Hindawi
Mobile Information Systems
Volume 2021, Article ID 5510055, 11 pages
https://doi.org/10.1155/2021/5510055

Research Article

Matching Sensor Ontologies with Simulated Annealing Particle Swarm Optimization

Hai Zhu,[1] **Xingsi Xue**,[2] **Aifeng Geng**,[3] **and He Ren**[3]

[1]*School of Network Engineering, Zhoukou Normal University, Zhoukou, Henan 466001, China*
[2]*Intelligent Information Processing Research Center, Fujian University of Technology, Fuzhou, Fujian 350118, China*
[3]*College of Electrical and Power Engineering, Taiyuan University of Technology, Taiyuan 030024, China*

Correspondence should be addressed to Xingsi Xue; jack8375@gmail.com

Received 10 February 2021; Revised 12 March 2021; Accepted 15 March 2021; Published 24 March 2021

Academic Editor: Hsu-Yang Kung

In recent years, innovative positioning and mobile communication techniques have been developing to achieve Location-Based Services (LBSs). With the help of sensors, LBS is able to detect and sense the information from the outside world to provide location-related services. To implement the intelligent LBS, it is necessary to develop the Semantic Sensor Web (SSW), which makes use of the sensor ontologies to implement the sensor data interoperability, information sharing, and knowledge fusion among intelligence systems. Due to the subjectivity of sensor ontology engineers, the heterogeneity problem is introduced, which hampers the communications among these sensor ontologies. To address this problem, sensor ontology matching is introduced to establish the corresponding relationship between different sensor terms. Among all ontology matching technologies, Particle Swarm Optimization (PSO) can represent a contributing method to deal with the low-quality ontology alignment problem. For the purpose of further enhancing the quality of matching results, in our work, sensor ontology matching is modeled as the meta-matching problem firstly, and then based on this model, aiming at various similarity measures, a Simulated Annealing PSO (SAPSO) is proposed to optimize their aggregation weights and the threshold. In particular, the approximate evaluation metrics for evaluating quality of alignment without reference are proposed, and a Simulated Annealing (SA) strategy is applied to PSO's evolving process, which is able to help the algorithm avoid the local optima and enhance the quality of solution. The well-known Ontology Alignment Evaluation Initiative's benchmark (OAEI's benchmark) and three real sensor ontologies are used to verify the effectiveness of SAPSO. The experimental results show that SAPSO is able to effectively match the sensor ontologies.

1. Introduction

In recent years, innovative positioning and mobile communication techniques have been developing to achieve Location-Based Services (LBSs) [1, 2]. With the help of sensors, LBS is able to detect and sense the information from the outside world to provide location-related services. To implement the intelligent LBS, it is necessary to develop the Semantic Sensor Web (SSW) [3, 4]; as the kernel technique of the SSW, sensor ontology is a standard information exchange model, which serves as the basis for different machines to understand semantics and implement the sensor data interoperability, information sharing, and knowledge fusion among intelligence systems.

Due to the subjectivity of sensor ontology engineers, they might make use of various concepts to mean the same thing, or one concept might have more than one meaning, yielding the problem of heterogeneity that affects semantic interoperability between ontologies. Ontology matching [5–7] can be seen as a powerful tool to face this challenge, which has been widely applied in different application domains, such as Artificial Internet of Things (AIoT) [8, 9] and biomedical domain [10]. Sensor ontology matching can be used to discover the semantic relationships of different sensor ontologies, which is capable of determining the correspondences between concepts of heterogeneous sensor. The similarity measure is critical for a sensor ontology matching technique. Due to the complicated semantic relationships among the sensor data, a single similarity measure cannot ensure that it is able to distinguish all the semantically identical entities in any matching context. Thus, several different similarity measures are usually aggregated to enhance the result's confidence. Ontology matching is generally interpreted as how to find a set of appropriate weights and threshold to achieve high-quality ontology alignments.

Particle Swarm Optimization (PSO) [11] is a contributing methodology for determining high-quality ontology alignments [12]. Although PSO converges fast, it is apt to fall into the local optima, which makes it unable to find the global optimal solution. To overcome this drawback, in this work, aiming at various similarity measures, a Simulated Annealing PSO (SAPSO) is proposed to optimize their aggregation weights and the threshold. Particularly, in the process of evolving process, SAPSO introduces a Simulated Annealing (SA) strategy to further enhance the quality of solution. The innovation points of this work are as follows:

(1) An approximate evaluation metric on ontology alignment is proposed, and an optimization model for the sensor ontology meta-matching problem is constructed.

(2) To effectively solve the problem of sensor ontology meta-matching, an ontology meta-matching framework and a SAPSO algorithm are proposed.

This paper is organized as follows. Section 2 presents the related work. Section 3 gives the formal definitions on the sensor ontology and similarity measure. Section 4 constructs the optimization model for sensor meta-matching problem. Section 5 presents the SAPSO. Section 6 shows the experimental results and the corresponding analysis. Finally, Section 7 draws the conclusions and puts forward the future research directions.

2. Swarm Intelligence Algorithm-Based Ontology Matching Technique

In different sensor ontologies, due to the subjectivity of the designer, conceptual name in the sensor system may have different naming methods and definition methods, thus causing the problem of communication inconvenience between different sensor ontologies [13, 14]. Due to the complex intrinsic nature of matching two ontologies, swarm intelligence algorithms, such as PSO, Parallel Compact Cuckoo Search Algorithm (PCCSA) [15], Artificial Bee Colony (ABC) algorithm [16], Firefly Algorithm (FA) [10, 17], and Evolutionary Algorithm (EA) [18, 19], have become effective methods to determine the ontology alignments.

Bock et al. [20] used a discrete PSO algorithm to optimize the results of ontology entity matching, which does not require the computation of large similarity matrices. He et al. [16] used the ABC-based matcher to solve the ontology meta-matching problem, whose results can be proved more effective. Xue et al. [17] proposed a Compact Cooperative Firefly Algorithm- (CCFA-) based ontology matching system, which can improve the search efficiency effectively by using a new mechanism. Xue et al. [12] also proposed a compact multiobjective PSO to solve the matching problem of large-scale biomedical ontology. In addition, they [10] also proposed a Compact Firefly Algorithm (CFA), which greatly reduced the running time and memory consumption by two compact movement operators. Chu et al. [21] first built an ontology model in vector space and proposed a Compact Evolutionary Algorithm (CEA) to solve the ontology matching problem. In this work, we further introduce SA into PSO's evolving process to trade off its exploration and exploitation, which is able to effectively help the algorithm to jump out of the local optima.

