Centroids : a decentralized approach

Antoine Dutot, Damien Olivier, Guilhelm Savin

Proceedings of EPNACS 2011 within ECCS'11
Emergent Properties in Natural and Artificial Complex Systems
Vienna, Austria - September 15, 2011

CENTROIDS: A DECENTRALIZED APPROACH

Antoine Dutot*, Damien Olivier*and Guilhelm Savin*†

Abstract. The centroid of a graph is a structure composed of nodes closest from all others. This suggests the presence of center of mass average of all edges, weighted by the local density or specific weight. To compute this centroid in a *classic* way needs a global view of the graph environment. In this paper, we propose an algorithm using ant colony is proposed to compute an approximate solution of the centroid using a local view of the graph. This allows to study the centroids of complex networks such as protein-protein interaction networks and also those generated by social interactions or Internet, for example.

Keywords. complex system, self-organization, centroid, ant algorithm

1 Introduction

Complex systems are constituted by interactive numerous entities. These entities and their interactions define an interaction network which evolves in the time. We can use a dynamic graph [1, 2] to model this network and to describe its evolution. Through this graph, we can detect some properties emerging from interactions between entities.

A lot of real networks exhibit non-trivial properties. These complex networks present generally patterns of connection between their entities such as particular degree distribution, high clustering coefficient, hierarchical structure or communities. These features emerge from interactions and evolve in the time, notably, the self-organization defining a group of entities whose members maintain privileged interactions. The challenge is to detect them [3] and to define metrics. We assume that the notion of centers and centroids can help us in this approach.

It is impossible to make a reification of the whole graph because of numerous elements which consitute itself, however that is possible to distributed this graph over a set of machines. In such cases, each machine only has a *local view* of the graph. This local view raises problems when trying to use algorithms needing a *global view* of the graph.

Let us add another problem, the dynamic of the environment. In fact nodes and edges can appear or disappear, and attributes of elements can evolve, that makes obsolete the previously computed solution. Most algorithms are not able to use a previous solution to update it and they have to compute a new solution from scratch.

Ant System paradigm [4] allows to define metaheuristic where computation is distributed over a set of agents called *ants*. These ants explore the graph and make local action on crossed nodes and edges. The well-known *Traveling Salesman Problem* is given as practical example of problem that can be solved with an ant system.

We are interested here by finding a structure called *centroid* of a graph using an ant system. This paper is divided into two parts: section 2 is a state of art about centroid in literature while section 3 describes the ant heuristic and shows results on several graph categories.

2 Centers and Centroids

The term *centroid* is used in several scientific fields such as geometry or physics. Centroid is also used in human sciences to summarize, for example, a set of spatial points. The centroid of an object is its barycenter or its geometric center. In the case of a weighted object with uniform density, the centroid of this object is its *center of mass*. Centroid evokes the idea of a balancing point of the object. The following describes the concept of centroid applied to graphs. We not provide here a full referencing of properties or theorems about centroids but only an approach of what is a centroid in a graph. The concept of *center* being associated to the one of centroid, it is described to.

Centers and *centroids* are in literature the object of many works. From C. Jordan giving a proof of their existence in every tree [5] to P. J. Slater generalizing their definitions [6], through parts of graph theory books [7, 8].

In the following, some notations are used and have to be defined first. G designates a connected graph with $V(G)$ its node set and $E(G)$ its edge set. T designates a tree. $op(e, u)$, with $e = (u, v)$ an edge, designates the vertex v, the opposite of the vertex u according to e. $d(a, b)$ is the distance between two nodes a and b, *i.e* the number of edges in the shortest path between these two nodes. $e(v)$ is the *eccentricity* [8] of a node v and it is

*L.I.T.I.S., University of Le Havre, France,
authors are sorted alphabetically
†corresponding author: guilhelm.savin@litislab.fr

