# ON THE DECENTRALIZED DYNAMIC GRAPH COLORING PROBLEM

Antoine Dutot, Frédéric Guinand, Damien Olivier and Yoann Pigné
LITIS - University of Le Havre
25 rue Ph. Leblon - BP 540
76058 Le Havre Cedex - France
Email: Surname.Name@univ-lehavre.fr

## INTRODUCTION

The central topic of this contribution is the conception of decentralized methods for dynamic graphs processing. The discussion is illustrated by the well-known graph coloring problem. In the wide domain of Complex Systems, graphs are omnipresent. Above all, this is due to the nature of these systems. Indeed, the emergent properties of complex systems arise from the dynamics of the interactions between entities belonging to the system and between these entities and the environment. Domains so different as, Biology, Physics, Social Sciences or Computer Science share these notions. If anchovy shoals, starling flocks, people crowds or mobile ad hoc networks are under study, the same universal model may be considered: dynamic graphs. Each entity, whatever its type, person, animal or machine, is autonomous and interact locally with other entities located in its neighborhood of communication. The communication may be visual, oral or of another nature like electromagnetic waves for instance. However, not all computer networks are comparable with other kinds of interaction networks. We think that DT-MANETs, Delay-Tolerant Mobile Ad-Hoc Networks are especially close to natural interaction networks. Such computer networks are made of mobile communicating devices that may be switched on or off at any moment and in a sudden way, and during its evolution this network may also be partitionned as it may be the case with people crowds in human societies or birds flocks. Whatever the considered domain, the main sources of dynamics are: the volatility of entities and the dynamic of the topology of the interaction network. The volatility of entities may have many differents reasons. A machine may be subject to a failure or may have been voluntarily switched off/on for energy saving for instance. Biological entities may appear and disappear according to life cycle (death/birth), or may be disconnected from their neighborhood for a while (sleep). Then, the obtention as a side-effect, by a decentralized method, of global-range structures in dynamic graphs, seems to raise from the same mechanisms aa emergent properties in natural complex systems. And we hoppe that a better understanding of such kind of mobile networks may lead to a better understanding basic mechanisms of emergence in natural complex systems.

In the context of dynamic graphs, the conception of decentralized algorithms for reaching a particular goal (broadcasting, routing, counting, stations coupling, search for particular paths, etc) prove to be of another nature that in a static setting. Indeed, in a dynamic framework, the algorithm has not only to build a structure but it also has to maintain this structure. So the algorithm itself has to behave in a dynamic way. This situation occur also in distributed systems, when some tasks like file transfert or dynamic scheduling have to be performed in a continuous way. Some of these tasks correspond to the classical graph coloring problem. Within this paper, a self-organized approach is described and its performances in term of graph coloring are compared with those obtained using classical static algorithms like Welsh and Powell algorithm. These algorithms are statically applied to the whole graph between to steps of dynamic events.

## GRAPH COLORING PROBLEM

The aim of the graph coloring problem consists in using as few colors as possible such that two adjacent vertices own a different color. A formulation of the $k$-colorability problem might be found in [GJ79]:

> **Problem Name :** *Graph k-coloring*.
>
> **Parameter Description :** a colored graph $G = (V, E, C)$, such that $V$ denotes the set of vertices of the graph, $E$ the set of edges belonging to the graph and $C$ the colors associated to vertices.
>
> **Question :** Is it possible to color the graph using only $k$ colors such that two adjacent vertices do not share the same color ?

This problem formulated in a static context has been proved to be NP-complete in 1972 by Richard Karp. For dynamic graphs, this problem can be formulated in a very similar way, except that time appears in the description:

> **Problem Name :** *Dynamic graph coloring*.
>
> **Parameter Description :** a dynamic colored graph $G(t) = (V(t), E(t), C(t))$, such that $V(t)$ denotes the vertices of the graph at time $t$, $E(t)$ the edges belonging to the graph at time $t$ and $C(t)$ the colors of the vertices at time $t$.
>
> **Question :** Is it possible to color the graph using only $k(t)$ colors such that at $t$ two adjacent vertices do not have the same color ?

Of course, as the static version of the problem is NP-complete, so is the dynamic one.

We are willing to solve this problem using a decentralized approach. Indeed, we would like the applicability of the method to be as wide as possible. For instance, if we consider this problem in the context of mobile ad hoc networks (MANETs), no device has a global view of the network and only strategies relying on local information are applicable.

For summing up, we address the problem of coloring a graph, that may change during the time it is considered, using as few colors as possible, by means of a decentralized strategy. In addition, we want to maintain the coloring property at any time in the graph, that is, no two adjacent nodes may have the same color at the same time. We consider that two vertices are connected as soon as at least one station is aware of the existence of the other.

