Establishing Dynamic Ontology for Agriculture Domain

Susan F. Ellakwa[*1], Passent El-Kafrawy[**2]

[*]*Central Laboratory of Agricultural Expert Systems, Cairo, Egypt*

[**]*Mathematics and CS Department, Faculty of Science, Menoufia University*

[1]fisalsusan@yahoo.com

[2]passentmk@gmail.com

Abstract: ***This paper presents semi-automated system for establishing integrated ontology by merging two ontologies. It uses two processes: matching and merging. Matching process uses string-based technique, this technique uses four methods: exact method to detect identical terms, and substring, suffix and prefix methods to compare between terms. Using these four methods altogether improve the effectiveness of matching process, matching process uses also language-based techniques; this technique uses WordNet Method to detect terms that have the same meaning. This technique improves also the effectiveness of matching process. The proposed system presents a merging method of taxonomies in effective way. The system solves redundancy and inconsistency problem in integrated ontology.The proposed system is applied on the agricultural domain for Faba Bean crop to get an integrated ontology, it can be applied also on all crops whatever field crops or horticulture crops. The evaluation of the system shows that the performance of the system has high quality. The comparison of the proposed system and other systems shows that the proposed system has advantage of using five matching methods for mapping between terms that make the mapping between terms more perfect and efficient. The merger algorithm solves problems which appeared in other systems.***

Keywords: ***Ontology, Knowledge Representation, Matching and Merging Ontology***

1 INTRODUCTION

'Ontology matching is a key interoperability enabler for the semantic web, as well as a useful tactic in some classical data integration tasks dealing with the semantic heterogeneity problem. It takes ontologies as input and determines as output an alignment, that is, a set of correspondences between the semantically related entities of those ontologies. These correspondences can be used for various tasks, such as ontology merging, data translation, query answering or navigation on the web of data. Thus matching ontologies enables the knowledge and data expressed in the matched ontologies to interoperate'[1].

Multiple ontologies of different systems for the same domain may be dissimilar, thus, various parties with different ontologies do not fully understand each other, in spite of the fact that these ontologies are for the same domain. To solve this problem, it is necessary to integrate these ontologies. But integrating ontologies faces types of heterogeneity problems like: syntactic heterogeneity, terminological heterogeneity, conceptual heterogeneity, and semiotic heterogeneity. In this paper, we deal with terminological heterogeneity; it occurs due to variations in names when referring to the same entities in different ontologies. This can be caused by the use of synonyms, e.g., Paper vs. Article, abbreviations or scientific terms of common terms.

This paper presents a system to integrate ontologies and solve the terminological problem. The system is based on matching and merging. Matching is the process of finding relationships or correspondences between entities of different ontologies. Matching process is based on string-based technique and language-based technique.

String-based technique[2] compares strings depending on the way the string is viewed: for example, as an exact sequence of letters, an erroneous sequence of letters, a set of letters, and a set of words. String-based technique consists of exactmethod, substring Method, prefix and suffix method. Language-based technique [2] relies on using Natural Language Processing (NLP) technique to helpextract the meaningful terms from an ontology. This technique uses external resources such as WordNet thesaurus. It also uses tokenization method andstopword elimination method.

These techniques help the proposed system to extract similarities between two ontologies to detect matched terms to reduce redundancy in the output of the merged ontology.

Merging is the process of creating a new single coherent ontology from two or more existing source ontologies related to the same domain, the new ontology replaces the source ontologies. Merging process uses intersection operation, union operation and merger algorithm.

2 RELATED WORK

This paper solve Redundancy problem, it is due to heterogeneity[3]of ontologies. Heterogeneity can be solved by matching and merging. Common matching techniques use a single matching criterion to analyze concepts which does not semantically align concepts correctly. These techniques cannot detect kinds of similarities in source ontologies and produces integrated ontology has duplicate terms (redundancy). For example,

Anchor prompt[4]system uses matching technique for detecting terms which are identical terms, but terms which are close syntactically cannot be detected such as abbreviation, so the integrated ontology may have duplicate terms such as (*irrig*, *irrigation*) and this system cannot detect terms have same meaning such as (*base, stem*).

DKP-AOM[5]system uses matching technique for detecting identical terms, terms have same meaning, and it also detects similarities based on base method, but this system cannot deal with terms that have composite words. So the integrated ontology may have duplicate terms such as (*irrigation system*, *water system*).

For example, if there are two source ontologies taxonomies for two views of two knowledge engineers: an *insect* taxonomy of first *pest* ontology (see figure1) has matched with *insects* taxonomy of second *pest* ontology (see figure2), the integrated taxonomy will be as shown in figure3 which is inconsistent because the correct integrated ontology should be as shown in figure4. *Leaves insect* should be parent of *leafminer, thrips,* and *leaf hoppers*, also *aphids* should be parent of green *aphids* and *black aphids*.

SKAT [6]system does not use WordNet method, the system cannot detect the terms that have the same meaning. So the integrated ontology may have duplicate terms such as (*base, stem*).

Therefore we need a multi-matching technique to solve the redundancy problem. This technique should have variety of matchers which deal with abbreviation, composite words and semantic terms to detect all similarities between source ontologies. These similarities should be filtered by threshold and confirmed by the user.

This paper solve also inconsistency problem which has appeared during handling hierarchies ontologies lead to an inconsistent merged ontology, this means that some concepts in the integrated ontology are not in the correct place in the taxonomy; this problem is not explained in common systems.