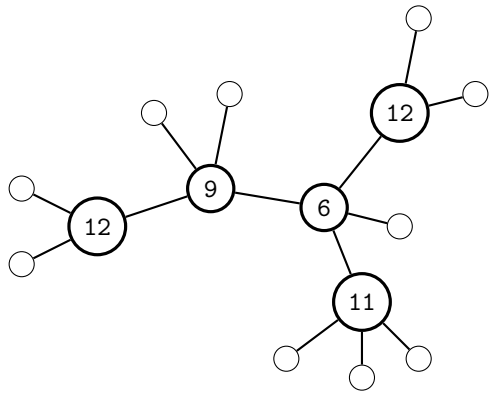

Figure 1: *Weights of non-leaf nodes in a tree as defined in [8].*

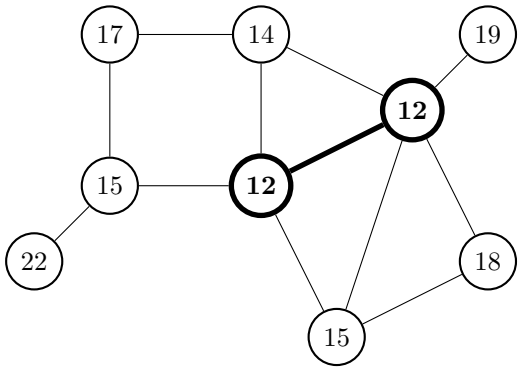

Figure 2: *Distances of nodes. Bold structure is the centroid of the graph.*

the maximal distance between v and any other node as defined in equation 1. The distance $d(v)$ of a node v is the sum of distance between v and the other nodes as defined in equation 2. The weight of a node v in a tree T is the maximum number of nodes in any branch of v. The figure 1 shows the weight of non-leaf nodes.

$$e(v) = \max_{x \in V(G)} (d(v,x)) \tag{1}$$

$$d(v) = \sum_{x \in V(G)} d(v,x) \tag{2}$$

The concept of *center* and *centroid* have been first defined for trees [5, 7, 8] and then extended to graphs.

A node c_0 of a graph G is considered as a *central point* of G [8] if its eccentricity $e(c_0)$ is minimal for G as defined in equation 3. The set of these central points define the *center* of G.

$$e(c_0) = \min_{v \in V(G)} (d(v)) \tag{3}$$

There are usually qualified centers. *Center of gravity* of a graph G is the set of nodes that minimize the function $m()$ defined in equation 4. A *mass center* of a tree T is a node with minimal weight.

$$m(v) = \frac{1}{|V(G)|} \sum_{x \in V(G)} d(v,x)^2 \tag{4}$$

The *centroid* of a graph G is a set of nodes called *centroid points* [8]. One defines the centroid as the subgraph induced by these centroid points [9], *i.e* the subgraph H of G with $V(H)$ is the set of centroid points and $E(H) = \{(u,v) | (u,v) \in E(G); u,v \in V(H)\}$. There are several functions in literature to characterize centroid points. In [8], the author defines centroid points of a tree T as the nodes with minimal weight, *i.e* the mass centers of the tree. One defines centroid points of a finite, connected, undirected graph G [6] using distance of a node (equation 2): centroid points are nodes with minimal distance. The figure 2 shows distance value of nodes. The subgraph induced by nodes with minimal distance is the centroid of G.

Slater defines the *k-centrum* of G [6] allowing to link definitions of center and centroid. "The sum of the k-largest vertex distances to" a node v is denoted $r_k(v)$ and is defined in equation 5. The set of nodes of a graph G that minimize is called the *k-centrum* of G and is denoted $C(G;k)$. When $k = 1$, $r_1(v)$ is the eccentricity of v so $C(G;1)$ is the center of G. When $k = n$ with $n = |V(G)|$, $r_n(v)$ is the distance of v so $C(G;n)$ is the centroid of G.

$$r_k(u) = \max_{S \subseteq V(G); |S|=k} \left\{ \sum_{s \in S} d(u,s) \right\} \tag{5}$$

3 Decentralized algorithm

3.1 Definition

We are interested here by describing an *heuristic* that aims to find centroid of a graph using only local informations. This is an *ant-based* algorithm [4] where an ants colony explores a graph, dropping informations on elements, nodes and edges. These dropped informations are a form of indirect communication, called *stigmergy*, which allows ants to collaborate to find a solution. In such algorithms, one defines a behavior for ants that we denoted here the *step* of ants. The algorithm consists in a loop in which the step method of each ant is triggered.

The goal of this approach is to provide an algorithm that does not need a global reification of the environment (the graph) and that can handle dynamic of the environment by updating the solution. This algorithm computes an approximate centroid.

We consider that each node has a mass, which is initially the same for all nodes and ants are able to take a part of node mass and to distribute the taken mass on other nodes. The final aim is to increase mass of centroid

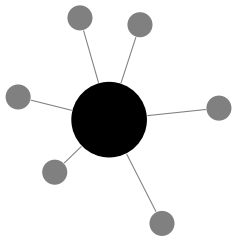

Figure 3: *The local view of the environment of an ant: the black node is the current position and gray nodes are its neighborhood.*

nodes and decrease mass of outlying nodes, the most massive nodes being parts of the centroid.

