Journal of Network Intelligence ©2022 ISSN 2414-8105 (Online)
Taiwan Ubiquitous Information Volume **7**, Number **1**, February **2022**

Aggregating Similarity Measures for Optimizing Ontology Alignment

Wenyu Liu

School of Computer Science and Mathematics
Intelligent Information Processing Research Center
Fujian University of Technology
No.3 Xueyuan Road, Fuzhou City, Fujian Province, 350118, China
wenyuliu1983@hotmail.com

Xingsi Xue*

Intelligent Information Processing Research Center
Fujian University of Technology
No.3 Xueyuan Road, Fuzhou City, Fujian Province, 350118, China
Guangxi Key Laboratory of Automatic Detecting Technology and Instruments
Guilin University of Electronic Technology
No.1 Jinji Road, Guilin City, Guangxi Province, 541004, China
*Corresponding Author: jack8375@gmail.com

Zhenhong Wu

School of Computer Science and Mathematics
Fujian University of Technology
No.33 Xuefu South Road, Fuzhou City, Fujian Province, 350118, China
a1994261965@hotmail.com

Vaci Istanda

Indigenous Peoples Commission
Taipei City Government
No.1 Shifu Road, Taibei City, Taiwan
Biungsu@yahoo.com.tw

Received August 2021; revised October 2021
(Communicated by Xingsi Xue)

Abstract. *Ontology is able to build a shared knowledge model for an application domain to overcome the data heterogeneity issue, but they also suffer from the heterogeneity issue. Finding identical entities in two ontologies, which is the so-called ontology matching, is regarded as a solution to this issue. During the matching process, it is important to use the Similarity Measure (SM) to distinguish the heterogeneous entities. However, due to the complex semantic relationships among concepts, there is no such SM that is effective on all matching tasks. How to aggregate different SMs to make their advantages and disadvantages complement each other, and on this basis, the ontology alignment can be optimized, is the so-called ontology meta-matching problem. Due to the semantic richness of the concepts and their relationships, ontology meta-matching problem remains a challenge in the ontology matching domain. Inspired by the success of Genetic Algorithm (GA) in addressing various complex optimization problem, this work proposes a GA based ontology meta-matching technique to tune the aggregating weights. In particular, we construct a novel ontology meta-matching framework based on the weighted average strategy, and build the mathematical model for the meta-matching problem, propose a problem-specific GA to optimize the weight set for aggregating various similarity measures. The experiment utilizes the famous benchmark in ontology matching domain for testing purposes, and the experimental results show that our approach is able to effectively find high-quality alignment.*

Keywords: Ontology Matching, Similarity Measure, Genetic Algorithm

1. Introduction.

Ontology is able to build a shared knowledge model to overcome the data heterogeneity issue [1, 2], and integrate the data in the various domain such as Wireless Sensor Network (WSN) [3]. However, different ontologies have different application requirements and bias interests, which makes the domains ontologies themselves suffer from the heterogeneity problem. Finding identical entities in two ontologies, which is the so-called ontology matching, is a solution to this issue [4, 5]. When matching two ontologies, it is important to use the Similarity Measure (SM) to distinguish the heterogeneous entities. However, due to the complex semantic relationships among concepts, there is no such SM that is effective on all matching tasks. Thus, it is meaningful to aggregate different SMs to make their advantages and disadvantages complement each other, and on this basis, the ontology alignment can be optimized.

Currently, the parallel framework is the most flexible ways of aggregating SMs, which assigns a weight for each SM to obtain the final alignment. During this procedure, each SM's corresponding similarity matrix is first calculated, whose row and column respectively represent two ontology's entities and the elements inside are the corresponding entities' similarity value. After that, the aggregated matrix is determined by aggregating all the matrices with the weighted mean strategy. Finally, a threshold is used to filter the elements with low similarity values to obtain the final matrix, which is decoded to the ontology alignment. It is a complex problem to determine the optimal aggregating weight set for SMs, since it has many local optimal solutions. Genetic Algorithm (GA) [6, 7] is a popular global optimization algorithm, and being inspired by its success in various domains [8, 9], we make the following contributions:

- a novel ontology meta-matching framework is constructed, which select the weighted average strategy to enhance the flexibility of the aggregating process,
- a mathematical model is built to define the ontology matching problem, which takes f-measure as the objective function to better describe the essence of the problem,

- a problem-specific GA is proposed to tune the aggregating weight set, which takes a problem-specific encoding mechanism to encode GA's individual to efficiently execute its evolutionary operators.