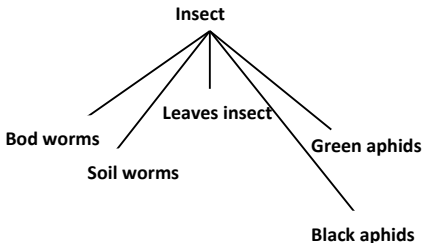

Figure 1: Part of First Pest Ontology

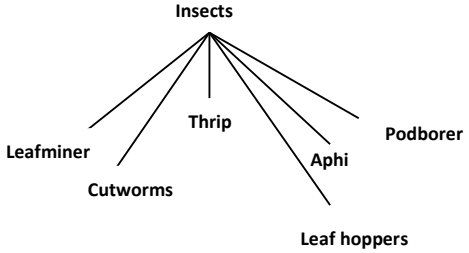

Figure 2: Part of Second Pest Ontology

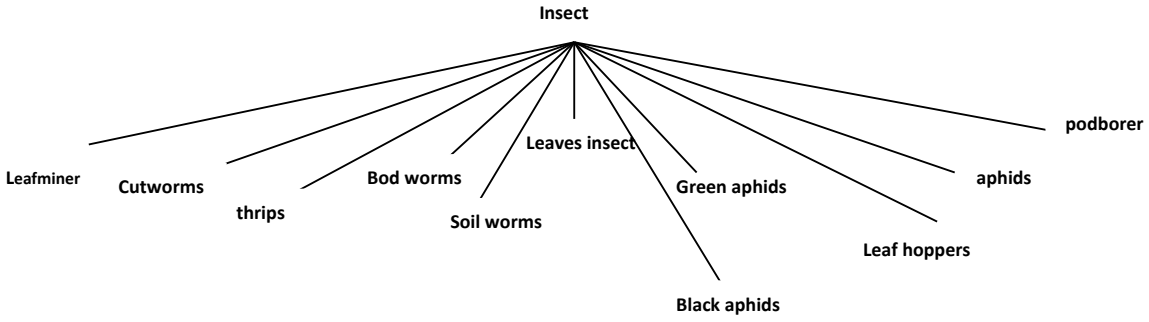

Figure 3: Part of Inconsistent Merged Pest ontology

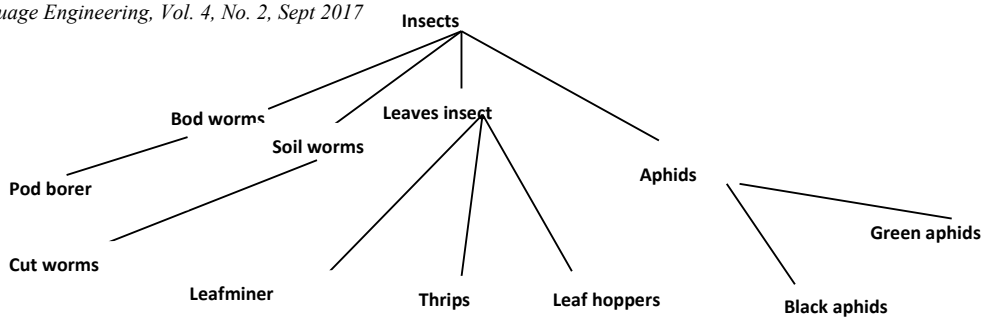

Figure 4: Consistent Merged Pest Ontology

3 PROPOSED SYSTEM DETAILS

This section presents a semi-automated system, Multi-Matching and Merging System (MMMS), for designing an integrated ontology by merging pre-existing ontologies. This system consists of two main components: matching process and merging process.

A. System Structure

The structure of the two main components, matching process and merging process, are shown in figure 5. Matching process consists of three operations: the first operation is *Concept Matcher* to extract concept alignment (matched concepts), Ac, the second operation is *Property Matcher* to extract property alignment (matched properties), Ap, and the third operation is *Value Matcher* to extract value alignment (matched values) Av. Merging process consists of *Merger* to integrate matched ontologies and obtain merged ontology, O'

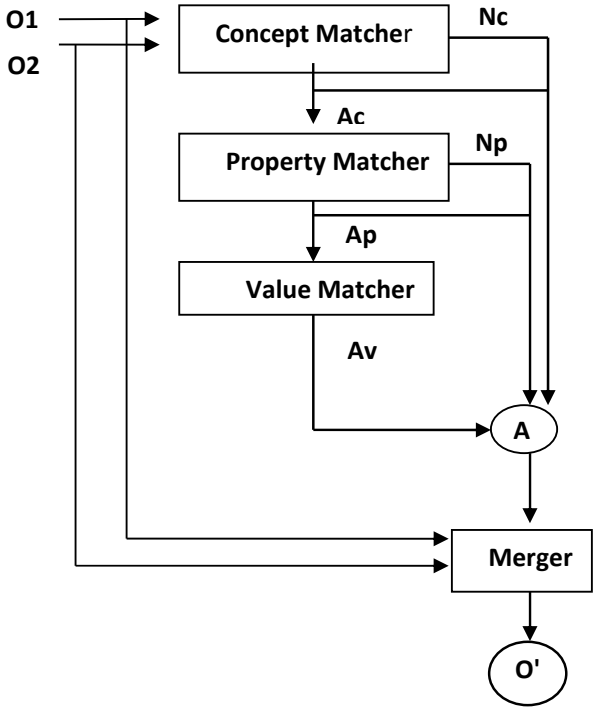

Figure 5: Multi-Matching and Merging System (MMMS)

A. Matching Process

The matching process receives source ontologies (O1, O2) and computes similarities between the ontological entities (concepts, properties, values). Each operation (*Concept Matcher, Property Matcher* or *Value Matcher*) is processed by matcher component (see figure 5). Matcher component consists of five matchers: exact, substring, prefix, suffix, and WordNet. Matcher component is applied on the three kinds of entities: concepts, properties and values from the source ontologies.

Matcher1 based on exact string method, it searches for identical terms, for example (*frost* alignsto *frost*), the output of Matcher1 is M1 (correspondences of Matcher1). Matcher2 based on substring method, for example (*location* aligns to *suitable-location*).

The input of this matcher is the unmatched entities of previous matcher, the output is M2 (correspondences of Matcher2). M2 is to be filtered according to a threshold (the threshold is a value that determines the minimum accepted similarity between two entities, it may be determined by the user otherwise the system uses the built-in value).The user confirms the filtered correspondences.

Matcher3 based on prefix method for example (*whitefly* align to *whiteflies*). The input of this matcher is the unmatched entities of previous matcher (M2), the output is M3 (correspondences of Matcher3). M3 is to be filtered according to a threshold. The user confirms the filtered correspondences.

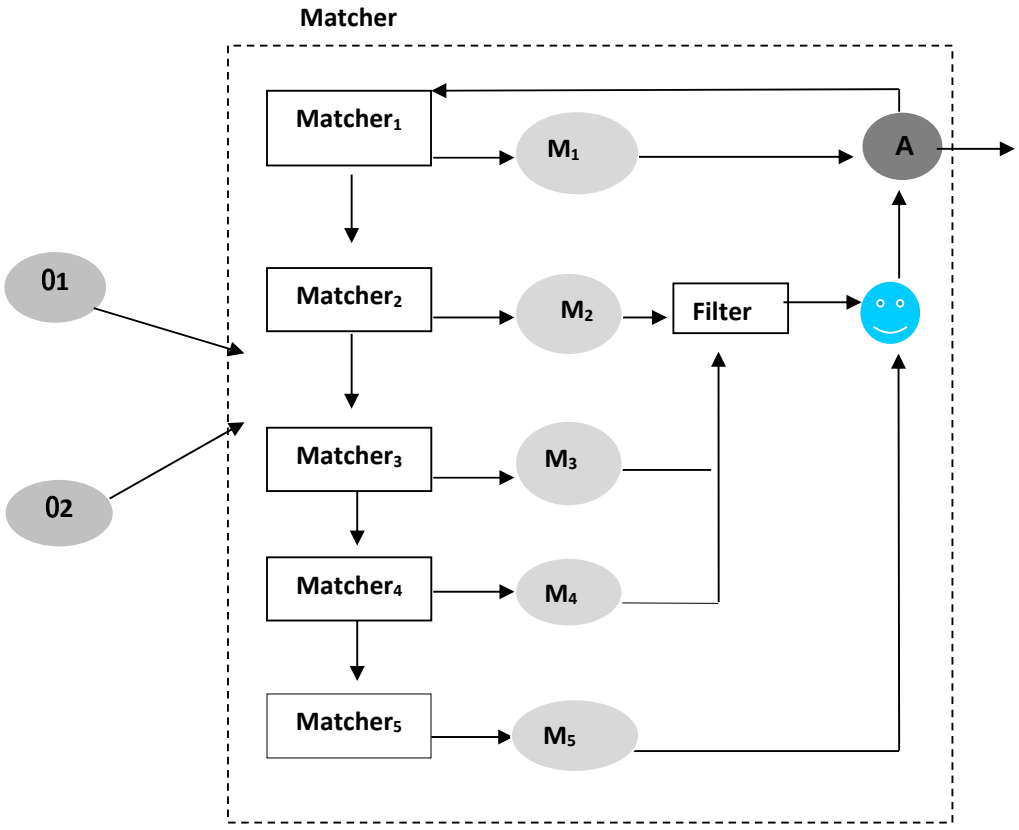

Figure 5: Flowchart of Matcher Component

Matcher4 based on suffix method for example (water-schedule aligns to *irrigation-schedule*). The input of this matcher is the unmatched entities of previous matcher (M3), the output is M4 (correspondences of Matcher4). M4 is to be filtered according to a threshold. The user confirms the filtered correspondences.

