ISSN No. 0976-5697

Volume 5, No. 6, July-August 2014

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

# Software Component Clustering using SPARROW Algorithm

Y. Mohana Roopa
Research scholar, Dept. of CSE
JNTUA, Anantapur A.P.India

Dr.A. Rama Mohan Reddy
Professor, Dept. of CSE
SVU CE, SVUniversity Tirupati, A.P,India

***Abstract***: Component Based Software Development (CBSD) provides a high efficient and low cost way to construct software systems by integrating reusable software components. Although CBSD has already become a widely accepted paradigm, it is still beyond possibility to assemble components easily from COTS components into one application system. In real world, such as automation domain, this probability is unacceptable because additional measures, time, efforts, and costs are required to minimize its impacts. Many general clustering approaches have been proposed in literature to manage the composition of system at early stage of development. This paper investigates to identify the component clusters in parallel using a multi-agent adaptive algorithm called SPARROW algorithm. The results of this study are important since it will be used to develop an efficient Component Based Software Architecture.

***Keywords:*** Component-Based software development, component clustering, software architecture, SPARROW Algorithm.

## I. INTRODUCTION

Component-based software development approach is based on the idea to develop software systems by selecting appropriate off-the-shelf components and then to assemble them with a well-defined software architecture. In recent years, there have been increasing interests in using Component-Based System Development (CBSD) approach, particularly COTS (Commercial Off The Shelf) components [1], to develop large complex applications. Both software consumers and developers share the interest for the CBSD approach because of the clear advantages. Some advantages are but not limited to:

The efficiency of development increased, the product becomes more reliable, need for maintenance is radically decreased, the development time decreases, and the usability of the products increases. Although it promises faster time-to-market and increased productivity [2], many risks has been introduced when developing COTS-based systems such as failure to satisfy the quality attributes. The use of good quality components to develop system does not grantee to obtain system with the satisfied quality. Indeed, bad quality components will not produce high quality product, and even good quality components can damage a good product if the composition is not managed appropriately. Consequently, the failure to satisfy the quality attribute such as reliability means a financial loss, increased expenses of hardware, higher cost of software development, and harder than that, the loss of relationships with consumers. Whenever, quality issues are addressed at implementation or integration time, correction of problems impacts the cost, schedule, and quality of the software.

The clustering methods can be classified into partitioning methods, hierarchical methods, density-based methods, and Grid-based methods [3]. Recently, other algorithms based on biological models have been proposed to solve the clustering problem [4]. These algorithms are characterized by the interaction of a large number of simple agents sensing and changing their environment locally. They exhibit complex, emergent behavior that is robust compared to the failure of individual agents. Ants colonies, flocks of birds, termites, swarms of bees etc. are agent-based insect models that exhibit a collective intelligent behavior (swarm intelligence) and may be used to define new algorithms of clustering.

The multi-agent based models such as Ant colonies, bird and swarms of bees, exhibit a collective behavior. Based on these biological models, many new algorithms have been devised to solve the complex problems in the area of computer science. These algorithms are characterized by the interaction of a large number of agents which following simple rules. One of the first collective behavior models is the flocking model, which is used in popular applications like animation. Normally, flocking is considered as an Artificial Life algorithm because of its budding property [5]. The flocking algorithm takes advantage of the collective search mechanism a flock implies, by which if a member of a flock finds an area of interest; the mechanics of the flock will drive other members to scan that area in more detail. A standard clustering algorithm is used to scan the entire dataset in order to discover the clusters.

The main goal of the proposed work is to efficiently identify the component clusters in parallel using a multi-agent adaptive algorithm called SPARROW algorithm. In this proposed work, some test cases are generated for component values like cost, time etc. According to these test cases [6],[7], this proposed approach will effectively generates the clusters in parallel.

