Abstract
This article proposes a framework $P3E$ for evaluating entity extraction problem over cyber-physical system data. As known by us, this article is the first work focusing on this problem, which is an important step for identifying entities in cyber-physical system data. Directed by the initial entities, utilizing the relaxation and verification techniques, $P3E$ provides a path expression–based solution for entity extraction problem, which has following characteristics. First, using path expressions, $P3E$ provides a condensed presentation for entity locations whose size may get very large when scaling up the data size. Second, requiring only one entity example to indicate the interests, using relaxation technique, $P3E$ can discover other similar entities automatically. Third, by adjusting the example given to $P3E$, users can specify their own interesting entities and control the entities discovered by $P3E$. Besides, utilizing the idea of sharing computations, by extending previous automaton techniques, an efficient implementation of $P3E$ is provided. Experimental results are reported, which show that $P3E$ can provide an effective and efficient solution to the entity extraction problem.

Keywords
Cyber-physical system, entity extraction, path expressions, relaxation

Introduction
There have been lots of research interests on cyber-physical system (CPS for short) since it has been widely used in real applications and produced huge influence on the way of obtaining and exchanging information of the physical world. Typically, there are at least two kinds of nodes in CPS, sensors and actuators. The duty of sensors in CPS is to collect valuable information about the physical world and transfer the information to cyber side of CPS (e.g. a cloud computing platform for analyzing sensed information) by the universal network. The duty of actuators is to change the physical world based on the advices given by cyber side of CPS. Obviously, the cyber side of CPS plays a key role between CPS users and physical world, and it can help users analyze and understand the information obtained by sensors. Data collection and analysis have been identified to be the essential problems in CPS.\(^1\)\(^-\)\(^3\) Previous research works in CPS area mainly focus on the data collection and in-network computing. Recently, more and more research efforts in designing efficient and effective computing methods on cyber side for CPS have been made.

In typical CPSs, different nodes are connected through a universal network, and data collection is achieved by deploying different kinds of sensors in physical world and transferring sensor data in the network. Sensors will monitor the physical world, produce sensed data, and send the data to the network.

College of Computer Science and Technology, Harbin Engineering University, Harbin, P.R. China

Corresponding author:
Lijie Li, College of Computer Science and Technology, Harbin Engineering University, No. 145, Nantong Street, Harbin 150001, P.R. China.
Email: lilijie@hrbeu.edu.cn
Different kinds of sensors will produce different kinds of data which can describe the physical world from different aspects. Also, different sensors may produce data about the same object or place. For example, to monitor the weather in some place, one sensor is used to measure the temperature, while another sensor is used to measure the humidity. The network providing connection for sensor nodes is built on advanced communication techniques and organized independent from the deployment of sensor nodes. Also, the network connects sensor nodes with some computing center on cyber side which will process the data collected by sensors and provide useful information to users. Usually, the feedback can be made automatically by computing center or artificially by expert users and will be translated to commands which can be executed directly on actuator nodes. The feedback commands are distributed to the actuator nodes by the universal network. Finally, according to the commands received, the actuator nodes perform special actions to change the physical world.

An example of CPS topology structure is shown in Figure 1. Assume that the CPS is deployed in some building and the goal is to monitor the environment status of the rooms in the building. Each node in the graph represents some sensor node in the CPS, and the communication and management relations between nodes are represented by edges. The types of sensors are labeled on the nodes. In the conceptual view, at each time slot, each node will produce one data, and thus, the topology structure of CPS network is indeed the structure of CPS data. In this example, two entities in Figure 1 can be recognized by identifying the nodes labeled with infrared. Intuitively, each entity represents the sensors monitoring one room, and the entity information can be obtained by collecting all data of the nodes in every time slot. The key problem focused by this article is just how to extract such entities efficiently. Also, it can be found that the “anchor” node infrared in the above example is rather useful for extracting the entities.

The typical system architecture of CPS as described above has many advantages such as high robustness and fine compatibility. However, such architectures cause more challenges for CPS data to be processed and understood on entity level by the processing center. Intuitively, the core problem of entity extraction for CPS data is to identify substructures of the whole topology structure of CPS such that the information generated by each substructure identified is related to the unique object or entity in physical world. Obviously, since one original goal of CPS is just obtaining information of the physical world, the entity extraction problem is essential for CPS. The challenges can be described as follows. (1) To extract entity information from CPS data, usually too much communication cost is spent since we do not know where the entities are distributed. (2) There might be many different communication methods which can find the entities, and proper ones should be selected to reduce the communication cost. (3) The coverage range of nodes and the topology structure of CPS data may vary, and we need to adapt those cases.

This article, to overcome the above challenges, focuses on the entity extraction problem. The entity extraction problem is divided into two parts,

\[ \text{Figure 1. CPS data topology example.} \]
conceptually, that is, discovering entity locations and the related attributes in CPS data. Intuitively, a location is an anchor node which is associated with an entity and describes it using information of nodes around, and it can be treated as the ID of the entity. The nodes providing descriptive information about the location node are attribute nodes. For example, considering the CPS data shown in Figure 1, the nodes labeled with infrared are locations which represent two rooms whose information is collected by the sensors, and the nodes around the left infrared node (e.g. temperature and Global Positioning System (GPS)) are attribute nodes. As known by us, there are still no research efforts on entity extraction problem for CPS data. The key idea of this article can be summarized as follows. First, we use expressions to represent entities in CPS data, and an initial set of entity expressions is generated manually or automatically. Then, more candidate entities are obtained with the help of expression relaxation techniques. Finally, the candidates are verified under the guide of entity attributes.

In this article, according to the idea introduced above, an entity extraction framework, \( P3E \) (Path Expression-based Entity Extraction for short), is proposed. By utilizing path expressions, in \( P3E \), the mechanism to declare interesting entity locations and specify the characteristics of the interesting entities is provided. It is unreasonable to assume that all interesting locations can be specified by users because of the irregularity of locations; to solve this problem, taking the locations identified by the users as an initial set, \( P3E \) can discover other interesting locations by extending the location set iteratively with the help of relaxation and verification techniques. The main challenges and contributions can be summarized as follows:

- **The first contribution is to propose the \( P3E \) framework to investigate the feasibility and explore useful techniques for entity extraction of CPS data. In \( P3E \), path expression is used to specify the interesting entity locations and describe the characteristics needed by interesting entities.**

- **The second contribution is to identify two main problems in \( P3E \) framework. The first one is to discover the candidate locations in view of the irregularity of localizations. It means that, even for entities of the same kind (e.g. rooms monitored by CPS), many different representations are needed to cover their locations, but user specifications can only provide some of them; therefore, automatic techniques for discovering the remaining locations are required. The second one is the performance of \( P3E \), that is, to propose practical implementation of \( P3E \) framework to provide an effective and efficient method for entity extraction problem.**

- **The third contribution is a method of using relaxation techniques to adapt for the irregularity of localizations in \( P3E \). Given an entity location expression \( q \), relaxation technique can discover other candidate location expressions by extending \( q \) using similar expressions, and the quality of generated candidate location expressions can be controlled by limiting the cost of relaxation operations.**

- **The fourth contribution is to propose an effective and efficient solution for entity extraction problem. For the effectiveness, measuring functions for determining the interesting degrees of entity locations and location expressions are introduced. For the efficiency, an algorithm IMP2E is proposed based on the idea of sharing computations and reducing redundancies among queries.**

- **The last contribution is to give a detail experimental study of IMP2E algorithm, which shows the effectiveness and efficiency of the algorithm on both synthetic and real data.**

The following part of this article is organized as follows. In section “Preliminary,” some background knowledge are introduced. The \( P3E \) framework and the formalization of entity extraction problem are described in section “The framework \( P3E \) for entity extraction in CPS.” Two algorithms NAI2E and IMP2E are proposed in section “Algorithms for entity extraction of CPS data.” In section “Experimental evaluation,” the details of experimental results are introduced. The final parts are related work and conclusion.

**Preliminary**

**Network model**

A network can be represented by a graph \( G = (V, E, T, l, l) \), where the parameters can be explained as follows:

- **\( V \) is the node set representing sensor nodes in the CPS network.**

- **\( E \subseteq V \times V \) is the edge set representing the connection relation between nodes. If \((u, v) \in E\), we can know that \( u \) and \( v \) can communicate with each other directly. Obviously, \( E \) is symmetric.**

- **\( T \) is another edge set which defines a tree over \( V \), which represents the topology structure of routing strategy in the network. Obviously, we have \( T \subseteq E \). Usually, we use \( r_{T} \) to represent the root of \( T \) which can be explained to be the processing center. For \((v_{1}, v_{2}) \in T\), \( v_{1} \) is called parent of \( v_{2} \) and \( v_{2} \) is one child of \( v_{1} \).**
• $L$ is the set of node labels, which represents the different types of sensors.
• $l$ is a function which defines a mapping from $V$ to $L$. Intuitively, $l$ returns the type of given sensor.

Given a network $G$, we use $G_T$, $G_E$, $G_R$, $G_L$, and $G_l$ to represent the elements in $G$, respectively. Here, w.l.o.g., it is assumed that each node in CPS has sensing and acting abilities and can be sensor and actuator at the same time.

