Organization Detection Using Emergent Computing

Cyrille Bertelle Antoine Dutot Frédéric Guinand Damien Olivier

LITIS – Université du Havre,
25 Rue Philippe Lebon, BP 540
76058 Le Havre Cedex - FRANCE
Email : {Cyrille.Bertelle ... Damien.Olivier}@univ-lehavre.fr

Abstract: Organization is a central concept in systems. In this paper an ant algorithm for detecting organizations is presented. In a discrete-time context, at each time-step, an organization corresponds to a set of closely interacting entities in a system. This system is mapped to a graph where nodes represent entities and edges represent interrelations. Several colonies of ants compete, and inside each colony, ants collaborate in order to colonize the graph. Detected organizations emerge from the global behavior of the ants. The proposed approach is compared to other methods on a graph where the organizations are already known. It is then tested on two real world graphs studied in the related literature.

Keywords: Organization, community, dynamic graph, dynamic network, ant algorithm.

1. Introduction

System: "Organized whole, made of interdependent elements that can only be defined in relation to each other, according to their place in this whole" *F. de Saussure* [10].

Ferdinand de Saussure proposes an interesting definition of the term "system". He highlights the concept of organization, connecting it with the whole and the interdependency. In other words, interrelations between elements, events or individuals, as soon as they are stable or regular, become organization generators, creators, attractive and eventually stable on the long term. The organization then connects elements, events or individuals by interrelations so that they become components of a whole. It assumes the solidarity and robustness of these links, and ensures that the system will eventually be long lasting despite random perturbations. The organization, therefore: transforms, produces, ties and maintains. Edgard Morin in "La Méthode" [25] proposes indeed the neologism "organisaction" to highlight the active feature. Furthermore, this organization can be produced by the system itself, and so it can exhibit auto-organizational traits, that is the ability to "auto-create" itself by producing its own *organisaction* principles in a continuous way. Matura and Varela go deeper with the term "autopoiesis", that is the property of living systems to auto-produce themselves permanently, to create continuously and without interruption their own living conditions. Results of the organization and of the functioning of the autopoeitic living are the ones that produce its organization and operation. Autopoiesis, or continuous reorganization, is a category that can apply to the whole biologic order, and by extension to the human social order.

In the domain of Artificial Intelligence, Swarm Intelligence refers to the collective intelligence of insect societies as described by ethologists. It consits in the building of simple reactive artificial individuals societies. Such individuals are able to collectively produce a complex response despite the fact they only have a local vision and have a simple behavior. The process is intrinsically decentralized. Problem resolution is obtained from the interactions and the dynamics of the system: intelligence appears collectively. The global result of the system is therefore emergent, made of a succession of "reflex"-like behaviors. Such systems are characterized by their adaptivity and their robustness. Indeed, due to the decentralized control, each agent, according to its own perceptions, reacts to changes in its environment and is able to continuously adapt to variations of it. Furthermore, the agent count, their interchangeability and the absence of any centralized control make such a system fault and failure tolerant. A system like this is able to change its behavior during execution to adapt to the environment evolution, and so take into account its dynamics.

Organization is therefore a central concept in systems. They are represented as eventually evolving graphs (dynamic graphs [7]) whose vertices are the elements and edges represent interactions.

In this paper a new detection method based on an emergent computation mechanism [17] implemented by an ant algorithm is presented. Using ant algorithm for emergent computing has been already successfully applied. In [24] such an approach is used for addressing the problem of Word Sense Disambiguation (WSD) in the domain of computational linguistics. The corresponding application graph is static. Vertices correspond to word meanings and the initial topology is given by the morphosyntactic tree of the analyzed sentence. During the resolution process, distinct colonies compete for imposing their meanings. They strike against colonies associated to opposite senses and collaborate with colonies associated to close meanings.

The same algorithm has been applied for addressing the dynamic load balancing problem [8]. The target application is a simulation of an ecological Individual-Based Model (IBM). The vertices of the considered dynamic graph are biological entities (different species) as well as inorganic particles, and

the edges represent the interactions between these entities.

The underlying organizations are groups of biological entities. Indeed, during the execution, entities self-organize themselves in groups according to different situations. As conditions change during the simulation, groups may appear or disappear in an unpredictable way. The idea was to use these groups as the basis for performing efficiently load balancing. The same method was also applied for building clusters of proteins from an homology graph. In such a static graph, vertices correspond to proteins and edges to closeness with respect to homology [6]. Based on some recent works [33, 31], a method to interface the algorithm with other simulations taking into account both inorganic particles organization and biological entities organizations in order to detect specialized trophic chains is currently studied.

The rest of this paper is organized as follows: the next section presents other comparable methods for organization or community detection. Section 3. describes in detail the approach, the implemented method and proposes a model for it. Results that can be expected by the method are presented in section 4., and some perspectives are drawn in the conclusion.

2. State of the Art

Finding organizations in graphs has been studied recently [2, 26], but two domains have already worked in this direction:

- Graph partitioning;
- Hierarchical clustering.

More recently, Girvan and Newman [20] suggested an elegant global algorithm based on hierarchical clustering that extended the concept of vertex betweenness centrality of Freeman [18] also to edges. Newman also suggested a stochastic approaches, like the random walk [29]. For a general review an discusion on the existing approaches see [9].

2.1 Graph Partitioning

Graph partitioning is a problem which appears in many different applications as VLSI design, data-mining, finite element and parallel computing, but the problem of finding organization can be considered.