The rest is organized as follows: after defining ontology matching problem (Section 2), various SMs are introduced (Section 3), then the problem-specific GA is presented (Section 4), and the experimental results are shown (Section 5). Finally, the conclusion is drawn (Section 6).

2. **Ontology Matching Problem.** An ontology consists of concepts, properties and axioms, and an ontology alignment is a mapping set. A mapping is a 3-tuple $(c_1, c_2, simValue)$ where c_1 and c_2 are respectively two ontologies' entities, and $simValue \in [0, 1]$ is their similarity [10, 11]. An ontology's quality can be measured with f-measure [12]:

$$recall(A) = \frac{|A \cap RA|}{|RA|} \tag{1}$$

$$precision(A) = \frac{|A \cap RA|}{|A|} \tag{2}$$

$$f - measure(A) = \frac{2 \times recall(A) \times precision(A)}{recall(A) + precision(A)} \tag{3}$$

where A and RA are respectively an alignment and reference alignment, and $||$ is the set's cardinalities. On this basis, the ontology matching problem is defined as follows: the objective function is to maximize the f-measure, and the decision variable $X = \{x_1, x_2, \ldots\}^T$ where x_i is the ith aggregating weight, and $\sum x_i = 1$, $x_i \in [0, 1]$.

3. **Similarity Measure.** Generally, there are three broad categories of SM, i.e. Syntactic SM, Linguistic SM and Taxonomy SM [13, 14]. In the following, we shall describe them one by one.

Syntactic SM calculates two strings' similarity through their edit distance. In this work, we use the Levenshtein distance [15], which is defined as follows:

$$sim_{Levenshtein}(s_1, s_2) = max\Big(0, \frac{min(|s_1|, |s_2|) - d(s_1, s_2))}{min(|s_1|, |s_2|)}\Big) \tag{4}$$

where $|s_1|$ and $|s_2|$ are respectively the character numbers of two strings s_1 and s_2, $d(s_1, s_2)$ is their edit distance.

Linguistic SM utilizes an electronic dictionary to measure two words' similarity. In this work, we select Wordnet [16, 17] as the electronic knowledge base, and given two entities' label $label_1, label_2$, their linguistic similarity value is defined as follows:

$$sim_{Linguistic}(label_1, label_2) = \begin{cases} 1, & \text{if } label_1 \text{ and } label_2 \text{ are synonymous;} \\ 0.5, & \text{if } label_1 \text{ and } label_2 \text{ are hyponymous or hypernymous;} \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

Taxonomy SM uses two concepts c_1 and c_2's context to determine their similarity, which is defined as follows [18, 19]:

$$sim_{SF}(c_1, c_2) = \frac{sim_{Levenshtein}(super_1, super_2) + \sum sim_{Levenshtein}(sub_i, sub_j)}{2} \tag{6}$$

where $super_1$ and $super_2$ are respectively c_1 and c_2's super classes, and sub_i and sub_j are respectively their ith and jth sub classes.

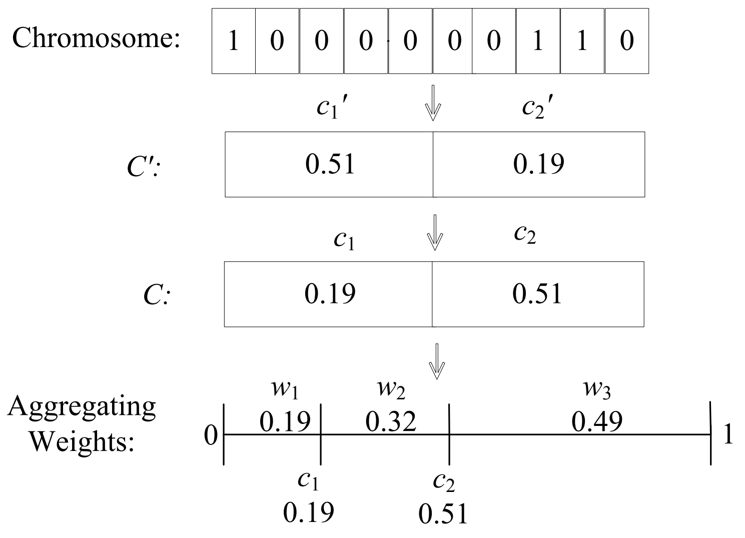

FIGURE 1. An example of encoding mechanism.

