Dynamic Search Optimization for Semantic Webs Using Imperialistic Competitive Algorithm

Iman Rasekh, PhD Student
Institute of computer science, University of Philippines at Los-Banos
Los-Banos, Laguna, Philippines
iman.rasekh@gmail.com

Abstract—**Web 3.0 is known as next generation of web technology after linear presentation of information (Web 1.0) and multi-linear presentation of information (Web 2.0) .Semantic Web is a new collaborative movement toward Web3.0 that led by the World Wide Web Consortium (W3C) .the Semantic Web aims at converting the current web of unstructured documents into a "web of data". Searching is a big challenge in semantic web. Its searching strategy is graph structured search (GSS). It is quite different with the current structure of Web documents. GSS defines as a graph of relationships between anchor nodes but still there is no sufficient search engine designed for it. In this paper we tried to use imperialistic Competitive Algorithm (ICA) to improve Semantic Web Searching. By using ICA, a structured scheme, a powerful semantic search engine could be designed.**

Keywords-Web 3.0, Semantic Webs; imperialistic competitive algorithm; Dynamic search; Graph based search

I. INTRODUCTION

Imperialist Competitive Algorithm (ICA) is a new socio-politically motivated global search strategy that has recently been introduced for dealing with different optimization tasks. This algorithm starts with a given solutions (*countries*) some of the best countries (imperialists) rule the rest of countries (colonies). After competition between Imperialists one empire (optimized answer) remains. This algorithm can be very useful to optimize dynamic search strategy in *Semantic Webs*. Searching in *Semantic Webs* is a tree based searching strategy based on user's query (considered as anchor nodes of the graph). A dynamic algorithm (BFS) can be used to traverse the search tree and its sub graphs [5].

In the proposed method in this article, an Imperialistic approach is used to optimize Dynamic BFS algorithm.

The rest of the paper is organized as follows: Section 1 introduces the Imperialist Competitive Algorithm that was used. In Section 2; semantic Web is introduced and dynamic tree based is discussed. .our proposed method is discussed in section 3 and finally, Conclusions and remarks are explained in Section 4.

1. IMPERIALIST COMPETITIVE ALGORITHM

This algorithm starts with an initial population. Each individual of the population is called a *country*. Some of the best countries (in optimisation terminology, countries with the least cost) are selected to be the imperialist states and the rest form the colonies of these imperialists. All the colonies of initial countries are divided among the mentioned imperialists based on their power. The imperialist states together with their colonies form some empires.

A. *Creation of Initial Empires*

The best country that is the country with the best combination of socio-political characteristics such as *culture*, *language*, *economical policy*, and religion is selected as an Empire. From optimisation point of view this leads to find the *optimal solution* of the problem, the solution with least cost value.
To form the initial empires, the colonies are divided among imperialists based on their power. That is, the initial number of colonies of an empire should be directly proportionate to its power [6].

B. *Pursuing assimilation policy*

The imperialist states tried to absorb their colonies and make them a part of themselves. More precisely, the imperialist states made their colonies to move toward themselves along different socio-political axis such as mentioned. To model this fact and to increase the ability of searching more area around the imperialist, a random amount of deviation is added to the direction of movement. Figure 1 shows the new direction. In this figure θ is a parameter with uniform (or any proper) distribution.

$$\theta \sim U(-\gamma, \gamma) \tag{1}$$

Where γ is a parameter that adjusts the deviation from the original direction. Nevertheless the values of β and γ are arbitrary, in most of implementations a value of about 2 for β and about $\pi/4$ (Rad) for γ results in good convergence of countries to the global minimum [1].

