Dessin de Graphes

Romain Bourqui

Maître de Conférences

romain.bourqui@labri.fr

Dessin de Graphes

Plan

- Introduction
- Dessin de graphes planaires
- Dessin hiérarchique
- Dessin par analogie physique

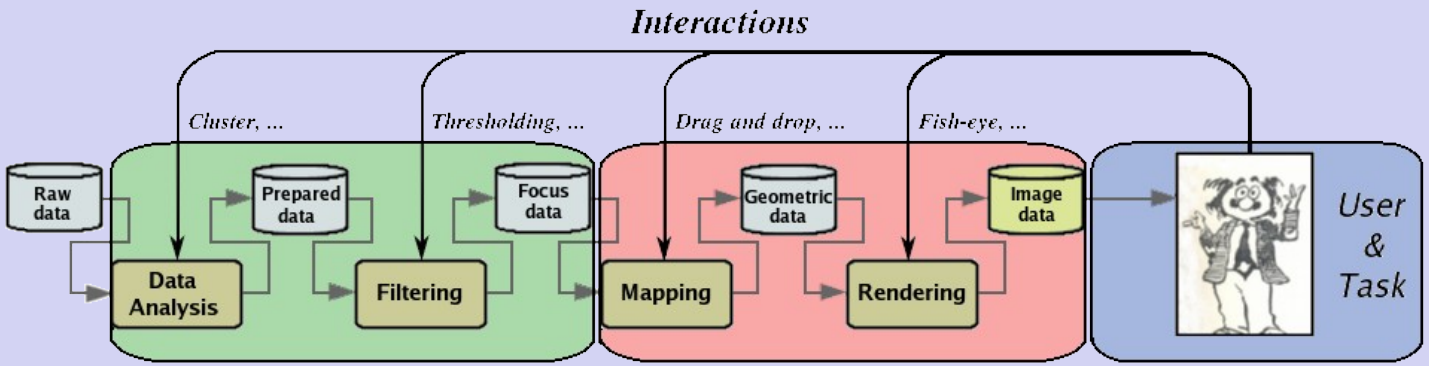
Dessin de Graphes

Plan

- Introduction
- Dessin de graphes planaires
- Dessin hiérarchique
- Dessin par analogie physique

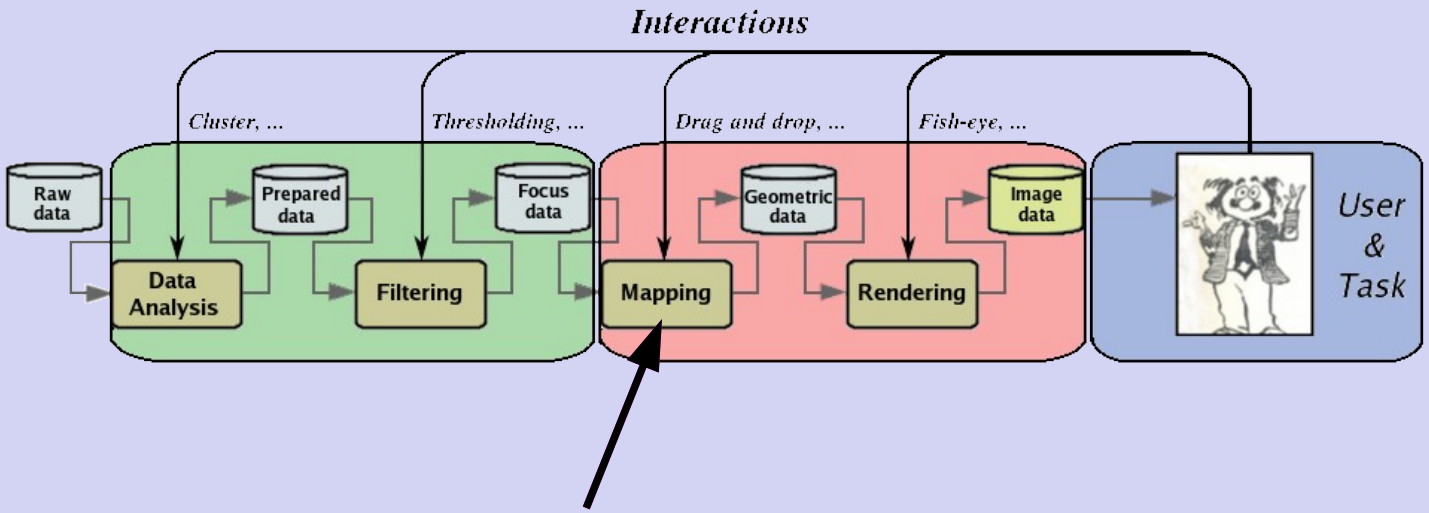
Dessin de Graphes

Visualization Pipeline [dos Santos et Brodlie 04]



Dessin de Graphes

Visualization Pipeline [dos Santos et Brodlie 04]



Dessin de Graphes

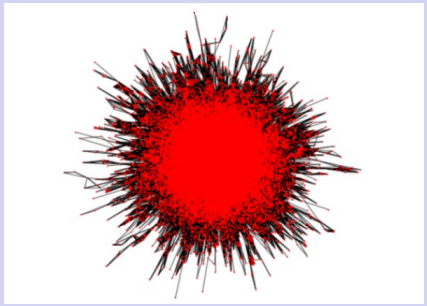
Dessin de Graphes

Introduction

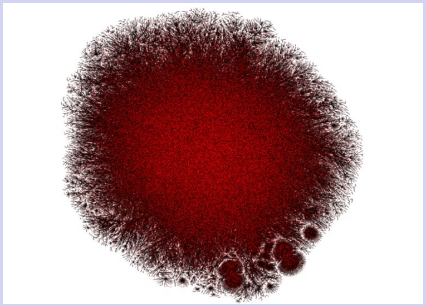
- La représentation automatique de graphe est un domaine de recherche à part entière.
  - Origine dans la construction de circuit imprimés
  - Mais aussi pour la visualisation des informations relationnelles (réseaux d'interactions, réseaux sociaux, réseaux internet/informatiques,...)

