

Featureless similarities for terms clustering using tree-traversing ants

W. Wong, W. Liu and M. Bennamoun

School of Computer Science and Software Engineering
University of Western Australia
Crawley WA 6009
{wilson,wei,bennamou}@csse.uwa.edu.au

Abstract. Besides being difficult to scale between different domains and to handle knowledge fluctuations, the results of terms clustering presented by existing ontology engineering systems are far from desirable. In this paper, we propose a new version of ant-based method for clustering terms known as *Tree-Traversing Ants* (TTA). With the help of the *Normalized Google Distance* (NGD) and *n° of Wikipedia* ($n^\circ W$) as measures for similarity and distance between terms, we attempt to achieve an adaptable clustering method that is highly scalable across domains. Initial experiments with two datasets show promising results and demonstrated several advantages that are not simultaneously present in standard ant-based and other conventional clustering methods.

Keywords. Terms clustering, ant-based clustering method, featureless similarity measures

1 Introduction

Ontology provides inter-operable semantics to an increasing number of applications. Terms are the basic building blocks in ontology engineering. They are considered as the lexical realizations of everything relevant and important to a domain. However, raw terms alone are not suitable for constructing and maintaining an ontology since different terms may represent the same entity (i.e. synonyms), and the same term can represent many different things (i.e. homonyms). Consequently, the task of grouping together variants of terms to form concepts and separating unrelated ones (known as *terms clustering*) constitutes a crucial fundamental step in ontology engineering.

Unlike documents [1], webpages [2], and pixels in image segmentation and object recognition [3], terms alone are lexically featureless. The absence of features requires certain adjustments to be made with regard to the methods for terms clustering. One of the most evident required changes is the use of an appropriate similarity measure. A recent state-of-the-art survey in ontology engineering [?] reveals that all reviewed systems which apply clustering methods rely on the contextual cues surrounding the terms as features (details in Section

2.1). The large collection of documents, and predefined patterns and templates that are required for the extraction of contextual cues makes the portability of such ontology engineering systems difficult. Consequently, non-feature similarity measures are fast becoming a necessity for terms clustering, especially when there are no features to rely on. Along the same line of thought, Lagus et al. [4] state “*In principle a document might be encoded as a histogram of its words...symbolic words as such retain no information of their relatedness*”. Besides the conventional feature-based similarity measure, the existing approaches in ontology engineering are largely reliant on either partitional or hierarchical methods for clustering the extracted terms. Some of these widely adopted conventional methods such as K-means and average-link agglomerative have been shown to yield to a mediocre performance during data clustering in comparison with the ant-based method [5]. Despite the obvious potentials of the ant-based method in data clustering (details in Section 2.2), it remains relatively unexplored for potential applications in ontology engineering.

In this paper, we report our initial attempt to introduce the strengths of ant-based clustering together with featureless similarity into ontology engineering. We propose a new version of ant-based method called *Tree-Traversing Ant* (TTA). TTA employs two featureless similarity measures known as *Normalized Google Distance* (NGD) and *n° of Wikipedia* (n°W) for clustering terms in a two-pass approach. The well-established NGD is employed in the first-pass to create the initial tree of clusters which is later refined during the second-pass using the new measure n°W. Initial experiments with two datasets show promising results and demonstrated several advantages that are not simultaneously present in standard ant-based and other conventional clustering methods. In Section 2, we will give an introduction to the existing techniques employed for term clustering in ontology engineering, and the introduction to standard ant-based clustering and its enhancements to-date. In Section 3, we will present the TTA, and how NGD and n°W are employed to support terms clustering. In Section 4, we will summarize some findings from our initial experiments of clustering terms using TTAs, NGD and n°W. Finally, we conclude this paper with an outlook to future work in Section 5.

2 Related Work

2.1 Some Existing Techniques for Terms Clustering

Faure & Nedellec [?] presented a corpus-based conceptual clustering method as part of an ontology engineering system called ASIUM. Their clustering method is designed for aggregating terms which have already been grouped into basic classes using *sub-categorization frames*. The aggregation of these basic classes is conducted bottom-up using a distance measure inspired by the Hamming distance. Only pairs of basic classes with distance less than the user-defined threshold are aggregated. Maedche & Volz [6] presented a bottom-up hierarchical clustering method that functions as part of the ontology engineering system Text-to-Onto. This clustering method rely on an all-knowing *oracle*, denoted by

H , which is capable of returning possible hypernyms for a given term. The oracle is constructed with the help of WordNet and *lexico-syntactic patterns*. The similarity between each pair terms is computed using a cosine measure with the help of the oracle. For this purpose, the *syntactic surface dependencies* of each term, in the form of modifiers of various types of phrases, are extracted and used as the features for that term. Shamsfard & Barforoush [7] presented two clustering algorithms as part of their ontology engineering system Hasti. Concepts have to be formed prior to the clustering phase. It suffices to know that the process for creating the concepts and extracting relations that are used as features for clustering involve a knowledge extractor where “*the knowledge extractor is a combination of logical, template driven and semantic analysis methods*” [7]. In short, these systems rely on the contextual cues surrounding the terms as features. The large collection of documents, and predefined patterns and templates that are required for the extraction of contextual cues makes the portability of these systems difficult.