**Entity-based data view and path expressions**

On the physical side of CPS, the data obtained by each sensor are relatively independent. For example, suppose that there are two sensors $u$ and $v$ deployed in different positions of one room $A$, and $u$ can measure the temperature and $v$ can measure the humidity value, and at time $t$, the data obtained from $u$ and $v$ can be denoted by $u_t$ and $v_t$, respectively. From the physical view of CPS, since $u$ and $v$ are different types of sensors and deployed in different positions, $u_t$ and $v_t$ are independent and two separated data.

However, on the cyber side of CPS, the data are expected to be explained based on real entities in physical world by end users. Continued from the example above, the CPS users are used to treating $u_t$ and $v_t$ to be the temperature and humidity values of the same room. That is, two independent data obtained from physical side belong to the same entity on cyber side.

Obviously, the construction of entity-based data view is a key problem of CPS which can improve the usability of data obtained from CPS. Still continued from the example above, a typical task is to return the information (temperature and humidity) of all rooms. Given the data from physical side, we need efficient methods to find all temperature and humidity pairs for the rooms.

In this article, we construct the entity-based data view based on the topology structure of CPS network. The intuitive idea can be explained by the following observations. First, if node $u$ is near node $v$ in $T_G$, it is highly possible that they are used to monitor the same entity. Second, usually, the sensors are deployed in special strategies for the monitoring task, and the sensors of same entities are also distributed according to special rules in $T_G$. For some special entity $e$, we use routing paths on $T_G$ to represent the sensors related to $e$. For example, the information obtained from room $A$ can be represented by $\{/r/a/b, /r/a/c/d\}$, where $/r/a/b$ means that starting from $r$ we can reach a $b$-type sensor through some $a$-type sensor. Obviously, according to the routing path information, it is easy to extract entity-based data view. However, real CPS may be composed of huge amounts of sensor nodes; it is impossible to enumerate all such paths, and we need to find a concise representation method for the routing paths.

**Definition 1**

**Path expressions.** Path expressions can be defined by the syntax $p :: = E \mid p/ e \mid p// e$. Here, $e$ belongs to the set $L$, $/$ means directly connecting two sub-expressions, and $//$ means connecting two sub-expressions using a line of elements with arbitrary length.

In a tree $T$, two nodes $n_1$ and $n_2$ satisfy “/” (“///”) relation if they have parent–child (ancestor–descent) relation, and the semantics of a path expression $p$ can be defined similarly by iteratively applying the two basic operations. Given query $p$ and node $n$, $n(p)$, or $p(n)$ is used to represent the nodes which can be reached from $n$ by matching path $p$. Then, the entity can be represented concisely as follows.

**Definition 2**

**Entities in CPS.** Given a CPS network $G$, we can use the following expression to represent and extract entities in CPS

$$\varphi = (Q, \{P_1, \ldots, P_k\})$$

where $Q$ and $P_j$s are all path expressions. $Q$ is called target expression, and $P_j$s are called attribute expressions. We use target ($\varphi$) and attribute ($\varphi$) to represent them, respectively.

Intuitively, the entity representation in CPS is routing-oriented, the sophisticated routing protocols are based on paths in network, and the information gathered in such networks can be treated to be organized based paths. Thus, the path expression $Q$ can find sensor nodes which can collect information of some entity by forwarding and collecting messages to its children nodes, and the expressions $\{P_j\}$ can tell us how to find all kinds of information of the entity.

**Path expression relaxation**

Usually, the topology structure of CPS network is built by network protocols in randomized way; thus, the entities in CPS have similar path structures but not necessary same. To support efficient entity extraction from CPS data, we need the help of relaxation techniques for path expressions. Next, we will introduce the relaxation techniques used in this article.

The problem of path expression relaxation is to return a set of expressions with similar semantics of the given path expression. In this article, it is implemented by applying rewriting rules repeatedly over the given path. The rewriting rule set includes the following:

- **Axis relaxation:** Replace one “/” operator in the path with “///,” denoted as $r_1$. 

• **Label relaxation**: Replace one node’s label with another label available in the data, denoted as $r_2$.
• **Star relaxation**: Replace one node’s label with $\ast$, denoted as $r_3$.

It should be noted that the rules used in this article are different from the ones in previous works of relaxation. Previous works focused on finding a variant $q'$ of original expression $q$ such that $q'$ has non-empty result and $q \subseteq q'$. With that goal, $r_2$ was not considered by them because it cannot make sure that generated expression is larger than $q$.

In the following part, the rule set composed of those three rules is denoted as $R$. For path expression $q$ with length $n$, $i \in [1, n]$, and $R$, $r_i(q, i)$ means replace the operator in position $i$ with $\sim$; $r_2(q, i, l)$ means replace the label in position $i$ with $l$, and $r_3(q, i)$ means replace the label in position $i$ with $\ast$. Those three rules can be unified as the representation $(r_i, pos, label)$ of triple operation executed on some expression, where $r_i$ is the rule, $pos$ is the position $r_i$ is applied, and $label$ is the parameter of $r_2$. When $i$ is not 2, the triple can be shortened as $(r_i, pos)$. A rule sequence is defined as $\vec{r} = \tau_1 \tau_2 \cdots \tau_k$ where each $\tau_x$ for $x \in [1, k]$ represents some triple $(r, pos, label)$.

Given $q_1$ and $q_2$, it is denoted as $q_1 \xrightarrow{\vec{r}} q_2$ if $q_2$ can be obtained by applying rules in $\vec{r}$ one by one in order. It is denoted as $q_1 \xrightarrow{\vec{r}} q_2$ if $q_2$ can be obtained by applying one rule $r_i$ in $R$ at most once over $q_1$. It is denoted as $q_1 \overset{\mathcal{R}}{\longrightarrow} q_2$, if $q_2$ can be obtained by applying rules in $\mathcal{R}$ multiple times over query $q_1$. For a special expression $q$, if $q \overset{\mathcal{R}}{\longrightarrow} q'$, $q'$ is said to be an extended path of $q$. Denoted as $E(q)$, all extended paths of $q$ can be defined as $E(q) = \{q' | q \overset{\mathcal{R}}{\longrightarrow} q'\}$.

### The framework $\mathcal{P3E}$ for entity extraction in CPS

In this section, a framework called $\mathcal{P3E}$ for solving entity extraction problem in CPS will be introduced. Using the path expressions introduced above to represent entity locations and attributes, the problem can be solved step by step after answering the following questions. (1) What is entity and entity location in $\mathcal{P3E}$? (2) How does $\mathcal{P3E}$ represent the locations? (3) How can we find the entity locations and their attributes? After providing an overview of $\mathcal{P3E}$, the formal definition of entity extraction problem is stated finally.

### Entity locations in CPS data

Generally, there are no universal definitions for entity and entity locations in CPS data; in this part, the meanings of those two concepts in this article are explained.

For CPS data, entity has no definitions formal enough to be directly used. Even worse, in general, there are no general definitions for entity, and different research areas usually use different definitions. (For example, in Wiki, entity is defined as something that has a distinct and separate existence. In a DBMS (database management system), entity is defined as a thing in the modeled world, and in a system, entity means a component.) In this article, an entity in CPS data is assumed to be a sub-tree fragment, whose nodes store the information about the corresponding real-world entity. Considering the CPS data $T$ shown in Figure 1 again, those sub-trees enclosed with dashed lines are entities stored in $T$.

For a special entity in CPS $G$, which actually means a sub-tree of $G_T$, entity location is defined as the root node of the sub-tree. For example, for the entity $T_2$ shown in Figure 1, node 2 is the location of $T_2$, it can be seen as the anchor node of entity $T_2$, and the other nodes under it in $T_2$ represent the attributes of this entity.

### Representing entity locations and attributes

Based on the explanation of entity location, the essential problem of entity extraction in CPS is to extract one set of interesting locations (nodes) in CPS data. In $\mathcal{P3E}$, path expression is used to provide condensed representations for entity locations. For example, for the data tree $T$ shown in Figure 1, the path expression/center/infrared may represent one room whose information (temperature, humidity, and so on) is monitored by the CPS. Although there may be a large number of nodes of same type, only one path expression is enough to cover all of those entity locations. Using path expressions to cover locations provides a method to present entities concisely and takes the advantage of sophisticated algorithms for path expression evaluation. Similarly, the attributes of entities can also be represented by path expressions as discussed above.

### Extracting entities from CPS data

Since entity locations are represented by path expressions, the problem of extracting entities from CPS data is indeed the problem of discovering path expressions which can represent entity locations and attributes efficiently. The idea of this article can be summarized as following steps. First, some initial entities (represented by path expressions) of CPS data are generated by experts or processing center. Then, for each entity expression $\varphi$, the set of candidate extended entities, $\mathcal{E}_\varphi$, is obtained by relaxing the expressions in $\varphi$. Finally, for each candidate entity expression $q \in \mathcal{E}_\varphi$, a verification procedure is run to determine whether $q$ is useful.
Generating initial entities. As discussed above, the concept of entities is general and subjective, and usually, it depends on how the CPS data are treated and organized by users. Therefore, we need some initial expressions of entities to tell the system which entities are interests of users. For general CPS, the initial expressions can be provided by expert users. For example, let us consider the data shown in Figure 1 again. Expert users can only select the attribute nodes of some entity they focus on and return the nodes to the system. Suppose that the left infrared nodes are selected. Initial entity expression can be built trivially by enumerating the path from root node to the attribute nodes and using the lowest ancestor node of those attribute nodes as the entity location. That is, \( \varphi = /\text{center}/\text{infrared} \). It is easy to found that the target \( \varphi \) path refers to the interesting entity location, and attribute \( \varphi \) refers to the attributes which describe the entity from different aspects. In some cases, we can generate the initial entities automatically by utilizing additional information. For example, if the CPS is well deployed and the entities focused by users are highly related to the geographic locations, we can collect the initial entities by simply clustering nodes near enough in the geographic view.