Matcher5 based on WordNet method for example (*base* aligns to *stem*), it searches for terms which have the same meaning. The input of this matcher is the unmatched entities of previous matcher (M4), the output is M5 (correspondences of Matcher5), this matcher uses tokenization method and stopword elimination method. The user confirms the correspondences.

The input of a matcher is the unmatched entities of the last matcher. The matched entities are to be aggregated in final alignment A, which is the alignment of the triple concept, property, value.

A new approach for matching ontologies based on syntactic and semantic methods. It consists of five matchers (exact, substring, prefix, suffix, and WordNet methods) that are used sequentially. Using these five matchers reduce redundancies and increase matching efficiency. This approach gives the end user (or the expert) the capability to confirm the alignments and select the appropriate name of the matching terms.

B. Merging Process

The proposed system presents a merging method of taxonomies in an effective way. It merges taxonomies of the ontologies by the merger defined in figure 6.

The input of this process is the two source ontologies o1, o2, and the alignment A (the output of the matching process). The output is the merged ontology O'.

The merger consists of five operations; *Select Concepts* operation searches for a matching concept in the correspondence for the input concept. Collect Properties determines properties of the selected concept from its correspondence. *Collect Values* determines values of a property from its correspondence.

Merge Hierarchical Classification determines concept location in the hierarchy structure.

Merging Process consists of building the new ontology, O', from the two matched ontologies, O1, O2; by defining the correct hierarchy of each concept.

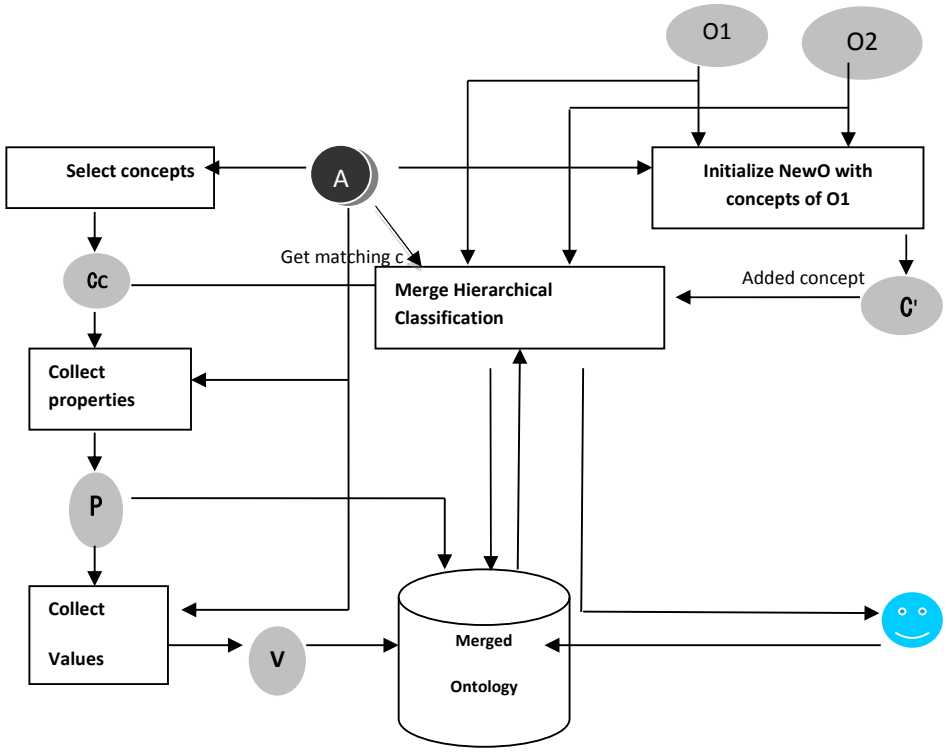

Figure 6: Merge Algorithm

4 EVALUATION OF PROPOSED SYSTEM

This section presents an evaluation for the proposed methodology, I experiment the efficiency and reliability of the MMMS system. A comparison of this work with other systems is presented to evaluate the proposed system.

A. Evaluation of Matching Process

This section valuates the multi-matching technique which solves the redundancy problem. This technique has a variety of matchers which deal with abbreviation, composite words and semantic terms to detect all similarities between source ontologies. These similarities are filtered by threshold and confirmed by the user. But common matching techniques use a single matching criterion to analyze concepts which does not semantically align concepts correctly. These techniques cannot detect different types of similarities in source ontologies and produce an integrated ontology that has duplicate terms (redundancy). We applied the system on *fababean* ontologies, these ontologies have five types of items: (Concepts, Properties, Values, Taxonomies, Relations)

The evaluation methodology uses the following indicators: Precision, Recall, and F-measure. Figure 7 illustrates the idea of the matching comparison.

Precision is a value in the range [0, 1]; the higher the value, the fewer the wrong mappings (false positives) computed, the precision measure is defined as follows:

$$\text{Precision} = \frac{|B|}{|B|+|C|}$$

B represents true positives and C represents false positives.

Recall is a value in the range [0, 1]; the higher this value, the smaller the set of correct mappings which are not found (true positives), the recall measure is defined as follows:

$$\text{Recall} = \frac{|B|}{|A|+|B|}$$

A represents the set of false negatives.

F-measure is a value in the range [0, 1], which is a global measure of the matching

$$\text{F-measure} = \frac{2\times\text{Precision}\times\text{recall}}{\text{Precision}+\text{Recall}}$$

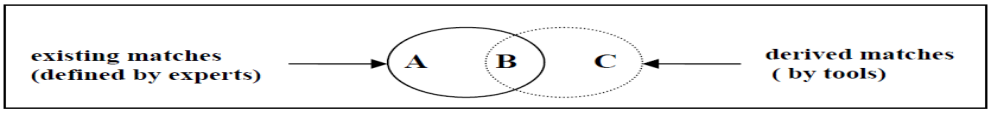

Figure 7: Comparing Existing Matches and Derived Matches

$$\text{Precision} = \frac{\text{Number-of-correct-found-alignments (by system)}}{\text{Number-of-found-alignments (by system)}}$$

$$\text{Recall} = \frac{\text{Number-of-correct-found-alignments (by system)}}{\text{Number-of-existing-alignments (by expert)}}$$

Tables1 shows results of first iteration of matching process, the first iteration executes five matchers on concepts to extract alignments (matching concepts) between two source ontologies, first we experiment the matching process with all matchers, the first row presents number of alignments are detected by the system is 70, while number of correct alignments of them is 58 and number of alignments executed manually by expert is 58,soprecision is 0.79, recall is 1, and f-measure is 0.88 (see figure 8). Second we experiment the matching process without substring matching, the second row presents number of alignments are detected by the system is 65, while number of correct alignments of them is 50 and number of alignments executed manually by expert is 58, so precision is 0.76, recall is 0.86, and f-measure is 0.80 (see figure 8). Third we experiment the matching process without prefix matching, the third row presents number of alignments are detected by the system is 64, while number of correct alignments of them is 52 and number of alignments executed manually by expert is 58, so precision is 0.81, recall is 0.89, and f-measure is 0. 84 (see figure 8). Fourth we experiment the matching process without suffix matching, the fourth row presents number of alignments are detected by the system is 67, while number of correct alignments of them is 49 and number of alignments executed manually by expert is 58, so precision is 0.73, recall is 0.84, and f-measure is 0.78 (see figure 8). In conclusion, the experiments show that the quality of matching increases using the five matchers.