2.2 Ant-based Clustering

The idea of ant-based method was first proposed by Deneubourg et al. [8]. During clustering, ants are represented as agents that move around the environment, a square grid, in random. Objects are randomly placed in this environment and ants can pick-up an object, move it and drop it. These three basic operations are influenced by the distribution of objects in the environment. The probability of picking up and dropping of objects are influenced by the following probabilities:

$$P_{pick}(i) = \left(\frac{k_p}{k_p + f(i)} \right)^2$$

$$P_{drop}(i) = \left(\frac{f(i)}{k_d + f(i)} \right)^2$$

where $f(i)$ is an estimation of the distribution density of the objects in the ants’ immediate environment (i.e. local neighbourhood). As $f(i)$ decreases below k_p , the probability of picking up the object becomes very high. Similarly, as $f(i)$ exceeds k_d , the ants will tend to give up the object. Since Deneubourg et al., a small number of researchers have introduced improvements over the basic ants. Gutowitz [9] proposed a variant known as *complexity-seeking ants* that are capable of sensing local complexity. These ants can accomplish their task faster because they are more inclined to manipulate objects in regions with higher complexity. Lumer & Faieta [10] represented the objects in ant-based clustering in terms of numerical vectors and the distance between the vectors is computed using an Euclidean distance. Given that $\delta(i, j) \in [0, 1]$ as the Euclidean distance between object i and every other neighbouring objects j , the neighbourhood function $f(i)$ is defined as [10]:

$$f(i) = \begin{cases} \frac{1}{s^2} \sum_j 1 - \frac{\delta(i,j)}{\alpha} & \text{if } f(i) > 0 \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

where s^2 is the size of the local neighbourhood, and $\alpha \in [0, 1]$ is the constant for scaling the distance among objects. An ant will have to consider the average similarity of object i with respect to all other objects j in the local neighbourhood prior to the pick and drop operation. The radius of perception (i.e. the extent to which objects are taken into consideration for $f(i)$) of each ant at the centre of the local neighbourhood is given by $\frac{s-1}{2}$.

Vizine et al. [11] proposed an adaptive ant clustering algorithm. The authors introduced two major modifications, namely, *progressive vision scheme* and the *use of pheromones* on grid cells. The progressive vision scheme allows the dynamic adjustment of s^2 , while the use of pheromones increases the probability of deconstruction of relatively smaller regions, and increases the probability of dropping objects at denser clusters. Handl & Meyer [12] introduced the concept of *eager ants*. Idle phases are avoided by having the ants to immediately pickup objects as soon as existing ones are dropped. The authors also proposed the notion of *stagnant control* where ants are forced to drop whatever they are carrying after a certain number of unsuccessful drops. In a separate paper [5], Handl et al. demonstrated that ant-based clustering outperforms many of the conventional clustering methods. Some of the most appealing advantages are 1) the tolerance to different cluster sizes, 2) the ability to identify the number of clusters, and 3) graceful degradation in face of overlapping clusters. Nonetheless, the authors have also highlighted two shortcomings of ant-based clustering, namely, the inability to divide more refined clusters within coarser level ones, and the inability to specify the number of clusters.

3 Terms Clustering using Tree-Traversing Ants

Unlike the standard ant-based clustering where the ants walk on grids, the structure employed by the proposed TTA is a dynamic tree. The dynamic tree begins with one root node r_0 consisting of all terms $r_0 = \{t_1, \dots, t_n\}$, and will further branch out to new sub-nodes as required. The first snapshot in Figure 1 shows the start of the algorithm with the root node r_0 initialized with the terms $t_1, \dots, t_{n=10}$. Essentially, each node in the tree is a set of terms $r_u = \{t_1, \dots, t_q\}$. The size of each new sub-node $|r_u| = q$ reduces as less and less terms are assigned to them in the process of creating nodes with higher intra-node similarity. The processing starts with only one ant, while a theoretically infinite number of ants awaits to work at each new sub-node created. In the third snapshot in Figure 1, while the original ant moves on to work at the left sub-node r_{01} , a new ant proceeds to process the right sub-node r_{02} . The exact number of new sub-nodes that can be grown for each main node r_m (i.e branching factor) for this version of TTA is two (i.e. r_{m1}, r_{m2}). TTAs employ pheromones to mark trails and as a positive feedback mechanism from its environment. Similar to some of the standard ants,