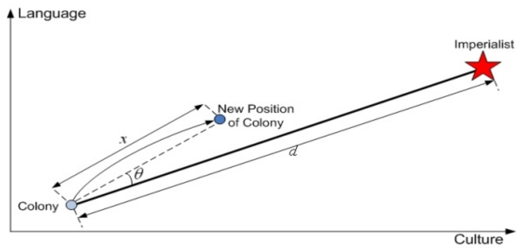

Figure 1. Movement of colonies toward their relevant imperialist

C. *Revolution*

In the terminology of ICA, revolution causes a country to suddenly change its socio-political characteristics. That is, instead of being assimilated by an imperialist, the colony randomly changes its position in the socio-political axis. Figure 2 shows the revolution in Culture-Language axis. In our simulations the revolution rate is 0.3. That means that 30 percent of colonies in the empires change their positions randomly [3].

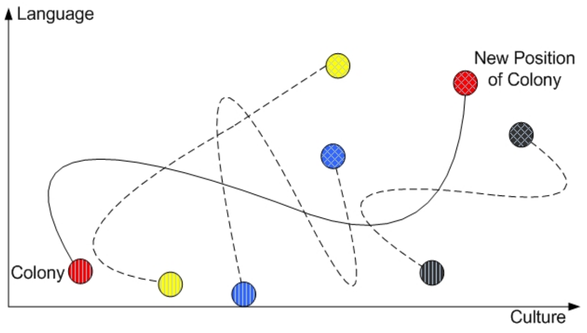

Figure 2. Revolution

D. *Exchanging Positions of the Imperialist and a Colony*

While moving toward the imperialist, a colony might reach to a position with lower cost than the imperialist. In this case, the imperialist and the colony change their positions. Then the algorithm will continue by the imperialist in the new position and the colonies will be assimilated by the imperialist in its new position [1, 4].

E. *Uniting Similar Empires*

If the distance between two imperialists becomes less than threshold distance, they both will form a new empire which is a combination of these empires [3]

F. *Total Power of an Empire*

Total power of an empire is mainly affected by the power of imperialist country. However the power of the colonies of an empire has an effect (Albeit negligible, on the total power of that empire) this fact is modelled by defining the total cost of an empire by

$$T.C = Cost\,(\mathrm{Im}\ prialist_{\,n}) + \xi mean\,\{Cost\,(Colonisofe\ \ mpire_{\,n})\} \qquad (2)$$

Where $T.C._n$ is the total cost of the n^{th} empire and ξ is a positive small number. A little value for ξ causes the total power of the empire to be determined by just the imperialist and increasing it will increase to the role of the colonies in determining the total power of an empire. The value of 0.1 for ξ has shown good results in most of the implementations.

G. *Imperialistic Competition*

All empires try to take the possession of colonies of other empires and control them. The imperialistic competition gradually brings about a decrease in the power of weaker empires and an increase in the power of more powerful ones. The imperialistic competition is modelled by just picking some (usually one) of the weakest colonies of the weakest empire and making a competition among all empires to possess these (this) colonies. To start the competition, first a colony of the weakest empire is chosen and then the possession probability of each empire is found. The possession probability P_P is proportionate to the total power of the empire. The normalized total cost of an empire is simply obtained by

$$N.T.C_n = T.C_n - \max_i\{T.C_i\} \qquad (3)$$

Where, $T.C_n$ and $N.T.C_n$ are the total cost and the normalized total cost of n^th empire, respectively. Having the normalized total cost, the possession probability of each empire is given by

$$P_{P_n} = \left| \frac{N.T.C_n}{\sum_{i=1}^{inp} N.T.C_i} \right| \qquad (4)$$

To divide the mentioned colonies among empires vector **P** is formed as following

$$P = [P_{P1}, P_{P2}, P_{P3}, ..., P_{Pn}] \qquad (5)$$

Then the vector **R** with the same size as P whose elements are uniformly distributed random numbers is created

$$R = [r_1, r_2, r_3, ..., r_n] \qquad (6)$$

$$r_1, r_2, r_3, ..., r_n \sim U(0,1)$$

Then vector **D** is formed by subtracting **R** from **P**

$$D = P - R = [D_1, D_2, D_3, ..., D_n]$$

$$= [P_{P1} - r_1, P_{P2} - r_2, P_{P3} - r_3, ..., P_{PN} - r_n] \qquad (7)$$

Referring to vector **D** the mentioned colony (colonies) is handed to an empire whose relevant index in **D** is maximized [2].