3. Sensor Ontology and Similarity Measure

3.1. Sensor Ontology. In the computer and information science field, ontology is a formal list of all the concepts and their relationships in a particular domain [18]. With respect to the SSW, a sensor ontology is used as the most important and extensive model for describing the concepts related to sensors and the IoT [22, 23], such as the sensor's output, observations, observation characteristics, and so on. For ease of description, a set of triples (C, P, I) [24] is used to represent a sensor ontology, where C, P, and I represent the sets of class or concept, property, and instance, respectively. An example of sensor ontology is shown in Figure 1, where an ellipse represents a class and the arrows between the ellipses represent the class' properties. A class is a collection of instances, and each element in I is an instance of a class. Generally, classes, properties, and instances are collectively called entities.

The goal of sensor ontology matching [25] is to establish correspondences between heterogeneous entities and find the set of entity correspondences, the so-called sensor ontology alignment [26]. Here, an entity correspondence is a five-tuple <*id, e, e*$'$*,c, R* >, where $i\,d$ refers to the identifier of entity correspondence; e and $e\prime$ are the entities of two ontologies, respectively; c is the degree of confidence between e and $e\prime$ that can be matched, usually at [0, 1]; and R is the equivalence relationship between e and $e\prime$. The process of matching two sensor ontologies is shown in Figure 2, where O_1 and O_2, respectively, represent the two sensor ontologies to be aligned, A_I is the input alignment, p is a set of parameters, r represents some external resources, and A_N is the obtained alignment.

A similarity measure uses particular information to calculate to what extent two entities are similar. Generally, the similarity measures can be composed of three types, which are described in detail in Section 3.2.

3.2. Similarity Measure

3.2.1. Syntax-Based Similarity Measure. A syntactic measure calculates the string distance between entities of different ontologies. In our work, we use the N-Gram distance, which is an effective syntactic metric in the ontology matching domain. N-Gram has an obvious advantage in comparing the similarity between two strings [27, 28]. Given two

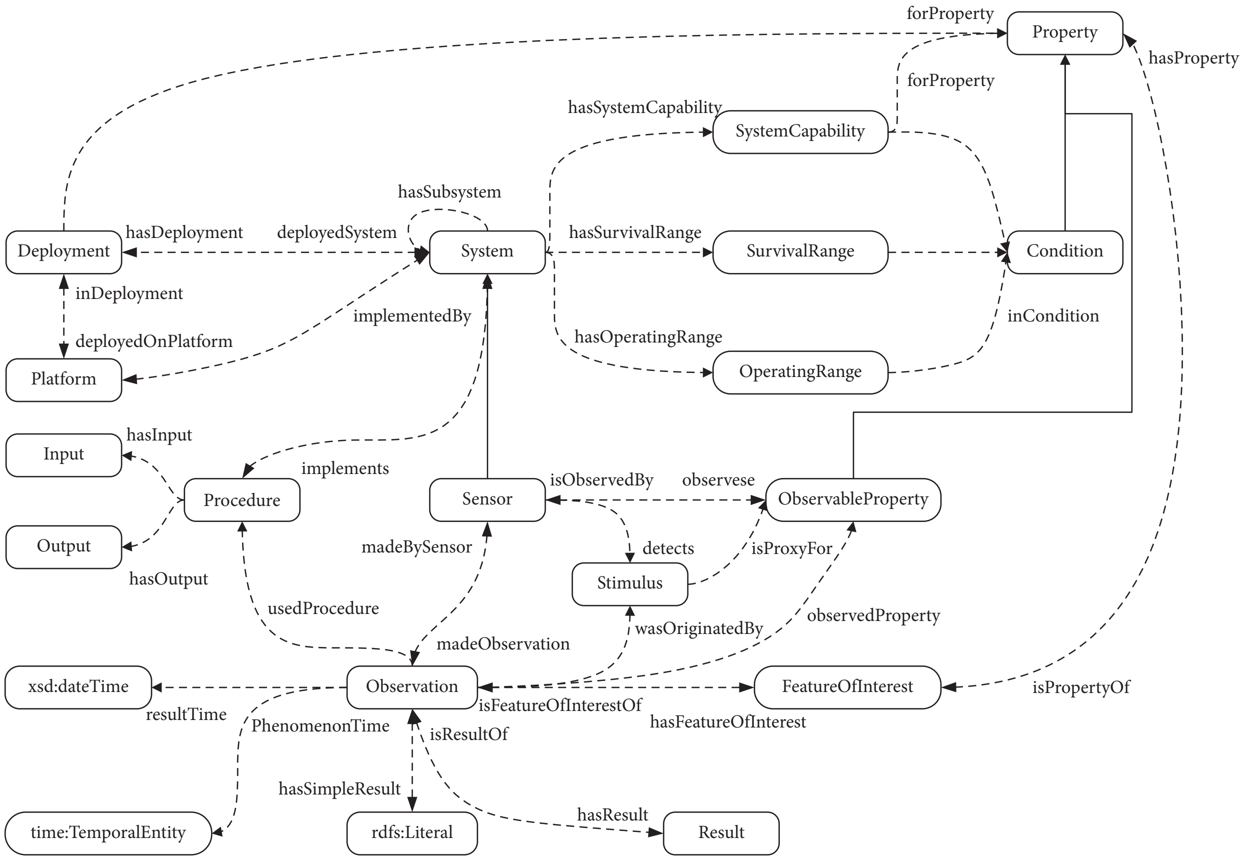

Figure 1: An example of sensor ontology.