Generating candidate entities by relaxation. Using only the naive strategy shown in generating initial entities, there may be interesting entities lost in practical applications. Suppose that the entity expression \( \varphi = (/r/a, /\{b, /c\}) \) is found in the first step. Using \( \varphi \), we can only extract entities locating at \( /r/a \) and containing two attributes nodes under paths \( /b \) and \( /c \). However, because the construction of topology structure of CPS data is usually in randomized way, there may be entities of the same type locating at \( /r/x/a \) with attribute nodes \( /y \), \( /b \), and \( /c \). In this case, the method shown above will lose those useful entities. Additionally, we cannot expect, in that case, expert users can provide all available entity expressions because either (1) the size of such expressions may be too large to be enumerated completely by users, or (2) users may have no enough knowledge about the CPS data to give complete specification.

To solve this problem, \( \mathcal{P}\mathcal{E} \) provides a mechanism to automatically discover the locations of those entities not covered by user specifications, even if users cannot specify enough entities, and the power of this mechanism can be adjusted by users. The idea of \( \mathcal{P}\mathcal{E} \) is using the relaxation techniques.

To complement the user specifications, relaxation techniques are utilized based on the following observation. In real CPS, although the same kind entities may exist under different incoming paths, it is usually that their incoming paths are similar on the syntax structures. Then, for a given entity expression \( q \) generated by the first step, the idea is using relaxation techniques to discover the query \( q' \) with similar syntax structures as candidate entity expressions, which is expected to have similar semantic information too. The rules \( r_1, r_2, \) and \( r_3 \) shown in section “Path expression relaxation” are used in this article to guarantee that relaxed expressions are similar to the original one.

To adjust the power of relaxation, a threshold is used to limit the application of rewriting rules. Intuitively, if rewriting rules are allowed to be applied arbitrarily, given entity expression \( q \), all possible entity expressions can become the candidates, within which there may be entities totally “far” from \( q \) such that they belong to different type from \( q \) and users do not want them. Obviously, the more times the relaxation rules are applied, the less similar the relaxed expression is with original one. Therefore, to ensure that the distance between relaxed expression and the original one is within a certain range, a cost function is defined for the relaxation rules and used to define the distance between expressions. Given \( n \)-length entity expression \( p \) and rules set \( \mathcal{P} \), the cost of applying rules can be defined as \( \text{COST}(r_k(p, i)) = 2 \) for star relaxation operation and 1 for other cases. Based on the rule cost function, a distance measure \( \mathcal{F} \) for two path expressions \( p \) and \( p' \) can be defined.

Definition 3

Distance \( \mathcal{F} \). For path expressions \( p \) and \( p' \), and a rule sequence \( \tau = \tau_1\tau_2\cdots\tau_k \), s.t. \( p \rightarrow p' \), the distance between them is defined as the sum of \( \text{COST} \) for each rule applied in the rewriting sequence. More formally, \( \mathcal{F}_\tau(p, p') \), which is the cost of obtaining \( p' \) by applying \( \tau \) to \( p \), can be defined recursively as follows

\[
\mathcal{F}_\tau(p, p') = \begin{cases} 
\text{COST}(r(p, i)), & \text{if } |\tau| = 1 \text{ and } \tau_1 = (r, i, l); \\
\mathcal{F}_{\tau\{l\}}(p, p') + \text{COST}(r(p'', i)), & \text{if } \tau = \tau_{\{l\}} \cdot \tau_n \cdot p \rightarrow p'' \text{ and } \tau_n = (r, i, l); 
\end{cases}
\]

Since there are many rewriting sequences taking \( p \) to \( p' \), the sequence taking minimal cost for rewriting operations is considered to be the distance between two path expressions. Formally, the distance between \( p \) and \( p' \) can be defined as \( \mathcal{F}(p, p') = \min \{ \mathcal{F}_\tau(p, p') | p \rightarrow p' \} \).

Based on the relaxation techniques and distance \( \mathcal{F} \) introduced above, given path expressions \( p \) for entity locations or attributes determined by user specification, \( \mathcal{P}\mathcal{E} \) can discover a set of candidate entities \( Q \), which are syntactically similar to \( p \) and possibly indicate the locations or attributes of the same kind entities interested by users. To control the quality of generated entities, a threshold \( k \) can be given to limit the distance
between generated expressions and \( p \), which means for \( p' \in Q \), there should be \( F(p, p') \leq k \).

**Example 1.** Let us consider the following example. Suppose that the path expression \( p = /r/a/b \) has been obtained to indicate the entity locations. Using relaxation, we can also find \( p' = /r/a/bb \). However, without the limitation of threshold \( k \), expressions such as \( /r/*/ \) and \( /*/*/ \) can also be found, for which too many non-interesting entities will be extracted too. Setting the relaxation threshold \( k \) as 1, those two expressions can be pruned because they are too “far” from \( p \).

However, even \( k \) is limited, and some entity expressions which are not useful are still generated. Therefore, it is not enough to use only \( k \) to control the quality of relaxed expressions, and further techniques will be introduced to solve such problems.

**Verifying candidate entities.** As shown in Example 1, even if the relaxed entity expressions generated by limiting the \( F \) distance are not larger than \( k \), they still may bring non-interesting entities. Therefore, for each new candidate, further verifications for determining whether it is indeed useful are needed.

The intuitive idea can be explained as follows. In perfect cases, useful entities of same type will have same set of attributes, while in real applications useful entities will have similar set of attributes. Also, to simplify the discussions, it is assumed that the types of sensor nodes are determined by the node labels, and nodes with different labels will belong to different types. Next, a measure function \( M_q \) is proposed to determine whether an entity is similar enough; then, based on \( M_e \), \( M_q \) is proposed to determine whether an entity expression is useful enough.

Given an entity location \( e \) and an initial entity expression \( \psi \), \( M_e \) can be defined over \( e \) and the attribute expressions attribute \( \psi \) of \( \psi \) as \( M_e(e, \psi) = |f(\text{attribute}\psi, e)|/|\text{attribute}\psi| \), where \( f(S, e) \) is the size of attributes of \( S \) appearing in \( e \). \( M_e \) is used together with a parameter \( \alpha \), which is the threshold to determine whether entity \( e \) is the kind described by the attributes of \( \psi \). In detail, if the value of \( M_e \) is smaller than \( \alpha \), the answer is negative; otherwise, the answer is positive.

For candidate entity expression \( \varphi \), key rule \( \psi \), and CPS data \( T \), based on the \( M_e \) and its parameter \( \alpha \), the candidate entity measure function \( M_q \) can be defined to measure the usefulness of \( \varphi \) for entity extraction problem as \( M_q(\varphi, \psi, T_D, \alpha) = |\{e | e \in \text{target}(\varphi(T_D)) \wedge M_e(e, \psi) \geq \alpha\}|/|\text{target}(\varphi(T_D))| \). \( M_q \) is used together with a parameter \( \beta \), which is the threshold to determine whether a given expression \( \varphi \) is proper for the entity extraction problem. In detail, if the value of \( M_q \) is smaller than \( \beta \), the answer is negative; otherwise, the answer is positive.

**Formal definition of entity extraction problem in framework \( P3E \)**

The entity extraction framework of \( P3E \) can be explained as follows. Given CPS data \( D \) and initial entity expression set \( \Sigma \), \( P3E \) is expected to automatically extract entities and return them to users. For each expression \( \varphi \in \Sigma, \text{relaxation module} \) relaxes the entity location expressions generated by \( \varphi \) with constraint \( k \) and then outputs the relaxed candidate location expressions. Then, given a relaxed expression \( p \) and attribute expressions \( \{p_1, \ldots, p_n\} \), taking parameters \( \alpha \) and \( \beta \), the verification module filters the expressions not satisfying \( M_q \) out and returns the suitable expressions to users. The two modules, relaxation and verification, are essential procedures in \( P3E \), and they provide the solution to the entity extraction problem. Then, the entity extraction problem in \( P3E \) can be formalized as the following definition.

**Definition 4**

**Entity extraction problem.** Given CPS data \( T \) and an initial entity expression \( \varphi \), also the three parameters, relaxation constraint \( k \), and \((\alpha, \beta)\) for verification measure \( M_q \), the entity extraction problem is to return the set of entity expressions \( Q \), such that for \( \varphi' \in Q \), (1) the distance between \( \text{target}(\varphi') \) and \( \text{target}(\varphi) \), \( F(\text{target}(\varphi'), \text{target}(\varphi)) \), is not larger than \( k \), and (2) the measure \( M_q(\varphi', \varphi, T_D, \alpha) \) is not smaller than \( \beta \).