## II. NEED FOR CLUSTERING SOFTWARE COMPONENTS

When an architect starts building a new CBSD application, he has many options to do this task. Each probable solution is arranging from a mixture of distinctive components. All those possible alternatives are called Design Options. The combination that satisfied the performance requirements is the target of the architect. However, design options are proportional with the degree of freedom. The degrees of freedom are resulted due to the following [8]: Components, the selection of one component from number of components instances with the same functionality but different performance specifications; Resource Allocation, due to the fact that, the selection of hardware does not impact the functional of components, its configuration could be changed during search. Therefore, hardware environment are modeled separately from the common assembly. In fact, manual or/and mismanaging composition lead to undetected problems in the system. Researchers have proposed Software component clustering Approaches [9] to avoid such problem since it provides early evaluation for architecture.

## III. A MULTI-AGENT ADAPTIVE ALGORITHM

SPARROW is a multi-agent algorithm where agents use modified rules of Reynolds' standard flock algorithm to transform a boid into a hunter foraging for clusters in spatial data A parallel spatial clustering algorithm SPARROW (***SPAtial ClusteRing AlgoRithm thrOugh SWarm Intelligence***), which is based on an adaptive flocking algorithm combined with a density-based cluster algorithm, to discover clusters of arbitrary shape and size in spatial data. SPARROW uses the stochastic and exploratory principles of a flock of birds for detecting clusters in parallel according to the density-based principles of the DBSCAN algorithm, and a parallel iterative procedure to merge the clusters discovered [10].

It begins with a fixed number of agents that take up a randomly generated position. Then, a core point is identified as each agent moves around the spatial data testing the neighbor of each location. The neighbors of the identified core point are given a temporary label. These labels are updated as multiple clusters. Contiguous points belonging to the same cluster take the label corresponding to the smallest label in the group of contiguous points. The movements of the agents are all described in Reynolds's model.

The color is used as a communication device between the flock agents to indicate the roadmap they need to follow. The roadmap is adaptively attuned as the agents alter their color moving to explore data until they reach the goal. Consider a d dimension search space in which the flocks move. Different agents are characterized by a different color: red, revealing similar test case values, green, a medium one, yellow, a low one, and white, indicating a total absence of values. The main idea behind this approach is to take advantage of the colored agent in order to explore more accurately the most interesting regions (signaled by the red agents) and avoid the ones without interesting points (signaled by the white agents). Red and white agents stop moving in order to signal this type of region to the others, while green and yellow ones fly to find denser zones. Indeed, each flying agent computes its heading by taking the weighted average of alignment, separation and cohesion (as illustrated in fig 1).

The following are the key features of our model, which is different from Reynold's:

a. Alignment and cohesion do not consider yellow boids, since they don't move in a very attractive zone.
b. Cohesion is the resultant of the heading towards the average position of the green flock mates (centroid), of the attraction towards reds, and of the repulsion from whites.
c. A separation distance is maintained from all the agents, without considering their color.

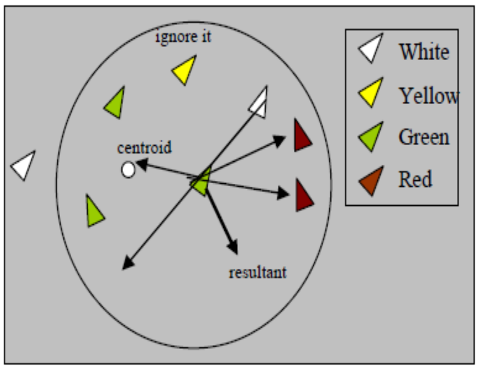

Figure 1. Computing the direction of a green agent

Yellow and green agents will compute their direction, according to the rules previously described, and will move following this direction and with the speed corresponding to their color. Note that the color of the agent is assigned on the basis of the desired property at the point in which it falls; the assignment is made on a scale going from white (property = 0) to red (property > threshold), passing for yellow and green, corresponding to intermediate values.

Agents will move towards the computed direction with a speed depending on their color: green agents more slowly than yellow agents since they will explore more interesting regions. An agent will speed up to leave an empty or uninteresting region whereas it will slow down to investigate an interesting region more carefully.