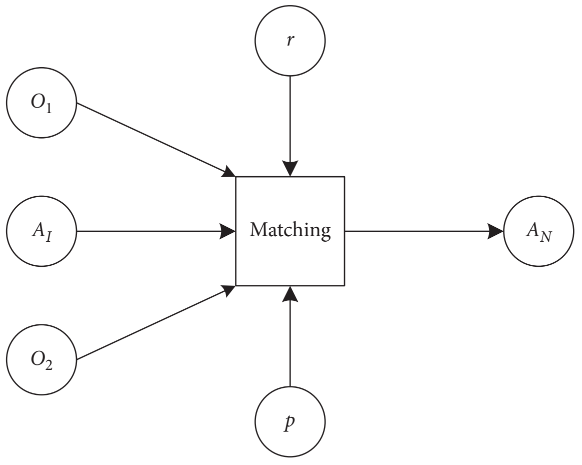

Figure 2: The process of sensor ontology matching.

strings, their N-Gram distance is calculated by measuring the number of common substrings they have. To be specific, the N-Gram distance is defined as follows:

$$N - \text{Gram}(s_1, s_2) = \frac{2 \times C(s_1, s_2)}{n_{s_1} + n_{s_2}}, \quad (1)$$

where s_1, s_2 are two strings to be computed, respectively; N stands for the length of each substring after splitting the original string, which is generally set to 2 or 3 (the lower the value, the higher their similarity; the value of N in this work is 3); $C(s_1, s_2)$ is the number of their common substrings; and n_{s_1} and n_{s_2} are their lengths, respectively.

3.2.2. Linguistic-Based Similarity Measure. Semantic similarity calculates the similarity between entities according to the semantic context. In our approach, we use the Wu–Palmer [29] similarity measure, in particular, it returns a fraction to indicate the degree of similarity between the two words. In this work, we use the WordNet [30], which is an English dictionary based on cognitive linguistics, to calculate the related variables in Wu – Palmer. Here, we choose Wu – Palmer because it is the most popular WordNet-based similarity measure, which calculates the semantic similarity between two strings by considering not only the conceptual depth in the hierarchical semantic structure of WordNet but also their context information. To be specific, it is defined as follows:

$$\text{Wu} - \text{Palmer}(s_1, s_2) = \frac{2 \times \text{depth}(lcs(s_1, s_2))}{\text{depth}(s_1) + \text{depth}(s_2)}, \quad (2)$$

where depth denotes the depth of the word in WordNet's the hierarchical semantic structure and $lcs(s_1, s_2)$ is the closest common parent concept of s_1 and s_2.

3.2.3. Structure-Based Similarity Measure. The main idea of structure-based similarity measure is to determine two entities' similarity through neighborhood entities (superclass and subclass relationship). In general, matched entities have similar structures, that is, they have the same number of superclass and subclass; conversely, if two entities have the same number of superclass and subclass, they are considered similar. In our work, the structure-based similarity measure that we use is called Out-In degree, which calculates the similarity according to the number of superclasses and subclasses of entities in different ontologies, which is defined as follows:

$$\mathrm{Struc}\left(s_1, s_2\right) = \begin{cases} 1, & \text{if two entities have the same number of subclasses and super classes,} \\ 0, & \text{if two entities have different number of subclasses and super classes.} \end{cases} \quad (3)$$

Based on three similarity measures, we can get three similarity matrices, respectively. The similarity matrix is defined as a matrix of $m \times n$, where m and n are, respectively, the number of entities in the original ontology and the target ontology. Each element of the matrix is the similarity value of two corresponding entities determined by the similarity measure. After that, through assigning an aggregating weight for each similarity matrix, we can obtain an aggregated matrix, which is filtered by using a similarity threshold to determine the final matrix. The ontology meta-matching problem can be defined as determining the optimal aggregating weights and the threshold to get a high-quality ontology alignment, which will be formally defined in the following.

4. Sensor Ontology Meta-Matching Problem

In general, optimization problems can be divided into unconstrained optimization problems and constrained optimization problems; the classification criteria are whether there are constraints. In this paper, the problem of sensor ontology meta-matching is modeled as a constrained continuous optimization problem, and its constraints are the sum of the weights and the threshold of the similarity measure, which is explained in more detail in the following. There are three points to consider when building an optimization model: constraint conditions, decision variables, and objective function.

4.1. Constraint Conditions and Decision Variables. For convenience, the process of sensor ontology meta-matching can be described as a seven-tuples (O_1, O_2, n, M,ω, thres, A), where O_1 and O_2 represent the source ontology and target ontology, respectively; n represents the number of similarity measures; M represents a set of similarity matrices;ω is the set of aggregating weights;thres is the similarity threshold; and A is the obtained sensor ontology alignment. In particular, M and ω are, respectively, defined as follows:

$$\begin{gathered} M = \sum_{i=1}^{n} \omega_i \times M_i, \\ \sum_{i=1}^{n} \omega_i = 1, \quad \omega_i \in [0, 1]. \end{gathered} \quad (4)$$

The framework of sensor ontology meta-matching is shown in Figure 3, where $m_1, m_2, \ldots, m_n$ are the similarity measures; $M_1, M_2, \ldots, M_n$ are the similarity matrices; $\omega_1, \omega_2, \ldots, \omega_n$ are aggregating weights on the similarity matrices, respectively;M is the aggregated matrix; A is alignment determined by M; and threshold is the threshold. As can be seen from the figure, the ultimate goal of sensor ontology meta-matching is to find a suitable weight for each similarity matrix and a suitable threshold value for the comprehensive similarity matrix, which is able to ensure the quality of the alignment.