TABLE 1: CONCEPT ALIGNMENTS

	Number-of-correct-found alignments(system) (B)	Number-of-existing-alignments(by experts) (A+B)	Number-of-found-alignments (by system) (B+C)
MMMS with all matchers	58	58	70
Without substring matching	50	58	65
Without prefix matching	52	58	64
Without suffix matching	49	58	67

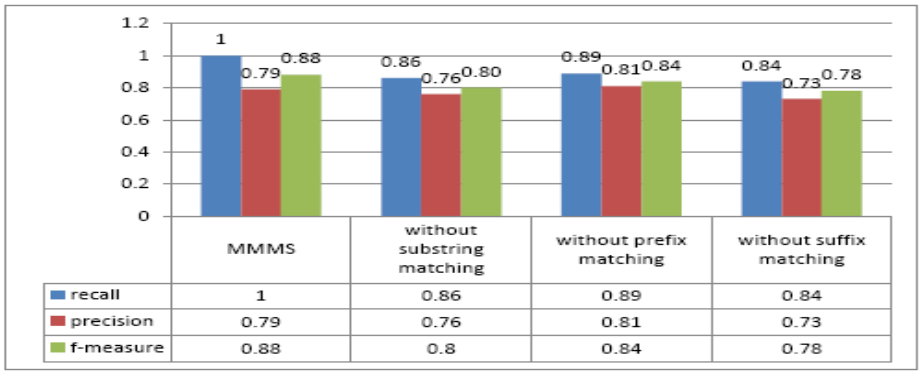

Figure 8: F-measure of Concept Alignments

Table 2 and table 3 show results of second and third iteration of matching process and presents properties, values alignments information respectively. Table 4 shows all alignments. The evaluation results are shown in figures 9, 10 and 11. These figures compare between the performance of the proposed system and the proposed system without a matcher (substring, prefix, or suffix) of properties, and values alignments respectively. The results show that the performance of the proposed system (using all matchers) is better than the performance without substring, prefix or suffix matcher.

TABLE 2: PROPERTY ALIGNMENTS

	Number-of-correct-found alignments(system) (B)	Number-of-existing-alignments(byexperts) (A+B)	Number-of-found-alignments(by system) (B+C)
MMMS with all matchers	82	82	110
Without substring matching	27	82	101
Without prefix matching	43	82	104
Without suffix matching	59	82	100

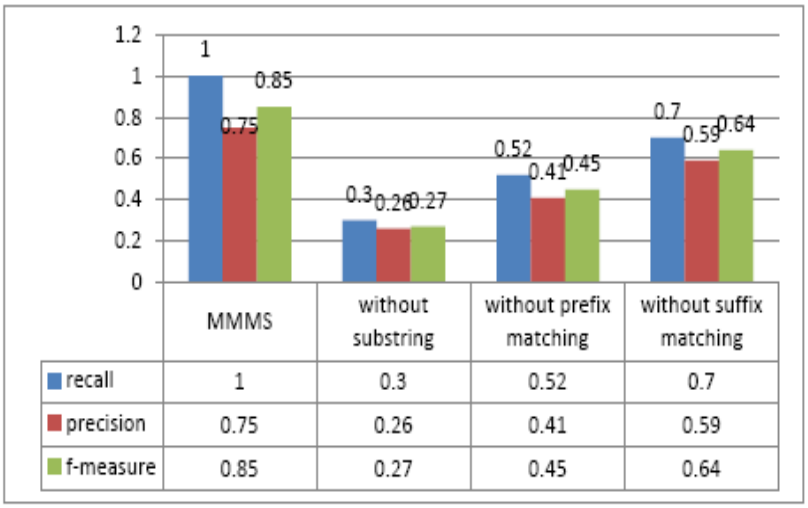

Figure 9: F-measure of Property Alignments

TABLE 3: VALUE ALIGNMENTS

	Number-of-correct-found alignments(system) (B)	Number-of-existing-alignments(by experts) (A+B)	Number-of-found-alignments (by system) (B+C)
MMMS with all matchers	136	136	168
Without substring matching	24	136	35
Without prefix matching	111	136	154
Without suffix matching	111	136	155

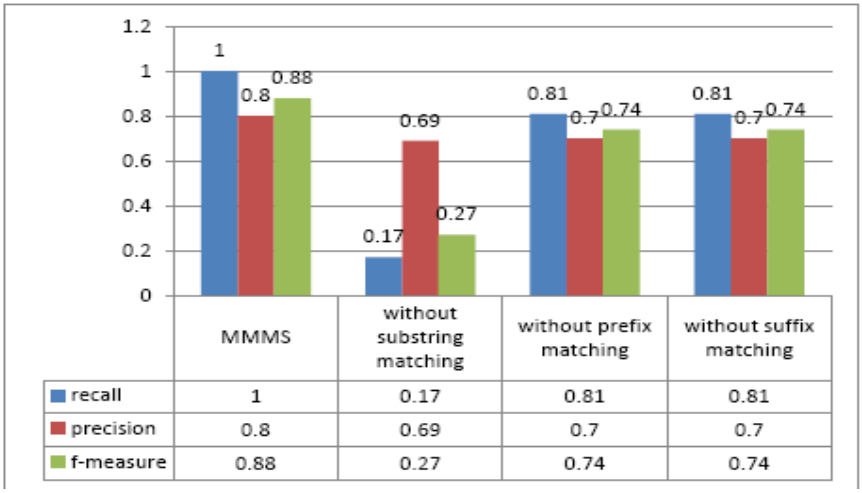

Figure 10: F-measure of Value Alignments

TABLE 4: ALL ALIGNMENTS (CONCEPTS, PROPERTIES, VALUES)

	Number-of-correct-found alignments(system) (B)	Number-of-existing-alignments(by experts) (A+B)	Number-of-found-alignments (by system) (B+C)
MMMS with all matchers	276	276	348
Without substring matching	101	276	201
Without prefix matching	206	276	322
Without suffix matching	219	276	322