Given a graph $G = (V, E)$, where V is the set of vertex and E the set of edges that determines the connectivity between the nodes. Both vertex and edges can be weighted, where $|v|$ is the weight of a vertex v, and $|e|$ is the weight of edge e even if in most of the literature, they are given unit weights. The graph partitioning problem consists on dividing G into k disjoint partitions. The goal is to minimize the number of cuts in the edges of the partition, and on the other hand reduce the imbalance of the weight of the subdomains. The weight of a subdomain is the sum of the weights of the vertex allocated in it.

As this is an NP-complete problem [19], for graphs with a large order it is not possible to guarantee the achievement of the best solution in a reasonable computation time: when the size of the graph increases, the execution time of an algorithm capable of solving the problem can be assumed to grow exponentially. Therefore the problem is practically unsolvable for most graphs and heuristic and probabilistic methods are considered to obtain solutions.

The methods presented below (2.1.1 and 2.1.2) use recursive bisection. First the graph is divided in two partitions, then each cluster is divided in two, and so on. This method needs $log_2(k)$ steps to divide the graph in k partitions.

2.1.1 Spectral Methods

A spectral method (i.e., methods using the eigenvalues and eigenvectors of a matrix representation of a graph) is based on the analysis of the adjacency matrix $A = [a_{ij}]$ [16, 11]. It consists to analyze properties of A. In particular, it is necessary to calculate the Laplacian matrix $L = D - A$ and the normal matrix $N = D^{-1}A$ where D is the diagonal degree matrix of vertices, if k_{ij} is an element of the matrix D,

$$k_{ii} = \sum_{k=1}^{card(V)} a_{ik} \text{ and } a_{ik} = 0 \text{ when } i \neq k.$$

The smallest eigenvalue of L is 0, the second eigenvalue of the Laplacian matrix of a graph, $\lambda_2(L(G))$, is the algebraic connectivity of graph G. Thus the vector v_2 corresponding to $\lambda_2(L(G))$ is the eigenvector of algebraic connectivity, and has special properties. It is possible to divide the graph into two cluster with the minimal amount of connectivity between them with the following scheme: If $v_2(n) < 0$, place n in G_1. If not, place n in G_2.

2.1.2 Iterative Improvement

The Kernighan/Lin Algorithm [21] addresses the problem by providing a method to improve upon an initial allocation of nodes between two subgraphs. Thus the algorithm starts with balanced partition and exchanges vertices in each subgraph to improve cutsize. Till the cutsize is improved, the vertex pairs (one in each subgraph) which give the largest decrease in cutsize are exchanged. These vertices are locked. If no improvement is possible and some vertices are still unlocked, the vertices which give the smallest increase are exchanged. The algorithm by Fiduccia and Mattheyses [15] improves this method, providing an extremely efficient linear-time technique.

2.2 Hierarchical Clustering

These methods are generally used to analyze data [14]. These are statistical methods for finding relatively homogeneous clusters of cases based on measured characteristics. The method starts with each case in a separate cluster and then combines the clusters sequentially, reducing the number of clusters at each step until only one cluster is left. When there are N cases, this involves $N - 1$ clustering steps, or fusions. This hierarchical clustering process can be represented as a tree, or dendrogram and are generally based on an agglomeration algorithm.

This kind of methods can be used on graph. The main goal is to group the vertices in subsets which represent the organizations. A similarity measure d_{ij} based on the graph structure is introduced between each pair of vertices. The algorithm begins with one organizations for one vertex, then in the following steps, the distance between each organizations is computed and the closest is merged.

2.3 Stochastic Approaches and Collective Explorations

Stochastic approach are generally based on random walk in graphs [29] and its derivatives. A random walk is a simple stochastic process. It is a formalization of the intuitive idea of taking successive steps, each in a random direction. Thus if a simple random walk in an undirected graph G is considered, at each time step, if the walk is located at vertex u, it moves forward to a vertex v chosen uniformly at random from the neighbors of u. To detect organizations, the main idea lays on the fact that small random walks tend to be trapped into them [30, 32]. A dissimilarity index [36] between two nearest-neighboring vertices of the graph is measured by a random walk. It is used to determine if nearest-neighboring vertices must be in the same organization. It integrates both the local and the global structural information of the given graph and bias can be introduced.

Multi-agents system can be used to detect organizations, Young *et al*. [34] proposed the following method : agents perform slightly biased walks on a graph and retain a list of the vertices they traverse. Based on this information, the agent then follow a simple voting routine whereby nodes are merged into organizations and excluded from others. After this process, a simple clean up scheme can be employed to deliver the desired number of organizations.

3. The Model

The goal is to find organizations inside a graph $G = (V, E)$ representing entities V, and their interactions E. If interactions have various degrees of importance, edges can be weighted, this weight being noted $|e|$ for the edge e. Organizations are informally defined in the graph G as: *a set of entities more closely connected with each other than with the other parts of the graph.* Note that this definition is very close to the various definitions of a community found in the literature. More closely connected means here that there can be more edges, but also that these edges may have stronger weights.

3.1 The Method

Several colonies of ants that travel inside the graph to detect organizations are used. As said above, ants are simple entities that collectively exhibit intelligent behavior. In our method collaboration between ants of a same colony allow to detect organizations, whereas competition between distinct colonies allows to separate organizations.

The algorithm used can be referred to as an *ant algorithm*, however it departs from the usual ACO (Ant Colony Optimization) as described in [12] since it never globally evaluates solutions, that is, it does not use an objective function. In our algorithm, ants auto-organize themselves and produce solutions that can be observed in the graph, but these solutions are never retrofitted in the algorithm.

The ant algorithm use several colonies of ants, each of a distinct color. Ants travel inside the graph an lay down pheromones, information that can be detected by other ants. Pheromones are also colored. Ants tend to be repulsed by pheromones of other colors. Furthermore, in the case of a weighted graph, ants tend to favor edges with important weights.

