community where the bottleneck is located. The distance in hyperbolic space used in clustering is as follows: for a d-dimensional Poincare ball $B^{d,c} = \{\boldsymbol{x} \in \mathbb{R}^d : ||\boldsymbol{x}||^2 < \frac{1}{c}\}$, $\boldsymbol{x}$ and $\mathbf{y}$ are two points in this Poincare ball. The hyperbolic distance between $\boldsymbol{x}$ and $\mathbf{y}$ on $B^{d,c}$ has the explicit formula:

$$d^c(\boldsymbol{x}, \boldsymbol{y}) = \frac{2}{\sqrt{c}} arctanh(\sqrt{c}||-\boldsymbol{x} \oplus_c \boldsymbol{y}||) \tag{12}$$

where $||\cdot||$ denotes the L_2 norm.

The overall procedure of root cause analysis of bottlenecks shows in Algorithm 2.

Algorithm 2: Root cause analysis of bottlenecks (BRCA)

Input: Design process document set $D = \{d_1, d_2, \dots, d_{N_d}\}$, cluster number (task number) K,
Output: community graph

1. Preprocess the input data D, including word segmentation, named entity recognition and relation extraction, and get triples set $[h, t, r]$ of data D, including n triples.
2. Encode the entity and relation of the triples set and get the encoded triples list $\boldsymbol{kg}$ (size:$n \times 3$).
3. Input the initial embeddings of triples list $\boldsymbol{kg}$ into HCNN to train it, and get the final knowledge graph embedding $\boldsymbol{E} = (\boldsymbol{e_1}, \boldsymbol{e_2}, \dots, \boldsymbol{e_n})$, $\boldsymbol{e_i}$ is the embedding of the ith triple.
4. Select K triples from $\boldsymbol{kg}$ as the initial cluster centres, $c_e, e = 1, \dots, K$
   Repeat:
   Calculate the distance between each triple and c_e according to the distance function (12)
   Update cluster centres c_e.
   Until:
   Cluster centre c_e is no longer changed.
5. Find the community where the bottlenecks locate and build its community graph.

4. Experimental Study

The experimental study was conducted using the historical data collected from a real design process hosted by a university. The dynamic and complex environment and the inherently uncertain nature of innovative design processes lead to an industrial reality in which bottlenecks are widespread. Identifying the bottleneck and analysing its root cause can support designers in avoiding these bottlenecks in future design projects and reduce design risks. Design process analysis is a multi-dimensional problem, including analysing activities, control flow, decisions made throughout the workflow, and changes in the design process [44].

The design process is a knowledge-intensive process in which most data is unstructured. The project in this case study aims to design an Ants transportation system to track the traffic wave problem in the highway system. During the design process, the activities of all participants are recorded in the communication emails. During the two years of the project, all 569 emails were required to be sent to a shared address and stored as an XLM file. This XLM file contained all information during the design process, such as activities, resources and personnel interactions. An

initial experimental study was conducted to identify the bottleneck and analyse its root cause from a process perspective.

4.1 Data Acquisition and Preparation

The data in this study was collected from a design process. The raw data regarding the design process were emails extracted from Outlook as an XML file. The raw data is unstructured text containing process information, including design activities, participants, resources, and time. After filtering blank emails and useless information such as links and marker symbols, 357 emails were kept for subsequent analysis and the vocabulary size $|V| = 2928$. For the two goals in this study, the bottleneck identification and its root cause analysis, some preprocessing was taken.

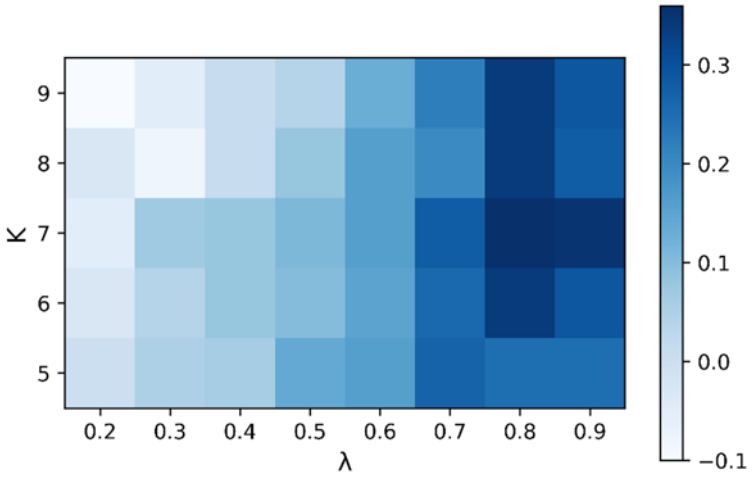
Bottleneck identification is mainly realised by the topic model BTMDW and the fusion-based K-means. In BTMDW, the word filter sets the semantic similarity threshold to increase the probability of sampling similar words and enhance the topic clustering effect and the difference between selected topics. To calculate the word similarity, preprocessing was conducted to transform words into 100-dimension vectors using the Glove algorithm [45]. The root cause analysis of bottlenecks is mainly realised by the process knowledge graph. The construction of the process knowledge graph needs to transform the data into triples. Therefore, named entity recognition and relation extraction are carried out to obtain triples like [head-entity, relation, tail-entity].

4.2 Experimental Setup

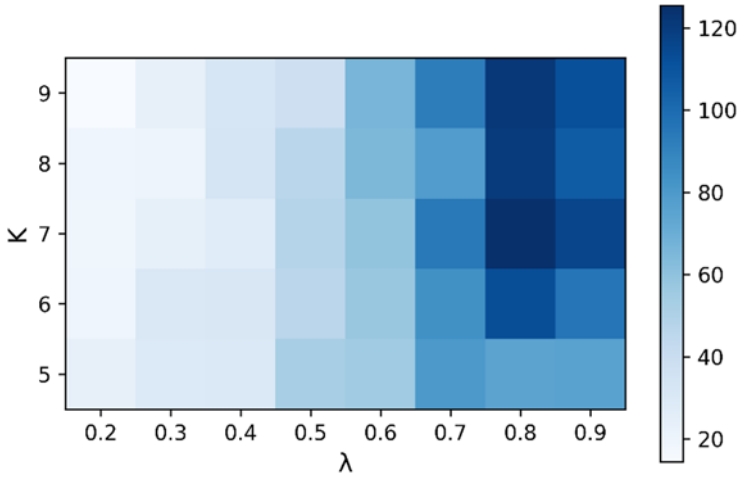
This paper proposed a bottleneck identification algorithm BTMDW-Fusion K-means and a root cause analysis algorithm based on the process knowledge graph. Three experimental studies were set up to examine proposed algorithms. The first experimental study aims to identify design bottlenecks and prove the proposed algorithm's effectiveness. The second experimental study aims to build the design process knowledge graph from extracted triples to represent the design process

documents as a graph structure. The third experimental study analyses the root cause of the bottleneck identified in the first experimental study through clustering-based community detection.