4. **Genetic Algorithm.**

4.1. **Encoding Mechanism.**

GA uses the binary coding mechanism [20], which is of help to reduce the evolutionary operation's computational complexity. Considering that the coding information needs to contain the weight set of SMs, we store them in disguise form by storing the cutting points in the coding information. Given a set of cut points $C' = \{c_1, c_2, \ldots, c_n\}$, we first sort it in ascending order as $C = \{c_1, c_2, \ldots, c_n\}$, and then we can get the corresponding weight set through the following equation:

$$w_k = \begin{cases} c_1, & k = 1 \\ c_k - c_{k-1}, & 1 < k < p \\ 1 - c_{p-1}, & k = p \end{cases} \tag{7}$$

Through calculation, we can use n cutting points to obtain $n+1$ aggregating weights. Particularly, this work selects three SMs, so we need to encode two cutting points' information. In addition, we use 10 gene bits to represent a cutting point, and thus, the length of a chromosome is 20 gene bits. Figure 1 shows an example of encoding mechanism. As can be seen from the figure, two cutting points are used to represent the aggregating weights of the SM, and the gene bits for encoding each cutting point is 5. As shown in the figure, firstly, a chromosome is decoded to decimal to obtain the cut point set C', then C' is sorted to obtain the cutting point set C, and finally, three weights w_1, $w_2$and w_3 are calculated according to Eq. 7.

4.2. **Selection Operator.**

In the process of biological evolution in nature, individuals with strong fitness have strong survival ability. Similarly, the selection operator should ensure that solutions with high quality should have more opportunities to survive in the next generation. However, solutions with low fitness values should not be completely ignored, because they are an important prerequisite to ensure population diversity. The selection operator used in this paper is the roulette selection operator. The probability of selecting an individual is obtained by the ratio of the fitness value of this individual to the sum of the all individuals's fitness values. This makes the probability of each individual being selected proportional to the fitness value, and also makes each individual have the opportunity to be selected. Given the fitness value of ith solution F_i, the selection probability of the ith solution is $\frac{F_i}{\sum F_i}$.

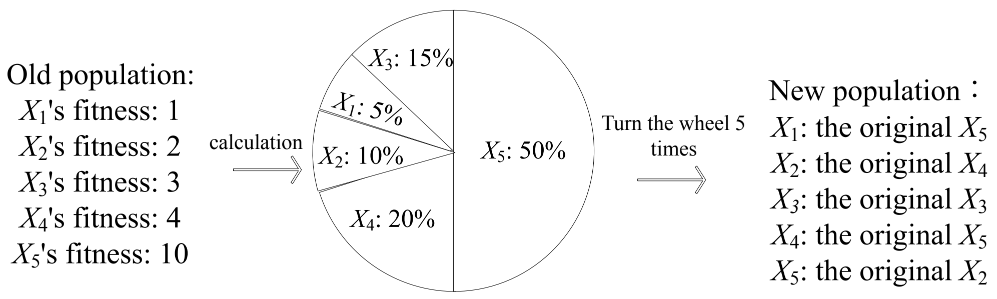

FIGURE 2. An example of selection operator.

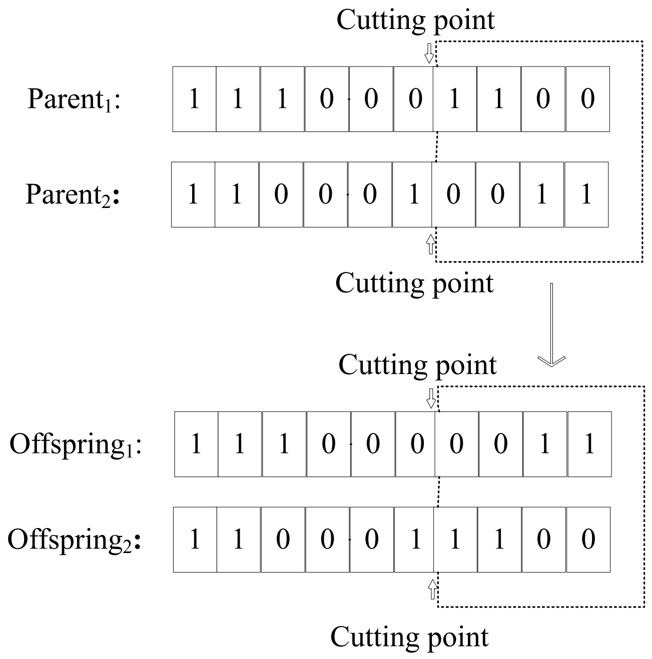
Cutting point
Parent$_1$: 1 1 1 0 0 0 1 1 0 0
Parent$_2$: 1 1 0 0 0 1 0 0 1 1
Cutting point
Cutting point
Offspring$_1$: 1 1 1 0 0 0 0 0 1 1
Offspring$_2$: 1 1 0 0 0 1 1 1 0 0
Cutting point

FIGURE 3. An example of crossover operator.