A ant a is composed of a few attributes:

- pos_a, the node position of the ant;
- $phn_{max}(a)$ and $mass_{max}(a)$ that are described in the following;
- $tmass_a$, the mass transported by a;
- a tabu list used to avoid to back track.

While ants are exploring the graph, they drop pheromones on edges they crossed. For an edge e, the measure $phe(e)$ is the load of pheromones of e and for a node n, $phn(n)$ is the sum of $phe(e_i)$ where e_i is an edge adjacent to n. These pheromones *evaporate* through time according to a factor ρ, so we have:

$$phe_{i+1}(e) = phe_i(e) \times \rho, \forall e \in E(G)$$

where i is the iteration of the algorithm. Each ant a remembers maximum pheromone load $phn_{max}(a)$ and maximum node mass $mass_{max}(a)$ that it has been found. Like pheromones, these two values evaporate through time to keep an up-to-date value reflecting the current state of the environment. The choice of the next edge to cross e_i adjacent to the ant position has a probability distribution $P(e_i)$ based on pheromone load and described in equation 6. The constant α helps to control the importance of pheromone load in choosing the next edge to cross. The function $\eta_a(e)$ defined in equation 7 is used to influence ants choice in the aim of leading them to massive nodes according to a parameter $\lambda \in [0;1]$: if λ is close to zero, then the probability of edges to be choosen is all the higher as the mass of the opposite node is high, else if λ is close to one, mass of node has no effect in the choice of the next edge.

$$P(e) = phe(e)^{\alpha} \times \eta_a(e) \tag{6}$$

$$\eta_a(e) = \lambda + (1 - \lambda) \times \frac{mass(op(e, pos_a))}{mass_{max}(a)} \tag{7}$$

The second part of the ant behavior is the ability to distribute node mass on other nodes. Each ant a has an attribute $tmass_a$ corresponding to the mass being transported. If $tmass_a$ is null, the ant takes a quantity q of the mass of its current position bounded between parameters $tmass_{min}$ and $tmass_{max}$, else the ant a computes a quantity $drop(a, n)$, defined in equation 8 to drop on its current position n.

$$drop(a, n) = tmass_a \times \frac{phn(n)}{phn_{max}} \tag{8}$$

The algorithm is a loop where evaporation of pheromones is made and where the behavior of each ant is triggered as described in algorithm 1. This behavior consists in choosing an edge to cross, cross it and then to transfer some mass. The algorithm describing this behavior is algorithm 2.

Algorithm 1: Global ant algorithm

```
Data:
    - G, the graph, E(G) is the edge set
    - A, set of ants
begin
    repeat
        forall edge e ∈ E(G) do
            phe(e) ← phe(e)×ρ
        forall ant a ∈ A do
            execute step of a
    until stop condition is reached
end
```

Algorithm 2: Step of an ant

```
Data:
    - a, the ant
    - t, mass transported by a
    - p, position of a
    - m, mass of p
begin
    forall edge i adjacent to p do
        compute P[i]
    randomly choose edge according to P
    p = ope(e, p) /* cross chosen edge */
    if t == 0 then
        alea ← random(tmass_min, tmass_max)
        q ← min(alea, m)
        t ← t + q
        m ← m- q
    else
        q = drop(a, p)
        t ← t- q
        m ← m + q
end
```

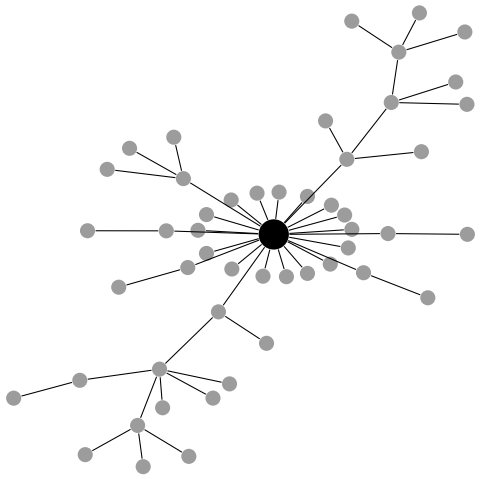

Figure 4: *In this tree of 50 nodes, ants have detected the optimal centroid (the biggest black node), i.e the centroid computed by the classic algorithm.*

3.2 Results

We compare here results of our ant algorithm with result of a *classic* algorithm consisting in computing all shortest paths of the graph using a Floyd-Warshall algorithm [10] and then to use the previous results to compute the distance of each node as defined previously in equation 2. The complexity of this algorithm is $O(n^3)$ with n is the amount of nodes. Values of parameters used in tests are available on table 5.