4.2. Objective Function. The quality of the results of sensor ontology meta-matching is usually measured by f-measure, whose value is related to both recall and precision. Traditional recall, precision, and f-measure [9] are defined in equations (5)–(7):

$$\text{recall} = \frac{|R \cap A|}{|R|}, \quad (5)$$

$$\text{precision} = \frac{|R \cap A|}{|A|}, \quad (6)$$

$$f - \text{measure} = \frac{2 \times \text{recall} \times \text{precision}}{\text{recall} + \text{precision}}, \quad (7)$$

where R is the standard alignment; A is the alignment determined by some matching techniques; recall divides true positive correspondences we find by the number of all correct matching pairs, which represents whether the matching results found by us are complete or not; and precision divides the number of true positive correspondences we find by the cardinality of found alignment and represents whether our match is accurate. Their values are between 0 and 1, and the quality of the results is judged by these values, but neither recall nor precision can evaluate the alignment effectively because a high recall value does not mean that our results are accurate and a high precision value does not mean that our results are complete. Therefore, in order to consider the evaluation results of recall and precision, we use f − measure to combine these two indicators. But the traditional evaluation index needs to work with reference matching results, which is impossible to obtain in advance in most cases. To overcome this drawback, in the following, we propose three new quality evaluation metrics [31] on sensor ontology alignment, i.e., ApproximateRecall, Approximate Precision, Approximate Fmeasure, to approximate traditional recall, precision, and f-measure:

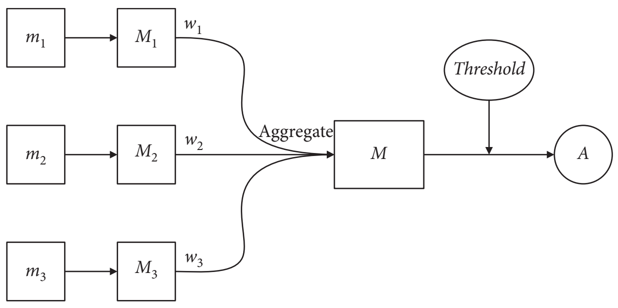

FIGURE 3: Sensor ontology meta-matching framework.

$$\text{Approximate Recall}(M) = \frac{2\times\sum_{i=1}^{m}\sum_{j=1}^{n}\Phi\left(\left|M_{ij}\right|\right)}{m+n},$$

$$\Phi\left(\left|M_{ij}\right|\right) = \begin{cases} 1, & \text{when element } M_{ij} \text{is largest in } ith \text{ row and } jth \text{ column,} \\ \text{in } M \text{ matrix,} & \text{and satisfies threshold criteria,} \\ 0, & \text{otherwise,} \end{cases} \tag{8}$$

where M represents the composite similarity matrix and $|M_{ij}|$ is the value of row i, column j of the composite similarity matrix M.

$$\text{Approximate Precision}(M) = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n}\left|M_{ij}\right|}{\sum_{i=1}^{m}\sum_{j=1}^{n}\Psi\left(\left|M_{ij}\right|\right)},$$

$$\Psi\left(\left|M_{ij}\right|\right) = \begin{cases} 1, & \text{when element } M_{ij} \text{ is largest in } ith \text{ row and } jth \text{ column,} \\ 0, & \text{otherwise,} \end{cases} \tag{9}$$

where M represents the composite similarity matrix and $|M_{ij}|$ is the value of row i, column j of the composite similarity matrix M.

$$\text{ApproximateFmeasure}(M) = \frac{2\times\text{ApproximateRecall}(M)\times\text{ApproximatePrecision}(M)}{\text{ApproximateRecall}(M)+\text{ApproximatePrecision}(M)}, \tag{10}$$

and finally, the objective function we need to optimize is defined as follows:

$$F(M) = \max\text{ApproximateFmeasure}(M). \tag{11}$$

5. Sensor Ontology Meta-Matching with Simulated Annealing Particle Swarm Optimization

5.1. Particle Swarm Optimization. PSO is an algorithm based on swarm cooperation, which is developed by simulating the birds' foraging behavior [32]. PSO initializes a set of random particles (stochastic solutions) and iteratively searches the optimal solution; in each iteration, the particles update themselves by tracking two extremes. The formula for updating the speed and position of PSO is as follows:

$$\begin{aligned} v_i^{t+1} &= v_i^t + c_1\times\text{rand}()\times\left(\text{pbest}_i^t - \text{present}_i^t\right) \\ &\quad + c_2\times\text{rand}()\times\left(\text{gbest}^t - \text{present}_i^t\right), \end{aligned} \tag{12}$$

$$\text{present}_i^{t+1} = \text{present}_i^t + v_i^{t+1}, \tag{13}$$

where i means the ith particle, $i\in[1,n]$, n is the size of population, t is the number of iterations, v is the speed of particles,c_1 and c_2 are learning factors,rand is a random number in $[0,1]$, pbest is the extremum of an individual, the best solution found by the particle itself, gbest is the global extremum, and present is the current position of the particle. Compared with other swarm intelligence algorithms, PSO has such advantage as only one-way information flow, i.e., all the particles are able to converge quickly, but it tends to fall into the local optima. To solve this problem, the SA strategy is introduced into the evolutionary process of PSO to make it better optimized.

5.2. Encoding Mechanism. A decimal encoding method is used in this work to encode a solution, which encodes a set of weights and a threshold into each particle. With respect to the encoding process on n aggregating weights and one threshold, first, n real numbers are generated in [0, 1] randomly, which are, respectively, denoted as $r_1, r_2, \ldots, r_{n-1}, r_n$, represents the encoding information of a particle. Then, the first $n-1$ numbers $r_1, r_2, \ldots, r_{n-1}$ are sorted in the ascending order, and we get $r'_1, r'_2, \ldots, r'_{n-1}$. In particular, the final number r_n is the threshold for filtering the final alignment. Finally, n aggregating weights are obtained as follows:

$$\omega_t = \begin{cases} r'_1, & t = 1, \\ r'_t - r'_{t-1}, & 1 < t < n, \\ 1 - r'_{n-1}, & t = n. \end{cases} \tag{14}$$

Each particle in the population contains a set of weights and a threshold. An example of encoding process on aggregating weights is shown in Figure 4.

This encoding mechanism on the aggregating weights meets their constraints defined in equation (4), and it is also of help to reduce the solution's dimension, and it ensures that different groups of numbers correspond to different aggregating weights.