Figure 2 shows the process of an old population getting the population through roulette. Among them, the old population first calculates the selection probability of each individual, so as to obtain a roulette. Then, the disc was rotated 5 times by probability simulation, and the new species group was obtained according to the results of 5 times.

4.3. **Crossover Operator.** The crossover operator generates a new individual by mixing the genes of two parent solutions. Crossover is executed according to a probability, which is the so-called crossover probability. This work uses the single-point crossover operator [21], which works as follows: first, randomly select a point in the genotype of the parent individual as the cutting point, which dividing two parents into left and right parts, and then, two new children are generated by exchanging the right genes of two parents. Figure 3 shows an example of a single-point crossover operator, and the chromosome length is 10.

4.4. **Mutation Operator.** The mutation operator maintains the diversity of the population, which is critical to the algorithm's searching ability. The mutation operator used in this paper is the locus mutation operator [22], which judges whether a gene value should be mutated by comparing a random number in [0,1] with the mutation probability. If the random number generated is smaller, the value of the gene will be flipped. Figure 4

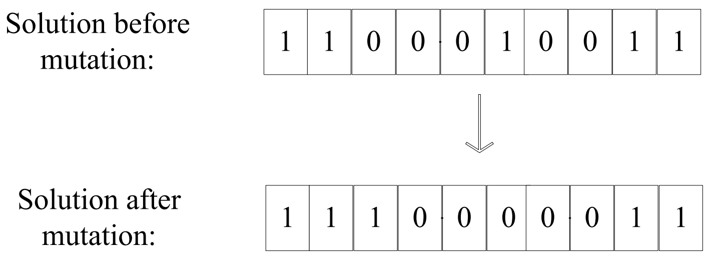

FIGURE 4. An example of mutation operator.

shows an example of the mutation operator. In this example, the chromosome length of an individual is 10, and all the genes are flipped according to the mutation rate.

4.5. **Pseudo-code of Genetic Algorithm.** Given the maximum generation *maxGen*, Algorithm 1 shows GA's pseudo-code.

Algorithm 1 Genetic Algorithm

```
*******Initialization**********
for $i = 0; i < population.len; i++$ do
  for $j = 0; j < solution.len; i++$ do
    $gene_{i,j} = random\{0,1\}$;
  end for
end for
*******Evaluation**********
for $i = 0; i < population.len; i++$ do
  $evaluation()$;
end for
$gen = 0$;
*******Evolution**********
while $gen < maxGen$ do
  $crossover()$;
  $mutation()$;
  for $i = 0; i < population.len; i++$ do
    $evaluation()$;
  end for
  $selection()$;
  $saveElite()$;
  $gen = gen + 1$;
end while
```

In Algorithm 1, the chromosome of the population is initialized with the random numbers in $\{0, 1\}$, and then the fitness of each solution is evaluated. In each generation, the crossover operator and mutation operator are successively performed on the current population, and then each solution is re-evaluated. After that, the selection operator is used to select a new population. Finally, we will replace the worst individual in the current generation, with the individual with the highest fitness value found so far, i.e. the elite solution.

5. **Experimental Studies and Analysis.** In this work, the famous benchmark [1] is utilized for testing purpose. A threshold for filtering the final alignment is set as 0.9, and

[1]http://oaei.ontologymatching.org/2016/benchmarks/index.html

Table 1. Comparison with OAEI's participants on benchmark in terms of recall and precision.