H. *Algorithm termination*

After eliminating the powerless empires if stop conditions satisfied means only one empire was exist the algorithm would be stopped.

II. SEMANTIC WEBS

The Semantic Web is a major research initiative of the World Wide Web Consortium (W3C) to create a metadata-rich Web of resources that can describe themselves not only by how they should be displayed (HTML) or syntactically (XML[1]), but also by the meaning of the metadata readers. Semantic Web is a "Man-made woven web of data" that facilitates machines to understand the semantics, or meaning, of information on the World Wide Web. World Wide Web

[1] Extensible Markup Language

Consortium ("W3C") extends the network of hyperlinked human-readable web pages by inserting machine-readable *metadata* about pages and how they are related to each other, enabling *Semantic Web Automated Agents* (SWAA) to access the Web more intelligently and perform tasks on behalf of users. Semantic Web data models build on and enhance XML—they are valid and well-formed, but also imply additional information to cognize applications. Figure 1 illustrates the Semantic Web stack of building block technologies, going from the grammatical rules of XML to the simplest semantic rules system (RDF[2]) to more sophisticated systems, culminating in OWL Full. DAML[3] and OIL[4] are predecessor investigations that investigated more sophisticated semantic and logic capabilities, which feed into OWL[5] in the form of lessons learned [7].

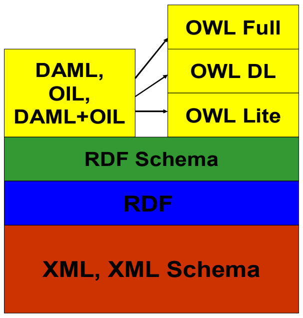

Figure 3. The Semantic Stack

A. *Searching in semantic Webs*

Documents in the Semantic Web are graph structured; Graph Structured Web (GSW) is quite different with the current structure of Web documents. In other words, the Semantic web is a web of documents and the relationships between them in real world, for example figure 4 shows a graph model for a singer named Yu-Yu Each expression in the Semantic Web Includes a subject, a predicate and an object (RDF triple) [12] the relationship between them is expressed as shown in figure 5

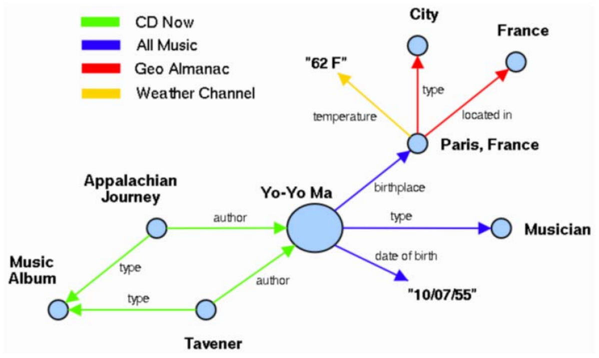

Figure 4. semantic web for Yu-Yu

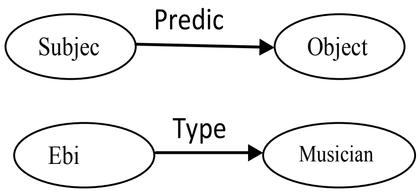

Figure 5. Relationships in Semantic Web

B. *Dynamic Searching strategy in semantic web*

First, we must specify the anchor nodes. Anchor nodes are nodes that determined based on user' s query , for implementing a Dynamic Search first we should Find a adjacency graph of the anchor nodes then we should find the scan order of sub graphs of the adjacency graph, To achieve this target , first of all, we consider all the nodes at the same priority, In the second step we use BFS algorithm to scan each adjacency graph of each node, In the third step, we choose the N first RDF triples, N is a predefined number; in fact, N is the average branching factor of the anchor node[9].