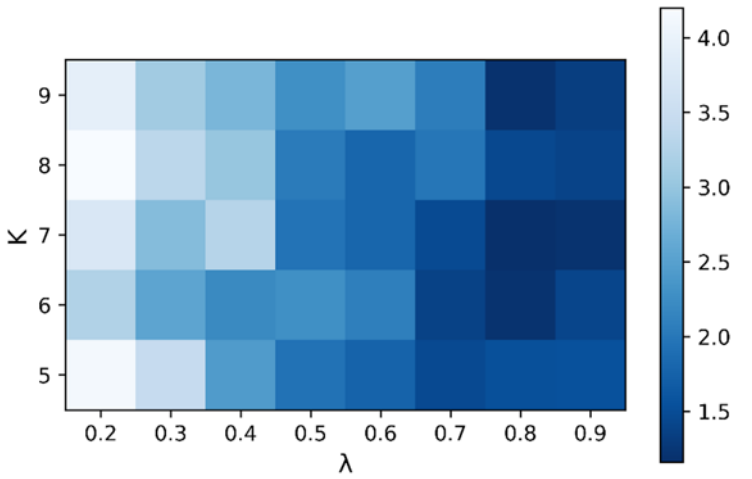
In the first experimental study, BTMDW extracted topics from design process emails. Then the fusion-based K-means divides these topics into several clusters. In BTMDW, a dynamic sliding window is introduced to select biterms from documents in different lengths. For different lengths, the size of the sliding window is also different. We set a document length threshold τ and a sliding window size win. If the document length is less than the $\tau = 40$, $win = 1$. Otherwise, $win = 0.4$. Only when the distance between two words is less than $win * document\ length$, select these two words as a biterm. According to expert experience, set the number of selected top topics $pz = 5$ and the topic vector dimension $pwz = 10$. In topic clustering, based on previous experience, the number of design stages in this design process is usually between five and nine, so the cluster number K was selected from 5 to 9. Another parameter, the fusion coefficient parameter λ was selected from 0.2 to 0.9 [40]. To determine optimal K and λ, three measures evaluation metrics: silhouette-score (S score), Calinski-Harabaz score (CH score), and Davies-Bouldin score (DBI score) [45], are utilised. From the analysis result in Figure 2, when K=7 and $\lambda = 0.8$, the performance is best. Under the optimal parameters, the S score is 0.36, the CH score is 125.43, the DBI score is 1.16, and the values indicate the clusters are effective.



(a) S score



(b) CH score



(c) DBI score

Figure 2. S value, CH value and DB value for different parameters K and λ value ((a) is the S score, (b) is the CH score, and (c) is the DBI score)

The second experimental study used entity recognition and relation extraction to extract triples from design process texts. A design process knowledge graph was built from these triples. The whole process is shown in Figure 3:

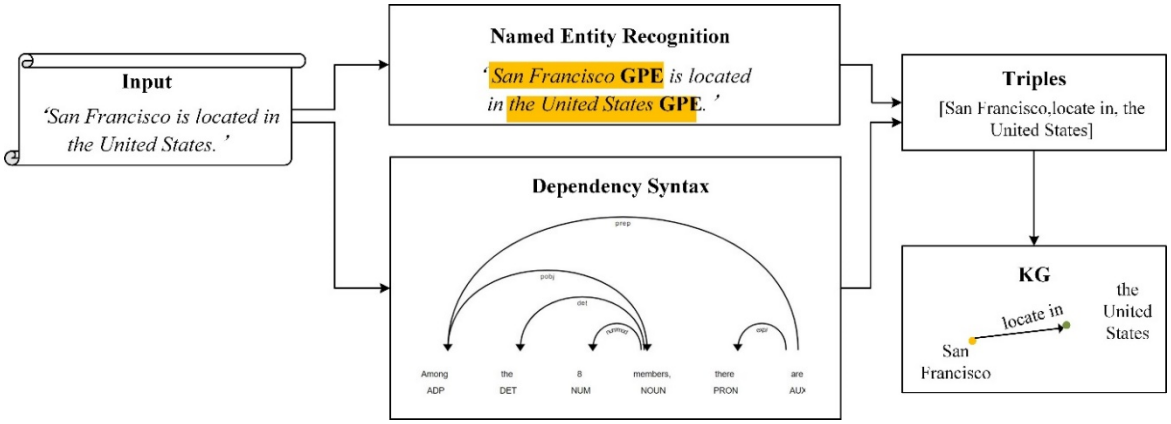


Figure 3. The Knowledge graph building

According to Figure 3, an email shown in Figure 4 can be divided into five sentences based on punctuation, such as the period and the question mark. Then, extracting and labelling named entities in each sentence utilising the named entity recognition algorithm. For example, in Table 1, entities including person name (PER), place name (LOR), city name (GPE) and organization (ORG) are labelled. Furthermore, the dependency syntactic parsing analysis is performed on the sentence structure, and the syntactic relation between labelled entities is analysed, as shown in Figure 5. For example, the first sentence in Table 1 includes two named entities, "Zhou Quan" and "DCC-FTS-Group-5". From the dependency tree (a) in Figure 5, the prepositional case (PREP) "from" connects the named entities, which represents the relation. Combining the above two points, every two entities and their relationships form a triple, such as in Table 2. These triples were used as input to machine learning to build a design process knowledge graph.

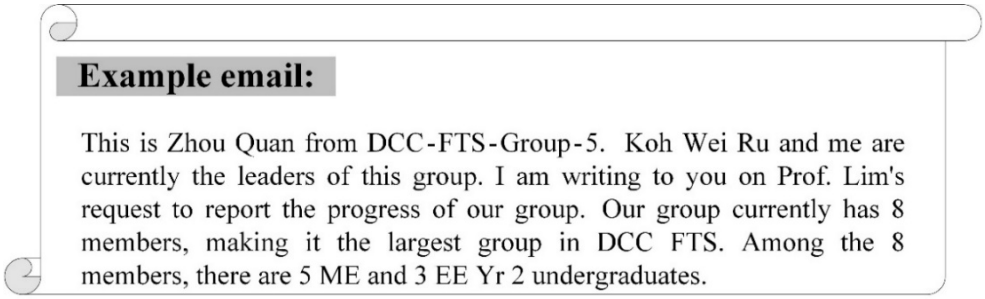


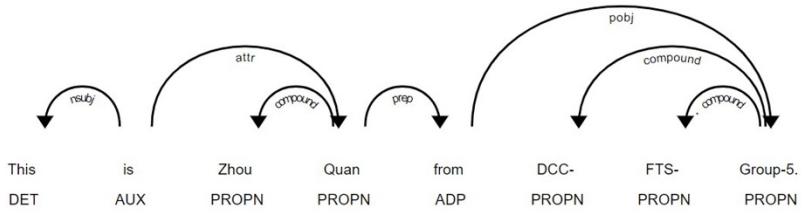
Figure 4. Example email

Table 1. Sentence and named entity extraction

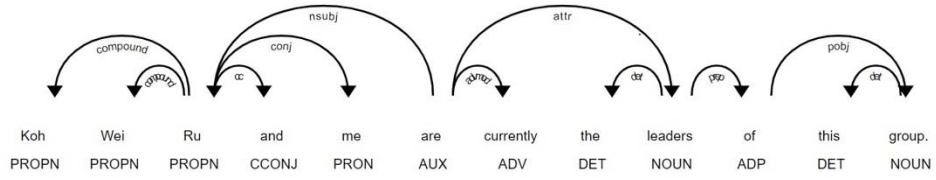
Sentences and named entity extraction
• This is ***Zhou Quan*** from ***DCC-FTS-Group-5***.
• ***Koh Wei Ru*** and ***I*** are currently the ***leaders*** of ***this group***.
• ***I*** am writing to ***you*** on ***Prof. Lim***'s request to report ***the progress*** of ***our group***.
• ***Our group*** currently has ***8 members***, making it the largest group in ***DCC FTS***.
• Among the ***8 members***, there are ***5 ME*** and ***3 EE undergraduates***.