Matching System	1XX *precision*	1XX *recall*	2XX *precision*	2XX *recall*
Edna	0.64	1.00	0.62	0.84
LogMap	0.94	0.96	0.90	0.81
LogMapLt	0.56	0.99	0.53	0.83
LogMapBio	0.50	0.56	0.52	0.65
GMap	0.97	1.00	0.88	0.85
LogMap-C	0.58	0.96	0.57	0.81
Mamba	0.90	0.84	0.79	0.76
AOT_2014	0.97	0.97	0.93	0.83
OReasoner	0.87	1.00	0.74	0.84
CIDER-CL	1.00	1.00	0.78	0.91
HerTUDA	0.89	1.00	0.90	0.85
MapSSS	0.89	0.34	0.87	0.27
RIMIOM2013	0.84	1.00	0.63	0.88
SerevOMap	0.95	1.00	0.67	0.56
StringsAuto	0.89	0.34	0.87	0.27
Synthesis	0.94	1.00	0.81	0.86
XMapGen	0.84	1.00	0.67	0.78
XMapSig	0.84	1.00	0.70	0.84
ASE	0.58	1.00	0.61	0.85
GOMMA	0.84	1.00	0.70	0.87
MEDLEY	0.72	1.00	0.68	0.84
Optima	1.00	1.00	0.85	0.83
ServOMap	1.00	0.98	0.91	0.76
ServOMaplt	1.00	0.28	1.00	0.45
Wmath	0.84	1.00	0.73	0.85
GA	1.00	1.00	0.93	0.84

the configuration of GA is empirically set as follows:

- Maximum Generation: 2000;
- Crossover Rate: 0.8;
- Mutation Rate: 0.05;

In the experiment, we compare GA-based ontology matching technique with OAEI's participants [2]. Table 1 compares GA with OAEI's participants to benchmark in terms of recall and precision, and Figure 5 compares GA with OAEI's participants to benchmark in terms of f-measure. Here, 1XX and 2XX are respectively the testing cases with IDs beginning with 1 and 2. With respect to 1XX, two ontologies under alignment are exactly the same except different OWL restrictions, while regarding 2XX, two ontologies are heterogeneous in terms of the entity name, the concept's hierarchical structure or both of them.

As shown in the table, GA's precision and recall values are in general higher than OAEI's participants, which shows that GA is able to effectively tune the aggregating weights for SMs and search for the correct correspondences. In addition, from the figure,

[2]http://oaei.ontologymatching.org/

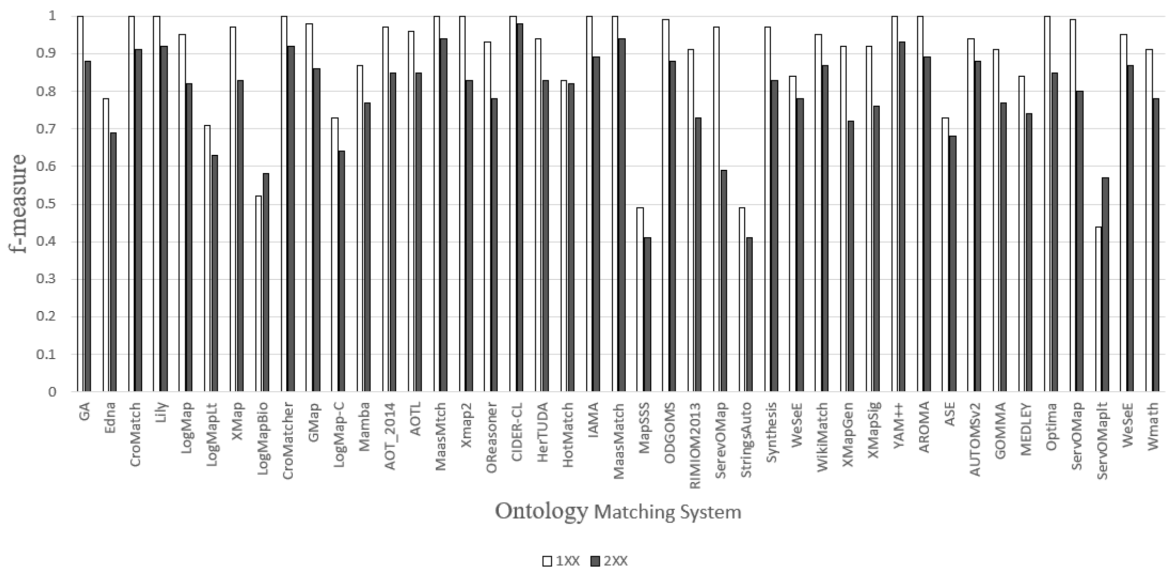

FIGURE 5. Comparison with OAEI's participants on benchmark in terms of f-measure.

we can see that GA's f-measure is also better than all OAEI's participants on both 1XX (the mean f-measure is 1.00) and 2XX (the mean f-measure is 0.88) testing cases, which show that GA is able to effectively optimize the ontology alignment's quality.