C. *Heuristic Method*

Heuristic Method is a Method to improve the above Three-step method, in this method we choose M, RDF triples that have a common source and edge labels, M is a factor based on bushiness of anchor nodes or based on a distance Function of node from the anchor node, Finally we choose M Triple with common source. Heuristic Method causes a priority for nodes that are closer to anchor node. The possible problem in heuristic method is when the user‘s query include two or more anchor nodes, In this case we can find some path between anchor nodes rather than adjacency sub graphs of anchor nodes and then add adjacency sub graphs of this paths to our output and finally RDF triples are sorted based on sources and neighborhood with anchor node and ultimately based on cost of edges.

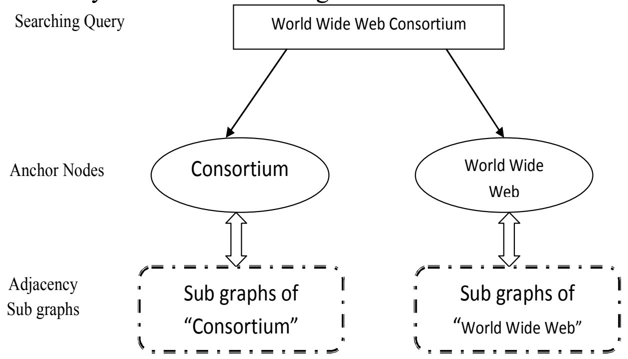
Searching Query
World Wide Web Consortium
Anchor Nodes
Consortium
World Wide Web
Adjacency Sub graphs
Sub graphs of "Consortium"
Sub graphs of "World Wide Web"

Figure 6. Three steps of Dynamic Searching strategy for the phrase "World Wide Web Consortium"

[2] Resource Description Framework
[3] DARPA Agent Markup Language
[4] Ontology Interchange Language
[5] Web Ontology Language

III. DYNAMIC SEARCH USING IMPERIALISTIC COMPETITIVE ALGORITHM

At First we should select the Empires and Colonies. Each anchor node in the input query can be selected as an imperialist so empires can be initialized by determining the anchor nodes. There is no need to select some random points or use a random function. Each part of input query can be considered as an empire and all the adjacency graphs of each node are considered as colonies of those empires (That means that each node in an adjacency graph considered as a colony) But here we need a strong web mining algorithm to determine the anchor nodes because we can't consider each separate word as an anchor, sometimes an anchor is made of some separate words for example for the phrase **"***World Wide Web Consortium***"** can be divided into only two phrases "*World Wide Web "*and *"Consortium"* because *"World Wide Web "is* a known phrase figure -7 shows that how can it possible.

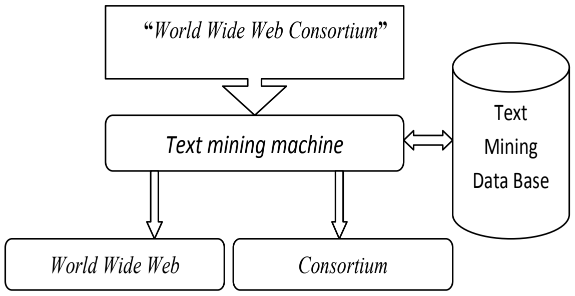

Figure -7 text-mining approaches for "*World Wide Web Consortium*"

All the data from the adjacency graphs (colonies) should move toward their relevant imperialist, this is the Assimilation policy that can be implem ented in semantics webs. In semantic webs Data from each sub-adjacency-graph should move toward its anchor node because the analysis only can be done at interpretation machine at anchor nodes, figure 8 shows this movement. This kind of implementation can improve the time complexity. Since *Semantic Web* uses metadata each anchor node can determine the distance of other nodes from itself. Better colonies are closer to the imperialist (anchor node) that reduce the time complexity of BFS with uniform distribution. (using formula 1)

