Un algorithme de visualisation de graphes plans

Frédéric Papadopoulos et Jean-Christophe Janodet

Laboratoire IBISC, Université d'Evry, France

14 Octobre 2015

Contexte

Objectif de nos travaux

Être capable de classifier des *graphes plans* via des techniques d'inférence grammaticale

Un graphe plan

Est un *dessin* de graphe planaire sans croisement d'arête avec des faces connues et fixées.
Facilite la résolution de problèmes algorithmiques NP-difficiles sur les graphes planaires quelconques

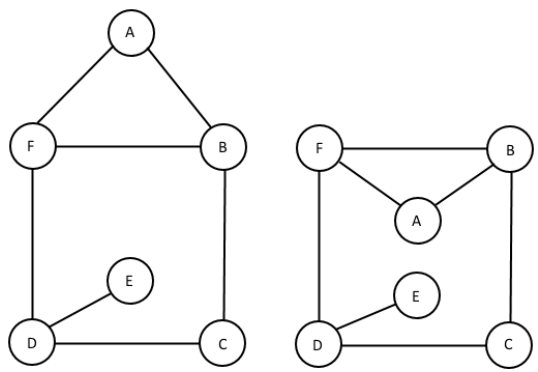

Table 1: Deux graphes planaires *isomorphes* dont les *dessins* ne sont pas identiques

Contexte (2)

Comment faire de la classification de graphes plans

On utilise des grammaires de cartes combinatoires probabilistes. On apprend une seule grammaire par classe et on attribue la classe la plus probable à une carte

Nouvelle problématique

Les classes générés par la grammaire décrivent la topologie, mais sans représenter les objets. Pour le confort de l'utilisateur il faut disposer d'un moyen efficace de représenter les graphes en s'appuyant sur nos connaissances

Etat de l'art

Dessiner des graphes est un problème largement étudié sous différents aspects. On trouve notamment :

- Des diagrammes d'arcs (Saaty 1964)
- Des agencements circulaires (Madden & Madden 1997)

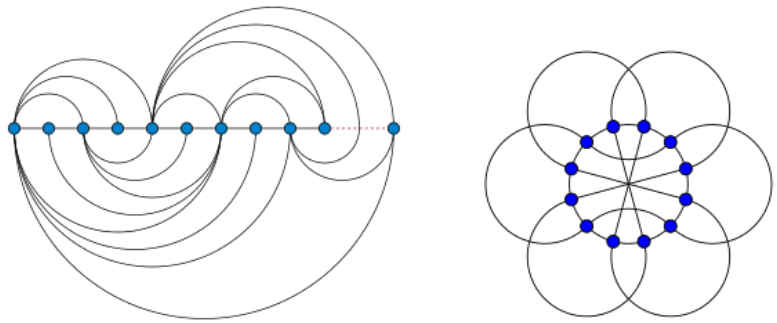

Table 2: A gauche un exemple de diagramme d'arcs, à droite un exemple d'agencement circulaire

- Des graphes de dominances (Di Battista et al. 1994)
- Des méthodes par vecteurs propres (Beckman 1994)
- Basés sur les forces
- et bien d'autres...

Algorithmes basés sur les forces

Algorithme présenté ici basé sur les algorithmes dits *force-based* ou *force-directed* (Di Battista et al. 1994)

Principe général

Des forces sont calculés en fonction de la position respectives des sommets, puis leur position évolue en fonction des forces qui s'exercent sur eux. Après un temps les forces d'attraction et de répulsion s'équilibrent et on obtient un dessin de graphe stable

Algorithmes basés sur les forces (2)

Quelques variations :

- Système à ressorts + champs électriques (Kobourov 2012)
- Contractions des sommets pour dessiner une première approximation (Goodrich et Kobourov 2004)

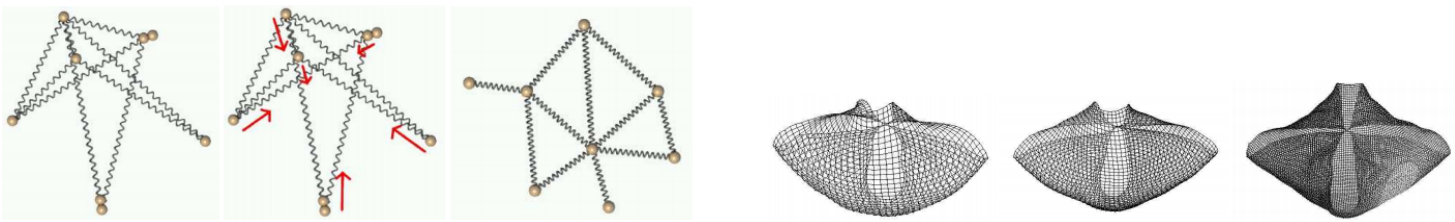

Table 3: A gauche un exemple de système à ressort, à droite un exemple de contraction de sommets