5.3. Simulated Annealing. Simulated annealing algorithm is an algorithm that introduces random factors into the search process. The simulated annealing algorithm does not completely reject the worse solution, which greatly improves the probability of getting rid of the local optimal solution. Generally, SA contains two parts, which are metropolis algorithm and annealing process. Metropolis algorithm aims at helping the solution jump out of the local optima, which accepts new solutions with a certain probability. Annealing is a process in which T, the parameter of the probability of accepting the worse solution, decreases with the iteration, so that as the iteration proceeds, the probability of accepting a worse solution gradually decreases. Assuming that a system's previous solution is denoted as $s(n)$ and the current solution is denoted as $s(n+1)$, where n is the current iteration number, the acceptance probability P of the system on changing from $s(n)$ to $s(n+1)$ is

$$P = \begin{cases} 1, & \text{if } F(n+1) \geq F(n), \\ e^{-(F(n) - F(n+1)/T)}, & \text{if } F(n+1) < F(n), \end{cases} \tag{15}$$

$$T = \frac{1}{\sqrt{1 + 0.1 \times n}} \times T_0, \tag{16}$$

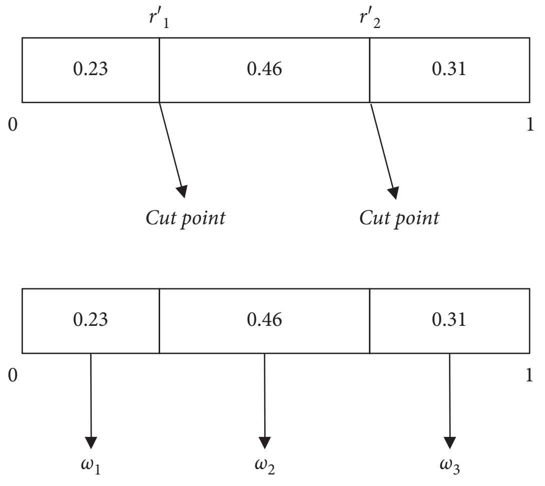

FIGURE 4: An example of encoding process on aggregating weights.

where $F(n)$ and $F(n+1)$ are the fitness of the previous solution and current solution, respectively. T is a parameter that represents the annealing temperature. Here, the initial temperature T_0 should be large, and as the iteration goes on, the temperature T would be gradually reduced, so as to ensure that the probability of state transition is gradually reduced from 1. In such situation, any solution can be accepted at the beginning of the iteration, and the current solution stays unchanged at the end of the iteration. Therefore, SA not only avoids the algorithm falling into local optimization too quick but also guarantees the algorithm's convergence.

For the sake of clarity, the pseudocode of SAPSO is presented in Algorithm 1.

First, the particles are initialized, and each particle generates three random numbers r_1, r_2, and r_3 on the [0, 1] interval, representing the cut points of the two weights and a threshold, respectively. And each particle also generates three initial velocities. Consider the cut points of weights and a threshold contained by the particle as the best cut points and threshold of individual history for each particle, denoted as pbest_1, pbest_2, and pbest_3, respectively, and calculate the fitness values for each particle (line 8). The two cut points and one threshold of each particle are denoted as the three dimensions of the particle. Find out the best one of each dimension of all particles, denoted as gbest_0,gbest_1, and gbest_2, respectively. Initialize the temperature T_0 and calculate/update the annealing temperature T_k (line 16) at the beginning of the iteration based on equation (16), update the cut points and threshold for each particle with the PSO formulas (lines 19 and 20), and get the updated fitness values based on the formulas in Section 4.2 (line 21). Then, there is the key to simulated annealing: if the updated particle has greater fitness than its predecessor, then the solution transition probability is set to 1 (line 23), and the new particle is considered as the *pbest*; otherwise, it is accepted at a certain probability according to equation (15). If the probability condition is satisfied, the new particle is considered as the *pbest* in the next generation to update the velocity and position with PSO (lines 19 and 20). Finally, the

```
**Input**: Source and target ontologies $O_1$ and $O_2$, number of iteration $k_{\max}$, initial temperature $T_0$,
population size $n$
**for** $(i = 0;\ i < n;\ i{+}{+})$
    **for** $(j = 0;\ j < 3;\ j{+}{+})$
        $v_j[i] = \text{random}(0, 1)$
        $r_j[i] = \text{random}(0, 1)$
        $\text{pbest}_j[i] = r_j[i]$
        calculate $\textit{fitness}\ [i]$
        $f\ [i] = \textit{fitness}[i]$
    **end for**
**end for**
$\text{gbest}_0 = \max\ \{\text{pbest}_0[i]\}$
$\text{gbest}_1 = \max\ \{\text{pbest}_1[i]\}$
$\text{gbest}_2 = \max\ \{\text{pbest}_2[i]\}$
**while** $k < k_{\max}$ **do**
    $T_k = 1/\sqrt{1 + 0.1 \times k} \times T_0$
    **for** $(i = 0;\ i < n;\ i{+}{+})$
        **for** $(j = 0;\ j < 3;\ j{+}{+})$
            $v_j[i] = v_j[i] + c_1 \times \text{random}(0, 1) \times (\text{pbest}_j[i] - r_j[i]) + c_2 \times \text{random}(0, 1) \times (\text{gbest}_j - r_j[i])$
            $r_j[i] = r_j[i] + v_j[i]$
            update $\textit{fitness}[i]$
            **if** $(\textit{fitness}[i] \geq f\ [i])$
                $P = 1$
                $\text{pbest}_j[i] = r_j[i]$
            else
                $P = e^{-(f[i] - \text{fitness}[i]/T_k)}$
                **if** $(P \geq \text{random}(0, 1))$
                    $\text{pbest}_j[i] = r_j[i]$
                **end if**
            **end if**
            $f\ [i] = \textit{fitness}[i]$
        **end for**
    **end for**
    $\text{gbest}_0 = \max\ \{\text{pbest}_0[i]\}$
    $\text{gbest}_1 = \max\ \{\text{pbest}_1[i]\}$
    $\text{gbest}_2 = \max\ \{\text{pbest}_2[i]\}$
    $\text{Gbest} = \max\ \{f\ [i]\}$
**end while**
**Output** Gbest
```

ALGORITHM 1: The pseudocode of SAPSO.

pbest particle whose fitness value is the largest is treated as the global best particle of the population for the next generation of PSO updating. If the end condition is not met, the loop executes the program until the end condition is met and the globally optimal fitness value, *f*-measure, is output.