6. Conclusion.

To find a suitable way of aggregating different similarity measures, in this work, a GA-based ontology matching technique is proposed. We first re-define the ontology matching problem in the parallel framework, and a problem-specific GA is proposed to address it. The experiment utilizes OAEI's benchmark for testing, and the experimental results show that our approach is effective in various matching tasks.

Acknowledgment. This work is supported by the National Natural Science Foundation of China (No. 62172095), the Natural Science Foundation of Fujian Province (No. 2020J01875), and the Guangxi Key Laboratory of Automatic Detecting Technology and Instruments (No. YQ20206).

REFERENCES

[1] P. Shvaiko, J. Euzenat, Ontology matching: state of the art and future challenges, *IEEE Transactions on knowledge and data engineering*, vol. 25, no. 1, pp.158–176, 2013.
[2] I. Osman, S. B. Yahia, G. Diallo, Ontology integration: Approaches and challenging issues, *Information Fusion*, vol. 71, pp. 38-63, 2021.
[3] C.-M. Chen, Y.-H. Lin, Y.-C. Lin, H.-M. Sun, RCDA: Recoverable concealed data aggregation for data integrity in wireless sensor networks, *IEEE Transactions on parallel and distributed systems*, vol. 23, no. 4, pp. 727-734, 2011.
[4] 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.
[5] N. Ferranti, S. S. R. F. Soares, J. F. de Souza, Metaheuristics-based ontology meta-matching approaches, *Expert Systems with Applications*, vol. 173, pp. 1-15, 2021.
[6] D. Whitley, A genetic algorithm tutorial, *Statistics and computing*, vol. 4, no. 2, pp. 65-85, 1994.
[7] S. Katoch, S. S. Chauhan, V. Kumar, A review on genetic algorithm: past, present, and future, *Multimedia Tools and Applications*, vol. 80, no. 5, pp. 8091-8126, 2021.
[8] C. Kim, R. Batra, L. Chen, H. Tran, R. Ramprasad, Polymer design using genetic algorithm and machine learning, *Computational Materials Science*, vol. 186, pp. 1-6, 2021.
[9] X. Xue, 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.

[10] G. Acampora, V. Loia and A. Vitiello, Enhancing ontology alignment through a memetic aggregation of similarity measures, *Information Sciences*, vol.250, pp.1-20, 2013.

[11] X. Xue, Matching Biomedical Ontologies Through Compact Hybrid Evolutionary Algorithm, *Journal of Information Hiding and Multimedia Signal Processing*, vol. 10, no. 1, pp. 110-117 , 2019.

[12] C. J. Van Rijsberge, *Information Retrieval*, University of Glasgow, Butterworth, London, 1975.

[13] 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.

[14] S. Mani, S. Annadurai, Explicit Link Discovery Scheme Optimized with Ontology Mapping using Improved Machine Learning Approach, *Studies in Informatics and Control*, vol. 30, no. 1, pp. 67-75, 2021.

[15] V. I. Levenshtein, Binary codes capable of correcting deletions, insertions, and reversals, *Soviet physics doklady*, vol.10, no.8, pp.707–710, 1966.

[16] G. A. Miller, WordNet: a lexical database for English, *Communications of the ACM*, vol. 38, no. 11, pp. 39-41, 1995

[17] E. Geller, M. Gajek, A. Reibach, Z. Lapa, Applicability of wordnet architecture in lexical borrowing studies, *International Journal of Lexicography*, vol. 34, no. 1, pp. 92-111, 2021.

[18] X. Xue and J.-S. Pan, A Segment-based Approach for Large-scale Ontology Matching, *Knowledge and Information Systems*, vol. 52, no. 2, pp. 467-484, 2017.

[19] M. AlMousa, R. Benlamri, R. Khoury, Exploiting non-taxonomic relations for measuring semantic similarity and relatedness in WordNet, *Knowledge-Based Systems*, vol. 212, pp. 1-27, 2021.

[20] Y. Xue, H. Zhu, J. Liang J, A. stowik, Adaptive crossover operator based multi-objective binary genetic algorithm for feature selection in classification, *Knowledge-Based Systems*, vol. 227, pp. 1-17, 2021.

[21] F. A. Zainuddin, M. F. Abd Samad, Comparison of Crossover in Genetic Algorithm for Discrete-Time System Identification, *International Review of Mechanical Engineering*, Vol. 15, No. 2, pp. 59-66, 2021.

[22] J. Al-Afandi, A. Horvath, Adaptive Gene Level Mutation, *Algorithms*, vol. 14, no. 1, pp. 1-18, 2021.