# Dynamic Decentralized Graph Coloring

## Hypotheses

When dealing with decentralized approaches for graph coloring, there exist, at least, two ways of performing the task. On the one hand, one node initiates the process, on the other hand, the process starts on every node at a given date $t_{\text{start}}$, or as soon as the node is created, or appears in the network. In the sequel, we will refer to *broadcasting coloring* for the first scenario, and to *generalized coloring* to the second one. In all cases, it is supposed that:

1. no color set is initially known by the nodes,
2. colors are not comparable, that is there is no order on the set of colors,
3. when a node needs to use another color, this color is obtained by a random process that may produce either a new color or a color already existing,
4. finally, each vertex knows its neighborhood and may access to the color of his neighbors.

In the broadcasting coloring, one node begins with a new color and broadcast this color to his neighbors. Then, asynchronously, neighbors perform a coloring of themselves by either creating a new color or by choosing in the received color tab an existing color that do not belong to their neighbors. In a static context, each node may be processed only once, but in a dynamic setting the neighborhood of a vertex may change during the process such that the constraints are no more respected. So, each vertex should be both proactive and reactive. Proactivity is necessary for involving new nodes in the process, and reactivity is required for verifying the coloring constraint. In the generalized coloring scenario, each node comes with its own color. There is a high probability that the coloring constraint is verified at the beginning. The goal of each node, while preserving the coloring constraint, consists in trying to switch to another color already used in the network in order to reduce the number of used colors.

## Algorithm

Here is the proposed algorithm. It is executed by each vertex.

```
while true do
    if invitation to start is received from a neighbor then
      coloring()
    endIf
    if change in neighborhood detected then
      coloring()
    endIf
    if request for myColor then
      send myColor
    endIf
    if request for my color tab then
      send AllColors[]
    endIf
endWhile


coloring()
    Impossible[] ← request neighbors to their color
    AllColors[] ← request neighbors to their color tables
    if there is a color c such that c ∈ AllColors
    and c ∉ Impossible then
      myColor ← c
    else
      AllColors[] ← AllColors[] ⋃ new color c
      myColor ← c
    endIf
```

# Experiments

Preliminary results are reported in this section. The experiments were led using GraphStream, a library for visualizing and manipulating dynamic graphs [DGOP07].

First we note that the dynamic decentralized algorithm can self-adapt to the situation since it is able to increase and decrease the number of colors according to the dynamic of the graph.

Second, it appears that the performances in terms of number of used colors of the dynamic decentralized algorithm is almost as good as the Welsh and Powell algorithm, as it can be seen on the pictures 1 and 2.

# References


[DGOP07] Antoine Dutot, Frédéric Guinand, Damien Olivier, and Yoann Pigné. Graphstream: A tool for bridging the gap between complex systems and dynamic graphs. In *EPNACS'2007 workshop of ECCS'07*, 2007.

[GJ79] M.R. Garey and D.S. Johnson. *Computers and intractability: A guide to the theory of NP-completeness*. W.H. Freeman, New York, 1979.

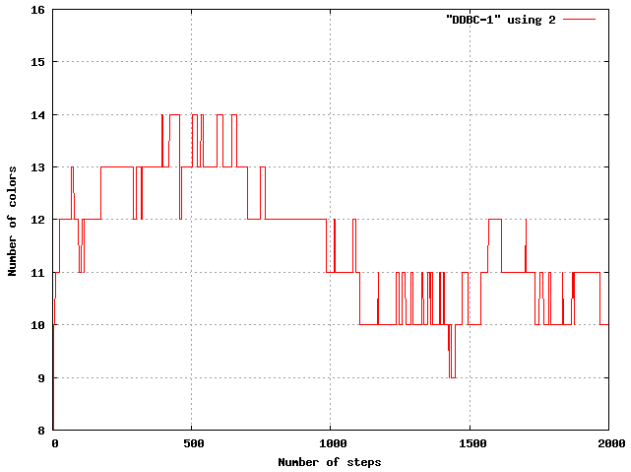


Figure 1: Results obtained using the Dynamic Decentralized Broadcasting Coloring algorithm. We can observe that the number of used colors in the graph may increase for constraint satisfaction, but it may also decrease according to the dynamic of the graph.

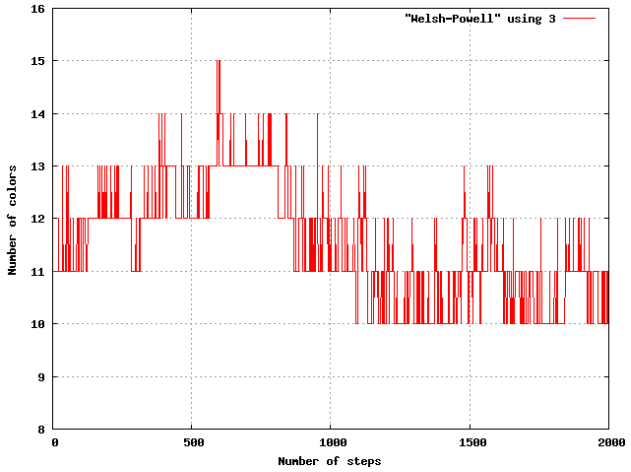


Figure 2: Results obtained for the same series of graphs by applying the Welsh and Powell static algorithm at each time step.