Essentially, the entity extraction problem is to find entity expressions most of whose results are similar to initial entities by utilizing relaxation in limited steps. To limit the using of relaxing operations, a parameter restricting the number of relaxation steps is utilized, and to guarantee the quality of expressions obtained further, verifications based on known information are utilized.

**Algorithms for entity extraction of CPS data**

In this section, the entity extraction algorithms of this article are introduced, which indeed implement the entity extraction of \( P3E \). First, a naive implementation is given as the baseline. Then, a method based on automaton techniques is provided.

**Naive algorithm**

A naive algorithm for entity extraction problem is shown in Figure 2. The main idea of NAI2E is to (1) enumerate all relaxed expressions by applying rewriting
rules on the original expression $q$ with relaxing constraint $k$ and then (2) for each relaxed expression $q'$, evaluate $q'$ over data tree $T$, collect the information about locations extracted by $q'$, and determine whether $q'$ is an interesting location expression.

The details of NAI2E can be explained as follows. First, according to the initial entity expression $\varphi$, target expression $q$ describing the locations of entities and attribute expressions $Q$ describing the attributes of the entity are generated (lines 1–2). Then, procedure ERELAX is invoked to obtain the relaxed expression set $C$ satisfying the constraint $k$ (line 3). Third, procedure EVer is invoked to filter out those expressions not satisfying $M_q$ (line 4). Finally, the result $Q$ is returned (line 5). For procedure ERELAX, a FIFO (first-in first-out) queue $FQueue$ is maintained to store the current relaxed expressions. First, $C$ is initialized to be empty, and target expression $q$ is inserted into the queue (lines 1–2). Then, following operations are executed until $FQueue$ becomes empty (lines 3–12). (1) In each round, an expression $q'$ in $FQueue$ is polled (line 4). (2) For $q'$, relaxed expressions are obtained by trying
all possible relaxing operations (lines 5–11). The relaxa-
tions having no effects are ignored (lines 8–9). If some
relaxed expression \( q' \) has \( F \) value not larger than \( k \), it
will be inserted into \( FQueue \) (lines 10–11). Here, an
abbreviated form of the triple relaxation is used (line
7), and if \( r \) is \( r_2, \ r_3(q', \ i) \) means the set \( \{ \allowbreak r_2(q', \ i, \ l) \} \) is a
valid label in \( Ty \). (3) \( q' \) is added to the result \( C \) (line
12). Finally, relaxed expressions set \( C \) is returned (line
13). For procedure EVER, the candidate expressions in
\( C \) are verified one by one, and the collection of expres-
sion evaluation results is used to compute the \( M_e \) and
\( M_q \) values which are the measures for filtering the can-
didate expressions. For each expression \( q' \in C \), follow-
ing operations are executed. First, a location set \( gres \) is
obtained by evaluating \( q' \) on \( T \) (line 2), and a counter \( c \)
is initialized to be 0 (line 3). Then, the \( M_e \) value of each
node \( n \in gres \) is evaluated by checking expressions in
attribute expressions \( Q \), and by comparing \( M_e \) with \( \alpha \),
counter \( c \) recorded the number of satisfying nodes
(lines 4–8). Finally, the value \( M_q \) is computed and com-
pared with \( \beta \) to determine whether \( q' \) belongs to
the final result (lines 9–11).

**IMP2E algorithm**

In this part, the IMP2E algorithm is proposed to solve
entity extraction problem. Comparing with NAI2E, in
the relaxation module, the number of candidate expres-
sions generated is reduced by utilizing the normal form
relaxations, and in the verification module, the cost is
reduced by sharing the computation of multiple candi-
date expressions. An overview of IMP2E will be given
first and then two procedures invoked by IMP2E will
be introduced in details.

**Overview of IMP2E.** As shown in Figure 3, IMP2E algo-
rum can be divided into three parts. First, by given
initial entity expression \( \varphi \), the target expression and
attribute expressions are generated (lines 1–2). Then,
RERELAX is invoked to generate candidate expres-
sions by relaxing operations (line 3). Here, by relaxing
expressions into the standard forms, the redundant
expressions in the candidates can be reduced. Finally,
SEVER is invoked to verify all candidate expressions,
and the result is returned (lines 4–5). Here, by sharing
the computations between expressions, the efficiency of
verification procedure is improved.

**Reduced expression relaxation.** According to the rewriting
rules in \( \mathcal{R} \), for special expression \( q \), the naive idea to
find all relaxed expressions with \( F \) constraints is to
enumerate all possible rewriting sequences, just like in
NAI2E algorithm. Such a simple strategy may produce
redundant candidate expressions because for given
expression \( q \) and some relaxed expression \( q' \), there may
be several rule sequences which are able to transform \( q \)
to \( q' \). For example, given \( q \) as \( /a/b \) and \( q' \) as \( //*/c \), con-
sidering the rule sequences defined as follows, obvi-
ously, both of these two different rule sequences can
transform \( q \) to \( q' \). When relaxing expressions, such rule
sequences let the relaxation procedure return redu-
dant relaxed expressions

\[
\bar{r}_1 : q \xrightarrow{r_1} //a/b \xrightarrow{r_1} //*/b \xrightarrow{r_2} \bar{c} q' \\
\bar{r}_2 : q \xrightarrow{r_2} /a/* \xrightarrow{r_1} //a/* \xrightarrow{r_1} //*/b \xrightarrow{r_2} \bar{c} q'
\]

To avoid the redundant candidate expressions, a
standard form of rule applying sequence is given. For
rule triple \( \tau = (r_i, \text{pos}, \text{label}) \), let \( \tau.r = i \), \( \tau.pos = \text{pos} \),
and \( \tau.label = \text{label} \). A rule sequence \( \bar{r} = \tau_1 \tau_2 \cdots \tau_k \),
where the rules comes from \( \mathcal{R} \), is standard if (1) for
\( 1 \leq i \leq j \leq k \), \( \tau_i.pos \leq \tau_j.pos \); (2) for \( 1 \leq i \leq j \leq k \), if
\( \tau_i.pos = \tau_j.pos \), then \( \tau_i.r < \tau_j.r \); and (3) there is no
\( 1 \leq i \leq j \leq k \) such that \( \tau_i.pos = \tau_j.pos \), \( \tau_i.r = 2 \), and
\( \tau_j.r = 3 \).

Following properties guarantee that the standard
rule sequence can reduce the redundancy when relaxing
expressions.

**Proposition 1.** For any expression pair \( q \) and \( q' \) such that
\( q \xrightarrow{r} q' \), there is one and only one standard rewriting
sequence \( \bar{r} \) generating \( q' \) from \( q \).

**Proposition 2.** For any expression pair \( q \) and \( q' \) such that
\( q \xrightarrow{r} q' \), where \( \bar{r} \) is a standard rewriting sequence, we
have \( \mathcal{F} \rightarrow (q, q') = \mathcal{F}(q, q') \). That is, standard rule
sequence produces the shortest relaxing plan for given
expressions.

Based on the properties above, procedure
RERELAX shown in Figure 3 will generate the candi-
dates by relaxing expressions according to the standard
rewriting sequence.

**Sharing expression verification.** For given input parameters
\( C, Q, T, \alpha, \beta \), expression verification procedure is to
determine whether a candidate expression \( q_i \in C \)
satisfies the conditions defined by \( M_q \), that is, the ratio
of nodes satisfying \( M_e \geq \alpha \) to all nodes of \( q_i \)’s result is

---

**Figure 3.** IMP2E algorithm, part a.

---

**Procedure Imp2E (Improved Expression Generation)\**

Input: An initial entity expression \( \varphi \), CPS data tree \( T \), parameters \( \alpha, \beta \) and \( \delta \).
Output: An PE expression set \( Q \).

1. \( q \leftarrow \text{target}(\varphi) \);
2. \( Q = \text{attribute}(\varphi) \);
3. \( C = \text{RERELAX}(Q, \delta) \);
4. \( Q = \text{SEVER}(C, Q, T, \alpha, \beta) \);
5. return \( Q \).
not smaller than $\beta$. Therefore, for each node $n \in T$, it needs to determine (1) whether $n$ satisfies the target expression $q_i$, and (2) for some attribute expression $qa_j \in Q$, whether $n$ has descendant nodes satisfying $qa_j$. Let $g_i : T_n \rightarrow \{0,1\}$ represent the value whether a node satisfies $q_i$, where integer $i \in [1,|C|]$, and $f_j : T_n \rightarrow \{0,1\}$ represent whether a node satisfies $qa_j$ where integer $j \in [1,|Q|]$. Considering that trivially evaluating $g_i$ and $f_j$ is expensive, the idea of this article is to improve the efficiency by sharing the evaluations between those functions. For each $g_i$, a corresponding automaton $A_{g_i}$ is built, and the computations between $A_{g_i}$s are shared by merging states of the automatons. For $f_j$s, the two strategies, TopDown and BottomUp, are considered.

In the following parts, first, the evaluation of $g_i$s is introduced; then, for evaluating $f_j$s, a TopDown strategy and a computation sharing–based BottomUp strategy are introduced, respectively, and discussed; finally, the algorithm for verification module is given.