Algorithmes basés sur les forces (3)

- Méthode Lombardi : les arêtes sont toutes des arcs, réparties de façon homogènes autour des sommets (Chernobelskiy et al 2011)

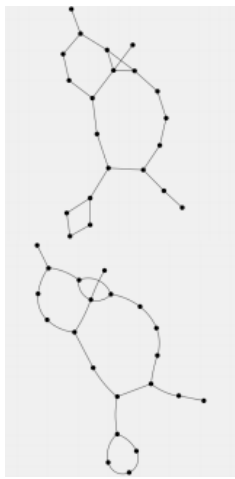

Table 4: En haut un graphe classique, en bas le même graphe dessiné en utilisant la méthode de Lombardi

Algorithmes basés sur les forces (3)

Inconvénients :

- Coûteux en temps (de l'ordre de $O(n^2)$ par itération)
- Converge vers un *minimum local*

Avantages :

- Principe simple et adaptable par l'ajout de règles
- Observation dynamique de l'évolution du graphe
- Répartition des sommets et longueur d'arêtes (relativement) uniformes
- Symétries interne au graphe

Principe de l'algorithme

Algorithme de type force-based qui s'appuie sur la connaissance des faces.
Forces exercées :

- Attraction : tous les sommets sont attirés par le barycentre de leurs faces
- Répulsion : influence d'un sommet s' sur un sommet s selon une force "loi gravitationnelle" inversée. Le sommet s est déplacé inversement proportionnellement au cube de sa distance à s'
- Répulsion : cas particulier de la face extérieure (facultative), force similaire à un ressort (comprimé)

Principe de l'algorithme (2)

Autres effets :

- Dépendance des forces de répulsions à la distance des sommets dans le graphe
- Fenêtre d'observation du graphe se repositionne systématiquement à une taille idéale

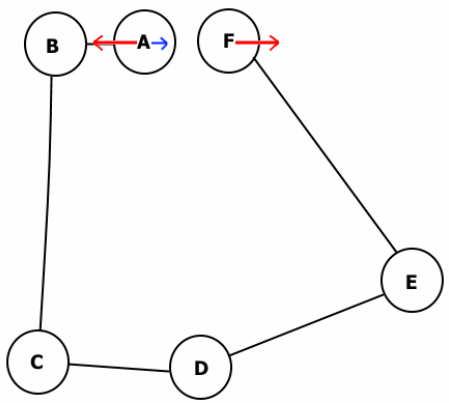

Table 5: Les forces de répulsions entre A et F sont très importantes à cause de leur distance dans le graphe (en rouge). En comparaison la répulsion de B sur A est beaucoup plus faible (en bleu)

Fonctionnalités du logiciel

Pour l'instant au stade expérimental, vise à être étendu aux cartes combinatoires (où les faces sont obligatoirement connues)

- Charger un graphe / générer aléatoirement
- Repositionner les sommets à la souris, forcer un polygone régulier, force le rapprochement d'une face
- Visionner les forces d'attraction/répulsion s'exerçant, les intersections générées, mettre en pause. Redimensionnement dynamique de la fenêtre
- Échanger les positions des sommets générant le plus d'intersection

Observations et conclusion

Observations

- Régularité de la solution (longueur d'arêtes, taille des faces, disposition des sommets)
- Peu (ou pas) de croisements
- Légères distorsions en périphérie (commune aux méthodes force-based)
- Vitesse de convergence acceptable pour les graphes de petite et moyenne taille.
- Convergence du graphe vers une solution dépendant fortement de la configuration initiale (aléatoire dans les test effectués ici)

Observations et conclusion (2)

Conclusion

- Algorithmes *force-based* : des simplifications doivent être envisagées pour les échantillons plus important (suppression des influences trop distantes, contractions de sommets ou notion de température).
- Positionnement initial "intelligent" pour améliorer le temps de convergence et la solution finale (en général un positionnement selon une simple lecture du graphe améliore drastiquement la convergence).
- Adaptabilité intéressante pour visualiser les résultat de classifications via des grammaires de cartes combinatoires (qui décrivent explicitement les faces)