The algorithm principle is to color organizations using pheromones. Each colony will collaborate to colonize zones, whereas colonies compete to maintain their own colored zone (see figure 1). Solutions will therefore emerge and be maintained by the ant behavior. The solutions will be the color of each vertex in the graph. Indeed, colored pheromones are deposited by ants on edges. The color of a vertex is obtained from the color having the largest proportion of pheromones on all incident edges.

3.2 The Ant Algorithm

Algorithm 1: Ant behavior

```
n: current node
t: current time
A: fear of hostile environment threshold
T: resting time
Δt: time counter
if degree(n)=0 then
    Jump randomly on another node
else
    w ← Sum of all weights on each incident edge to n
    τ ← Sum of all pheromones of all colors on each
    incident edge to n
    τ_c ← Sum of pheromones of the ant color on each
    incident edge to n
    a ← τ_c / τ
    if Δt < T then
        Choose an edge to cross in a weighted random
        fashion, using edges weight (if available)
        Lay down a small amount of pheromone of the ant
        color on this edge
        n ← vertex at the other end of the chosen edge
        Δt ← Δt + 1
    else
        if a < A then
            Jump randomly on another node
            Δt ← 0
        else
            Choose an edge to cross in a weighted random
            fashion, using edges weight (if available)
            Lay down a small amount of pheromone of
            the ant color on this edge
            n ← vertex at the other end of the chosen edge
```

A colored dynamic graph $G(t) = (E(t), V(t), C(t))$ is defined such that:

- $V(t)$ is the set of vertices at time t. Each vertex v is characterized by:
  - a color $c \in C(t)$,
- $E(t)$ is the set of edges at time t. Each edge e is characterized by:
  - a weight $|e| \in \mathbb{N}^+$ that corresponds to interaction importance between the elements at each end of edge e.
  - a quantity of pheromones of each color.
- $C(t)$ is a set of colors representing the ant colonies at time t.

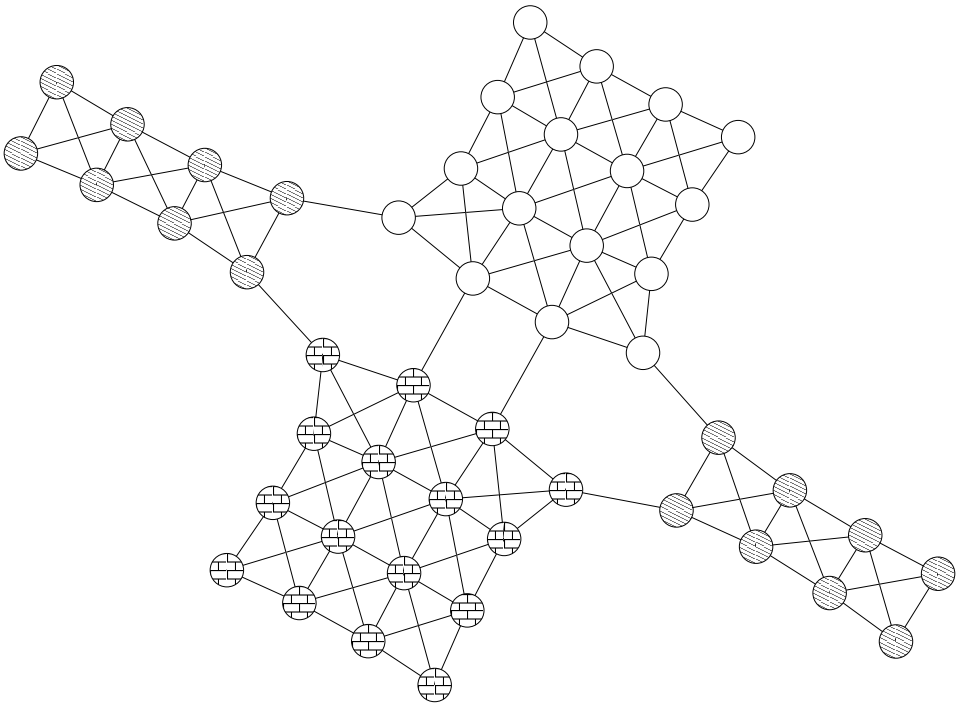


Fig. 1. Example of a colored dynamic graph.

The colored ants behavior is defined by algorithm 1. This algorithm is to be repeated iteratively for each ant. At each iteration the algorithm decides the node the ant will visit next. Two parameters are used T and A. A is a parameter that control competition among ants. When the proportion of pheromone having the same color as the ant compared to other colors is under A, the ant is in an hostile environment and flee to another zone of the graph chosen randomly. T is a stabilizing parameter avoiding that an ant flee all the time from node to node when they always fall in hostile environment. It allows them to better colonize zones. Usually, $T = 2$ and A is set to 1 on the number of colonies.

Given this algorithm each colony competes to create colonized areas. As shown on figure 1, the same colony can occupy two or more distinct zones, and ideally it would be possible to use a small number of colonies to discover all the organizations in the graph. However it is also possible that two organizations side by side be colonized by the same colony. To avoid this issue, the algorithm starts with only two colonies (or more if it can be estimated that the graph necessarily contains more organizations) and introduces new colonies, keeping the ant population constant, until it is no more possible to add more.

For each vertex of the graph a main color is computed and its proportion compared to other colors is determined (see figure 2). A vertex color is considered stable if it is larger than a given threshold ω. It is then possible to compute this proportion for all vertices of the graph and determine a global stability. This is this number that is used to know if it is possible to add more colonies or not. If the addition of another colony gives a stability that cannot reach ω, it is considered no more possible to add a colony. Usually, ω is set to 80%.

