Un algorithme de visualisation de graphes plans

Frédéric Papadopoulos*, Jean-Christophe Janodet*

*Laboratoire IBISC - Université d'Evry, France
frederic.papadopoulos@ibisc.univ-evry.fr,
janodet@ibisc.univ-evry.fr

Le contexte général de notre travail est la classification de graphes plans par des techniques d'inférence grammaticale. Par *graphe plan*, on entend des *dessins* de graphes planaires sans croisement d'arêtes ; on les obtient par exemple après segmentation d'images 2D (Damiand et al. 2014). Une des particularités de ces graphes plans est qu'ils ont des faces, comme tout graphe planaire, mais que celles-ci sont en outre connues et fixées dans le dessin ; cette propriété peut être avantageusement exploitée pour résoudre efficacement des problèmes algorithmiques qui sont NP-difficiles sur les graphes planaires quelconques (de la Higuera et al. 2013).

Pour attaquer notre problème de classification, nous avons introduit un nouveau type de *grammaire de graphes* (Eyraud et al. 2012). Notre technique vise à apprendre une grammaire probabiliste par classe puis à attribuer à un nouveau graphe plan la classe la plus probable. Cependant ces grammaires sont des générateurs de graphes plans, et il est important de pouvoir visualiser les graphes qu'elles produisent pour améliorer l'apprentissage.

Nous nous intéressons donc ici à un problème de dessin de graphes (*graph drawing*). De nombreuses techniques ont été proposées dans la littérature (Di Battista et al. 1994) : on trouve par exemple les algorithmes de *graphes de dominance* dans lesquels des graphes acycliques voient leur sommets hiérarchisés : un sommet v est atteignable depuis un sommet u si les coordonnées cartésiennes de v sont supérieures ou égales à celles de u. Il y a également les *diagrammes d'arcs* où les sommets du graphe sont alignés sur un plan euclidien et les arcs dessinés comme des demi-cercles, ce qui est utilisé pour l'étude de motifs de chaînes. On trouve encore les méthodes par vecteurs propres, les approches circulaires...

L'approche dite *forced-based graph* s'inspire quant à elle de la mécanique classique. Le principe consiste à faire évoluer les positions des sommets du graphe en fonction de forces qui s'exercent sur eux. Après un temps, les forces d'attractions et de répulsions s'équilibrent et l'on obtient un dessin de graphe stable. Bien que coûteuse en temps, cette technique comporte plusieurs avantages : le processus continu permet à l'utilisateur d'observer étape par étape les évolutions du graphe ; elle peut être adaptée à des cas particuliers de graphes en rajoutant de nouvelles forces ; les résultats sont souvent intéressants du point de vue de la répartition uniforme des sommets, des longueurs d'arêtes similaires et de la symétrie interne au graphe.

Disposant d'informations sur les faces des graphes plans, c'est cette méthode que nous avons adaptée. Concernant les forces de répulsion, l'algorithme fait se repousser les sommets du graphe selon une loi de type forces gravitationnelles : plus deux sommets sont proches dans le dessin courant, et plus la force de répulsion qui s'exerce sur eux est importante. Nous modulons cette règle en fonction de la distance entre les deux sommets dans le graphe (au sens du plus court chemin) : deux sommets liés par une

arête se repoussent moins fortement que deux sommets éloignés dans le graphe.

Concernant les forces d'attraction, et contrairement aux algorithmes de type *forced-based*, elles s'exercent non pas entre deux sommets liés par une arête, mais entre les sommets d'une même face : une force de type ressort tend à les ramener vers le barycentre de la face.

L'ensemble des forces d'attraction et de répulsion assure une convergence du dessin vers un état *localement* stable. Toutefois, pour accélérer la convergence de l'algorithme, nous avons développé deux variantes. Dans la première, les "bords" de la fenêtre de dessin exercent une force supplémentaire repoussant les sommets vers le centre ; c'est une force de répulsion supplémentaire qui s'exerce sur chaque sommet. Dans la seconde, l'utilisateur désigne la face extérieure du graphe à dessiner ; les sommets de cette face particulière sont repoussés (et non plus attirés) par le barycentre de la face.

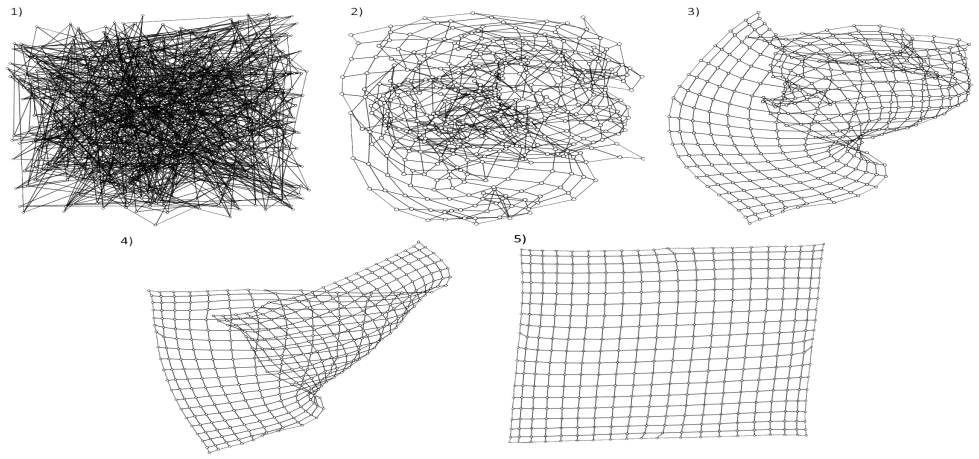

FIGURE 1 – Illustration du déroulement de l'algorithme. On part d'un maillage 20x20 ; les coordonnées des sommets sont fixées aléatoirement (1) ; les images (2) à (4) montrent des étapes successives jusqu'à convergence de l'algorithme (5).

Références

G. Damiand & P. Lienhard (2014), *Combinatorial Maps : Efficient Data Structures for Computer Graphics and Image Processing*, A K Peters/CRC Press.

G. Di Battista et al. (1994), *Algorithms for Drawing Graphs : an Annotated Bibliography*, Computational Geometry (4) 235-282.

R. Eyraud et al. (2012), *Learning Substitutable Binary Plane Graph Grammars*, Proc. ICGI'12, JMLR Conf. Proc. (21) 114-128.

C. de la Higuera et al. (2013), *Polynomial Algorithms for Open Plane Graph and Subgraph Isomorphism*, Theoretical Computer Science (498) 76-99.