The variable speed introduces an adaptive behavior into the algorithm. In fact, agents adapt their movement and change their behavior (speed) on the basis of their previous experience and on the position of the red and white agents. Indeed, red and white agents will stop signaling to the others respectively the interesting and desert regions. Note that, for any agent that has become red or white, a new agent will be generated in order to maintain a constant number of agents exploring the data.

In the first case (red), the new agent will be generated in a close random point, since the zone is considered interesting, while in the latter it will be generated in a random point over all the space .Anyway, this does not affect the overall performance of the system as the number of agents was not increased; in fact, white and red agents are not real agents, but only their position is stored. Finally, in the case where the agent falls in the same position as an older one it will be regenerated using the same policy described above.

## IV. USING SPARROW ALGORITHM FOR CLUSTERING COMPONENTS

Assume, N is the number of component values called test cases and t is the set of all the test cases [11],[12]. We use three color indications such as RED, GREEN and YELLOW. For coverage, the indication color is RED, for time its GREEN and number of errors are indicated by YELLOW color. Then, each agent is classified into above categories. The agents are clustered into groups by the SPARROW algorithm as shown in the Fig 2.

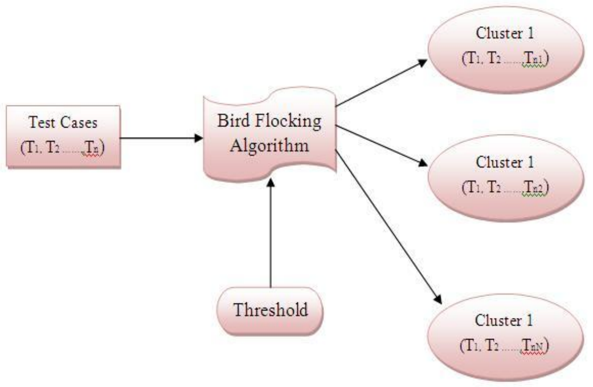

Figure 2: Test Cases Clustering using Birds Flocking Algorith*m*

Normally, a flock is a group of agents all staying close to each other, and the cohesion component is responsible for this action. Each agent watch the position of other agents to observe if it is within a specified neighbor radius, that is, it checks to see which other agents are close enough to be considered flock mates. The positions of the eligible neighbors are averaged and the agent moves towards that position. In this way, each one aims to move towards the center of the flock that results in, all of them are staying close together. Agents will move towards the desired destination with a speed depending on their color. The green agents travel more slowly than the yellow agents since they will discover denser zones of clusters.

An agent will increase the speed to depart a vacant or dull region but it slows down to explore an interesting region more carefully. The variable speed has established an adaptive behavior in the algorithm [13]. The movement and speed are changed by the agents based on their earlier experience represented from red and white agents. The cohesion component is computed by averaging the position of the all neighbors within the radius.

In the merging stage, two diverse cases are handled: when having never visited points in the circular neighborhood and when having points belonging to diverse clusters. In the first case, the points are labeled and a new cluster is formed, whereas in the second case, all the points will be pooled into the same cluster i.e., they will get the label of the cluster discovered first. A cage effect is occurred during the simulations, that is, some agents are detained inside the regions bounded by red or white agents and would have no way to depart, wasting some valuable resources for the exploration. Thus, to shun this effect, a limit is prescribed for their life. Hence, when their age goes beyond a determined value (max Life) they have been destroyed and regenerated in a new randomly selected location of the space.

We need to utilize the flocking algorithm [14],[15] to discover the multidimensional space searching point. A continuous data point can be represented in a multidimensional Euclidean space, by simply normalizing its attributes. In the following, we have given a proper depiction of the extension of the flocking algorithm to multidimensional space. Consider a multidimensional space with d as dimension. Each boid k can be denoted as a point in the space having coordinates and having directions, where gives the angle between the new direction of the boid k and axis i. Each boid will move according to the velocity. For each iteration t, the new position of the agent k is given by the following equation:

$$x_k(t+1) = x_{ki}(t) + v_k \times c_{ki} \quad \textbf{\textit{where i=1…d (1)}}$$

And represents the projection along the i axis of the direction of the boid k. Each component is determined by adding the respective components of alignment, separation and cohesion.