4. Results

In this section, a comparison of our algorithm results on a well known graph and then an exploration of graphs extracted from book databases is presented.

The Zachary Karate Club [35] is a real world data set already studied for example in [28, 29, 27, 3] that can be defined as a graph of 33 vertices. In these articles the authors propose several algorithms whose results are shown on figure 3(a). As it can be seen on figure 3(b), our algorithm matches exactly these results. In this representation, large circles drawn above edges represent the dominant pheromone color and the total pheromone intensity on the edge.

Browsing the Amazon.com book seller databases, it is possible, knowing the initial ISBN of a book to find all other books bought by customers at the same time. In the graphs shown on figure 4 and 5, each vertex represents a book, and each edge represents a link "customers having bought this book also bought". Books suggested this way are almost always on similar topics. The graph obtained therefore contains areas on wide topics like "the web", "programming", "architecture", or "systems".

The figure 4 is extracted starting at one edition of "The art of computer programming, volume 1" [22]. The algorithm found 9 distinct partitions of different sizes. Two movies of larger graphs can also be downloaded in [4, 5].

Our method was applied to a protein homology graph. In such a graph each vertex is a protein, and an edge exists between two proteins if they are considered close if they are functionally related. The graph has been taken from [1]. You can see on the movie [6] an application of our method to such a graph, exhibiting cluster of similar proteins.

5. Discussion and conclusion

An ant algorithm using emergent computing to detect organizations in graphs representing numerous elements in interaction was presented in this paper. The ant algorithm use both

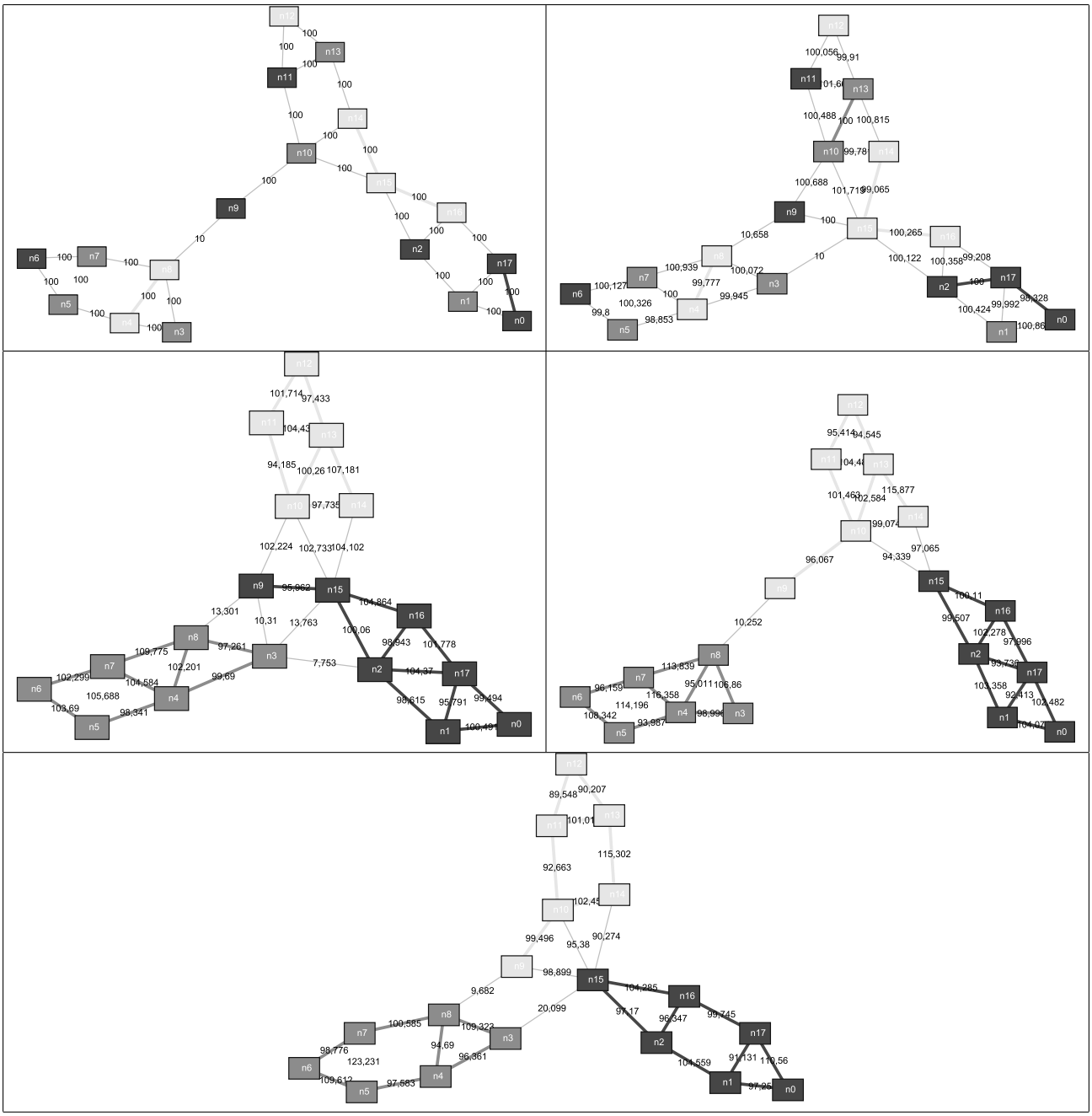


Fig. 2. Example of a dynamic colored graph at five stages of its evolution with organization detection.