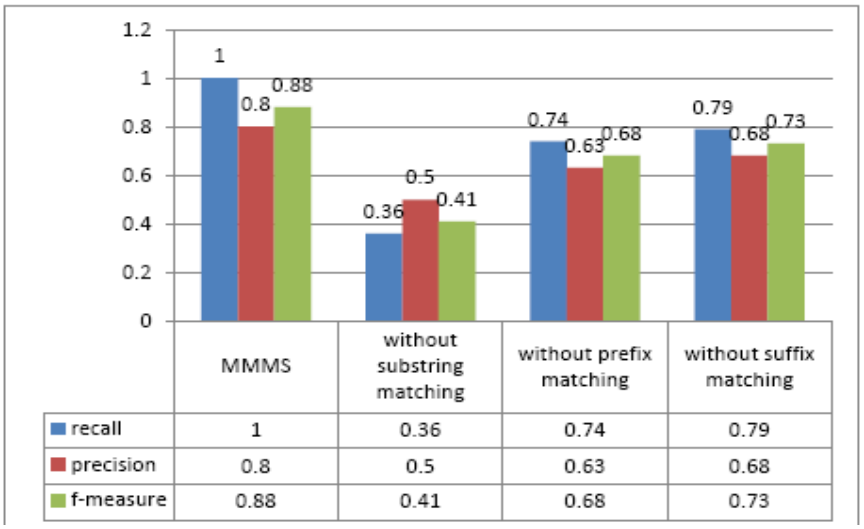

Figure 11: F-measure of All Alignments

1)Comparison between Proposed System and Other Systems

DKP-AOM[5]system has three components, *Match Manager, Consistency Checker*, and *Reasoner*. *Match Manager* consists of several matching algorithms, and uses linguistic and synonym based mappings between words. Linguistic analysis concept and properties discovered the base of words. For example, concept "*pupils*" the stem word is "*pupil*" and properties ("*Accepted*", "*accepting*", "*Accept*", and "*accepts*") the stem word is "*accept*". Synonym similarity is detected by using WorldNet dictionary. *Match Manager* sends the initial mappings to second component *Consistency Checker* for their validation. *Consistency Checker* validates mappings that detected in the initial stage so that the merged ontology stays consistent with reference to the source ontologies. When the initial mappings pass the consistency test, *Consistency Checker*, propagates the mappings to the third component '*Reasoner*'. Reasoner compiles the output as merged global ontology or final list of consistent mappings as required by the end user. In this step, it ensures the ultimate goal of achieving the contentment of merged ontology by checking the correctness and consistency of concepts, properties, and axioms of the generated global ontology. For example: if mapping list has '*programmer*' aligns with '*designer*' and the axioms have '*programmer'* not aligns with 'designer' then the system warns the situation and does not proceed merging with these mappings.

This system has missed some concept mappings according to the human expert. The incorrect or missed mappings are due to the concepts labels formed from composite words.

RiMOM[7]is alignment system uses string-based technique: edit-distance; an edit distance between two objects is the minimum number of insertions, deletions, and substitutions of characters required to transform one string into the other. Edit distances were designed for measuring similarity between strings that may contain spelling mistakes.

Anchor-Prompt[8], it is an algorithm for matching concepts. If there is a similarity betweentwo concepts in the source ontologies, and there are paths connecting these two concepts, then there can be similarities between these paths. Figure 12 shows that there is a match between concept A from one source ontology and concept B from source ontology. It also detects that there is a match between concept names H and J. So the tool suggests that there are some similarities between those concepts which lie between the two matches, such as concepts G and F, and that concepts E and D may share some properties with concept C.The toolsuggestssimilarities; the user may confirm the suggestion, and merge the concepts, or reject the suggestion.

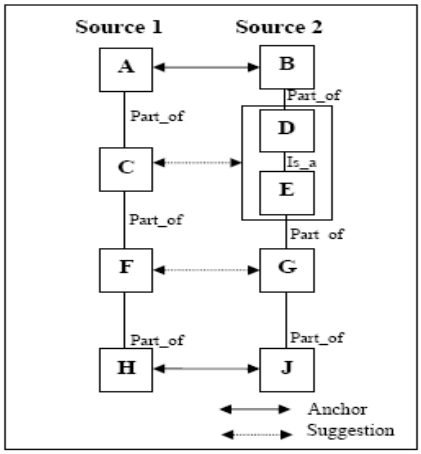

Figure 12: Prompt Algorithm

SKAT(Semantic Knowledge Articulation Tool)[3]. It is a rule-based tool which discovers similarities through a semi-automatic process. Rules provided byDomain experts,these rules are encoded in first order logic. Experts also specify desired similarities and dissimilarities. For example, the rule "*President* is equivalent to *Chancellor*" specifies that we want President to be an appropriate match for Chancellor. SKAT uses string matching as well as structure matching. In the structure matcher, SKAT matches graph slices, i.e. matching the nodes near the root in the first ontology against the nodes near the root in the second ontology.

The proposed system (MMMS) presents semi-automated system for establishing global ontology by merging two ontologies. It uses two processes: matching and merging. Matching process uses string-based technique and language-based techniques. The string-based technique uses four methods: exact method to detect identical terms, and substring, suffix and prefix methods to compare between terms. Using these four methods altogether improve the effectiveness of matching process, (see figure 8, figure 9, and figure 10 in section 4.1). Matching process uses also language-based techniques; this technique uses Word Net Method to detect terms that have the same meaning. This technique improves also the effectiveness of matching process. The proposed system uses a threshold of alignments to filter the similarities between terms. The proposed system can detect terms that other systems cannot for example: (*water-system*, *irrigation-system*), (*growth duration in days, maturity in days*), but DKP-AOM cannot detect these alignments because the proposed system uses suffix matching but DKP-AOM does not, this system has missed some concept mappings according to the human expert. The incorrect or missed mappings are due to the concepts labels formed from composite words. MMMS can deal with the composite words, and detects similarities between them, figure 13 shows that the f-measure of the matching process of MMMS is better than f-measure of the matching process of DKP-AOM. *Anchor-Prompt* uses exact matching but the proposed system uses exact matching substring matching; prefix matching, suffix matching and Word Net Method to detect more similarities and this reduces redundancies and due to consistent merged ontology. SKAT does not use WordNet Method to detect the terms that have the same meaning but the proposed system uses WordNet method. This system just matches ontologies but the proposed system matches and merges ontologies.

TABLE 5: COMPARISONS BETWEEN MMMS, DKP-AOM AND ANCHOR-PROMPT

	Number-of-correct-found alignments(system) (B)	Number-of-existing-alignments(by experts) (A+B)	Number-of-found-alignments (by system) (B+C)
MMMS	276	276	348
DKP-AOM	219	276	322
Anchor-Prompt	100	276	199

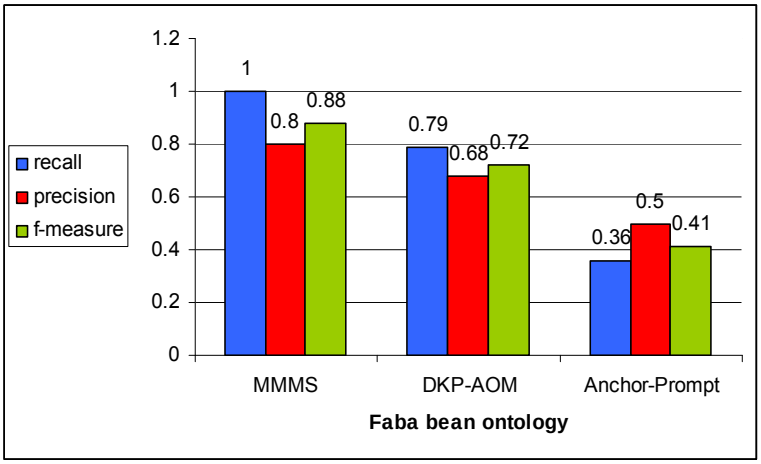

Figure 13: Comparisons between MMMS, DKP-AOM and Anchor-Prompt

MMMS, DKP-AOMand**Anchor-Prompt**are applied on *fababean* source ontologies, table 5 shows the output of matching process for MMMS, DKP-AOM and Anchor-Prompt; first row presents the output of MMMS, number of alignments by the systemis 348, number of correct alignment from them is 276, and number of alignment is obtained from expert manually is 276. So precision is 0.68, recall is 1, and f-measure is0.88, see figure 13

Second row presents the output of DKP-AOM, number of alignments by the system is 322, number of correct alignment from them is 219, and number of alignment is obtained from expert manually is 276. So precision is 0.68, recall is 0.79, and f-measure is 0.72, see figure 13

Third row presents the output of Anchor-Prompt, number of alignments is 199, number of correct alignment from them is 100, and number of alignment is obtained from expert manually is 276. So precision is 0.5, recall is 0.36, and f-measure is 0.41, see figure 13.