Table 2. Entity-relation-entity triples extracted from the example email.

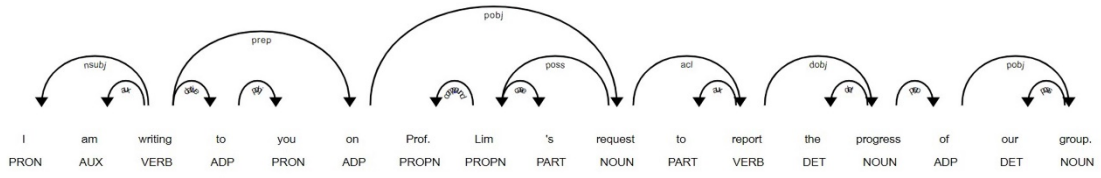
Triples extracted
[Zhou Quan, leader of, DCC FTS Group 5], [Koh Wei Ru, leader of, DCC FTS Group 5], [DCC FTS Group 5, report, process], [DCC FTS Group 5, has, 8 members], [DCC FTS Group 5, is in, DCC FTS], [8 members, has, 5 ME undergraduates], [8 members, has, 3 EE undergraduates]



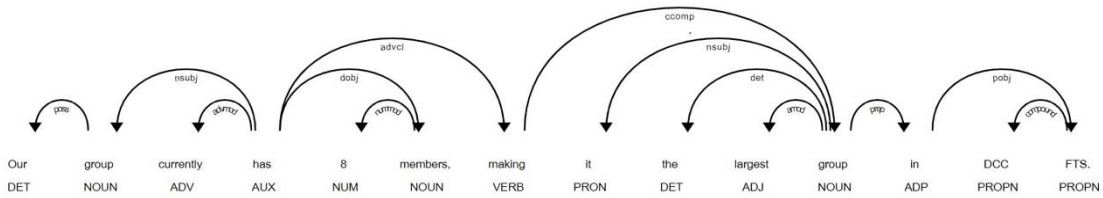
(a)



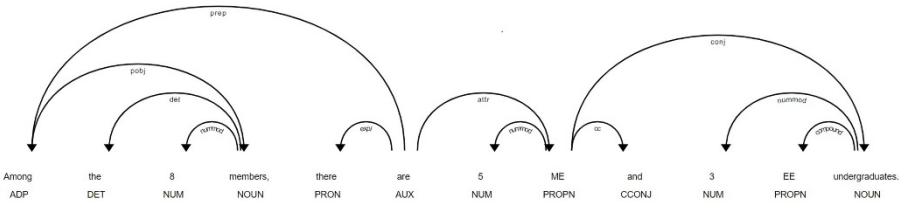
(b)



(c)



(d)



(e)

Figure 5. Syntactic dependency analysis (from (a) to (e) are the dependency analysis of the example email)

In the third experimental study, a hyperbolic convolution neural network was proposed to train the knowledge graph built in the second experimental study and get knowledge graph embeddings.

Then, hyperbolic clustering-based community detection was proposed to analyse the root cause of bottlenecks identified in the first experimental study.

This study uses a link prediction task to evaluate knowledge graph embeddings to optimise knowledge graph embeddings [46]. Following Bordes et al., invalid triples are needed in training. In invalid triples, either h or t is replaced by each of the other entities in E for each valid test triple (h, r, t) [46]. The initial entity and relation embeddings were produced by TransE (Translating Embeddings) [47]. According to [47], we train TransE for 500 epochs, and the hyperparameters are as follows: select the dimensionalities of embeddings k from $\{50, 100\}$, select the SGD (Stochastic Gradient Descent) learning rate from $\{1e^{-4}, 5e^{-4}, 1e^{-3}, 5e^{-3}\}$, using l_1 norm or l_2 norm, and select margin γ from $\{1,3,5,7\}$ [48].

We use Adam as the optimisation algorithm to learn the entity and relation embeddings, filters $\boldsymbol{\varphi}$ and the weight vector $\boldsymbol{W}$ [43]. The parameters setting is shown in Table 3. The maximum epoch is 200.

Table 3. Parameters setting of HCNN

Name	Value Ranges	Description
initial learning rate	$\{5e^{-6}, 1e^{-5}, 5e^{-5}, 1e^{-4}, 5e^{-4}\}$	-
g	ReLU	activation function
λ	0.001	loss function parameter
initial $\boldsymbol{\varphi}$	[0.1, 0.1, -0.1]	initial filter
β	$\{50, 100, 200, 400, 500\}$	the number of filters

5. Results and Discussion

5.1 Bottleneck Identification via BTMDW-Fusion K-means

Due to the value of K being 7, seven topic clusters are obtained. The seven main stages of the design process come from extracting the topics from these clusters. Each stage is represented by the top four words of topic clusters. According to the generation time of the emails related to stages,

the workflow of this design process is obtained. In Figure 6, we compared it with the planned workflow. It can be seen that both workflows include seven stages, and most of the stages and their sequence in the extracted workflow are consistent with the planned one. The extracted workflow combines two specific design phases in the planned workflow into one. Besides, some additional academic activities during the project have been considered because students also need to write related papers, which were also recorded in the emails. The result proved the workflow extracted based on BTMDW-Fusion K-means has high accuracy.