5.4. The Flowchart of SAPSO. SAPSO is a method using annealing strategy to avoid the local optimal solution of PSO algorithm. The flowchart of SAPSO is shown in Figure 5.

First, we initialize the entire population, including the parameters of each particle. The second step is to obtain the fitness value for each particle and then judge whether the iteration has reached the max iteration; if the max iteration is reached, the iteration process ends and the results are output; otherwise, the entire population will be optimized using PSO algorithm according to equations (12) and (13), and obtain each particle's new fitness; at this point, we use "state" to represent all the information that the particle contains, including its fitness value and encoding information, and the particle's fitness is used to indicate the particle's state; pbest state and gbest states are, respectively, the information of an individual's corresponding local best and the population's global best during the evolutionary process. And then it is going to judge whether the new state is better than that of the previous generation. If the new state is better than the previous generation state, the new state is accepted, which satisfies equation (15), which is the formula for simulated annealing. Using simulated annealing, if the particle accepts the new state, the particle treats the new state as pbest state; otherwise, the particle treats the original state as pbest state; then, gbest state is obtained by comparing the pbest state of each particle. The annealing temperature needs to be recalculated according to equation (16) before the next iteration, and then the process is looped until the end condition is met.

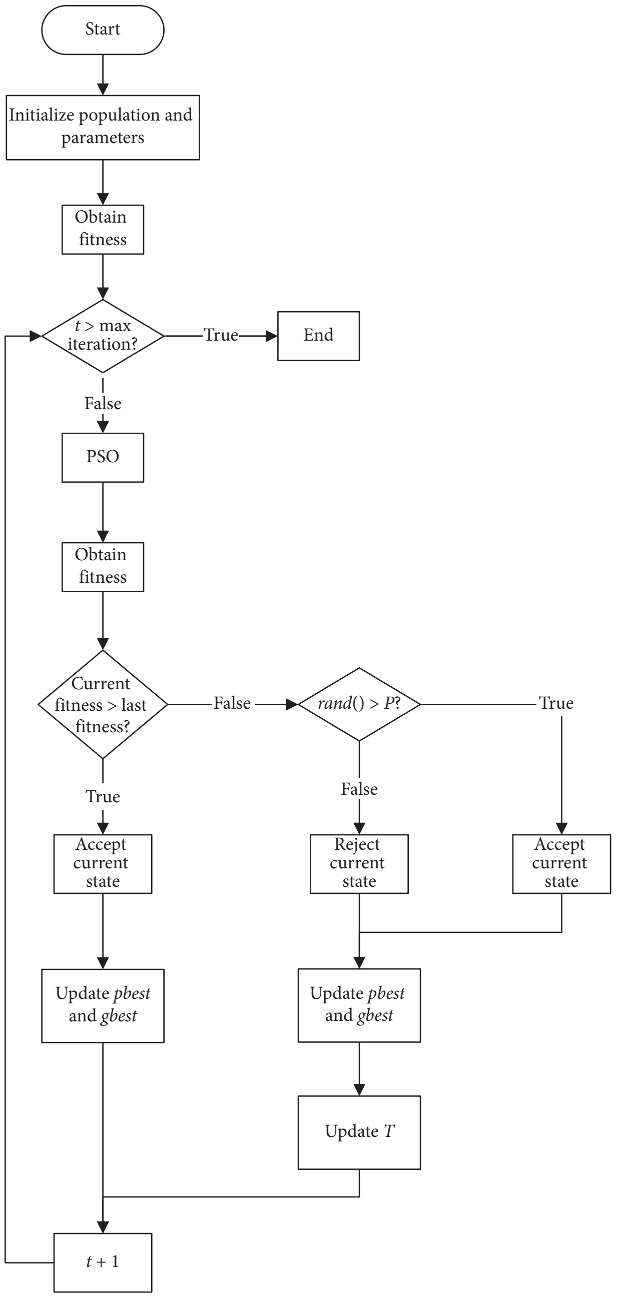

FIGURE 5: The flowchart of SAPSO.

6. Experiment Results and Analysis

In this experiment, to verify the effectiveness of SAPSO, we use the OAEI's benchmark and three real sensor ontologies, i.e., SOSA [33] and new SSN and old SSN ontology [22]. The test results of SAPSO and PSO shown in Tables 1 and 2 are the mean values of 30 independent runs.

6.1. Configuration. Similarity measures used in this experiment:

Syntactic-based measure: N-Gram

Linguistic-based measure: Wu–Palmer

Structure-based measure: Out-In Degree

The configuration on SPSO and PSO is as follows:

Population size: 60

Maximum number of iterations: 300

Learning factor c_1, c_2: 2

Initial temperature: 10.0

These parameters are determined in an empirical way, which is able to ensure the quality of the alignments in all testing cases.

6.2. Results and Analysis

6.2.1. OAEI Benchmark. The brief description of OAEI's benchmark is presented in Table 3. The first column in Table 3 is the ID of the testing cases, each corresponding to a testing ontology. We divide these test ontologies into five groups according to their specific characteristics, which is described in the second column of the table. We compare SAPSO with PSO-based ontology matching technique and OAEI's participants, i.e., edna, AML [34], LogMap [35], LogMapLt [35], XMap [36], and LogMapBio [35].

In Table 1, SAPSO's results are outperforming all the competitors except XMap on the testing cases 221–247. The reason is that on testing cases 221–247, the source ontology and the target ontology are identical in terms of lexical and semantic features but differ in terms of structural features, and our structure-based similarity measure is not effective, which reduces the *f*-measure. In particular, on all testing cases, SAPSO's results are all equal to or better than PSO, which shows that the introduction of SA is able to improve PSO's searching ability and improve the solution's quality. From the average of *f*-measure, SAPSO performs better than others, which shows that SAPSO plays an effective role in improving the quality of ontology matching.

6.2.2. Real Sensor Ontologies. SOSA (http://www.w3.org/ns/sosa/), the basic class and property of SSN (http://www.w3.org/ns/ssn/) ontology, represents the lightweight core of new SSN ontology. These sensor ontologies describe the function and performance of the sensor. They support many applications and use cases, such as signal detection in large-scale scientific exploration, home infrastructure monitoring, livelihood services, observation-driven ontology engineering, the World Wide Web, sensor data service system, and more [37]. The new SSN differs from the original SSN in that it simplifies the relationship between the device, platform, and system classes on the old SSN. We tested SAPSO on three real sensor ontologies with our sensor ontology meta-matching system and got their *f*-measure, recall, and precision values. Table 2 shows the matching results of SAPSO.