MMMS extract more similarities due to use five matchers, so each matcher detects kind of similarities, these similarities are filtered and then confirmed by users, but DKP-AOM system uses three matchers (exact, wordnet, base method) and cannot deal with composite terms, so there are similarities can not detect such as (water-system, irrigation-system) and (growth duration in days, maturity in days). Also Anchor-Prompt uses exact matcher, it also cannot detect kinds of similarities such as (base, stem). Figure 13 shows f-measure of matching processing of MMMS, DKP-AOM and Anchor-Promptis 0.88, 0.72 and 0.41. So the matching quality of MMMS is better than DKP-AOM and Anchor-Prompt.

We compute T-test for the proposed system (MMMS) and DKP-AOM for three groups: first f-measure of concept alignment, second f-measure of property alignment, and third f-measure of value alignment to compare the proposed system and the other system(see table 6), T is 22.4, this means that the MMMS algorithm is accepted by 0.99 significant.

$$T = \frac{\bar{X}_n - \bar{Y}_m}{\sqrt{\frac{s_x^2}{n} + \frac{s_y^2}{m}}}$$

Where $\bar{X}$ is the mean of three groups (n=3) of f-measures for MMMS, where $\bar{Y}$ is the mean of three groups (m=3) of f-measures for DKP-AOM, and Sx is standard deviation for f-measures of **MMMS** and and Sy is standard deviation for f-measures of **DKP-AOM**

TABLE 6: COMPARISONS BETWEEN F-MEASURES FOR MMMS AND DKP-AOM

	F-measure for concept alignment	F-measure for property alignment	F-measure for value alignment	mean	standard deviation (S)
MMMS(X)	0.88	0.85	0.88	0.87	0.0245(Sx)
DKP-AOM(Y)	0.78	0.64	0.74	0.72	0.14(Sy)

B. Evaluation of Merging Process

MMMS solves problems in hierarchy in merged ontology. It solves inconsistencies, redundancies and heterogeneities in merged ontology for example figure 14 and figure 15 are two source ontologies hierarchies for two *pest* ontologies: first ontology has five taxonomies eighteen concepts; the *pests* taxonomy has *insects, diseases, root rot nematode* and *lesion nematode*. *Insects* taxonomy has *leaves* insects, *bod worms, green aphids, black aphids* and *soil worms*. *Diseases* taxonomy has *viral* and *fungal*. *Viral* taxonomy has *bean yellow mosaic virus*, *leaf roll viral* and *necrotic viral*. *Fungal* taxonomy has wilt *diseases, powdery meldiew* and *downy meldiew*.

Second ontology has twenty two concepts and five taxonomies; *pest* taxonomy has insect, *virus, fungus* and *nematodes*. *Insect* taxonomy has *thrips, leafminer, cut worms, aphids, leaf hoppers* and *pod borer*. *Virus* taxonomy has *mottle virus, mosaic virus* and *necrotic viral*. Fungus taxonomy has *root rot, rust, alternaria leaf spot* and *downy mildew*. *Nematodes* taxonomy has *stem nematode, root rot nematode, cyst nematode* and *lesion nematode.*

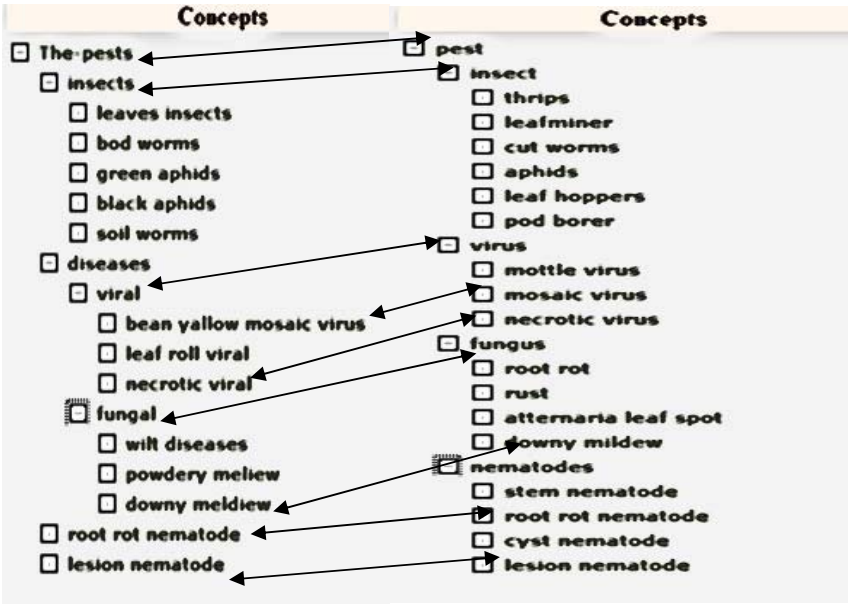

Figure 14: First Pest Ontology **Figure 15: Second Pest Ontology**

◄Correspondence between two matched concepts

The two ontologies has been matched and merged by MMMS system, first step they are matched by matching algorithm to produce similarities of two ontologies, then they confirmed by user (knowledge engineer) to produce alignments like (*the pests, pest*), (*insects, insect*), (*viral, virus*), (*bean yellow mosaic virus ,mosaic virus*), (fungal, fungus), (*downy mildew, downy mildew*), (*root rot nematode, root rot nematode*) and (*lesion nematode, lesion nematode*). In this step the user choose the concept from correspondence to appear in merged ontology, for example: the user selects insects from the correspondence (*insects, insect*) and 'insects' is appeared in merged ontology, so this step solves redundancies because it detects similarities between concepts by semantic and language matchers.

Figure 16: Pest Merged Ontology by MMMS (Merging process is semi-automatic)

Second step source ontologies are merged by merging algorithm (merger) to produce consistent merged ontology; merger solve problems in hierarchies and taxonomies, these problems can be classified into cases: (in the following cases, the **bold concept**means that the user chose it in the matching process)

Case1:

If two concepts('**X**', 'Y')of two ontologies are matched, their parents('**X1**', 'Y1')are matched then merger assigns 'X1' to be parent of 'X'; in hierarchy of merged ontology.

Case2:

If a concept 'C' has no matching concept, and its parent is 'X'. If (X, **Y**) are matching and 'Y' has no offspring then the merger assigns the concept 'Y' to be parent of concept 'C'.