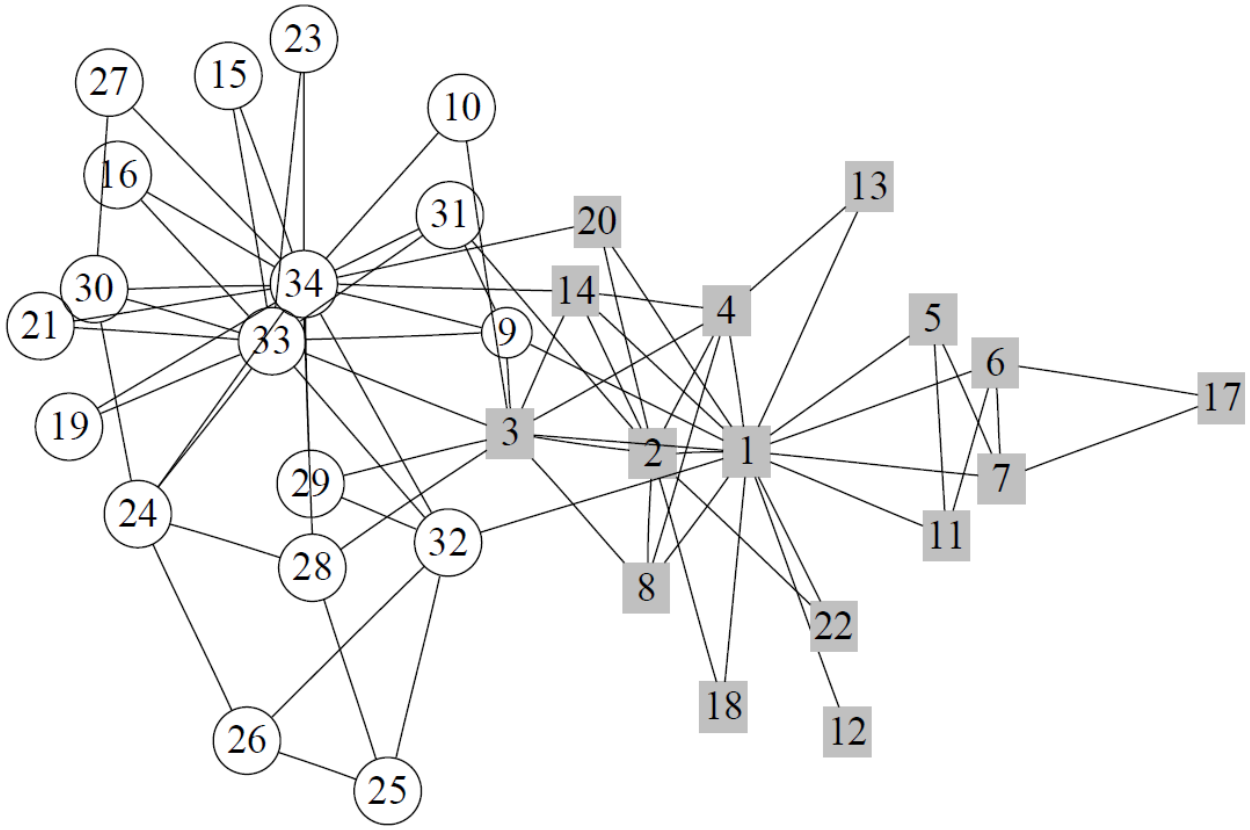
collaboration and competition mechanisms, and does not use any objective function. Solutions emerge from the ant behavior as colored areas in the graph.

Organizations are long lasting compared to their constituents. Identically the solutions proposed by the algorithm are maintained by the continuous exploration operated by ants. This implies that any change in the graph and therefore in the organization structure can be considered immediately, changing continuously the solution accordingly.

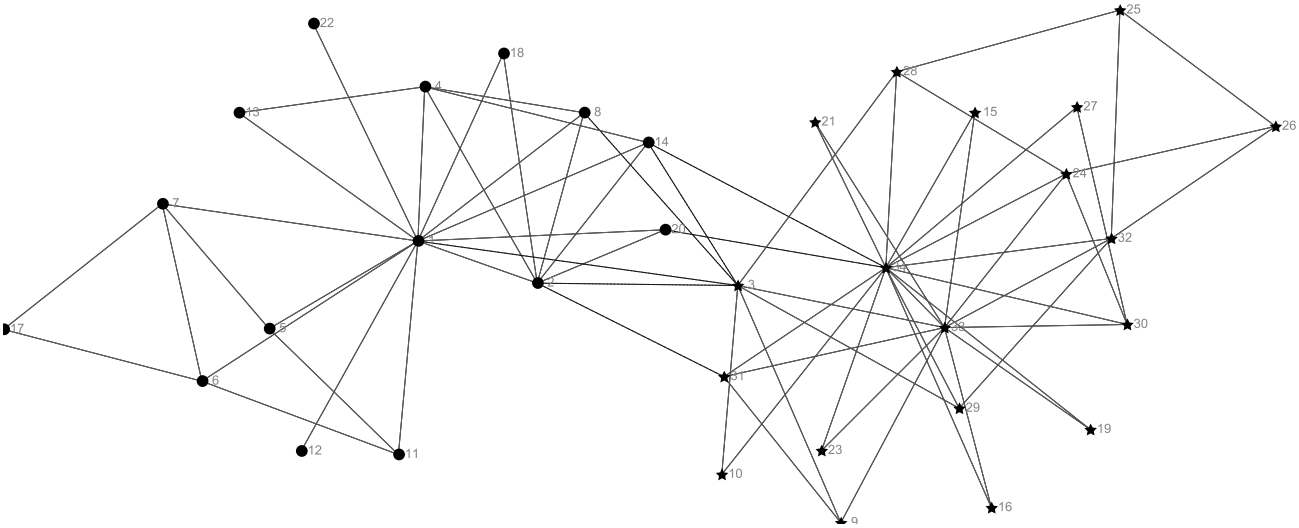
As vertices colors are given by the colored pheromone importance on all incident edges, it is possible to take into account several colors per edge (most prominent, second, third, *etc*.). This allows to detect areas where organizations overlap. Indeed organizations are not all completely distinct and such a feature could allow the detection of two closely related organizations where another method would only detect one.