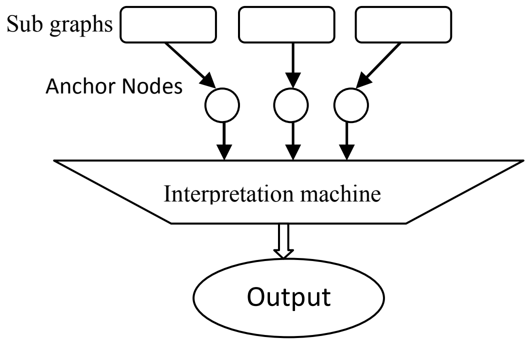

Figure 7. Data movement in semantic webs

Revolution also sometimes happens in semantic webs and the position of some colonies is changed randomly, because sometimes nodes in the graph should change their positions to create more sufficient sub graphs. That means that sometimes a lower sub graph in a *sub graph hierarchy tree* of an anchor node can direct us to a better answer than all its ancestor nodes (parents). In this condition the revolution happen and a node change it s position with one of his ancestors in *sub graph hierarchy tree.* Figure -8 shows this revolution. In Semantic *Web,* revolution can help us to avoid local optimum traps in dynamic search by restarting search from another suitable position in our search space. For example if the revolution rate is 0.3. That means that 30 percent of colonies in the empires change their positions randomly and it can help the search engine to achieve a better answer.

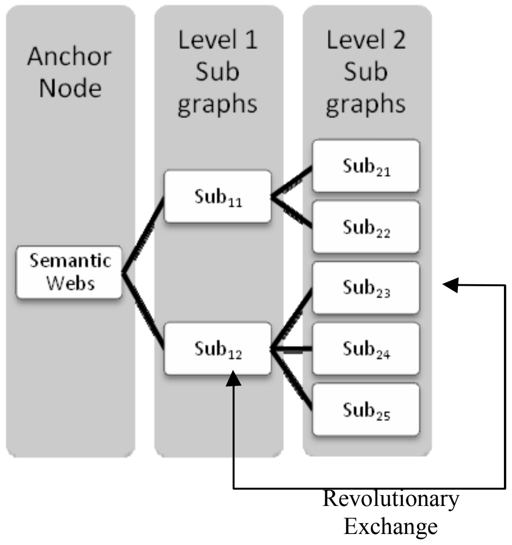

Figure 8. Revolutions in *sub graph hierarchy tree*

In traversing nodes of a sub graph if a node can lead us to an answer better than its father answer that means there is a colony (child node) in an empire which has lower cost than the imperialist; in this case the positions of that colony and the imperialist should be exchanged.

Sometimes we need to unite anchor nodes (similar empires) because sometimes some query parts should combine to achieve better answer.

Total cost of all anchor nodes (empires) can be computed by adding the cost of sub graphs of each anchor node.

- There is a king-size Imperialistic competition in semantic webs. The weakest sub graphs (colonies) from the weakest anchor nodes (empires) are occupied by one of the most powerful empires then the powerless empires should be eliminate, That process should be continued till conditions satisfied. Since each anchor node connected to several sub graphs the search may lead to different answers but competition can help us to reach a unique final answer.

IV. CONCLUSION

In this research, a BFS search system is proposed. An imperialistic competitive algorithm is used to optimize the dynamic search. But a strong web mining algorithm is needed to determine the anchor nodes because we can't consider each separate word as an anchor. The analysis only can be done at *interpretation machine* at anchor nodes so Data from each sub-adjacency-graph should move toward its anchor node. Moreover, we can manage all adjacency sub graphs of each anchor nodes we can change the position of each sub graphs through revolution and exchange position and this process should be continued till reaching the unique final answer.