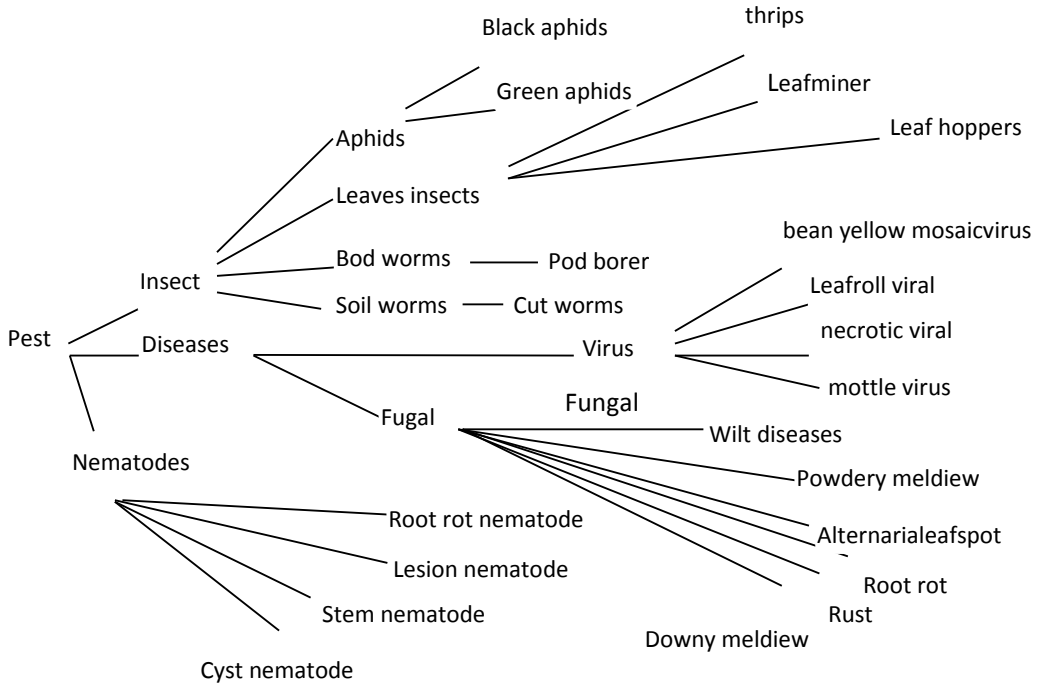

Figure 17: Pest Merged Ontology by Expert

Case3:

If a concept 'C', has no matching concept and its parent 'X' has a matching concept 'Y' then the merger ask user about parent of 'C'('Y' or one of its offspring), for example the user selected 'leaves insects' to be the parent of 'thrips' in the above example.

Case4:

If two concepts (**C1**, C2) are matched, their parents (X1, X2) are not matched and they are at the same level in hierarchies of source ontologies, then the merger ask user to choose one of them to be the parent of C1.

Case5:

If a concept 'X' has no matching and its parent 'Y' has no matching concept also then the merger assigns the 'X' to its parent; Y' in the above example this case has been demonstrated by stem nematode and nematodes.

Case6:

If two concepts (**C1**, C2) are matched, their parents (P1, P2) are not matched and they are at different levels in hierarchies of source ontologies then the merger assigns'C1' to the parent of the lower level. In the above example (***viral,*** *virus*) are matched, their parent (*diseases, pest*) are not matched but 'diseases' is at lower level than 'pest' then the merger assigns 'diseases' to be parent to 'viral'.

/

Figure 18: Pest Merged Ontology by MMMS

(Merging process is run full automatically)

MMMS has been run two times on the above example, one of them the system uses the above six cases (semi-automatic), see figure 16figure 17. In the second time the system did not ask the user in the merging process (i.e. the merging process is fully automatic), see figure 18.

To evaluate merging hierarchy of merged ontology by MMMS, it has been compared with the merged ontology by expert, see figure 17.

Merged ontology by MMMS (semi-automatic merging), see figure 16, has thirty one concepts and 10 taxonomies; the pests taxonomy has insects, diseases and nematodes. *Insects* taxonomy has *leaves insects, bod worms, soil worms* and *aphids*. *Disease* taxonomy has viral and fungal. *Nematodes* taxonomy has *root rot nematode*, lesion nematode, stem nematode and cyst nematode. Fungal taxonomy has *wilt diseases,powdery meldiew*, *root rot, downy meldiew, alternaria leaf spot* and *rust*. *Viral* taxonomy has *bean yellow mosaic virus, leaf roll viral, necrotic viral* and *mottle virus*. *Aphids* taxonomy has *green aphids* and *black aphids*. *Soil worm*'s taxonomy has *cut worms*. *Bod worms* has *pod borer*. *Leaves* insects has *thrips, leafminer* and *leaf* hoppers.

Merged ontology by the expert, see figure 17, has thirty one concepts and 10 taxonomies; *pest* taxonomy has *insect, diseases* and *nematodes*. *Insect* taxonomy has *leaves insects, bod worms, soil worms* and *aphids*. *Disease* taxonomy has virus and fungal. *Nematodes* taxonomy has *root rot nematode, lesion nematode, stem nematode* and *cyst nematode*. *Fungal* taxonomy has *wilt diseases, powdery meldiew, root rot, downy meldiew, atternaria leaf spot* and *rust*. *Virus* taxonomy has *bean yellow mosaic virus*, *leaf roll viral, necrotic*

viral and mottle virus. *Aphids* taxonomy has *green aphids* and *black aphids*. *Soil worms* taxonomy has cut worms. Bod worms has pod borer. *Leaves insects* has *thrips*, *leafminer* and *leaf hoppers*.

From two figures 16 and figures 17, we observe that the two merged ontologies are identical in taxonomies and there are difference only in labels of some concept according to the selection of the expert, so the MMMS is an efficient way to matching and merging ontology, so the ratio of correct concepts locations is 100% in merged ontology by MMMS with semi-automatic merging.

Merged ontology by MMMS using full-automatic merging, see figure 18, is not identical in taxonomies with one obtained from expert because the proposed system do not use cases which ask user in merging process, so some concepts has been located incorrectly in the merged hierarchy, for example, "*green aphid*", "*black aphids*" and" *cut worms*". In above example, the number of the total concepts is 31 concepts, 7 concepts has been located in incorrect taxonomies, the evaluation ratio in the case of full automatic merging is 77% and the evaluation ratio in the case of semi-automatic merging is 100%.

5 CONCLUSION

This paper presents a system that is used for integrating an ontology from other ontologies in the same domain. This system solves problems of integrating ontologies, first it solves redundancy by matching techniques which uses a variety of different matchers to detect more similarities between sources ontologies, so these matchers reduce duplicate terms. The proposed system applies a multi-matching technique while common matching techniques use a single matching criterion to analyze concepts which does not semantically align concepts correctly.

The system provides five matchers (exact method, substring method, prefix method, suffix method, WordNet method) sequentially to cover different kinds of alignments (matching entities) and to make the integrated ontology perfect with no redundancy. The system uses threshold in substring, prefix, and suffix methods to reduce useless correspondences and involves user to confirm alignments.