i.e., Thus, in a multidimensional space, the components are calculated as:

$$c_{k1} = \prod_{j=1}^{d-1} \cos(\beta_{kj})$$

$$c_{ki} = \sin(\beta_{ki-1}) \prod_{j=i}^{d-1} \cos(\beta_{kj}) \qquad i = 2\ldots d\,(2)$$

Once the clustering process using bird flocking algorithm is completed, the optimization of the clustered test cases is carried out using any optimization algorithm

## V. CONCLUSION

Architect needs to use clustering of components to avoid problem of quality dissatisfaction cause due to the late evaluation of developed system. A multi-agent adaptive algorithm called SPARROW algorithm provides efficient clustering of components for CBSA.

Once the clustering process using adaptive flocking algorithm is completed, the optimization of the clustered test cases is carried out using optimization algorithms. From this we can get the adaptive configuration of the components to fit in to the software architecture.

## VI. REFERENCES

[1]. Philippe Kruchten "Architectural Blueprints-The "4+1" View Model of Software Architecture" IEEE Software 12 November 1995, pp. 42-50.

[2]. Voas, J.COTS Software: The Economical Choice Software, IEEE, 1998. 15(2): pp. 16- 19.

[3]. Gianluigi Folino and Giandomenico Spezzano "An Adaptive Flocking Algorithm for Spatial Clustering" First International Conference on Emerging Trends in Engineering and Technology, IEEE (2008).

[4]. Chung-Horng Lung "Software Architecture Recovery and Restructuring through Clustering Techniques" Proc. of the 3rd International Software Architecture Workshop (ISAW), 1998, pp.101-104.

[5]. Bonabeau E., Dorigo M., Theraulaz G., "Swarm Intelligence: From Natural to Artificial Systems" Oxford University Press, 1999.

[6]. Gianluigi Folino and Giandomenico Spezzano "An Adaptive Flocking Algorithm for Spatial Clustering " google scholar.

[7]. Jose Carlos, Mario, Alberto, Francisco "A Strategy for Evaluating Feasible and Unfeasible Test Cases for the Evolutionary Testing of Object-Oriented Software" ACM (2008).

[8]. Nirmal Kumar Gupta and Dr. Mukesh Kumar Rohil "Using Genetic Algorithm for Unit Testing of Object- Oriented software", First International Conference on Emerging Trends in Engineering and Technology, IEEE (2008).

[9]. Martens, A. and H. Koziolek, Automatic, ”Model-Based Software Performance Improvement for Com ponent-Based Software Design”, Electronic Notes in Theoretical Computer Science, 2009. 253(1): p.77-93.

[10]. M. Wegmuller, J.P.von der Weid, P. Oberson, and N.Gisin, “High Resolution Fiber Distributed Measurements with Coherent OFDR”, in Proc. ECOC’00, 2000, paper 11.3.4, pp. 109.

[11]. Dr. Velur Rajappa, Arun Biradar, Satanik Panda “Effi cient Software Test Case generation Using Genetic al-gorithm based Graph theory”, International Conference on Emerging Trends in Engineering and Technology, pp. 298--303, IEEE (2008).

[12]. Jose Carlos, Mario, Alberto, Francisco “A Strategy for Evaluating Feasible and Unfeasible Test Cases for the Evolutionary Testing of Object-Oriented Software” ACM (2008).

[13]. Robert M .Patton, Annie S. Wu, and Gwendolyn H .Walton “A Genetic Algorithm Approach to Focused Software Usage Testing”, Annals Of software engineer ing, www.cs.ucf.edu/~ecl/papers/03.rmpatton.pdf.

[14]. Gianluigi Folino, Agostino Forestiero and Giandomenico Spezzano “An Adaptive Flocking Algorithm for Performing Approximate Clustering” Elsiever science May2009

[15]. Xiaohui Cui, Jinzhu Gao, Thomas E. Potok “A flocking Based Algorithm For Document Clustering Analysis”, Journal of systems and architectures, Elsiever 2006.