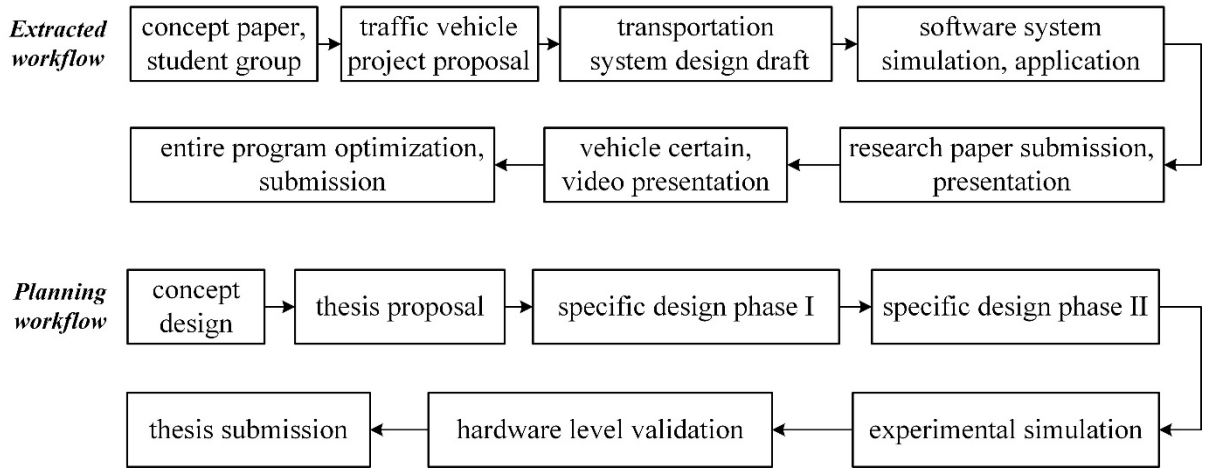
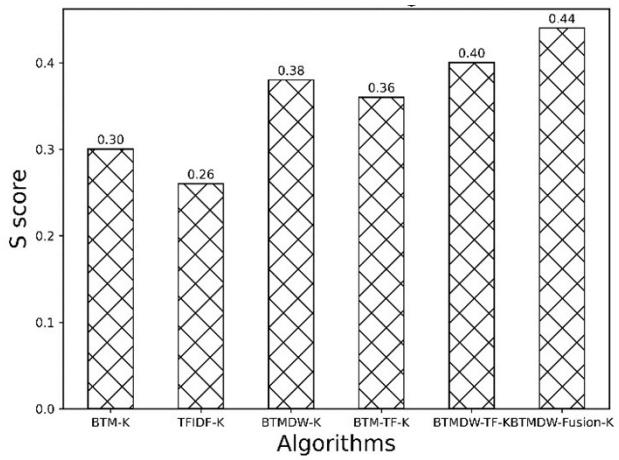


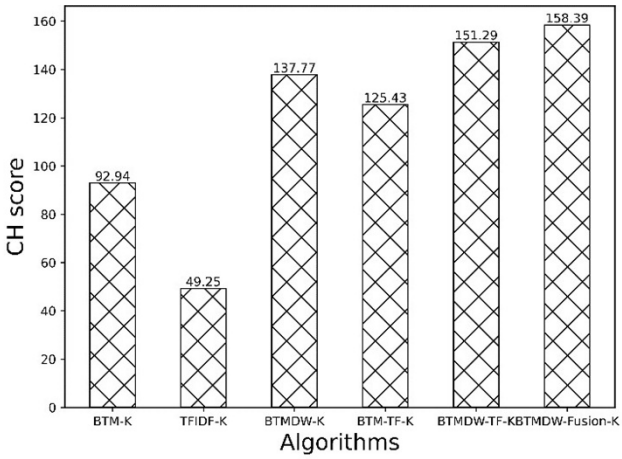
Figure 6. Comparison between extracted workflow and planning workflow

To verify the effectiveness of our algorithm BTMDW-Fusion K-means, we compared it with five advanced algorithms, BTM K-means, TFIDF K-means, BTMDW K-means, BTM-TF K-means [49] and BTMDW-TF K-means. Metrics S score, CH score, and DBI score were used to evaluate the quality of the extracted topic clusters. We believe a high-quality topic should contain top words with higher cohesion, corresponding to the metrics, a higher S score and CH score, and a lower DBI score. The result is shown in Figure 7. BTM K-means, TFIDF K-means and BTMDW K-means are three single-distance algorithms. Among them, BTMDW K-means shows the best performance according to the three metrics, proving that the topic model BTMDW can effectively improve the quality of document vectors. Comparing the performance of BTM-TF K-means, BTM K-means and TFIDF K-means, it can be found that the fusion distance can effectively improve the clustering performance. The performance of BTMDW-TF K-means, BTMDW K-means and

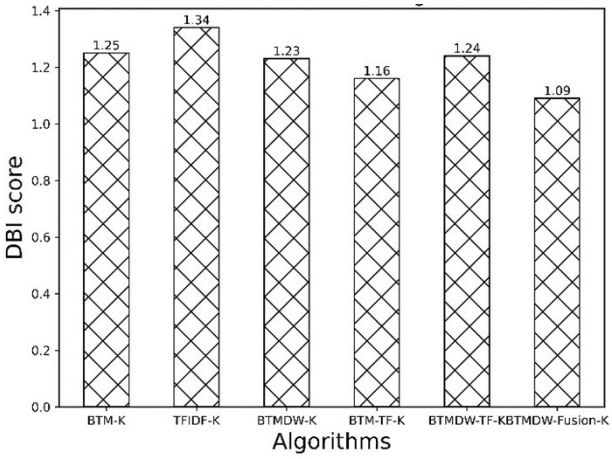
TFIDF K-means can also prove this conclusion. The performance of BTMDW-Fusion K-means is better than that of BTMDW-TF K-means, which indicates the fusion distance combining the statistical and semantic in this study can further improve the performance. We can see among these six algorithms, the algorithm BTMDW-Fusion K-means this paper proposed has the highest S score, CH score and lowest DBI score. Compared with the other five algorithms, BTMDW-Fusion K-means has significant advantages.



(a)



(b)



(c)

Figure 7. Comparison of evaluation metrics between BTMDW-Fusion K-means, BTM K-means, TFIDF K-means, BTMDW K-means, BTM-TF K-means and BTMDW-TF K-means ((a) is the S score, (b) is the CH score and (c) is the DBI score)

To identify the bottleneck, we analysed the execution period of each stage. The durations attached to each stage were used to discover and analyze bottlenecks. The stages where most time is spent and most reworks occur were highlighted, and cases in the stages will be further investigated to identify bottlenecks [6]. The result is shown in Figure 8. The design process lasted 23 months, and the project cycle is divided into 46 time periods based on half a month. The entire process can be divided into seven stages, and all stages are completed sequentially. In the plan, the previous stage will not appear in the later stage, and the specific design phase (task 3) will take the longest time.