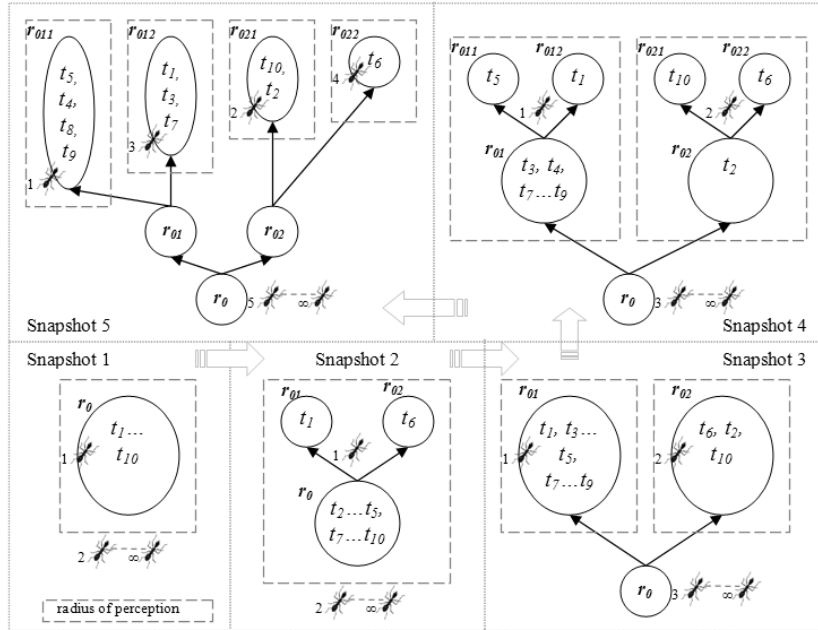


Fig. 1. Example of TTA at work

TTAs too are endowed with the ability of *short-term memory* for remembering similarities and distances acquired through their senses. We equip the newly proposed TTAs with two types of senses, namely, *Normalized Google Distance* (NGD) and *n° of Wikipedia* (n°W). The proposed TTAs work in a two-pass approach for terms clustering. During the first-pass, the TTAs recursively break nodes into sub-nodes and relocate terms until the ideal clusters are achieved. The resulting tree of clusters created in the first-pass are often good enough to reflect the natural clusters. Nonetheless, discrepancies do occur, not due to the TTAs, but rather due to certain unnatural co-occurrences of terms on the World Wide Web that manifest itself through NGD. Accordingly, a second-pass is proposed for relocating terms which are displaced due to NGD. n°W is employed for this purpose. The second-pass can be regarded as a refinement phase for producing clusters with higher quality.

3.1 First-Pass using Normalized Google Distance

The clustering process begins at the root node consisting of all n terms $r_0 = \{t_1, \dots, t_n\}$. Each term can be considered as an element in the node. The TTA randomly picks a term, and proceed on to sense its similarity with every other terms on that same node. The proposed TTA repeats this for all n terms until

the similarities of all possible pair of terms have been memorised. The similarity between two terms t_x and t_y is defined as:

$$s(t_x, t_y) = 1 - NGD(t_x, t_y)\alpha \quad (2)$$

where α is a constant for scaling the distance between the two terms, and $NGD(t_x, t_y)$ is the distance between term t_x and t_y estimated using the original NGD defined by Cilibrasi & Vitanyi [13].

Next, the algorithm will grow two new sub-nodes to accommodate the two least similar terms t_a and t_b . The TTA will move the first term t_a from the main node r_m to the first sub-node while emitting pheromones that traces back to t_b in the process. The TTA will then follow the trail pheromones back to the second term t_b to move it to the second sub-node. The second snapshot in Figure 1 shows two new sub-nodes r_{01} and r_{02} with the ant having moved the term t_1 to r_1 and the least similar term t_6 to r_{02} . Nonetheless, prior to the creation of the new sub-nodes and the moving of the two least similar terms to the new sub-nodes, an ideal intra-node similarity condition must be tested. Eventually, each leaf node will end up with only one term if the TTAs do not know when to stop. For this reason, we adopt an *ideal intra-node similarity threshold* s_T for controlling the extend of branching out. Whenever a TTA sense that the similarity between the two least similar terms exceeds s_T , no further sub-nodes is created and the current main node is left as it is. A high similarity (higher than s_T) between the two most dissimilar terms in the current main node provides a simple but effective indication that the intra-node similarity has reached an ideal stage. More refined factors such as the mean and standard deviation of intra-node similarity are possible but have not been considered here. If the similarity between the two most dissimilar terms is still less than s_T , further branching out is performed. The TTA repeatedly picks up the remaining terms on the current main node one by one and sense their similarities with every other term already located in the sub-nodes. Formally, we define the probability of picking up term t_i by the TTA in the first-pass as:

$$P_{pick}^1(t_i) = \begin{cases} 1 & \text{if } t_i \in r_m \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

where r_m is the set of terms in the main node. In other words, the probability of picking up terms by the TTAs is always 1 as long as there are still terms in the main node.

Each sub-node r_u of the main node r_m can be seen as a neighbourhood in which a TTA senses for the similarities between the term t_i it picked up from r_m with every other term $t_j \in r_u$. Each term $t_i \in r_m$ is moved to neighbourhood r_u that contains the term $t_j \in r_u$ having the highest similarity with t_i . In other words, a TTA considers multiple neighbourhoods prior to dropping a term. Snapshot 3 in Figure 1 illustrates the corresponding two sub-nodes r_{01} and r_{02} that have been filled with all the terms which are used to be located at the main node r_0 . The standard neighbourhood function $f(i)$ defined in (1) represents

the density of the neighbourhood as the average of the similarities between t_i with every other term in its immediate surrounding (i.e. local neighbourhood) confined by s^2 . As the extent to which a TTA can perceive covers all terms in two sub-nodes (i.e. multiple neighbourhoods) corresponding to the immediate main node, the new neighbourhood function $f_{TTA}^1(t_i, t_u)$ is defined as the maximum similarity between term $t_i \in r_m$ and the neighbourhood (i.e. sub-node) r_u . The maximum similarity between t_i and r_u is the highest similarity between t_i and all other terms $t_j \in r_u$. Formally, we define the density of neighbourhood r_u with respect to term t_i during the first-pass as:

$$f_{TTA}^1(t_i, r_u) = \text{maximum of } s(t_i, t_j) \text{ w.r.t } t_j \in r_u \quad (4)$$

where the similarity between the two terms $s(t_i, t_j)$ is computed using (2). Besides deciding on whether to drop an object or not, like in the case of basic ants, TTAs have to decide on one additional issue, namely, where to drop the object. A TTA will decide on where to drop a term based on the $f_{TTA}^1(t_i, r_u)$ that it has memorised for all sub-nodes $r_u \in \{r_{m1}, r_{m2}\}$ of main node r_m . Formally, the decision on whether to drop term $t_i \in r_m$ on sub-node r_v depends on:

$$P_{drop}^1(t_i, r_v) = \begin{cases} 1 & \text{if } (f_{TTA}^1(t_i, r_v) = \text{maximum of } f_{TTA}^1(t_i, r_u) \\ & \text{w.r.t. } r_u \in \{r_{m1}, r_{m2}\}) \\ 0 & \text{otherwise} \end{cases} \quad (5)$$

The function for the first-pass is summarised in Algorithm 1.

3.2 Second-Pass using n° of Wikipedia

The results of our initial experiments have shown that better clusters can be produced if we would allow the TTAs to revisit the nodes for the purpose of refinement. Instead of using NGD, we propose a different means to gauge the similarity between terms called *n° of Wikipedia* (n°W) during the second pass. n°W is inspired by a *game* for *Wikipedians* known as *6° of Wikipedia*¹. It is a task set out to study the character of Wikipedia in terms of the similarity between its articles. An article in Wikipedia can be regarded as an entry of encyclopedic information describing a particular topic. The articles are organized using categorical indices which eventually leads to the highest level, namely, “*Categories*”. Each article can appear under more than one category, making the organization of articles in Wikipedia to appear more as a directed acyclic graph with a root node, instead of a pure tree structure. The huge volume of articles in Wikipedia, the organization of articles in a graph structure, and the availability of the articles in electronic form makes it the ideal candidate for our endeavour.

¹ http://en.wikipedia.org/wiki/Six_Degrees_of_Wikipedia

Algorithm 1 *ant.ant_traverse*(r_m)

```
1: if  $|r_m| = 1$  then
2:   leave_trail( $r_m, r_0$ ) //for use in second-pass
3:   return //only one term left. return to root
4: end if
5:  $\{t_a, t_b\} := \text{find\_most\_dissimilar\_terms}(r_m)$ 
6: if  $s(t_a, t_b) > s_T$  then
7:   leave_trail( $r_m, r_0$ ) //for use in second-pass
8:   return //ideal cluster has been achieved. return to root node
9: else
10:   $\{r_{m1}, r_{m2}\} := \text{grow\_sub\_nodes}(r_m)$ 
11:  move_terms( $\{t_a, t_b\}, \{r_{m1}, r_{m2}\}$ )
12:  for each term  $t_i \in r_m$  do
13:    pick( $t_i$ )//based on 3
14:    for each  $r_u \in \{r_{m1}, r_{m2}\}$  do
15:      for each term  $t_j \in r_u$  do
16:         $s(t_i, t_j) := \text{sense\_similarity}(t_i, t_j)$  //based on 2
17:        remember_similarity( $s(t_i, t_j)$ )
18:      end for
19:       $f_{TTA}^1(t_i, r_u) := \text{sense\_neighbourhood}()$  //based on 4
20:      remember_neighbourhood( $f_{TTA}^1(t_i, r_u)$ )
21:    end for
22:     $\{f_{TTA}^1(t_i, r_{m1}), f_{TTA}^1(t_i, r_{m2})\} := \text{recall\_neighbourhood}()$ 
23:     $r_v := \text{decide\_drop}(\{f_{TTA}^1(t_i, r_{m1}), f_{TTA}^1(t_i, r_{m2})\})$  // based on 5
24:    drop( $\{t_i\}, \{r_v\}$ )
25:  end for
26: end if
27: new_ant1.ant_traverse( $r_{m1}$ ) //repeat the process recursively for each sub-node
28: new_ant2.ant_traverse( $r_{m2}$ ) //repeat the process recursively for each sub-node
```

We define Wikipedia as a directed graph $W := (V, E)$. W is essentially a network of linked-articles where $V = \{a_1, \dots, a_\omega\}$ is the set of articles. We limit the vertices to English articles. At the moment, $\omega = |V|$ is reported to be 1,384,729², making it the largest encyclopedia³ in merely five years since its conception. The inter-connections between articles are represented as the set of ordered pairs of vertices E . At the moment, the edges are uniformly assigned with weight 1. Each article can be considered as an elaboration of a particular event, entity or abstract idea. In other words, an article in Wikipedia can be regarded as a manifestation of the information encoded in the terms. Consequently, we can represent each term t_i using the corresponding article a_i in Wikipedia. Hence, finding the distance between two terms t_i, t_j can be reduced to the discovery of how closely situated are the two corresponding articles a_i, a_j in the Wikipedia categorical indices. The problem of finding the *degree of separation* between two articles can be addressed in terms of the single-source shortest path problem.