In Table 2, the first column refers to two matched sensor ontologies, and second, third, and fourth columns are, respectively, *f*-measure, recall, and precision of the alignments. It can be seen from the table that on the task of matching

Table 1: *F*-measure results comparison.

	edna	AML	LogMap	LogMapLt	XMap	LogMapBio	PSO	Our way
101	0.78	0	0.95	0.71	0.97	0.62	1	1
201–201–8	0.315	0.231	0.474	0.345	0.391	0.267	0.808	0.812
221–247	0.831	0.754	0.814	0.726	0.961	0.598	0.952	0.96
248–266	0.318	0.36	0.436	0.338	0.413	0.202	0.5432	0.5487
Average	0.41	0.41	0.51	0.409	0.51	0.28	0.6458	0.6514

Table 2: Measurement of *f*-measure, recall, and accuracy of SAPSO on three sensor ontologies.

	F-measure	Recall	Precision
Old SSN-new SSN	0.98	0.97	1.0
Old SSN-SOSA	0.98	0.97	1.0
New SSN-SOSA	1.0	1.0	1.0

Table 3: The relevant description of OAEI's benchmark.

ID	Relevant description
101–104	The two matched ontologies are identical in terms of structural, lexical, and linguistic features
201–210	Two matched ontologies are identical in terms of structural features but differ in terms of lexical and linguistic features
221–247	Two matched ontologies are identical in terms of lexical and linguistic features but differ in terms of structural features
248–266	Two matched ontologies are different in terms of lexical, linguistic, and structural features
301–304	Two matched ontologies are cases from the real world

new SSN and SOSA, SAPSO is able to determine the perfect alignment. With respect to the other two matching tasks, SAPSO's *f*-measure is also close to 1.0. Since there exist some complex correspondences in the reference alignment, i.e., one source concept corresponds to several target concepts, SAPSO fails to find them, which reduces its *f*-measure. In general, SAPSO is able to effectively match various sensor ontologies.

7. Conclusion and Future Work

LBS's architecture is widely used in the fields of vehicle speed estimation [38], vehicle travel time prediction system [39], and bus arrival time prediction system [40]. Technologies and applications of LBS cannot be separated from sensors. To implement the intelligent LBS, different sensor ontologies need to be integrated on SSW. To this end, in this work, the new quality evaluation metrics are proposed to evaluate the traditional three evaluation metrics. And a mathematical model on sensor ontology meta-matching problem is constructed; finally, a SAPSO is presented to address the problem, which uses SA to help the algorithm avoid the local optima. To verify the effectiveness of SAPSO, we use the OAEI's benchmark and three real sensor ontologies. Finally, the experiment proves that SAPSO is an effective method.

In the following work, the quality of the sensor ontology matching results would continue to be enhanced by taking into consideration those complex correspondences. At present, SAPSO still has some defects in determining the entity mappings with heterogeneous characteristics, which makes its *f*-measure relatively low in those testing cases with heterogeneous structure; at the same time, SAPSO has some limitations, for example, its performance is related to initial value and parameters are sensitive. Last but not least, it is necessary to improve the approximate evaluation metrics on ontology alignment to better guide the algorithm to search for the global optima.

Data Availability

The data used to support this study can be found in http://oaei.ontologymatching.org.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This study was supported by the National Natural Science Foundation of China (nos. 61801527 and 61103143) and the Natural Science Foundation of Fujian Province (no. 2020J01875).

References

[1] A. Bröring, J. Echterhoff, S. Jirka et al., "New generation sensor Web enablement," *Sensors*, vol. 11, no. 3, pp. 2652–2699, 2011.

[2] D. J. Russomanno, C. R. Kothari, and O. A. Thomas, "Building a sensor ontology: a practical approach leveraging ISO and ogc models," in *Proceedings of the 2005 International*

Conference on Artificial Intelligence (IC-AI 2005), pp. 637–643, Las Vegas, NV, USA, June 2005.

[3] A. Bröring, K. Janowicz, C. Stasch, and W. Kuhn, "Semantic challenges for sensor plug and play," *Web and Wireless Geographical Information Systems*, vol. 5886, pp. 72–86, 2009.

[4] J. C. W. Lin, Y. Shao, Y. Djenouri, and U. Yun, "ASRNN: A recurrent neural network with an attention model for sequence labeling," *Knowledge-Based Systems*, vol. 212, Article ID 106548, 2021.

[5] X. Xue and J. Chen, "Using compact evolutionary tabu search algorithm for matching sensor ontologies," *Swarm and Evolutionary Computation*, vol. 48, pp. 25–30, 2019.

[6] X. Xue and Y. Wang, "Using memetic algorithm for instance coreference resolution," *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 2, pp. 580–591, 2016.

[7] X. Xue and J. Chen, "Optimizing ontology alignment through hybrid population-based incremental learning algorithm," *Memetic Computing*, vol. 11, no. 2, pp. 209–217, 2019.

[8] X. Xue and J. Liu, "Collaborative ontology matching based on compact interactive evolutionary algorithm," *Knowledge-based Systems*, vol. 137, pp. 94–103, 2017.

[9] X. Xue and X. Yao, "Interactive ontology matching based on partial reference alignment," *Applied Soft Computing*, vol. 72, pp. 355–370, 2018.

[10] X. Xue, "A compact firefly algorithm for matching biomedical ontologies," *Knowledge and Information Systems*, vol. 62, no. 7, pp. 2855–2871, 2020.

[11] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *Proceedings of the Sixth International Symposium on Micro Machine and Human Science*, pp. 39–43, Nagoya, Japan, October 1995.

[12] X. Xue, X. Wu, and J. Chen, "Optimizing biomedical ontology alignment through a compact multiobjective particle swarm optimization algorithm driven by knee solution," *Discrete Dynamics in Nature and Society*, vol. 2020, Article ID 4716286, 2020.