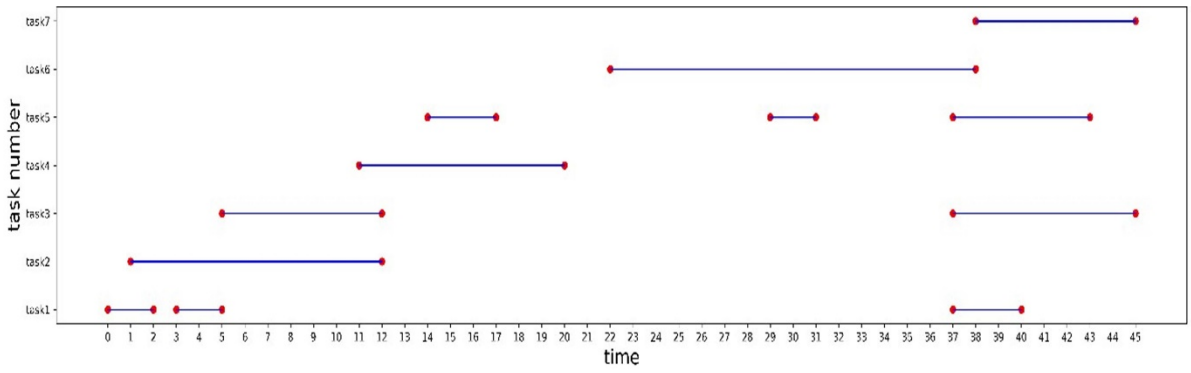
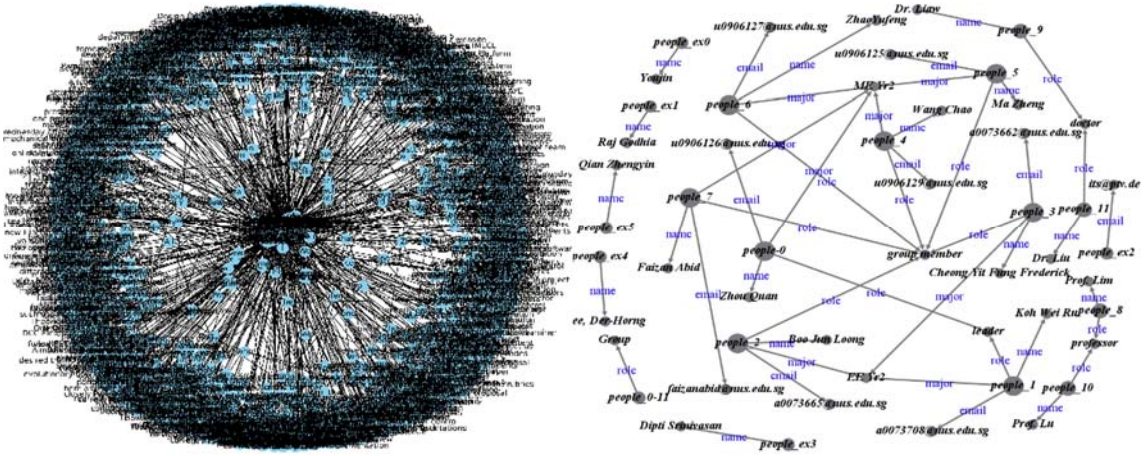


Figure 8. Last time of each task (Each scale on the time-axis represents half a month, a total of 23 months. The blue line in the figure indicates that the task corresponding to the y-axis is in progress during this time.)

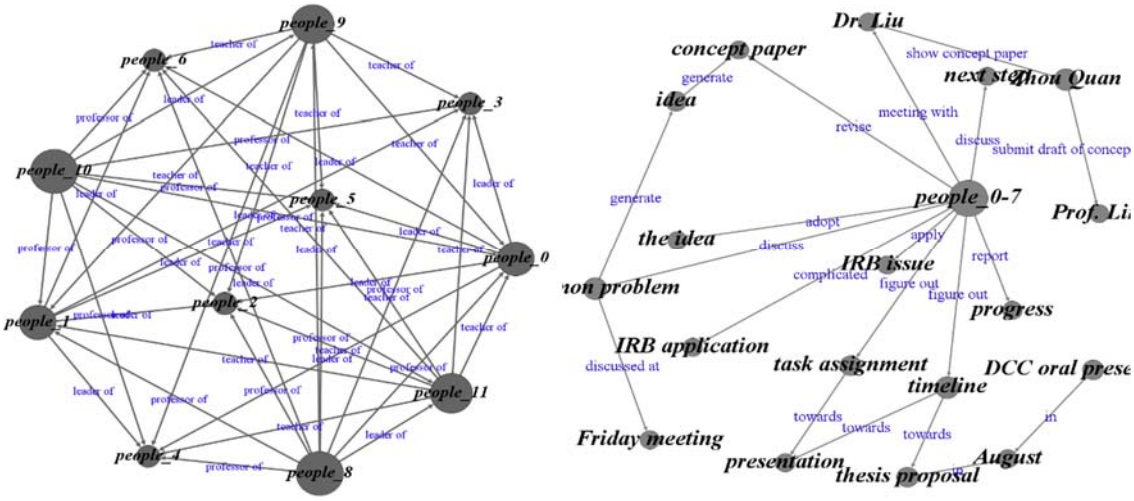
Figure 8 shows the actual completion of each stage. We can see that stage 1 lasted about four months, stage 2 lasted about five months, stage 3 lasted about seven months, stage 4 lasted about five months, stage 5 lasted about six months, stage 6 lasted about eight months, and stage 7 lasted about four months. From the point of lasting time, stage 6 consumed the longest time and exceeded the planned time. Besides, we can see that stages 1, 3 and 5 appear more than once. The second or third occurrence occurs during stage 6, which means some problems happened in the hardware level verification stage that requires changes to the previous design and re-simulation verification. From the point of causing rework, stage 6 is easy to cause the reworks of previous stages. From both metrics, bottlenecks occurred during stage 6 in this design process.

5.2 Process Knowledge Graph Building based on Rules

According to triples extracted from design process emails, a design process knowledge graph was built. The whole email contains 1476 sentences. At least one entity is detected in each sentence. The edge in the knowledge graph has three kinds: the attributes of people, the relations between people, and the task operation.



(a) Design process knowledge graph (b) Attributes of people



(c) Relations of people (d) Task operation

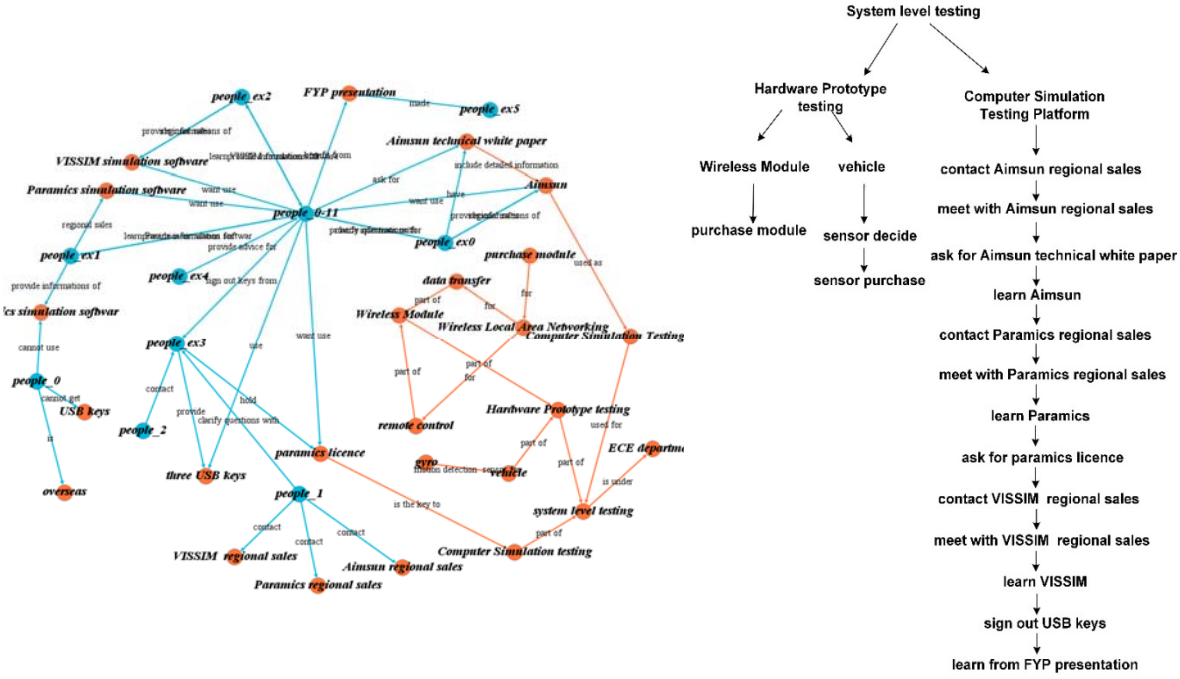
Figure 9. Process knowledge graph ((a) is the whole process knowledge graph, (b) is the part about attributes of some people, (c) is the part about relations of some people and (d) is the part about some task operations.)