² http://en.wikipedia.org/wiki/Wikipedia:Size_comparisons

³ http://en.wikipedia.org/wiki/Wikipedia:Largest_encyclopedia

Since the weights are all positive, we have resorted to Dijkstra’s Algorithm for finding the shortest-path between two vertices (i.e. articles). The benefits of various other existing algorithms for the shortest-path problem have yet to be scrutinized and considered. Formally, the distance between term t_x and t_y given by $n^\circ W$ is defined as

$$\delta(t_x, t_y) = n^\circ W(a_x, a_y) = \sum_{k=1}^{|SP|} c_{e_k} \quad (6)$$

where $n^\circ W(a_x, a_y)$ is the *degree of separation* between articles a_x and a_y which corresponds to the terms t_x and t_y respectively. The *degree of separation* is computed as the sum of the cost of all edges along the shortest path between articles a_x and a_y in the graph of Wikipedia articles W . SP is the set of edges along the shortest path and e_k is the k^{th} edge or element in set SP . $|SP|$ is the number of edges along the shortest path and c_{e_k} is the cost associated with the k^{th} edge. It is also worth pointing out that $\delta(t_x, t_y) \geq 1$ for $t_x \neq t_y$ but no upper bound can be ascertained as there are some Wikipedians who have shown that some articles can be separated by up to eight steps. This is the reason why we adopted the name *n° of Wikipedia* instead of *6° of Wikipedia*.

At the end of the first-pass of creating the sub-nodes, most of the terms would have been nicely grouped into nodes with intra-node similarities exceeding s_T . Only a small number of terms will be isolated into individual nodes. We refer to these terms as *isolated terms*. There are two possibilities of isolated terms in normal cases, one is that the term has been displaced during the first-pass due to discrepancies related to NGD, or the term is in fact an outlier. During the return of the TTAs to the root node at the end of the first-pass (as in line 2 and line 7 of Algorithm 1), trail pheromones will be released to mark the path from the root node to the leaf nodes. For the purpose of relocating the isolated terms to possibly more suitable nodes, one TTA will be allocated to work in the second-pass. The single TTA will jump to the leaf nodes following the pheromone trails. At each leaf node r_l , the probability of picking up a term t_i during the second-pass is 1 if the leaf node has only one isolated term:

$$P_{pick}^2(t_i) = \begin{cases} 1 & \text{if } |r_l| = 1 \wedge t_i \in r_l \\ 0 & \text{otherwise} \end{cases} \quad (7)$$

After picking up an isolated term, the single TTA will continue to jump from one leaf node to the next. In each leaf node, the TTA will determine whether that particular leaf node (i.e. neighbourhood) r_l is the most suitable one to house the isolated term t_i it is carrying based on the average distance between t_i and all other existing terms in r_l . Formally, we define the *density* of neighbourhood r_l with respect to the isolated term t_i during the second-pass as:

$$f_{TTA}^2(t_i, r_l) = \frac{\sum_{j=1}^{|r_l|} \delta(t_i, t_j)}{|r_l|} \quad (8)$$

where $|r_l|$ is the number of terms in the leaf node r_l and the distance between the two terms t_i and t_j is computed using (6). This process of sensing the distance of the isolated term with all other terms in each leaf node is carried out for all leaf nodes. The probability of the TTA dropping the isolated term t_i on the most suitable leaf node r_v is evaluated once the TTA returns to the original leaf node of t_i . Back at the original leaf node of t_i , the TTA will recall the neighbourhood density $f_{TTA}^2(t_i, r_l)$ that it has memorised for all neighbourhoods (i.e leaf nodes). In the case of t_i being an outlier, it will have very high average distances with all other leaf nodes. Without a threshold, the TTA will still relocate the outlier to the leaf node that has the minimum average distance, which by comparison with all other cases of non-outliers, is still considered extremely high. In short, the TTA will drop the isolated term t_i which it is carrying on the leaf node r_v if all terms in r_v collectively yields the minimum average distance with t_i that satisfies the *outlier discrimination threshold* δ_T . Formally,

$$P_{drop}^2(t_i, r_v) = \begin{cases} 1 & \text{if } (f_{TTA}^2(t_i, r_v) = \text{minimum of } f_{TTA}^2(t_i, r_l) \text{ w.r.t. } r_l \in L) \\ & \wedge (f_{TTA}^2(t_i, r_v) \leq \delta_T) \\ 0 & \text{otherwise} \end{cases} \quad (9)$$

where L is the set of all leaf nodes.