To compare results, one has to define a condition to stop the loop of the ant algorithm. For each node, the *stabilization* of the node is a measure described in equation 10 and computed from previous and current masses and bounded between 0 and 1 that indicates the evolution of the node mass : 0 means that the mass is strongly evolving, 1 means that the mass is stable. This allows the definition of the stabilization of the graph described in equation 11 which is the average value of node stabilization. The algorithm can stop when stabilization of the graph reaches a given threshold ϵ. For tests we used $\epsilon = 0.99$.

$$diff_t(n) = \frac{|mass_{t-1}(n) - mass_t(n)|}{mass_t(n)} \quad (9)$$

$$stab_t(n \in V(G)) = min(1, 1 - diff_t(n)) \quad (10)$$

$$stab_t(G) = \frac{\sum_{n \in V(G)} stab_t(n)}{|V(G)|} \quad (11)$$

Several kinds of graph have been considered for tests :

1. Trees built using a *preferential attachment* algorithm [11] as shown on figure 4;
2. Graphs built using a *Dorogotvsev Mendes* algorithm [12] that produces graphs with a power-low degree distribution ;
3. Two-dimensional grids ;
4. Two-dimensional grids with random holes.

Table 1: *Results for preferential attachment trees*

nodes	51	101	501	1001
edges	50	100	500	1000
$\overline{t_\Delta}$	-37.1%	-84.2%	-98.6%	-99.0%
$\sigma(t_\Delta)$	38.03%	5.07%	0.51%	0.00%
$\overline{s_\Delta}$	+0.5	+0.4	+0.5	+0.5
$\sigma(s_\Delta)$	1.13	0.77	0.77	0.87
$\overline{\Gamma}$	7.7%	6.7%	5.0%	4.0%
$\sigma(\Gamma)$	8.6%	7.3%	5.2%	4.1%

Table 2: *Results for dorogovtsev-mendes graphs*

nodes	53	103	503	1003
edges	103	203	1003	2003
$\overline{t_\Delta}$	-34.2%	-79.1%	-97.9%	-99.0%
$\sigma(t_\Delta)$	19.84%	5.28%	0.30%	0.00%
$\overline{s_\Delta}$	+1.5	+0.7	+0.6	+0.6
$\sigma(s_\Delta)$	2.67	1.13	0.83	0.68
$\overline{\Gamma}$	10.4%	6.2%	4.3%	3.7%
$\sigma(\Gamma)$	9.4%	5.8%	3.8%	3.4%

Comparison of results focuses on differences between computation time and between computed centroid. If $time_A$ is the computation time of the ant algorithm and $time_C$ the one of the classic algorithm, then

$$t_\Delta = \frac{time_A - time_C}{time_C}$$

express the gain or loss of time of the ant algorithm compared to the classic one. If C_A is the centroid computed by ants and C_C the one computed by the classic algorithm, then

$$s_\Delta = |C_A| - |C_C|$$

express the size difference between both centroids.

When a node v of C_A is not in C_C, we can define the shortest distance $\gamma(v)$ between v and any node of C_C normalized by the diameter of the graph. The measure Γ is the average value of $\gamma(v)$: if equals to 0 then both centroids are confused, else higher is the value, farther are the centroids.

3.3 Conclusion on results

Two points have to be considered in studying results. The first one is the computation time needed by the algorithm, the second one is the quality of the solution.

Table 3: *Results for grids*

nodes	36	121	256
edges	60	220	480
$\overline{t_\Delta}$	-15.9%	-89.3%	-96.7%
$\sigma(t_\Delta)$	31.99%	2.48%	0.78%
$\overline{s_\Delta}$	+2.9	+11.4	+13.3
$\sigma(s\Delta)$	3.51	6.83	9.69
$\overline{\Gamma}$	17.4%	17.8%	13.8%
$\sigma(\Gamma)$	4.7%	2.4%	1.8%

Table 4: *Results for incomplete grids*

nodes	22	93	214
edges	53	292	730
$\overline{t_\Delta}$	+197.8%	-77.8%	-94.4%
$\sigma(t_\Delta)$	122.77%	3.52%	0.88%
$\overline{s_\Delta}$	+4.1	+9.2	+13.7
$\sigma(s_\Delta)$	2.81	5.87	8.87
$\overline{\Gamma}$	25.3%	19.4%	15.1%
$\sigma(\Gamma)$	9.1%	5.2%	2.0%

Table 5: *Parameters value*

ϵ	0.99
λ	0.50
α	1.00
pheromon drop	0.03
pheromon evaporation	0.86

From results we can observe that the ant-algorithm is significantly faster than the classic algorithm. In addition, ant-algorithm is able to handle dynamic of graph by updating its current solution, whereas the classic algorithm needs to start from scratch. So, in these tests that use static graphs, benefits of the ant algorithm about the robustness of its solution within the ambit of a dynamic evolution of the graph.