**Evaluating target expressions.** Since the expressions evaluated in this step are all produced by the relaxation module, expressions having similar structures are very common among them, which brings chance for us to share the computations. For example, for $/a/b$ and relaxation constraint 1, expressions $/a/a$, $/a/c$, and $/a/d$ can be generated, and the first step /a can be evaluated only once and shared among those expressions. To formalize this idea, an automaton-based technique for evaluating target expressions is introduced.

The procedure for evaluating target expressions is composed of three parts. (1) For each target expression $q_i$, a nondeterministic finite automaton (NFA) $A_{q_i}$ is built using the method given by Diao et al.\textsuperscript{9} Briefly, the idea is to use the transformations shown in the following to build automaton. Each step of given expression $q_i$ is converted to the basic automatons first and then those basic automatons are linked together to form the $A_{q_i}$. For the accept states of $A_{q_i}$, an integer value $i$ is attached for indicating which expressions are accepted. (2) To share the redundant computations among expressions, different automatons of $q_i \in C$ are merged into $A_C$ by combining same prefix of expressions using the techniques in the work by Diao et al.\textsuperscript{9} Besides, another merging strategy utilizing the characteristics of the relaxed candidate expressions is applied before the prefix-sharing techniques. Considering the expression $/a/b/c$ and relaxation constraint 1, expressions $/a/a/c$, $/a/d/c$, and $/a/e/c$ can be generated. Not only the prefix of those expressions but also the postfix can be shared. Such expressions having chance to share both prefix and postfix are very common in the expressions generated by relaxations and only using prefix-sharing techniques is not enough. Thus, a special merging strategy, Similar Automatons Merging strategy, for such expressions is utilized in this article, which can be explained using an example. Consider the three expressions generated by $/a/b/c$ above, and they can be represented as just one regular expression $a(a + d + e)c$, which can be identified by a single NFA. Such a technique can merge the automatons of those very similar expressions into a condensed one. After that, the produced automaton will be merged with others using prefix-sharing techniques. Additionally, when merging the automatons, the associated information about expression index in accept states is also merged, so in $A_C$, each accept state has a set of expression indexes attached. (3) Finally, given the NFA $A_C$ generated above, by making a preorder traversal on tree $T$, $A_C$ can be evaluated. For node $n$, during the traversal, $g_i(n)$ is assigned to 1 if $A_C$ accepts at $n$, and $i$ is one of the associated expression indexes at the accepting state. There have been many sophisticated algorithms for this evaluation problem, and in this article, the lazy state method proposed by Diao et al.\textsuperscript{9} is used to evaluate the NFA $A_C$.

**Evaluating attribute expressions.** To evaluate the attribute expressions $Q$, both TopDown and BottomUp methods are considered and discussed in this part.

**TopDown-Evaluation.** This strategy uses the similar idea of evaluating target expressions. Intuitively, for attribute expression $p$ and each node $n$, to determine whether $n$ has $p$ is equivalent to determine whether $p(T_n)$ is empty. It can be done by executing a preorder traversal and evaluating $p$ on the sub-tree $T_n$. In detail, when visiting $n$, automatons for evaluating attribute expressions in $Q$ are initialized, and the states are attached to $n$. Then, those automatons are evaluated when visiting the nodes in sub-tree $T_n$. On some node, if the automaton for attribute expression $p_j$ accepts, the value of $f_j(n)$ is set to 1. After all nodes in $T$ have been visited, all values of functions $f_j$ $j \in [1,|Q|]$ are assigned correctly. It will take expensive cost if $f_j$ values of all nodes in $T$ are calculated. To optimize the evaluation cost of attribute expressions, let us consider the function $V$ again. For some node, if $g_i$ is 0, $f_i$ has no effects on the value of $V$. Therefore, when making the preorder traversal, only those nodes accepted by $A_C$ evaluate the automatons for attribute expressions. Such a strategy can reduce the size of nodes to be evaluated from $|T|$ to $|C(T)|$, where $C(T)$ is defined as $\{n | n \in T \wedge (\exists q \in C, n \in q(T))\}$.

**BottomUp-Evaluation.** This strategy evaluates attribute paths when making a postorder traversal. The intuitive idea can be explained by following example. Given an attribute path $p = /a//b$, it is said a node $n$ satisfies this condition if there is a path $L$ between $n$ and some leaf node $l$ such that $p(n)$ is not empty when being evaluated on $L$. If $L$ is processed in reverse order, that condition can be checked by
evaluating the expression $p' = /[b]/a/*$ on the path between $l$ and the root node and detecting whether $n \in p'(l)$. In detail, when visiting a new leaf node during the postorder traversal, automatons of the reverse variants of attribute expressions are initialized, and the states are attached to the node. Then, when visiting an internal node, for each child node $c$, each automaton is evaluated taking current node as input and states associated with $c$ as current states. After that, the union of all result states is attached to the current node. Thus, for some node $n$, when visiting $n$, all $f_j$ values can be computed by evaluating the automatons and checking whether its result states contain the accept ones. The computation sharing idea is reflected by the union operations of states. In fact, those states come from automatons for different branches, and it seems that they cannot be used by other automatons. However, because the attribute expressions are evaluated in an existential way, no matter which branch a state comes from, the current node can utilize it and continue the automaton evaluations. Following analysis will show what effects such a sharing strategy achieves.

Since neither of the two strategies can dominate the other one, in practice, to choose which one depends on the applications, essentially, depends on the values of the maximal size of open nodes and the selectivity of target expressions.

**Algorithm for sharing query verification.** To concatenate the evaluations of target and attribute expressions together and show how the related information is collected and compared with parameters $\alpha$ and $\beta$, the details of a BottomUp version of procedure SEVER are introduced in this part. The TopDown version is similar and omitted here. As shown in Figure 4, for given data $T$, SEVER visits $T$ using a depth-first access method and processes the related nodes. Before that, initialization of related data structures is finished (lines 1–5). First, for target expressions set $C$, the automaton $\mathcal{A}_C$ is constructed (line 1–2). Then, for each attribute expression in $Q$, the automaton for the reverse variant is constructed (lines 3–4). Finally, $Q$ for storing the result locating expressions is initialized using empty set (line 5). After that, nodes in $T$ will be processed one by one (lines 6–16). First, a states stack $S_C$ is initialized with the start state of $\mathcal{A}_C$ (line 6), and for each target expression $q_1$, two counters $C^1_q$ and $C^2_q$ are set to 0 (lines 7–8), where $C^1_q$ is the number of nodes satisfying $q_1$ and $C^2_q$ is the number of nodes satisfying both $q_1$ and $\mathcal{M}_e \geq \alpha$. For the first meet of node $n$, the function PUSHINFO is called (lines 11–12). In PUSHINFO shown in Figure 5, the automaton $\mathcal{A}_C$ is simulated by taking the label of node as input and the top element in $S_C$ as current state (lines 1–3), and the result states are pushed into $S_C$ (line 4), and for each target expression $q_1$ accepted in this step, the counter $C^2_q$ is increased by 1 (lines 5–6). Here, label returns the label of given element node, trans returns the states of automaton by running with the first parameter as current states and the second parameter as input, and accept returns the expressions attached to the accept states in given state set. For the second meet of node $n$, the function POPINFO is called (line 14). In POPINFO as shown in Figure 5, the automatons $\{A_{yi}\}$ are simulated. First, $cStates$ is obtained from the top of $S_C$, and current node $n$ and the label $l$ are initialized (lines 1–2). Then, if the current node is a leaf node, start states of automatons $\{A_{yi}\}$ are attached to $n$ (lines 3–5). Otherwise, the states associated with the children nodes of $n$ are collected, and based on those, states automatons $\{A_{yi}\}$ are simulated, and the result states are attached to $n$. During the simulation, $qCount$ is used to record the number of attribute expressions accepted (line 7). After the simulation of automatons, $\mathcal{M}_e$ is calculated by $qCount \times \gamma$ and compared with $\alpha$. If $\mathcal{M}_e$ is not smaller than $\alpha$, for each accepted expression $q_1$ in $cStates$, the counter $C^2_q$ is increased by 1 (lines 8–17). Finally, for each expression $q_1$ in $C$, the measure $\mathcal{M}_q$ is calculated by formula $C^2_q / C^1_q$ and compared with $\beta$, and the satisfying expressions are added into $Q$ (lines 15–16). After all events are processed, the final result $Q$ is returned (line 17).

**Remark 1.** Obviously, the algorithm shown above depends on the parameters hardly. Intuitively, the parameters $\alpha$ and $\beta$ control the quality of entities extracted and the parameter $k$ can adjust the ability of discovering entities with different structures. It is hard for users, even for experts, to decide the values of those three parameters. One possible solution is to use the feedback of the system to adjust the parameter values. The feedback can be organized based on entity quality measures or labels of experts. The entity quality measures used in the experiments can be utilized here. The labels of experts can distinguish “right” and “wrong” entities obtained by the algorithms. According to the experimental results, it is reasonable to design a method for selecting parameter values automatically using the idea of hill-climbing.

**Experimental evaluation**

In this section, an experimental study of the entity extraction algorithm on effectiveness and efficiency is reported.