The evaluation of the system shows that using five matching methods makes the mapping between terms more perfect and efficient. The evaluation of the proposed system shows that the performance of the matching is high (f-measure = 0.88). The comparison of the proposed system and other systems shows that the quality of matching of proposed system, f-measure, was higher. F-measure of matching step for DKP-AOM and Anchor-Prompt are 0.72 and 0.41.

Second the proposed system solves problems in merging hierarchies by merging concepts at the same levels of source ontologies or different levels of source ontologies. Merging process uses Merger algorithm which solves problems in handling hierarchies and taxonomies, these problems can be classified into cases: First case when a concept in taxonomy is not matched with any concept in the other one. The second case when the two matching concepts have different parents in the two taxonomies. The proposed system has a capability to set a concept in correct location of taxonomy of merged ontology, so the result integrated ontology is consistent and has no heterogeneity.

The hierarchy of integrated ontology by the system was compared with the hierarchy of an integrated ontology by the expert. The two results were close. User intervention is requested to confirm similarities and also to confirm location of a concept in new hierarchy of integrated ontology. The proposed system has been applied on the agricultural domain for Faba bean crop. It can be also applied on ontologies of other domains.The complexity of matching and merging algorithm is $O(n^2)$ and the algorithm is accepted by 0.99 significant.

REFERENCES

[1] Pavel Shvaiko, Jerome Euzenat, Ernesto Jimenez-Ruiz, Michelle Cheatham, Oktie Hassanzadeh, Ryutaro Ichise*Ontology Matching, Proceedings of the ISWC Workshop,* December 2016.

[2] J. Euzenat, P. Shvaiko: *Ontology matching*, Springer, 2nd edition, 2013, 511 p., 103 illus., Hardcover, ISBN 978-3-642-38720-3.

[3] A dilHameed, Alun Preece, and Derek Sleeman.In Steffen Staab and Rudi Studer, editors, Handbook on ontologies, chapter 12, pages231–250. Springer Verlag, Berlin (DE), 2004.

[4] Natalya Noy and Mark Musen,*Ontology versioning in an ontology management framework*, IEEE Intelligent Systems, 19(4):6–13, 2004.

[5] Muhammad Fahada, Nejib Moallaa, AbdelazizBouras,*Towards Ensuring Satisfiability of Merged Ontology*, Procedia Computer Science 4 (2011) 2216–2225

[6] P. Mitra, G. Wiederhold, J. Jannink, *Semi-automatic integration of knowledge sources*, In Proc. 2nd International Conference on Information Fusion,pp. 572-581,Sunnyvale (CA US), 1999.

[7] Juanzi Li, Jie Tang, Yi Li, and Qiong Luo, RiMOM: A Dynamic Multistrategy Ontology Alignment Framework. *IEEE Transactions on Knowledge and Data Engineering*,VOL.21, NO. 8, August 2009.

[8] Noy, N. and Musen M. *PROMPT: Algorithm and Tool for Automated Ontology Merging and Alignment*. In Proc.of the 17th National Conference on Artificial Intelligence (AAAI-2000), pp. 450-455, Austin, Texas, USA.

BIOGRAPHIES

Susan F. Ellakwa, Head of Knowledge Engineering and Expert System Tools Department in Central Laboratory for Agricultural Expert System (CLAES), Agricultural Research Center (ARC), Giza, Egypt. PhD degree in Computer Science, Faculty of science, Menoufia University, Menoufia, Egypt

Passent elKafrawy, Associate Professor, Faculty Science, Menofia University since 2013. She got her PhD from the University of Connecticut in United States on 2006 in Computer Science and Engineering. In the field of computational geometry as a branch of Artificial Intelligence. Then she taught in Eastern State University of Connecticut for one year. In 2007 she worked as a Teacher in Faculty of Science, Menoufia University, Mathematics and computer science department since that time till 2013.

إنشاء تبويب معرفة ديناميكى فى مجال الزراعة

سوزان فيصل أمين اللقوة*، بسنت الكفراوى**
*المعمل المركزى للنظم الزراعية الخبيرة مركز البحوث الزراعية
**قسم الرياضيات و نظم الحاسبات-كلية العلوم-جامعة المنوفية
[1]fisalsusan@yahoo.com
[2]passentmk@gmail.com

ملخص البحث- الانتولجى(تبويب المعرفة) يساعد علي توحيد وتداول المفاهيم في مجال معين بين المهتمون بهذا المجال وأيضا بين النظم التطبيقية له. ويمكن بناء الانتولوجى من عدة انتولوجيات مختلفة وموجودة مسبقا لهذا المجال كما يمكن أيضا بناء الانتولوجى بالكامل من البداية وبدون الاستعانة بانتولوجيات موجودة ولكن هذا يكون شاق و مكلف.العديد من الأنتولوجيات للنظم المختلفة قد تكون غير متماثلة بالرغم من أنها

لنفس المجال. ولتوحيد المفاهيم و المعرفة فى مجال معين فإنه من الضرورى أن تدمج هذه الأنتولوجيات وتصبح انتولوجي متكامل و موحد المفاهيم و المعرفة و متناسق. وهذا البحث يقدم نظام شبه آلى لبناء انتولجى متكامل عن طريق تطابق و دمج أنتولوجيات مختلفة وموجودة مسبقا، كما يمكن الإضافة أوالحذف أو التعديل لهذا الانتولوجى عن طريق النظام المعروض في البحث و بذلك فانه يكون ديناميكى يمكن التغييرفيه حسب مقتضيات الامور.

وقد طبق هذا النظام في المجال الزراعى علي محصول الفول البلدي وقد تم الحصول علي أنتولوجى متكامل حيث يمكن إضافة و حذف و تعديل بعض عناصره عند الاحتياج كما يمكن تطبيقه علي جميع المحاصيل الزراعية سواء المحاصيل الحقلية أو البستانية.

يوجد العديد من لغات الحاسوب لتنفيذ الأنتولوجى وقد تم استخدام لغة XML في هذا البحث كما تم استخدام منهجية ال CommonKADS في بناء الانتولوجي. وهذه المنهجية تتعامل مع العناصر الآتية : مفاهيم ، خصائص للمفاهيم، القيم المحتملة للخصائص. والنظام الذي يقدمه هذا البحث يستخدم تقنية لحل مشاكل تطابق و دمج الانتولوجيات باستخدام تقنية تعدد التطابق لايجاد المتناظرات بين المفردات في الانتولوجيات المختلفة ويستخدم أيضا تقنية الدمج حيث يتعامل مع العناصر المختلفة و التصنيفات ذو التسلسل الهرمي .

يعمل النظام على ثلاث مراحل ، كل مرحلة تتعامل مع أحد عناصر الأنتولوجى، المرحلة الأولى لمعالجة المفاهيم، المرحلة الثانية لمعالجة الخصائص، المرحلة الثالثة لمعالجة القيم. كل مرحلة تستخدم خمسة طرق تطبق بالتتالى وهي التطابق بالكامل ، التطابق الجزئي ، التطابق بالبادئة ، التطابق باللاحقة ، التطابق بالمعني وذلك لتغطية كل المتطابقات المحتملة وذلك لزيادة كفاءة التطابق ومنع التكرار ، ويقوم المستخدم بتأكيد أو نفي المتناظرات. وبذلك يكون الناتج النهائى لهذا النظام هو أنتولوجى متكامل ذات تصنيفات ذو تسلسل الهرمي للمفاهيم.