Quality of the solution is harder to assess because it can be dependent on how is used this solution. We focus on how the ant-algorithm solution S_a matches the classic one S_c by observing size difference between these two solutions and the average of shortest distance from nodes in S_a to any node of S_c, denoted Γ. Unlike time difference, size difference and Γ are more heterogeneous. Best matches is for trees (table 1) followed by graphs generated with Dorogovtsev-Mendes algorithm (table 2): size difference is less than one and Γ is low. For grids (table 3) and incomplete grids (table 4), S_a is far from S_c : ants find more centroid points which are not close to centroid points of S_c.

4 Conclusion

We have proposed in this paper a new heuristic to compute an approximate centroid of a graph. This algorithm uses an ant system with ants that explore the graph dropping pheromone on crossed edges and moving the mass of nodes.

The algorithm has been applied to several static graph categories and results compared with the one of a greedy deterministic algorithm. This comparison shows that the ant algorithm is significantly faster than the classic one and the quality of its solution depends on the graph category: the centroid of ants algorithm is really close of the classic one for trees but for grid solution is far.

However, we have made these tests on static graphs whereas this new ant algorithm aims to be applied on dynamic graphs, providing a robust solution. More, the ant algorithm provides a distributed and decentralized computation of the solution allowing to use it on massive graphs.

4.1 Further work

A first future work is to compute centroid of dynamic graphs using this algorithm to test the robustness of the solution.

Then we have to study the impact of parameter values on the algorithm solution and to find a way the determine an optimal value for these parameter according to the graph.

References

[1] A. Dutot, F. Guinand, D. Olivier, and Y. Pigné, "Graphstream: A tool for bridging the gap between complex systems and dynamic graphs," in *EPNACS: Emergent Properties in Natural and Artificial Complex Systems*, 2007.

[2] Y. Pigné, "Modélisation et traitement décentralisé des graphes dynamiques : Application aux réseaux mobiles ad hoc," Ph.D. dissertation, Université du Havre, 12 2008. [Online]. Available: http://tel.archives-ouvertes.fr/tel-00371962/PDF/these.pdf

[3] C. Bertelle, A. Dutot, F. Guinand, and D. Olivier, "Organization detection using emergent computing," *International Transactions on Systems Science and Applications*, vol. 2, no. 1, pp. 61–70, 2006. [Online]. Available: http://litis.univ-lehavre.fr/~dutot/biblio/ITSSA2006.pdf

[4] M. Dorigo, V. Maniezzo, and A. Colorni, "The ant system: Optimization by a colony of cooperating agents," *IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS-PART B*, vol. 26, no. 1, pp. 29–41, 1996.

[5] C. Jordan, "Sur les assemblages des lignes," *J. Reine Angew. Math*, vol. 70, pp. 185–190, 1869.

[6] P. J. Slater, "Centers to centroids in graphs," *Journal of Graph Theory*, vol. 2, pp. 209–222, 1978.

[7] O. Ore, *Theory of graphs*. American Mathematical Society, Providence,, 1962.

[8] F. Harary, *Graph Theory*. Westview Press, Oct. 1969. [Online]. Available: http://www.amazon.com/exec/obidos/redirect?tag=citeulike07-20\&path=ASIN/0201410338

[9] M. Truszczynski, "Centers and centroids of unicyclic graphs," *Mathematica Slovaca*, vol. 35, pp. 223–228, 1985.

[10] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*. MIT Press, 2009, ch. The Floyd-Warshall algorithm, pp. 558–565.

[11] A.-L. Barabási and R. Albert, "Emergence of scaling in random networks," *Science*, vol. 286, no. 5439, pp. 509–512, 1999. [Online]. Available: http://www.sciencemag.org/content/286/5439/509.abstract

[12] S. N. Dorogovtsev and J. F. F. Mendes, "Evolution of networks," in *Adv. Phys*, 2002, pp. 1079–1187.