**Experimental setting**

An area $R$ of $L \times L$ is considered and sensor nodes are deployed in this area. Two different deployment
Procedure SEVER (Sharing Expression Verification)
Input: Candidate expression set $C$, attribute expression set $Q$, CPS data tree $T$, $\alpha$ and $\beta$.
Output: Expressions satisfying the verification conditions $Q$.
1. for each expression $q_i$ in $C$ do construct NFA $A_{q_i}$;
2. build $A_C$ based on $\{A_{q_i}\}$;
3. for each expression $p_j$ in $Q$ do
4. construct NFA $A_{p_j}$ for the reverse variant;
5. $Q = \emptyset$;
6. initialize stack $S_C$ for $A_C$ with start state;
7. for each $q_i \in C$ do
8. initialize $C_{q_i}^1$ and $C_{q_i}^2$ as 0;
9. for each node $n$ met in DFS of $T$ do
10. if $n$ is first met then
11. for each expression $q_k \in \text{PUSHINFO}(curE\cdot\text{getCurrentNode}(),S_C,A_C)$ do
12. $C_{q_k}^1$ ++;
13. else
14. \text{POPINFO}(curE\cdot\text{getCurrentNode}(),S_C,A_C,Q);
15. for each $q_i \in C$ do
16. $M_q = C_{q_i}^2/C_{q_i}^1$; if $M_q \geq \beta$ then $Q = \{q\}$;
17. return $Q$;

Figure 4. IMP2E algorithm, part b.

Procedure PUSHINFO
Input: Node $n$, $S_C$, $A_C$.
Output: An PE expression set $eList$.
1. $eList = \emptyset$;
2. $\text{curStates} = S_C\cdot\text{peek}()$;
3. $\text{nextStates} = A_C\cdot\text{trans}(\text{curStates},\text{label}(n))$;
4. $S_C\cdot\text{push}(\text{nextStates})$;
5. for each expression $q_k \in \text{accept}($nextStates$)$ do
6. \text{eList.insert}(q_k)$;
7. return $eList$;

Procedure POPINFO
Input: Node $n$, $S_C$, $A_C$, $Q$.
Output: TRUE.
1. $cStates = S_C\cdot\text{pop}()$;
2. $l = \text{label}(n)$;
3. if $n\cdot\text{isLeaf}()$ do
4. for each expression $p_j \in Q$ do
5. assign start states of $A_{p_j}$ to $\text{states}(A_{p_j},n)$;
6. else
7. $q\text{Count} = 0$;
8. for each expression $p_j \in Q$ do
9. \text{curStates} = $\cup\{s \mid \exists m \in \text{children}(\text{node}) \land (s \in \text{states}(A_{p_j},m))\}$;
10. $\text{nextStates} = A_{p_j}\cdot\text{trans}(\text{curStates},l)$;
11. assign $\text{nextStates}$ to $\text{states}(A_{p_j},n)$;
12. if $\text{accept}($nextStates$) \neq \emptyset$ then
13. $q\text{Count}++$;
14. $M_e = q\text{Count}/|Q|$;
15. if $M_e \geq \alpha$ then
16. for each expression $q_k \in \text{accept}(cStates)$ do
17. $C_{q_k}^2$ ++;
18. return TRUE;

Figure 5. Functions used by SEVER.
strategies are considered to produce different types of CPS data. One strategy is that all nodes are randomly deployed in uniform way within this area, and the corresponding CPS data are called Type-I data. Based on the first strategy, the second one can be obtained by adding additional sensor nodes with different deployment method into the area. Added nodes are assumed to have enough powerful communication abilities, and they can communicate with each other in the range of \( R \). The CPS data with respect to the second strategy are denoted by Type-II data. It is easy to verify that Type-I data tend to have relatively large depth and regular structure, while Type-II data tend to have small depth and the fan-out of some nodes will be very large. The entity extraction algorithm, IMP2E, is implemented in JAVA, compiled in JDK 1.7, and all experiments were performed on a Windows 7 machine with Intel Core i5 Duo CPU with 2.4 GHz and 4 GB RAM. All experimental results involving the time cost were obtained by running algorithms 5 times and taking the average value.

**Experimental results and discussions**

**Effectiveness.** The effectiveness of the entity extraction algorithm referred in this article indicates its ability of extracting entities correctly. As usually did, precision and recall coefficients are used to measure the effectiveness. Besides, the \( F_1 \)-measure for combining them defined by

\[
F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}
\]

is also reported to comprehensively compare the performances of the algorithm under different settings. The three functions are often used in many fields (e.g. information retrieval\cite{10} and machine learning\cite{11}) for measuring the performance of querying, searching, classification, and so on. The following factors affecting the algorithm performance are considered in this part: the relaxation constraint \( k \), the threshold \( \alpha \), and \( \beta \).

**Type-I data.** For Type-I data, we generated 1M CPS data by adjusting the network simulation parameters. Entities are selected by users manually and the corresponding trivial rule is taken as the input of entity extraction algorithm. Fixing \( k = 3 \) and adjusting \( \alpha \) and \( \beta \), the algorithm is run and the results are recorded. As shown in Figure 6(a), we have following observations about the precision results of Type-I data. (1) The two parameters \( \alpha \) and \( \beta \) indeed affect the performance of the entity extraction algorithm, when they are changed from 0.1 to 1.0 and the precision of algorithm changed from 63\% to 100\%. (2) Precision of the algorithm can be improved by increasing either one parameter; when fixing one of the two parameters and increasing the other one, the precision ratio gets higher. (3) On the aspect of precision, Type-I data can be processed very well by setting proper parameters. As shown in Figure 6(e), we have following observations about the recall results of Type-I data. (1) The two parameters \( \alpha \) and \( \beta \) indeed affect the performance of the entity extraction algorithm; when they are increased, the recall of the algorithm also increased. (2) Recall of the algorithm can be improved by relaxing either one parameter; when fixing one of the two parameters and decreasing the other one, the recall ratio gets higher. (3) On the aspect of recall, Type-I data can be processed very well by setting proper parameters. As shown in Figure 6(e), we have following observations for Type-I data on measure \( F_1 \). (1) The performance of entity extraction algorithm can be affected by the two parameters, and the \( F_1 \) ratio ranged from 0 to 1. (2) For a fixed \( \alpha \) (or \( \beta \)) and increasing \( \beta \) (or \( \alpha \)), the \( F_1 \) ratio shows the up-then-down trend roughly. (3) The best \( F_1 \) ratio, that is, the best performance of algorithm, occurs when \( 0.5 \leq \alpha \leq 0.7, 0.7 \leq \beta \leq 0.9 \) or \( 0.3 \leq \alpha \leq 0.5, \beta \geq 0.7 \). This means that, if the parameters are taken properly, the entity extraction algorithm can process Type-I data very well on the task described in the experiments.

**Type-II data.** For Type-II data set, 1M CPS data are also generated and same experimental settings are prepared. The precision, recall, and \( F_1 \) results are shown in Figure 6(b), (d), and (f), respectively. The observations can be summarized briefly as follows. (1) Over Type-II data, two parameters can affect the performance of algorithm on all those three measures. (2) When \( \alpha \) and \( \beta \) become larger, both precision and recall ratio changes monotonically, but \( F_1 \) is a little different. When \( \alpha \) takes a small value and \( \beta \) takes a large one, the \( F_1 \) performance still keeps very well. This is because Type-II data set is generated by randomized methods, different entities have very special substructures, and even if only a few attributes are utilized, they can still be distinguished well. (3) By setting parameters properly, entity extraction algorithm can process Type-II data set well.

**Efficiency.** Efficiency is measured by the time cost taken by the algorithm in this article. Usually, the efficiency of algorithm determines whether it can work in practical applications, especially when the data size gets large. In this part, four dimensions affecting the algorithm efficiency will be considered, data size, \( k \), \( \alpha \), and \( \beta \).

**Data size.** For the dimension of data size, a series of combinations of the other three parameters is considered to test the performance of entity extraction algorithm when scaling the size of data. First, construct the data set with different sizes ranging from 10 to
100 MB. For Type-I data, the data sets are called to be I10M, I20M, and so on. For Type-II data, the data sets are called to be II10M, II20M, ..., II100M. Then, for different parameter combinations \( \{k = 1, k = 2, k = 3\} \times \{\alpha = \beta = 0.1, \alpha = \beta = 0.5, \alpha = \beta = 1.0\} \), the entity extraction algorithm was run over those two kinds of data sets, and the time cost results are reported.

As shown in Figure 7(a), the time cost results of Type-I data set have following three observations. First, as the data size changes from 10 to 100 MB, the time cost increases linearly for all parameter combinations. Second, for the same setting of \((\alpha, \beta)\), when \(k\) increases, the time cost gets larger, and this is because an increase in \(k\) will let algorithm consider more relaxations which will take more time cost to evaluate those queries. Finally, for fixed \(k\), different combinations of \((\alpha, \beta)\) nearly have no effects on the time performance of algorithm. As shown in Figure 7(b), the time performance of the algorithm over Type-II data has the similar observations. From the experimental results and observations above, it can be found that entity

Figure 6. Effectiveness performance when \(k = 3\): (a) Type I + precision, (b) Type II + precision, (c) Type I + recall, (d) Type II + recall, (e) Type I + \(F_1\), and (f) Type II + \(F_1\).
extraction algorithm scales well with data size under different combinations of the other three parameters.