The algorithm still presents some limitations. The major one is the number of colonies that must be fixed at the beginning and be less or equal to the number of organizations. Actually, to solve this problem the algorithm starts with a small number of colonies, and then introduces colonies one by one, while ω remains stable.

A method allowing to accelerate the solution emergent by adding weights to unweighted graphs using a graph layout is actually investigated. Graph layouts are geometrical descriptions of graphs that try to give a planar or tri-dimensional representation of the graph with the less edge intersection and node overlapping possible. Figures presented in this article use such a graph layout based on repulsive and attractive forces [13, 23]. As such force based layouts tend to push organizations (fol-



(a) Found by M. Newman in [29]



(b) Found by our algorithm

Fig. 3. Organizations in the Zachary Karate Club.

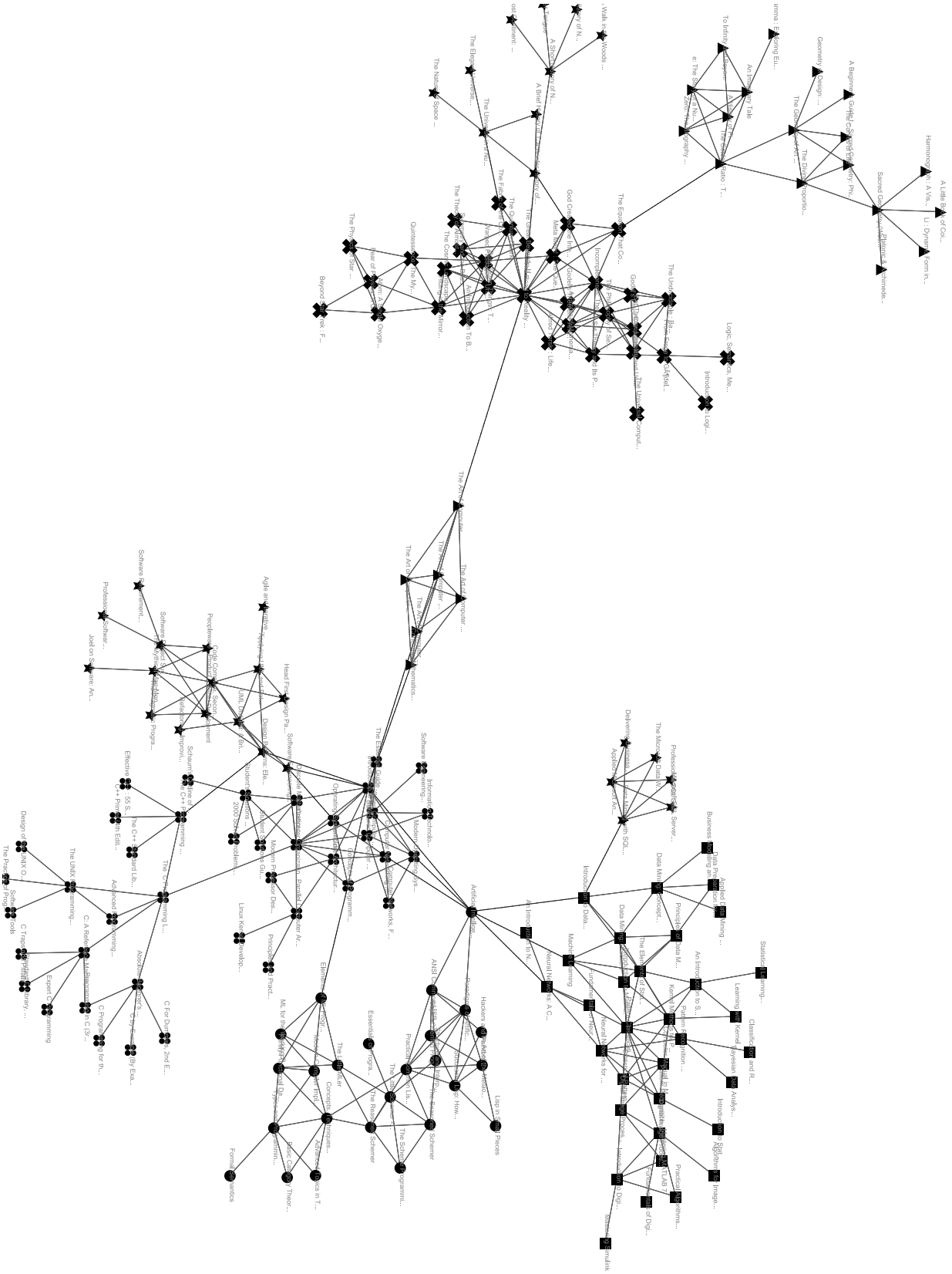


Fig. 4. Exploration of the Amazon.com databases starting from “The Art of Computer Programming” at 12 recursive levels.