Attributes of people reflect the basic information of students and teachers participating in this design project, including name, email, role and major, as shown in Figure 9 (b). There are 12 people in this research group, including eight students and four teachers. The attributes include name, email address, major and role in the project. Relations of people reflect the leadership between participants, as shown in Figure 9 (c). Among eight students, people_0 and people_1 are leaders. Among four teachers, people_8 and people_10 are professors and people_9 and people_11 are

teachers. Task operation reflects the task completion, as shown in Figure 9 (d), including the affiliation between the activities and the person executing the activity.

5.3 The Root cause Analysis Result via BRCA

Based on the above knowledge graph, a hyperbolic convolution neural network was used to train knowledge graph embeddings. Then, hyperbolic clustering-based community detection was applied to these knowledge graph embeddings, and several community graphs were obtained. According to the bottlenecks identified in 5.2, the community graph where the bottleneck stage is located can be found. In 5.2, the identified bottleneck stage is stage 6. The community graph where stage 6 is located was divided into three sub-community graphs from task, people-task and people-people perspectives, as shown in Figure 10 to Figure 12.



(a) (b)

Figure 10. Sub-community graph from the task perspective. (a) is the sub-community graph, the blue nodes are people, and the orange nodes are tasks. (b) is the task decomposition coming from the sub-community graph.

Figure 10 (a) is the task sub-community graph, and the tree diagram of task decomposition was extracted, as shown in Figure 10 (b). The nodes are divided into people and tasks. The task

nodes indicate that the computer simulation test occupies the main part of the task in the whole system-level testing. In the computer simulation test, multiple events requested the right to use some software, obtain software information and learn the use of this software which slowed down the whole process.

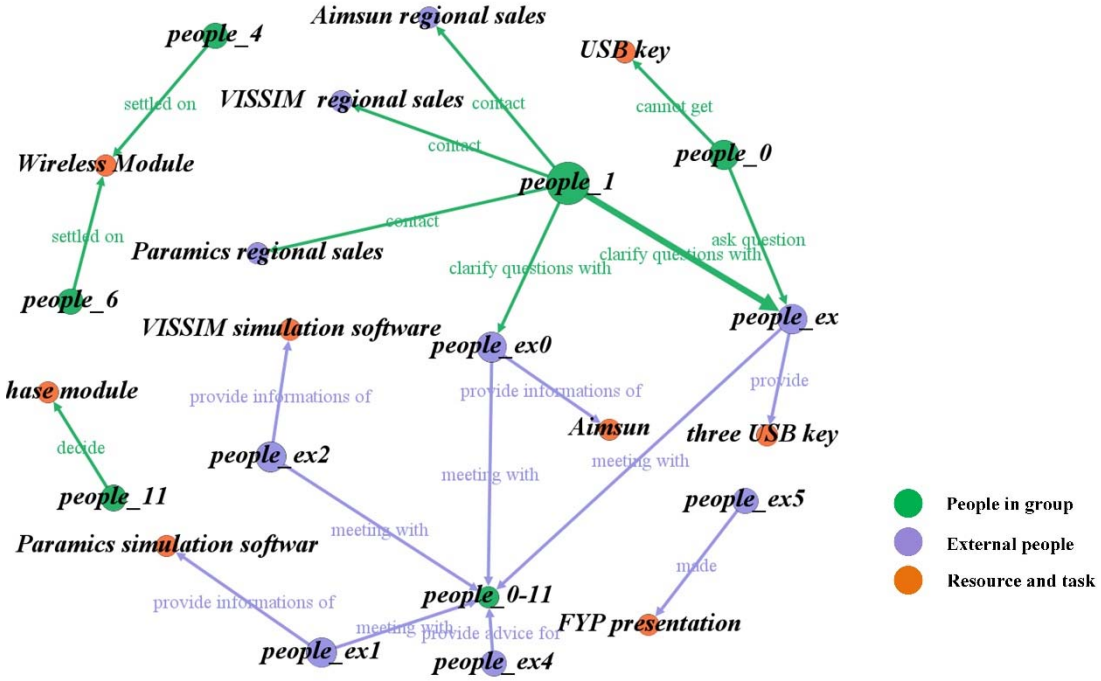


Figure 11. Sub-community graph from the people-task perspective.

Figure 11 is the task-people sub-community graph, the people_0-11 are teachers and students participating in this project, and people_ex0-5 are external people such as regional sales of some software. It can be seen that people_1 is the main person contacting external people. Through several meetings, external people provide software guidance for the teachers and students in the project team. People_4, people_6 and people_11 settled on the Wireless Module of the system and are responsible for purchasing related hardware. From the perspective of people-task, communication with external personnel is inefficient due to the lack of integration of information

among group members, which results in repeated and ineffective communication and slows down the whole process.