[13] X. Xue and X. Wu, "Optimizing biomedical ontology alignment in lexical vector space," *Journal of Intelligent & Fuzzy Systems*, vol. 38, no. 5, pp. 5609–5614, 2020.

[14] X. Xue and J. Liu, "Optimizing ontology alignment through compact MOEA/D," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 31, no. 04, Article ID 1759004, 2017.

[15] P. Song, J. S. Pan, and S. Chu, "A Parallel compact Cuckoo search algorithm for three-dimensional path planning," *Applied Soft Computing*, vol. 94, Article ID 106443, 2020.

[16] Y. He, X. Xue, and S. Zhang, "Using artificial bee Colony algorithm for optimizing ontology alignment," *Journal of Information Hiding and Multimedia Signal Processing*, vol. 8, pp. 766–773, 2017.

[17] X. Xue and J. Chen, "Optimizing sensor ontology alignment through compact co-firefly algorithm," *Sensors*, vol. 20, no. 7, 2020.

[18] X. Xue, J. Chen, and J. Pan, *Evolutionary Algorithm Based Ontology Matching Technique*, Science Press, Beijing, China, 2018.

[19] X. Xue and J. S. Pan, "An overview on evolutionary algorithm based ontology matching," *Journal of Information Hiding and Multimedia Signal Processing*, vol. 9, no. 1, pp. 75–88, 2018.

[20] J. Bock and J. Hettenhausen, "Discrete particle swarm optimisation for ontology alignment," *Information Sciences*, vol. 192, pp. 152–173, 2012.

[21] S. Chu, X. Xue, J. Pan, and X. Wu, "Optimizing ontology alignment in vector space," *Journal of Internet Technology*, vol. 21, no. 1, pp. 15–23, 2020.

[22] A. Sheth, C. Henson, and S. S. Sahoo, "Semantic sensor Web," *IEEE Internet Computing*, vol. 12, no. 4, pp. 78–83, 2008.

[23] M. Bermudez-Edo, T. Elsaleh, P. Barnaghi, and K. Taylor, "IoT-lite: a lightweight semantic model for the Internet of things and its use with dynamic semantics," *Personal and Ubiquitous Computing*, vol. 21, no. 3, pp. 475–487, 2017.

[24] X. Xue, J. Lu, C. Jiang, and Y. Huang, "Sensor ontology metamatching with heterogeneity measures," *Wireless Communications and Mobile Computing*, vol. 2020, no. 3, pp. 1–10, 2020.

[25] X. Xue, X. Wu, C. Jiang et al., "Integrating sensor ontologies with global and local alignment extractions," *Wireless Communications and Mobile Computing*, vol. 2021, Article ID 6625184, 2021.

[26] X. Xue, C. Yang, C. Jiang et al., "Optimizing ontology alignment through linkage learning on entity correspondences," *Complexity*, vol. 2021, Article ID 5574732, 2021.

[27] V. Mascardi, A. Locoro, and P. Rosso, "Automatic ontology matching via upper ontologies: a systematic evaluation," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 5, pp. 609–623, 2010.

[28] G. Stoilos, G. Stamou, and S. Kollias, "A string metric for ontology alignment," *The Semantic Web - ISWC 2005*, vol. 3729, pp. 624–637, 2005.

[29] Z. Wu and M. Palmer, "Verb semantics and lexical selection," in *Proceedings of the 32nd annual meeting on Association for Computational Linguistics*, pp. 133–138, Las Cruces, NM, USA, June 1994.

[30] G. A. Miller, "WordNet," *Communications of the ACM*, vol. 38, no. 11, pp. 39–41, 1995.

[31] X. Xue and Y. Wang, "Optimizing ontology alignments through a memetic algorithm using both MatchFmeasure and unanimous improvement ratio," *Artificial Intelligence*, vol. 223, pp. 65–81, 2015.

[32] J. Kennedy and R. C. Eberhart, "Particle swarm optimization," in *Proceedings of the International Conference on Networks*, vol. 4, pp. 1942–1948, Piscataway, NJ, USA, February 2002.

[33] K. Janowicz, A. Haller, S. Cox, D. Le Phuoc SOSA, and M. Lefrançois, "ASOSA: a lightweight ontology for sensors, observations, samples, and actuators," *Journal of Web Semantics*, vol. 56, pp. 1–10, 2019.

[34] F. Daniel, P. Catia, B. Booma et al., "OAEI 2016 results of AML," in *Proceedings of the Eleventh International Workshop on Ontology Matching*, vol. 1766, Kobe, Japan, October 2016.

[35] E. Jimenez-Ruiz, B. Cuenca Grau, and V. Cross, "LogMap family participation in the OAEI 2016," in *Proceedings of the 11th International Workshop on Ontology Matching Co-located with the 15th International Semantic Web Conference (ISWC 2016)*, Kobe, Japan, July 2016.

[36] W. Eddine-Djeddi, M. Tarek-Khadir, and S. Ben-Yahia, "XMap: results for OAEI 2016," in *Proceedings of the 11th International Workshop on Ontology Matching Co-located with the 15th International Semantic Web Conference (ISWC 2016)*, Kobe, Japan, July 2016.

[37] A. Haller, K. Janowicz, S. J. D. Cox et al., "The modular SSN ontology: a joint W3C and OGC standard specifying the semantics of sensors, observations, sampling, and actuation," *Semantic Web*, vol. 10, no. 1, pp. 9–32, 2019.

[38] C.-H. Chen, "A cell probe-based method for vehicle speed estimation," *IEICE Transactions on Fundamentals of*

Electronics, Communications and Computer Sciences, vol. E103.A-A, no. 1, pp. 265–267, 2020.

[39] C.-H. Chen, F.-J. Hwang, and H.-Y. Kung, "Travel time prediction system based on data clustering for waste collection vehicles," *IEICE Transactions on Information and Systems*, vol. E102.D, no. 7, pp. 1374–1383, 2019.

[40] C.-H. Chen, "An arrival time prediction method for bus system," *IEEE Internet of Things Journal*, vol. 5, no. 5, pp. 4231-4232, 2018.