Since dynamic search in semantic web is graph based and also we don't have any exact information about the total graph so a evolutionary algorithm can help us to reach a good answer, we test our proposed with a database including 500 words in 10 category then we compare the normal tree-based search (TBS) -BFS for each adjacency graph- and ICA table1 shows a comparison between the normal tree-based search (TBS) and ICA with different revolution rates 0.2 (ICA3),0.3(ICA3), 0.4(ICA3), 0.5(ICA5) and our runtime and efficiency was as follows .table1 shows the results of our simulation where efficiency is define as the percentage of correct answers.

TABLE I. COMPARING ICA AND TBS

Algorithms	Run time	Efficiency
TBS	0.65	66%
ICA2	0.55	73%
ICA3	0.53	73%
ICA4	0.56	68%
ICA5	0.57	68%

As you see in table the ICA implies better answer than TBS and ICA3 provides the best answer so 0.3 is the revolution rate for our simulation. But still need a good evaluation function to avoid local optimum traps in search and that is our goal in the future works

V. REFERENCES

[1] Atashpaz-Gargari, E., Hashemzadeh, F., Rajabioun, R. and Lucas, C. (2008). Colonial Competitive Algorithm, a novel approach for PID controller design in MIMO distillation column process, *International Journal of Intelligent Computing and Cybernetics,* 1 (3), 337–355.

[2] Atashpaz-Gargari, E., Lucas, C. (2007). Imperialist Competitive Algorithm: An algorithm for optimization inspired by imperialistic competition, *IEEE Congress on Evolutionary Computation*, 4661–4667.

[3] Biabangard-Oskouyi, A., Atashpaz-Gargari, E., Soltani, N., Lucas, C. (2008). Application of Imperialist Competitive Algorithm for materials property characterization from sharp indentation test. *To be appeared in the International Journal of Engineering Simulation*.

[4] Rajabioun, R., Hashemzadeh, F., Atashpaz-Gargari, E., Mesgari, B., Rajaei Salmasi, F. (2008). Identification of a MIMO evaporator and its decentralized PID controller tuning using Colonial Competitive Algorithm. *Accepted to be presented in IFAC World Congress*.

[5] Rajabioun, R., Atashpaz-Gargari, E., and Lucas, C. (2008). Colonial Competitive Algorithm as a Tool for Nash Equilibrium Point Achievement. *Lecture notes in computer science,* 5073, 680-695.

[6] Sepehri Rad, H., Lucas, C. (2008). Application of Imperialistic Competition Algorithm in Recommender Systems. In: 13th Int'l CSI Computer Conference (CSICC'08), Kish Island, Iran.

[7] F. Manola and E. Miller (eds), "RDF Primer." W3C Working Draft 23 January 2003. Available from http://www.w3.org/TR/rdf-primer/.

[8] Dublin Core Metadata Initiative (DCMI) Web Site: http://dublincore.org/.

[9] D. Brickley and R. V. Guha (eds), "RDF Vocabulary Description Language 1.0: RDF Schema." W3C Working Draft 23 January 2003.

[10] The DARPA Agent Markup Language Web Site: http://www.daml.org.

[11] R. Ouellet and U. Ogbuji, "Introduction to DAML: Part I". Available from http://www.xml.com/pub/a/2002/01/30/daml1.html.

[12] R. Ouellet and U. Ogbuji, "Introduction to DAML: Part II". Available from http://www.xml.com/pub/a/2002/03/13/daml.html.

[13] OIL Project Web Site: http://www.ontoknowledge.org/oil/.

[14] D. Fensel, F. van Harmelen, I. Horrocks, D. L. McGuinness, and P. F. Patel-Schneider, "OIL: An Ontology Infrastructure for the Semantic Web." IEEE Intelligent Systems, March/April 2001.

[15] Web-Ontology (WebOnt) Working Group (OWL) Web Site: http://www.w3.org/2001/sw/WebOnt/.

[16] D. L. McGuinness and F. van Harmelen (eds), "OWL Web Ontology Language Overview." W3C Working Draft 31 March 2003. Available from http://www.w3.org/TR/owl-features/.