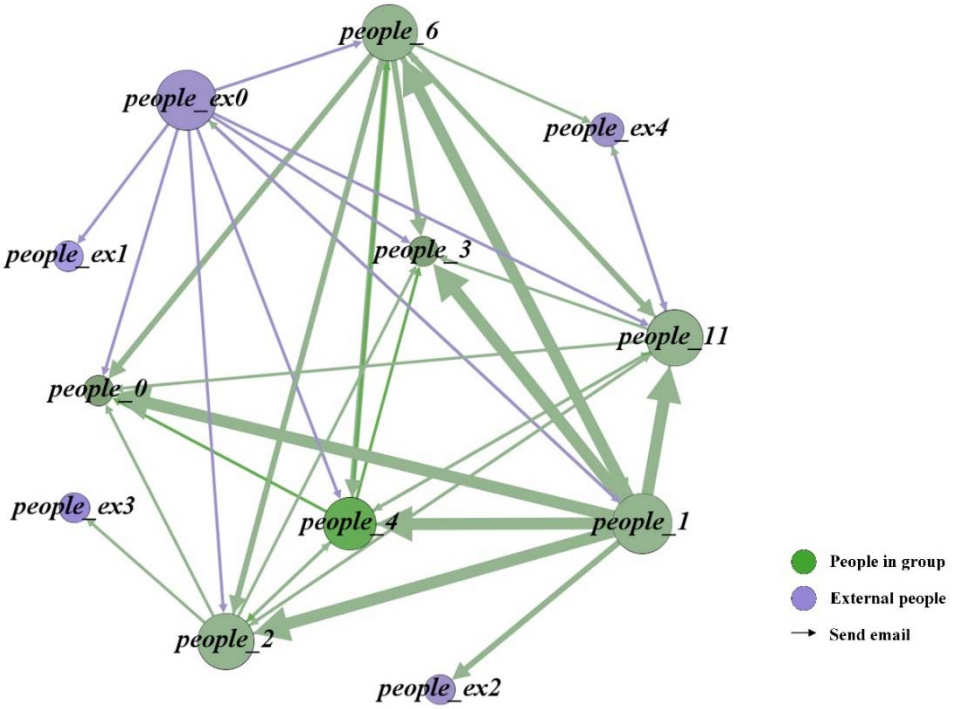


Figure 12. Sub-community graph from the people-people perspective

Figure 12 is the people-people sub-community graph. The weight of each edge represents the communication frequency, and the higher the weight, the greater the frequency of communication. It can be seen that people_1 and people_6 are in close contact with the rest of the students and teachers participating in this project. These two people are team leaders and maintain close contact with others. However, apart from them, there is a lack of communication among the other team members.

Three main root causes for this bottleneck from the three perspectives above exist. First, most students are unfamiliar with the relevant software and start learning them before needed. Second is the low efficiency of communication with external personnel. For example, the time of face-to-face meetings is difficult to coordinate. The third is that the division of the group is not clear, and the communication between team members is not enough.

5.4 Discussion

Bottleneck identification is crucial for improving processes and has been studied in specific domains. However, current studies still have some limitations in data and processes when extended to more domains. The information invisibility nature of unstructured data leads to the difficulty of extending quantitative metric-based approaches. A single local perspective can not satisfy process-oriented bottlenecks' root cause analysis. Hence, this paper focuses on automatically identifying the bottleneck from unstructured data without local quantitative metrics and analysing the root cause of the process-oriented bottleneck from multi perspectives.

For process-oriented bottleneck identification, experiment 1 indicates that the bottleneck identification framework can effectively extract tasks from process text and identify bottlenecks. However, this approach also has some limitations in the current stage. This paper only focuses on identifying bottlenecks from previous process data, while identifying bottlenecks of ongoing processes in time will provide more significant support to current processes. However, extending our approach to ongoing processes is difficult in which dynamic change (including new data and data distribution changes) is challenging. According to our research, incremental learning is a standard method for learning data streams. An incremental bottleneck identification approach may be a solution for ongoing processes.

For bottlenecks' root cause analysis, experiments 2 and 3 indicate that BRCA can analyse the root cause of bottlenecks identified from multiple perspectives and provide support for further bottleneck prediction and recommendation. This study introduced several knowledge graph-based technologies to help analyse root causes, which are seldom used in previous studies. The process knowledge graph helps enhance the feature representation of the data used in this study by integrating entity and relation features. Compared with the graph search method, the complexity of

the proposed clustering-based community detection is significantly reduced, and the basis of its realisation is knowledge graph embedding. The proposed knowledge graph embedding model based on a hyperbolic neural network can fit the hierarchical characteristics of data and extract higher quality features, which provide powerful support for discovering task-centric sub-knowledge graphs. These sub-knowledge graphs analyse the root cause of the bottlenecks from three perspectives: task-task, task-people and people-people. Although the introduction of knowledge opened a new door for root cause analysis, it also has some limitations. For big process data, the established process knowledge will also be complex, increasing the calculation cost of knowledge graph embedding. Perhaps divide-and-conquer training methods should be considered.

6. Conclusions

Bottleneck analysis is an important problem in process management and improvement. Identification and root cause analysis of process bottlenecks can bring significant benefits to process improvement and optimisation and can serve as the foundation for many other decision-support tasks. In the present study, to identify the bottleneck and analyse its root cause in the process, we have studied the possibility of achieving so by automatically identifying bottlenecks from process data that is not limited to event logs compared to previous studies. The proposed fusion-based clustering can identify bottlenecks in process documents with less human intervention and achieve better performance than the other prevailing algorithms. Furthermore, this paper has proposed and tested a knowledge graph and hyperbolic space-based multi perspectives root cause analysis approach, which enables critical support for further downstream tasks, e.g., bottleneck prediction and recommendation.

Acknowledgements

This work has been supported by the National Natural Science Foundation of China under grants 72171172 and 62088101, Shanghai Municipal Science and Technology, China Major

Project under grant 2021SHZDZX0100, Shanghai Municipal Commission of Science and Technology, and China Project under grant 19511132101.

References

[1]Wang Y, Zhao Q, Zheng D. Bottlenecks in production networks: An overview. Journal of Systems Science and Systems Engineering, 2005, 14(3): 347-363.
[2] Lei Q, Li T. Identification approach for bottleneck clusters in a job shop based on theory of constraints and sensitivity analysis. Journal of Engineering Manufacture, 2017, 231(6): 1091-1101.
[3] Lai X, Shui H, Ding D, et al. Data-driven dynamic bottleneck detection in complex manufacturing systems. Journal of Manufacturing Systems, 2021, 60: 662-675.
[4] Zhu F, Wang R, Wang C. Intelligent workshop bottleneck prediction based on complex network. Proceedings of the IEEE International Conference on Mechatronics and Automation (ICMA), 2019, 1682-1686.
[5] Li G Z, Xu Z G, Yan S L, et al. Bottleneck identification and alleviation in a blocked serial production line with discrete event simulation: A case study. Advances in Production Engineering & Management, 2020, 15(2).
[6] Van Der Aalst W. Process mining. 2016.
[7] Chen J, Yu Y, Pan M. A Method of Business Process Bottleneck Detection. Proceedings of the International Symposium on Parallel Architectures, Algorithms and Programming. Springer, Singapore, 2019: 249-261.
[8] Kir H, Erdogan N. A knowledge-intensive adaptive business process management framework. Information Systems, 2021, 95: 101639.
[9] Lijun Lan, Y Liu, Feng Lu W. Automatic discovery of design task structure using deep belief nets, Journal of Computing and Information Science in Engineering, 2018, 17(4), 041001.
[10] Han X, Wang Z, Xie M, et al. Remaining useful life prediction and predictive maintenance strategies for multi-state manufacturing systems considering functional dependence. Reliability Engineering & System Safety, 2021, 210: 107560.
[11] He Y, Zhu C, He Z, et al. Big data oriented root cause identification approach based on Axiomatic domain mapping and weighted association rule mining for product infant failure. Computers & Industrial Engineering, 2017, 109: 253-265.
[12] Chen J, Yu Y, Pan M. A Method of Business Process Bottleneck Detection. Proceedings of the International Symposium on Parallel Architectures, Algorithms and Programming. Springer, Singapore, 2019: 249-261.
[13] Werner M, Gehrke N. Multilevel process mining for financial audits. IEEE Transactions on Services Computing, 2015, 8(6): 820-832.
[14] Aydemir F, Pabuccu Y U, Basciftci F. A Hybrid Process Mining Approach for Business Processes in Financial Organizations. Procedia Computer Science, 2019, 158: 244-253.
[15] Munoz-Gama J, Martin N, Fernandez-Llatas C, et al. Process mining for healthcare: Characteristics and challenges. Journal of Biomedical Informatics, 2022, 127: 103994.
[16] Tamburis O, Esposito C. Process mining as support to simulation modeling: A hospital-based case study. Simulation Modelling Practice and Theory, 2020, 104: 102149.
[17] Aa H, Ciccio C D, Leopold H, et al. Extracting declarative process models from natural language. Proceedings of the International Conference on Advanced Information Systems Engineering. Springer, Cham, 2019: 365-382.
[18] Friedrich F, Mendling J, Puhlmann F. Process model generation from natural language text. Proceedings of the International Conference on Advanced Information Systems Engineering. Springer, Berlin, Heidelberg, 2011: 482-496.