Unlike the first-pass where the number of TTAs employed increases at the same rate as the creation of sub-nodes, we only need one TTA in the second-pass. Multiple TTAs revisiting the leaf nodes will introduce problems related to concurrency control such as the communication between TTAs and the possibility of conflict due to the sharing of the same address space (i.e. tree structure). After the TTA has visited all the leaf nodes and has failed to drop the isolated term, the term will be returned to its original location. The failure to drop the isolated term in a more suitable node indicates that it is an outlier. Referring back to the example in Figure 1, assume that snapshot 5 represents the end of the first-pass where the intra-node similarity of all nodes have satisfied s_T . While all other leaf nodes, namely r_{011} , r_{012} and r_{021} consist of more than one term, leaf node r_{022} contains only one term t_6 . Hence, at the end of the first-pass, all TTAs, namely, *ant1*, *ant2*, *ant3* and *ant4* will all retreat back to the root node r_0 . For the purpose of the second-pass, only one TTA will be deployed to relocate the isolated term t_6 from r_{022} to either leaf node r_{011} , r_{012} or r_{021} depending on the average distances of these leaf nodes with respect to t_6 .

4 Initial Experiments and Discussions

Two artificial datasets were used to carry out the initial experiments to evaluate the TTAs together with NGD and n°W. The first is the Animals dataset originally proposed for use with Self-Organizing Maps (SOMs) by Ritter & Kohonen [14]. The second dataset is manually compiled and consists of a good mixture of

31 terms from a large number of domains, namely, “*transport*”, “*animals*”, “*religious leaders*”, “*geographical territories*”, “*finance*”, “*politics*”, “*forestry*” and “*celebrities*”. During the experiments, snapshots were produced to show the results in two parts: results after the first-pass (using NGD) and results after the second-pass (using n°W).

The Animals dataset has been used to evaluate both the standard ant-based clustering (SACA) and an improved version of ant-based clustering (A²CA) by Vizine et al. [11]. The original dataset consists of 16 input vectors, each representing an animal with the binary feature attributes. Both SACA and A²CA discovered two natural clusters, one for “*mammals*” while the other for “*birds*”. While the SACA was inconsistent in its results, the A²CA yielded 100% recall rate over ten runs. The authors of A²CA stated that the dataset can also be represented as three natural groups. In the spirit of the evaluation by Vizine et al., we performed the clustering of the 16 animals using TTA over ten runs. In our case, no features were used. The 16 animals were clustered based on their names. Due to the deterministic nature (i.e. either 0 or 1) of the probability of picking up and dropping terms by the TTAs, the recall rate over the ten runs was 100%. As shown in Figure 2, by setting $s_T = 0.67$ and $\delta_T = 7$, the TTAs automatically discovered two natural clusters after the second-pass: “*birds*” and “*mammals*”. While the ant-based method has the intrinsic capability of identifying natural clusters, conventional clustering methods (e.g. K-means, average link agglomerative clustering) rely on the specification of the number of clusters [15]. Unlike both extremes, TTA has the **flexibility⁴ with regard to the discovery of natural clusters**. The granularity and number of natural clusters in TTA can be adjusted by modifying the threshold s_T . By setting higher s_T , the number of natural clusters for the Animals dataset has been increased to four as shown in Figure 3. A lower value of the desired *ideal intra-node similarity* s_T results in less branching out and hence less clusters, making the elements in the final leaf nodes more loosely coupled. Conversely, setting higher s_T produces more tightly coupled terms where the similarities between elements in leaf nodes are very high. In the experiment depicted in Figure 3, the value s_T was raised to 0.76 and more refined clusters were discovered: “*birds*”, “*hoofed mammals*”, “*mammals kept as pets*” and “*predatory mammals*”.

Another interesting aspect about the experiment in Figure 3 that reveals another advantage of TTA is the present of an outlier namely the term “*Google*”. Even though the definition of an outlier is more rigid from the statistical point of view [16], but in our case, an outlier can simply be considered as a term that does not fit into any of the clusters. According to [16], “*Legitimately, outliers can be viewed as legitimate records having abnormal behaviour. In general, clustering techniques do not distinguish between the two...*”. In the second-pass in Figure 3, the TTA successfully relocates other isolated terms (i.e. those in dotted-line ellipses) created during the first-pass to more suitable nodes while leaving behind the outlier term “*Google*” as it is. As similar terms are clustered into the same node, outliers will eventually be singled out as individual terms in leaf nodes

⁴ By flexibility, we mean that the method can either automatically discover the number of natural clusters or accept predefined value.

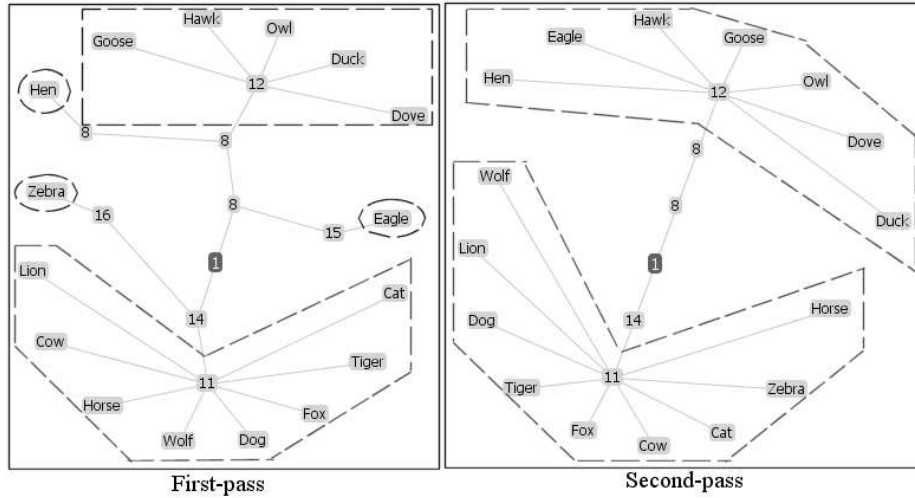


Fig. 2. Experiment 1 using Animal dataset consisting of 16 terms. The first-pass which uses NGD produces clusters with isolated terms. The refinement performed during the second-pass using $n^{\circ}W$ results in accurate clusters. The final result has 2 natural clusters achieved with the setting $s_T = 0.65$ and $\delta_T = 7$