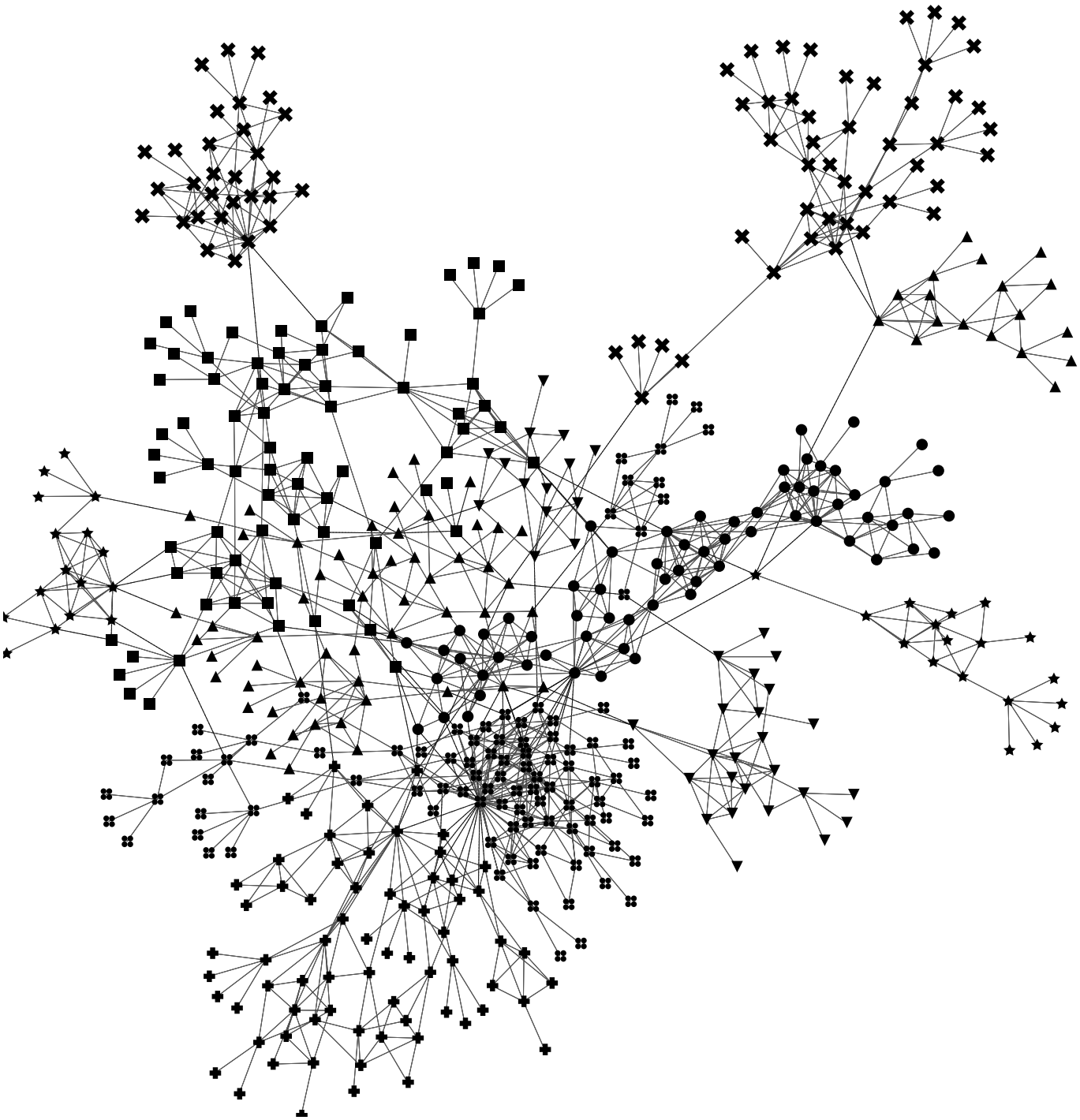


Fig. 5. Exploration of the Amazon.com databases starting from “The Ruby Way” at 20 recursive levels.

lowing our definition, set of nodes more closely tied together than with others) one from another, tightly connected nodes are closer and loosely connected nodes are farther (see for example figure 4). It is thus possible to use edge length to add weights used by ants to discover organizations.

Although the algorithm presented here is only dedicated to organization detection, variations of it has been used to dynamically and adaptively distribute simulations made of large number of interacting entities [8].

References

[1] A T Adai, S V Date, S Wieland, and E M Marcotte, Lgl: creating a map of protein function with an algorithm for visualizing very large biological networks, Molecular Biology 2004, Vol. 340, No 1 , pp. 179–190.

[2] R Albert and A L Barabasi, Statistical mechanics of complex networks, Reviews of Modern Physics 2002, Vol. 74, pp. 47–97.

[3] J-P Bagrow and E M Bollt, Local method for detecting communities, Physical Review E October 2005, Vol. 72 No 4 .

[4] C Bertelle, A Dutot, F Guinand, and D Olivier, Graph extracted from the amazon.com databases, colored with 5 colors, `http://www-lih.univ-lehavre.fr/~dutot/videos/videoGraphAmazon15_5colors.avi`.

[5] C Bertelle, A Dutot, F Guinand, and D Olivier, Graph extracted from the amazon.com databases, colored with 8 colors, `http://www-lih.univ-lehavre.fr/~dutot/videos/videoGraphAmazon30.avi`.

[6] C Bertelle, A Dutot, F Guinand, and D Olivier, Protein homology graph colored with 6 colors, `http://www-lih.univ-lehavre.fr/~dutot/videos/ProteinHomology1.avi`.

[7] C Bertelle, A Dutot, F Guinand, and D Olivier, Dynamic placement using ants for objects based simulations, In proceedings of DOA 2003. Catania (Sicily) Vol. 2888 of LNCS 2003 pp 1263–1274.

[8] A Cardon, A Dutot, F Guinand, and D Olivier, Competing Ants for Organization Detection: application to Dynamic Distribution Springer Verlag 2006, chapter 2, pp 27–54.