[19] Wang Y, Zhao Q, Zheng D. Bottlenecks in production networks: An overview. Journal of Systems Science and Systems Engineering, 2005, 14(3): 347-363.
[20] Bemthuis R, van Slooten N, Arachchige J J, et al. A Classification of Process Mining Bottleneck Analysis Techniques for Operational Support. Proceedings of the 18th International Conference on e-Business (ICE-B 2021). SCITEPRESS, 2021.
[21] Lira R, Salas-Morales J, Leiva L, et al. Process-oriented feedback through process mining for surgical procedures in medical training: The ultrasound-guided central venous catheter placement case. International journal of environmental research and public health, 2019, 16(11): 1877.
[22] Wang S, Geng G, Ma J, et al. Operational Bottleneck Identification Based Energy Storage Investment Requirement Analysis for Renewable Energy Integration. IEEE Transactions on Sustainable Energy, 2020, 12(1): 92-102.
[23] Sun H W, Liu W, Qi L, et al. A process mining algorithm to mixed multiple-concurrency short-loop structures. Information Sciences, 2021, 542: 453-475.
[24] Wang R, Nellippallil A B, Wang G, et al. A process knowledge representation approach for decision support in design of complex engineered systems. Advanced Engineering Informatics, 2021, 48: 101257.
[25] Wang R, Nellippallil A B, Wang G, et al. A process knowledge representation approach for decision support in design of complex engineered systems. Advanced Engineering Informatics, 2021, 48: 101257.
[26] Liu M, Li X, Li J, et al. A knowledge graph-based data representation approach for IIoT-enabled cognitive manufacturing. Advanced Engineering Informatics, 2022, 51: 101515.
[27] Li X, Lyu M, Wang Z, et al. Exploiting knowledge graphs in industrial products and services: a survey of key aspects, challenges, and future perspectives. Computers in Industry, 2021, 129: 103449.
[28] Bharadwaj A G, Starly B. Knowledge graph construction for product designs from large CAD model repositories. Advanced Engineering Informatics, 2022, 53: 101680.
[29] Zangeneh P, McCabe B. Ontology-based knowledge representation for industrial megaprojects analytics using linked data and the semantic web. Advanced Engineering Informatics, 2020, 46: 101164.
[30] Fortunato S. Community detection in graphs. Physics reports, 2010, 486(3-5): 75-174.
[31] Agrawal R, Arquam M, Singh A. Community detection in networks using graph embedding. Procedia Computer Science, 2020, 173: 372-381.
[32] Li H, Wu X, Wan X, et al. Time series clustering via matrix profile and community detection. Advanced Engineering Informatics, 2022, 54: 101771.
[33] Dai Y, Wang S, Xiong N N, et al. A survey on knowledge graph embedding: Approaches, applications and benchmarks. Electronics, 2020, 9(5): 750.
[34] Trouillon T, Welbl J, Riedel S, et al. Complex embeddings for simple link prediction. Proceedings of the International conference on machine learning. PMLR, 2016: 2071-2080.
[35] Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. Rotate: Knowledge graph embedding by relational rotation in complex space. Proceedings of the International Conference on Learning Representations, 2019, 1902.10197.
[36] Shuai Zhang, Yi Tay, Lina Yao, and Qi Liu. Quaternion knowledge graph embeddings. Advances in Neural Information Processing Systems, 2019, 2731–2741.
[37] Stierle M, Weinzierl S, Harl M, et al. A technique for determining relevance scores of process activities using graph-based neural networks. Decision Support Systems, 2021, 144: 113511.
[38] Balazevic I, Allen C, Hospedales T. Multi-relational poincaré graph embeddings. Advances in Neural Information Processing Systems, 2019, 32: 4463-4473.
[39] Chami I, Wolf A, Juan D C, et al. Low-dimensional hyperbolic knowledge graph embeddings. Proceedings of the Association for Computational Linguistics, 2020, 2005.00545.
[40] Tang J, Li L, Liu Y, et al. Automatic identification of bottleneck tasks for business process management using fusion-based text clustering. Proceedings of the 17th IFAC Symposium on Information Control Problems in Manufacturing INCOM, 2021, 54(1): 1200-1205.

[41] Yan X, Guo J, Lan Y. A biterm topic model for short texts, Proceedings of the 22nd international conference on World Wide Web, 2013, 1445-1456.
[42] Cheng X, Yan X, Lan Y, et al. Btm: Topic modeling over short texts. IEEE Transactions on Knowledge and Data Engineering, 2014, 26(12): 2928-2941.
[43] Khaire U M, Dhanalakshmi R. High-dimensional microarray dataset classification using an improved adam optimizer (iAdam). Journal of Ambient Intelligence and Humanized Computing, 2020, 11(11): 5187-5204.
[44] De Smedt J, Hasić F, vanden Broucke S K L M, et al. Holistic discovery of decision models from process execution data. Knowledge-Based Systems, 2019, 183: 104866.
[45] Maulik U, Bandyopadhyay S. Performance evaluation of some clustering algorithms and validity indices. IEEE Transactions on pattern analysis and machine intelligence, 2002, 24(12): 1650-1654.
[46] Bordes A, Usunier N, Garcia-Duran A, et al. Translating embeddings for modeling multi-relational data. Advances in neural information processing systems, 2013, 26.
[47] Wang Z, Zhang J, Feng J, et al. Knowledge graph embedding by translating on hyperplanes. Proceedings of the AAAI Conference on Artificial Intelligence. 2014, 28(1).
[48] Nguyen D Q, Nguyen T D, Nguyen D Q, et al. A novel embedding model for knowledge base completion based on convolutional neural network. Proceedings of the NAACL-HLT, 2018:1712.02121.
[49] Wu D, Zhang M, Shen C, et al. Btm and glove similarity linear fusion-based short text clustering algorithm for microblog hot topic discovery. IEEE Access, 2020, 8: 32215-32225.