after the second-pass. Consequently, unlike some conventional methods such as K-means, clustering using TTA is not susceptible to poor results due to outliers. In fact, there are two ways of looking at the term “Google”, one as an outlier as described above, and the second as an extremely small cluster with one term. Either way, the term “Google” demonstrates two abilities of TTA: the **capability in identifying and isolating outliers**, and **tolerance to differing cluster sizes**. In fact, Handl et al. [15] have shown through experiments that certain conventional clustering methods such as K-means and one-dimensional self-organizing maps perform poorly in face of increasing deviations between cluster sizes.

The results of the experiment using the second dataset is shown in Figure 4. Similar to the previous experiments, the first-pass produced considerably accurate results with some isolated terms over ten runs. The second-pass refined the tree by relocating the isolated terms to appropriate clusters. In this experiment using the second dataset, all the clusters formed by the TTA after the second-pass correspond precisely to their occurrences in real-life (i.e. natural clusters). Standard ant-based methods often produce clustering solutions that do not stabilise, fail to converge or inconsistent due to the probabilistic and random nature of the pick, move and drop operations. For example, in the evaluation by Vazine et al. [11], the standard ants were inconsistent in their performance over the ten runs using the Animals dataset. This is a very common problem in ant-based clustering when “they constantly construct and deconstruct clusters during the

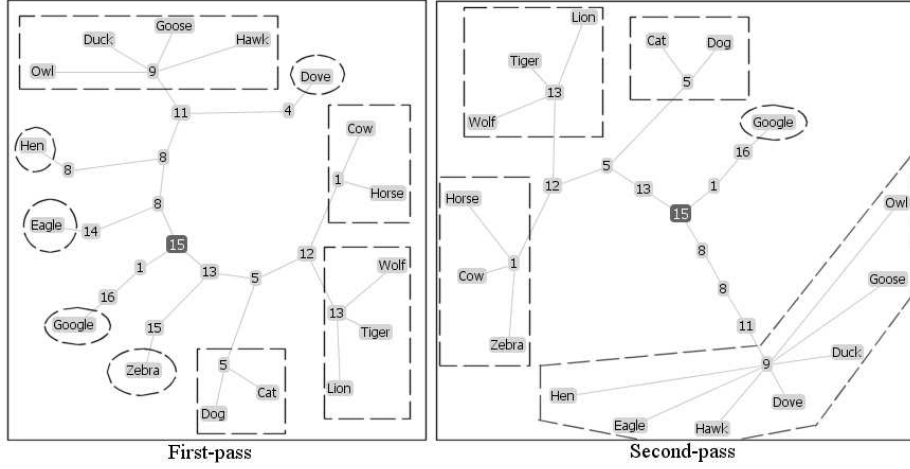


Fig. 3. Experiment 2 using Animal dataset consisting of 15 terms+1 outlier term. The final result has 4 natural clusters+1 outlier achieved with the setting $s_T = 0.76$ and $\delta_T = 5$

iterative procedure of adaptation [11]. The 100% recall rate and the convergence of results that reflect natural clusters for all experiments over ten runs demonstrates that TTAs have the virtue of **consistency and unsusceptibility to sub-optimal conditions**.

The quality of the clustering results is very much dependent on the choice of s_T and δ_T . As an effective rule-of-thumb, s_T should be set as high as possible. Higher s_T will result in more leaf nodes with each having possibly smaller number of terms but tightly coupled together. High s_T will also enable the isolation of potential outliers. The isolated terms and outliers generated by high s_T can then be further refined in the second-pass. The ideal range of s_T derived through experiments is within 7.0 – 7.8. Setting s_T too low will result in very coarse clusters like the ones we saw in Figure 2 where potentially hidden clusters were left uncovered. Regarding the value of δ_T , it is usually set inversely proportional to s_T . As can be witnessed from the three evaluations, as we set s_T higher, we decrease the value of δ_T . For the experiment in Figure 2, we set the $s_T = 0.65$ and $\delta_T = 7$ while we increase $s_T = 0.76$ and decrease $\delta_T = 5$ in Figure 3 to get finer clusters.