[9] L Danon, J Duch, A Diaz-Guilera, and A Arenas, Comparing community structure identification, Stat. Mech. 2005, Vol. P09008.

[10] F de Saussure, Cours de linguistique générale, Payot Geneva 1931.

[11] W E Donath and A J Hoffman, Lower bounds for partitioning of graphs, IBM J. Res. Develop. 1973, Vol. 17, pp. 420–425.

[12] M Dorigo and T Stützle, Ant Colony Optimization, MIT Press 2004.

[13] P Eades, A heuristic for graph drawing, In Congressus Numerantium Vol. 42 1984 pp 149–160.

[14] B S Everitt, S Landau, and M Leese, Cluster Analysis, H Arnold 4^{th} edition 2001.

[15] C Fiduccia and R Mattheyses, A linear time heuristic for improving network partitions, In Proceedings of 19^{th} Design Automation Conerence 1982.

[16] M Fiedler, Algebraic connectivity of graphs, Czechoslovak Mathematical Journal 1973, Vol. 23, No 98 , pp. 298–305.

[17] S Forrest, Emergent Computation chapter Emergent computation: self-organizing, collective, and cooperative phenomena innatural and artificial computing networks, MIT Press Cambridge, MA 1991, pp 1–11.

[18] L C Freeman, A set of measures of centrality based upon betweenness, Sociometry 1977, Vol. 40, pp. 35–41.

[19] M R Garey and D S Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, Number ISBN 0-7167-1044-7. W H Freeman and Company New York-San Francisco 1979.

[20] M Girvan and M E J Newman, Community structure in social and biological networks, Proceedings of the National Academy of Sciences of The United States of America 2002, Vol. 99, pp. 7821–7826.

[21] B Kernighan and S Lin, An efficient heuristic procedure for partitioning of electrical circuits, Bell Systems Technical Journal 1970, Vol. 1, pp. 291–307.

[22] D E Knuth, The Art of Computer Programming: volume 1, Fundamental Algorithms, Reading, Massachusetts: Addison-Wesley 1997.

[23] J Kruskal and J Seery, Designing network diagrams, In First General Conf. on Social Graphics 1980 pp 22–50.

[24] M Lafourcade and F Guinand, Algorithme de fourmis pour le traitement de la langue naturelle. To appear in International Journal of Computational Intelligence Research. In Proceedings of ROADEF 2005, pp 239–240,

[25] E Morin, La méthode - 1 - La nature de la nature, Seuil, Nouvelles Éditions 1981.

[26] M E J Newman, The structure and function of complex networks, SIAM Review 2003, Vol. 45, pp. 167–256.

[27] M E J Newman, Detecting community structure in networks, Eur. Phys. J. B. 2004, Vol. 38, pp. 321–330.

[28] M E J Newman, Fast algorithm for detecting community structure in networks, Physical Review E 2004, Vol. 69, pp. 066133.

[29] M E J Newman and M Girvan, Finding and evaluating community structure in networks, Physical Review E 2004, Vol. 69.

[30] P Pons and M Latapy, Computing communities in large networks using random walks, In Proceedings of the 20th International Symposium on Computer and Information Sciences (ISCIS'05) Vol. 3733 of Lecture Notes in Computer Science Istanbul, Turkey October 2005. Springer pp 284–293.

[31] G Prevost, Modélisation multi-niveaux d'écosystème aquatique par des approches mixtes., PhD thesis Université du Havre Décembre 2005.

[32] B Tadic, Exploring complex graphs by random walks, AIP Conference Proceedings 2003, Vol. 661, pp. 24–27.

[33] P Tranouez, Contribution à la modélisation et à la prise en compte informatique de niveaux de description multiples, PhD thesis Université du Havre 2005.

[34] M Young, J Sager, G Csardi, and P Haga, An agent-based algorithm for detecting community structure in networks, in arXiv at http://www.citebase.org/cgi-bin/citations?id=oai:arXiv.org:cond-mat/0408263 2004.

[35] W W Zachary, An information flow model for conflict and fission of small groups, Journal of Anthropological Reasearch 1977, Vol. 33, No 4 , pp. 452–473.

[36] H Zhou, Distance, dissimilarity index, and network community structure, Physical Review E 2003, Vol. 67, pp. 10.

Authors Bios

Cyrille Bertelle is professor in Computer Sciences in Le Havre University. His activities concern complex systems modelling: their conceptual formalization, their distributed implementation and their applications in various domains: aquatic ecosystems, game theory, logistic and cognitive sciences. He focuses his studies on emerging computing using collective intelligence methods. He manages a reserch master on complex systems modelling based on both mathematical and computer sciences approaches.

Antoine Dutot is an assistant professor in the University of Le Havre. He is a member of the Computer Science, Information Processing, and Systems Lab (LITIS). He is actually working on organization detection methods using distributed artificial intelligence, as well as dynamic load balancing.

Frédéric Guinand received his MSc degree in computer science in 1991, and his PhD degree in computer science in 1995 both from the Institut National Polytechnique de Grenoble (INPG - France). He is currently a professor in the Department of Computer Science and in the Computer Science Laboratory (LITIS) of Le Havre University. He previously worked as a research scientist in the LITH Laboratory of the Swiss Federal Institute of Technology (EPFL Lausanne - Switzerland). His main research interests are in the area of mobile and distributed Computing, bioinformatics and complex systems.

Damien Olivier is an assistant professor in the University of Le Havre. He is a member of the Computer Science, Information Processing, and Systems Lab (LITIS). His major fields of interest are distributed artificial intelligence and bioinformatic. Actually he works on swarm intelligence and emergent computing for dynamic load balancing and also on artificial immune systems. He has regularly published papers in academic journals.