**Relaxation constraint k.** For the dimension of relaxation constraint $k$, we fixed the data size and took several combinations of $(\alpha, \beta)$ and then tested the scalability of the algorithm to $k$. First, fix the data size as 50 MB, so the data sets I50M and II50M are used in this part. Then, for parameter combinations \( \{\alpha = \beta = 0.1, \alpha = \beta = 0.5, \alpha = \beta = 1.0\} \), the entity extraction algorithm was run for $k$ from 1 to 5 on I50M and II50M, and the *time cost* results are reported.

As shown in Table 1, the *time cost* results of Type-I data can produce following observations. First, as the constraint $k$ changes from 1 to 5, the *time cost* increases almost linearly for different combinations of $(\alpha, \beta)$. Theoretically, in the worst cases, the *time cost* increases faster than linear relation. However, in practical case, usually, most of expressions added by the relaxation using large $k$ produce only empty result and become invalid very early when evaluating them. Therefore,
they do not have obvious effects on the algorithm performance. Second, for fixed $k$, under different combinations of $(\alpha, \beta)$, the algorithms have similar time cost performance. As shown in Table 1, the results of Type-II data have similar observations. From the experimental results and observations above, it can be found that entity extraction algorithm is well scalable when increasing the relaxation constraint $k$.

**Parameters $\alpha$ and $\beta$.** In this part, both Type-I and Type-II data sizes are fixed as 50 MB. To study the effects of $\alpha$, several combinations of parameter $\beta$ and relaxation constraint $k$ are selected, increasing $\alpha$ from 0.1 to 1.0, the entity extraction algorithm is executed and the time cost results are reported. As shown in Figure 8(a) and (b), as $\alpha$ increases, the algorithm performances on the two data sets do not have significant change. Since $\alpha$ is designed to affect the determination of whether an entity is interesting, whose result can hardly affect the time performance, the above observation is normal and expected. For $\beta$, similar experiments are studied, and the results are shown in Figure 8(c) and (d). We have the similar observation that $\beta$ does not affect the efficiency performance significantly.

**Related work**

CPS is thought to be the next generation paradigm of computing, which integrates the abilities of computational intelligence, communication, and control. Monitoring and controlling the world is the final target of CPS. To represent the physical world on cyber side, several kinds of methods appeared have proposed a model for describing the attributes of physical world from multiple dimensions; it supports the analysis of both local and global events, proposing a model for describing the physical entities from time and space views. Based on lattice theory, Tan et al. proposed a model for entity description which supports the compositions of different types of events. Based on the CPS data, the cyber side of CPS can achieve the intelligent management of CPS data, and lots of research work
focus on this area. Based on the techniques of data management and wireless network, Kang and Son\textsuperscript{16} proposed a framework for data service which can integrate data management and network ability well. Loseu et al.\textsuperscript{17} consider the practical communication failure problem in CPS and propose a data management and analysis method supporting failure recovery. Besides the core problem of data management, Liu et al.\textsuperscript{18} identified other challenges of building the processing centers for CPS, including power management, network resource sharing, and so on. Also, Rao et al.\textsuperscript{19} and Parolini et al.\textsuperscript{20} proposed algorithms for power saving in data centers. For network management, the routing management problem is a key of CPS. For the selecting routing nodes problem in dynamic network settings, Lin\textsuperscript{21} proposed a practical method for network optimization which can decide the positions of routers and network connectivity by dynamic replacement of sensor nodes. By considering service priority of client nodes, Lin et al.\textsuperscript{22} proposed a simulation-based method to solve the router node replacement problem, Zhang et al.\textsuperscript{23} proposed a metric that can measure interference precisely among any set of paths in wireless sensor networks, even though these paths have some nodes or links in common, Zhang et al.\textsuperscript{24} proposed an efficient distributed algorithm to solve the MINimum-length-k-Disjoint-Paths (MIN-k-DP) problem in a sensor network. Entity based information management is a core problem of data management not limited to the area of CPS data management. One related area is entity identification which focus on how to extract entities from structured or non-structured data.\textsuperscript{25–27} In practical applications, more protocols\textsuperscript{28,29} are utilized in CPS to achieve efficient management of topology structures, which also needs further process of CPS data to extract entity based information and becomes another important motivation of this paper. The area of schema integration is also related to this paper, since it involves the problem of representing same entities using different organizations. To understand such different organizations, Spaccapietra and Parent\textsuperscript{30} extends the entity-relationship (ER) method to process the relationships between type, entity and attribute. Given the entities collected and the relations between them, sophisticated algorithms\textsuperscript{31–33} are proposed to distinguish entities. The relaxation techniques used in this paper also have been investigated very well. Fazzinga et al.\textsuperscript{34} considered the general definition for the relaxation problem, and proposed the concept of completeness for relaxation. Kanza and Sagiv\textsuperscript{35} considered the flexible and semi-flexible expression definition, which let the expressions can accept the complex schema of data. Koloniari and Pitoura\textsuperscript{36} considered the path expression relaxation problem. For the problem of security of the CPS and network resources consuming, Du et al.\textsuperscript{37} proposed an effective and efficient scheme that can defend such denial of service (DoS) attack on broadcast authentication. Yang et al.\textsuperscript{38} proposed a Polynomial-based Compromised-Resilient En-route Filtering scheme (PCREF), which can filter false injected data effectively and achieve a high resilience to the number of compromised nodes without relying on static routes and node localization. Zhang et al.\textsuperscript{39} proposed a Ring-based Privacy-Preserving Aggregation Scheme (RiPPAS) to resolve the data privacy problem in wireless sensor networks, and a publication method for histograms with outliers under differential privacy, Outlier-HistoPub, is proposed by Han et al.\textsuperscript{40} In practical applications,\textsuperscript{41,42} considering the security of implantable medical devices (IMDs), the writer designed a novel support vector machine (SVM)-based scheme to address the resource depletion (RD) attacks and the secure access control scheme based on basic biometric information of the patient and patient’s iris data. Smart grid is a new type of CPS that will provide reliable, secure, and efficient energy transmission and distribution. Lin et al.\textsuperscript{43} and Yang et al.\textsuperscript{44} proposed detection schemes to accurately identify attacks.

Conclusion

In this article, entity extraction problem is identified as an important problem to reduce the communication cost of extracting entities from CPS data. Based on the idea of using path expressions, a framework for entity extraction problem is proposed to explore the practical techniques. Using the relaxation and verification ideas, a formal definition of entity extraction problem is given in P3E. An efficient automaton-based algorithm, IMP2E, is proposed as a practical implementation of P3E. A detailed experimental study shows the effectiveness and efficiency of IMP2E on CPS data.

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This work was supported by the National Natural Science Foundation of China (Grant No. 61502116) and the Fundamental Research Funds for the Central Universities (Grant No. HEUCFM160601).

References

1. Li J, Cheng S, Gao H, et al. Approximate physical world reconstruction algorithms in sensor networks. IEEE Trans Parallel Distr Syst 2014; 25(12): 3099–3110.
2. Cheng S, Cai Z and Li J. Curve query processing in wireless sensor networks. *IEEE Trans Veh Tech* 2015; 64(11): 5198–5209.

3. Shi T, Cheng S, Cai Z, et al. Adaptive connected dominating set discovery algorithm in energy-harvest sensor networks. In: *Proceedings of the 35th annual IEEE international conference on computer communications (INFOCOM)*, San Francisco, CA, 10–14 April 2016, pp.1–9. New York: IEEE.

4. Guo L, Li Y and Cai Z. Minimum-latency aggregation scheduling in wireless sensor network. *J Combin Optim* 2016; 31(1): 279–310.

5. He Z, Cai Z, Cheng S, et al. Approximate aggregation for tracking quantiles in wireless sensor networks. In: *Proceedings of the 8th international conference on combinatorial optimization and applications (COCOA’14)*, Maui, HI, 19–21 December 2014, pp.161–172. Springer International Publishing.

6. He Z, Cai Z, Cheng S, et al. Approximate aggregation for tracking quantiles in wireless sensor networks. *Theor Comput Sci* 2015; 607(3): 381–390.

7. Amer-Yahia S, Cho S and Srivastava D. Tree pattern relaxation. In: *Proceedings of the 8th international conference on extending database technology: advances in database technology (EDBT ’02)*, Prague, 25–27 March 2002, pp.496–513. London: Springer-Verlag, http://dl.acm.org/citation.cfm?id=645340.650237

8. Amer-Yahia S, Lakshmanan LV S and Pandit S. Flex-path: flexible structure and full-text querying for XML. In: *Proceedings of the ACM SIGMOD international conference on management of data (SIGMOD ’04)*, Paris, 13–18 June 2004, pp.83–94. New York: ACM, http://doi.acm.org/10.1145/1007568.1007581

9. Diao Y, Altinel M, Franklin MJ, et al. Path sharing and predicate evaluation for high-performance xml filtering. *ACM Trans Database Syst* 2003; 28(4): 467–516.