5 Conclusion and Future Work

In this paper, we have presented a work in progress that seeks to introduce a decentralized multi-agent system for terms clustering in ontology engineering. For this purpose, we proposed a new version of ant-based clustering called *Tree-Traversing Ant* (TTA) that employs featureless similarity based on *Normalized*

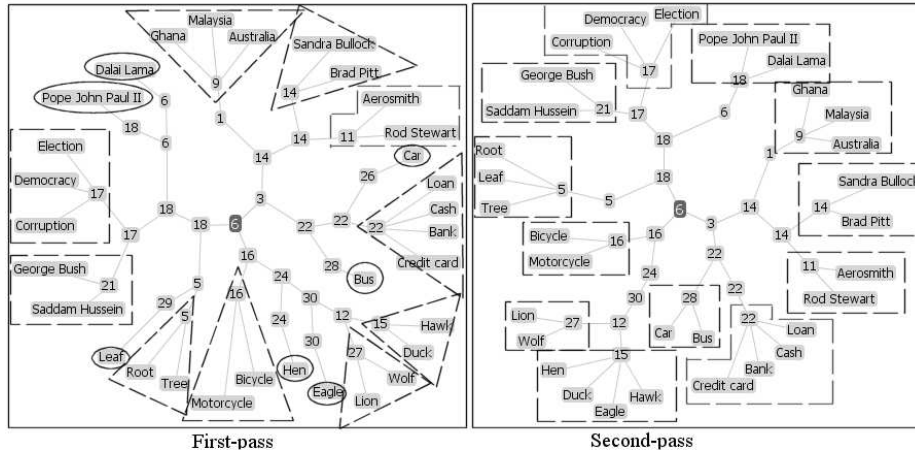


Fig. 4. Experiment 3 using dataset from multiple domains with 31 terms. The final result has 12 natural clusters achieved with the setting $s_T = 0.73$ and $\delta_T = 6$.

Google Distance (NGD) and *n° of Wikipedia* ($n^\circ W$). Our initial experiments have demonstrated promising results. In the course of the experiments, we have highlighted four notable advantages of the TTA that are not simultaneously present in the standard ant-based and conventional methods. These strengths are 1) flexibility with regard to the discovery of natural clusters, 2) capability in identifying and isolating outliers, 3) tolerance to differing cluster sizes, and 4) consistency and unsusceptibility to sub-optimal conditions.

One of the main future work we have in plan is to ascertain the validity and make good use of the implicit hierarchical relationships discovered by the TTAs. Labelling has always been a difficult problem in clustering and hence, we would like to investigate automatic ways to label the clusters and the nodes in the hierarchy. We are also keen on finding ways to automatically adapt s_T and δ_T to maximise the accuracy of the output. More evaluations using real-life, larger datasets will also allow us to study the scalability of the algorithm and its viability in real-world applications. In this aspect, we would like to assess and compare the time and space complexity of TTA with other existing methods.

Acknowledgement

This research was supported by the Australian Endeavour International Post-graduate Research Scholarship, and a Research Grant 2006 from the University of Western Australia.

References

1. Steinbach, M., Karypis, G., Kumar, V.: A comparison of document clustering techniques. Technical Report 00-034, University of Minnesota (2000)
2. Choi, B., Yao, Z.: Web page classification. In Chu, W., Lin, T., eds.: *Foundations and Advances in Data Mining*. Springer-Verlag (2005)
3. Jain, A., Murty, M., Flynn, P.: Data clustering: A review. *ACM Computing Survey* **31**(3) (1999) 264–323
4. Lagus, K., Honkela, T., Kaski, S., Kohonen, T.: Self-organizing maps of document collections: A new approach to interactive exploration. In: *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*. (1996)
5. Handl, J., Knowles, J., Dorigo, M.: Ant-based clustering: A comparative study of its relative performance with respect to k-means, average link and 1d-som. Technical Report TR/IRIDIA/2003-24, Universite Libre de Bruxelles (2003)
6. Maedche, A., Volz, R.: The ontology extraction & maintenance framework: Text-to-onto. In: *Proceedings of the IEEE International Conference on Data Mining*, California, USA (2001)
7. Shamsfard, M., Barforoush, A.: Learning ontologies from natural language texts. *International Journal of Human-Computer Studies* **60**(1) (2004) Page 17–63
8. Deneubourg, J., Goss, S., Franks, N., Sendova-Franks, A., Detrain, C., Chretien, L.: The dynamics of collective sorting: Robot-like ants and ant-like robots. In: *Proceedings of the 1st International Conference on Simulation of Adaptive Behavior: From Animals to Animats*, France (1991)
9. Gutowitz, H.: Complexity-seeking ants. In: *Proceedings of the 3rd European Conference on Artificial Life*. (1993)
10. Lumer, E., Faieta, B.: Diversity and adaptation in populations of clustering ants. In: *Proceedings of the 3rd International Conference on Simulation of Adaptive Behavior: From Animals to Animats 3*. (1994)
11. Vizine, A., deCastro, L., Hruschka, E., Gudwin, R.: Towards improving clustering ants: An adaptive ant clustering algorithm. *Informatica* **29**(2) (2005) 143–154
12. Handl, J., Meyer, B.: Improved ant-based clustering and sorting. In: *Proceedings of the 7th International Conference on Parallel Problem Solving from Nature*. (2002)
13. Ritter, H., Kohonen, T.: Self-organizing semantic maps. *Biological Cybernetics* **61**(1) (1989) 241–254
14. Handl, J., Knowles, J., Dorigo, M.: Ant-based clustering and topographic mapping. *Artificial Life* **12**(1) (2006) 35–61
15. Berkhin, P.: Survey of clustering data mining techniques. Technical report; Accrue Software (2002)