10. Beitzel SM. *On understanding and classifying web queries*. PhD Thesis, Illinois Institute of Technology, Chicago, IL, 2006.

11. Li X, Wang YY and Acero A. Learning query intent from regularized click graphs. In: *Proceedings of the 31st annual international ACM SIGIR conference on research and development in information retrieval (SIGIR ’08)*, Singapore, 20–24 July 2008, pp.339–346. New York: ACM, http://doi.acm.org/10.1145/1390334.1390393

12. Rajkumar RR, Lee I, Sha L, et al. Cyber-physical systems: the next computing revolution. In: *Proceedings of the 47th design automation conference (DAC ’10)*, Anaheim, CA, 13–18 June 2010, pp.731–736. New York: ACM, http://doi.acm.org/10.1145/1837274.1837461

13. Talcott C. Cyber-physical systems and events. In: M Wirasing, JP Banâtre, M Hölzl, et. (eds.) *Software-intensive systems and new computing paradigms: challenges and vision*. Berlin: Springer-Verlag, 2008, pp.101–115, http://dx.doi.org/10.1007/978-3-540-89437-7_6

14. Tan Y, Vuran MC and Goddard S. Spatio-temporal event model for cyber-physical systems. In: *Proceedings of the 29th IEEE international conference on distributed computing systems workshops (ICDCSW ’09)*, Montreal, QC, Canada, 22–26 June 2009, pp.44–50, Washington, DC: IEEE Computer Society, http://dx.doi.org/10.1109/ICDCSW.2009.82

15. Tan Y, Vuran MC, Goddard S, et al. A concept lattice-based event model for cyber-physical systems. In: *Proceedings of the 1st ACM/IEEE international conference on cyber-physical systems (ICCP’09)*, Stockholm, 13–15 April 2010, pp.50–60. New York: ACM, http://doi.acm.org/10.1145/1795194.1795202

16. Kang KD and Son SH. Real-time data services for cyber physical systems. In: *Proceedings of the 28th international conference on distributed computing systems workshops (ICDCSW ’08)*, Beijing, China, 17–20 June 2008, pp.483–488. Washington, DC: IEEE Computer Society, http://dx.doi.org/10.1109/ICDCSWorksshops.2008.21

17. Loseu V, Ghasemzadeh H and Jafari R. Toward power optimization for communication failure recovery in body sensor networks. In: *Proceedings of the 1st ACM/IEEE international conference on cyber-physical systems (ICCP’09)*, Stockholm, 13–15 April 2010, p.198. New York: ACM, http://doi.acm.org/10.1145/1795194.1795222

18. Liu J, Zhao F, Liu X, et al. Challenges towards elastic power management in Internet data centers. In: *Proceedings of the 29th IEEE international conference on distributed computing systems workshops (ICDCSW ’09)*, Montreal, QC, Canada, 22–26 June 2009, pp.65–72. Washington, DC: IEEE Computer Society, http://dx.doi.org/10.1109/ICDCSW.2009.44

19. Rao L, Liu X, Ilic M, et al. MEC-IDC: joint load balancing and power control for distributed internet data centers. In: *Proceedings of the 1st ACM/IEEE international conference on cyber-physical systems (ICCP’09)*, Stockholm, Sweden, 13–15 April 2010, pp.188–197. New York: ACM, http://doi.acm.org/10.1145/1795194.1795220

20. Parolini L, Tolia N, Sinopoli B, et al. A cyber-physical systems approach to energy management in data centers. In: *Proceedings of the 1st ACM/IEEE international conference on cyber-physical systems (ICCP’09)*, Stockholm, 13–15 April 2010, pp.168–177. New York: ACM, http://doi.acm.org/10.1145/1795194.1795218

21. Lin CC. Dynamic router node placement in wireless mesh networks: a PSO approach with constriction coefficient and its convergence analysis. *Inf Sci* 2013; 232: 294–308, http://dx.doi.org/10.1016/j.ins.2012.12.023

22. Lin CC, Shu L and Deng DJ. Router node placement with service priority in wireless mesh networks using simulated annealing with momentum terms. *IEEE Syst J* 2016; 10(4): 1402–1411.

23. Zhang K, Han Q, Cai Z, et al. DOAMI: a distributed on-line algorithm to minimize interference for routing in wireless sensor networks. *Theoretical Computer Science*. Epub ahead of print 6 June 2016. DOI: 10.1016/j.tcs.2016.05.043.

24. Zhang K, Han Q, Yin G, et al. OFDP: a distributed algorithm for finding disjoint paths with minimum total length in wireless sensor networks. *J Combin Optim* 2016; 31(4): 1623–1641.

25. Sarawagi S and Bhamidipaty A. Interactive deduplication using active learning. In: *Proceedings of the eighth ACM SIGKDD international conference on knowledge discovery
26. Cochinwala M, Kurien V, Lalk G, et al. Efficient data reconciliation. *Inform Sci* 2001; 137(1–4): 1–15.

27. Ananthakrishna R, Chaudhuri S and Ganti V. Eliminating fuzzy duplicates in data warehouses. In: *Proceedings of the 28th international conference on very large data bases (VLDB ’02)*, Hong Kong, China, 20–23 August 2002. pp.586–597, http://dl.acm.org/citation.cfm?id=1287369.857554

28. Qiu T, Liu X, Feng L, et al. An efficient tree-based self-organizing protocol for Internet of things. *IEEE Access* 2016; 4: 3535–3546.

29. Qiu T, Lv Y, Xia F, et al. ERGID: an efficient routing protocol for emergency response Internet of Things. *J Netw Comput Appl* 2016; 72: 104–112, http://dx.doi.org/10.1016/j.jnca.2016.06.009

30. Spaccapietra S and Parent C. View integration: a step forward in solving structural conflicts. *TKDE* 1994; 6(2): 258–274.

31. Palopoli L, Sacca D and Ursino D. Semi-automatic, semantic discovery of properties from database schemes. In: *Proceedings of the international symposium on database engineering and applications (IDEAS ’98)*, Cardiff, 8–10 July 1998, p.244. Washington, DC: IEEE Computer Society, http://dl.acm.org/citation.cfm?id=551941.857554

32. Liu X, Li J and Gao H. On the hardness of labeled correlation clustering problem. *Theor Comput Sci* 2016; 609(P3): 583–593, http://dx.doi.org/10.1016/j.tcs.2015.03.021

33. Liu X and Li J. Algorithms and complexity results for labeled correlation clustering problem. *J Combin Optim* 2015; 29(2): 488–501, http://dx.doi.org/10.1007/s10878-013-9607-4

34. Fazzinga B, Flesca S and Furfaro F. XPath query relaxation through rewriting rules. *IEEE T Knowl Data Eng* 2011; 23(10): 1583–1600.

35. Kanza Y and Sagiv Y. Flexible queries over semistructured data. In: *Proceedings of the twentieth ACM SIGMOD-SIGACT-SIGART symposium on principles of database systems (PODS ’01)*, Santa Barbara, CA, 21–23 May 2001, pp.40–51. New York: ACM, http://doi.acm.org/10.1145/375551.375558

36. Kolomiari G and Pitoura E. Distributed structural relaxation of XPath queries. In: *Proceedings of the IEEE 25th international conference on data engineering (ICDE ’09)*, Shanghai, China, 29 March–2 April 2009, pp.529–540. Washington, DC: IEEE Computer Society, http://dx.doi.org/10.1109/ICDE.2009.110

37. Du X, Guizani M, Xiao Y, et al. Defending DoS attacks on broadcast authentication in wireless sensor networks. In: *Proceedings of the IEEE international conference on communications (ICC’08)*, Beijing, China, 19–23 May 2008, pp.1653–1657. New York: IEEE.

38. Yang X, Lin J, Yu W, et al. A novel en-route filtering scheme against false data injection attacks in cyber-physical networked systems. *IEEE T Comput* 2015; 64(1): 4–18.

39. Zhang K, Han Q, Cai Z, et al. RiPPAS: a ring-based privacy-preserving aggregation scheme in wireless sensor networks. *Sensors* 2017; 17(2): 300.

40. Han Q, Shao B, Li L, et al. Publishing histograms with outliers under data differential privacy. *Secur Comm Network* 2016; 9(14): 2313–2322.

41. Hei X and Du X. Biometric-based two-level secure access control for implantable medical devices during emergencies. In: *Proceedings of the IEEE INFOCOM*, Shanghai, China, 10–15 April 2011, pp.346–350. New York: IEEE.

42. Hei X, Du X, Wu J, et al. Defending resource depletion attacks on implantable medical devices. In: *Proceedings of the IEEE global telecommunications conference (GLOBECOM 2010)*, Miami, FL, 6–10 December 2010, pp.1–5. New York: IEEE.

43. Lin J, Yu W, Yang X, et al. On false data injection attacks against distributed energy routing in smart grid. In: *Proceedings of the IEEE/ACM third international conference on cyber-physical systems*, Beijing, China, 17–19 April 2012, pp.183–192. New York: IEEE Computer Society.

44. Yang Q, Yang J, Yu W, et al. On false data-injection attacks against power system state estimation: modeling and countermeasures. *IEEE Trans Parallel Distr Syst* 2014; 25(3): 717–729.