Dessin de Graphes

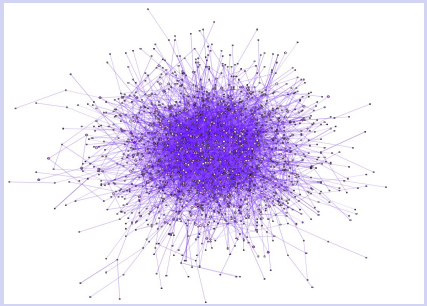
Introduction: Exemples



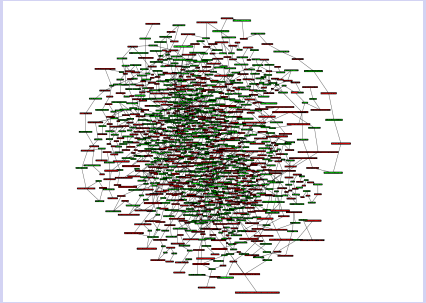
Réseau d'acteurs



Réseau de routeurs internet



Réseau d'interactions gène-gène



Réseau métabolique

Dessin de Graphes

Objectifs

- Entrée
  - ✔ Un graphe G=(V,E) avec un ensemble V de *n* sommets et un ensemble E de *m* arcs
- Sortie
  - ✔ Un tracé de G lisible et compréhensible par l'utilisateur
- 4 concepts de base [Di-Battista et al. 1999]

  1) Contraintes de supports
  2) Convention de dessin
  3) Critères esthétiques
  4) Contraintes sémantiques

Dessin de Graphes

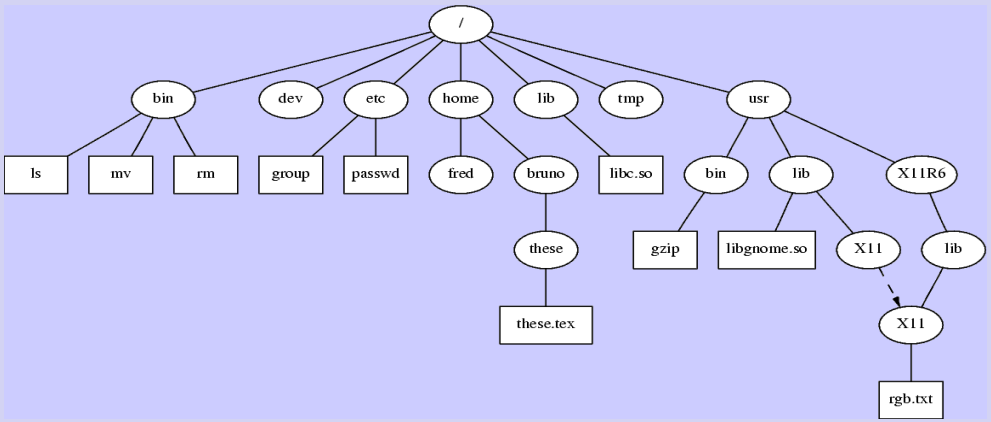
Objectifs

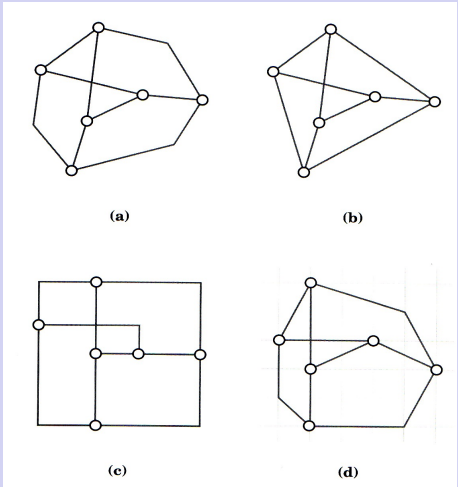
1) Contraintes de supports

- ✔ Exemples : feuille A4, écran d'ordinateur

2) Convention de dessin

- ✔ Exemples : lignes droites, brisées, orthogonal, planaire, orienté, ...





Dessin de Graphes

Objectifs

3) Critères "esthétiques"

Modélisent la lisibilité et la mémorisation des informations

- ✔ Nombre de croisements d'arêtes (un des plus importants [Purchase:2000])
- ✔ Somme de la longueur des arêtes
- ✔ Longueur d'arête maximale
- ✔ Longueur d'arête uniforme
- ✔ Nombre de points de contrôle
- ✔ Nombre max de points de contrôle sur une arête.
- ✔ Résolution angulaire : Angle minimum formé par deux arêtes
- ✔ Ratio : Proportion entre la hauteur et la largeur du dessin
- ✔ Symétrie : Mise en évidence des symétrie dans le graphe.

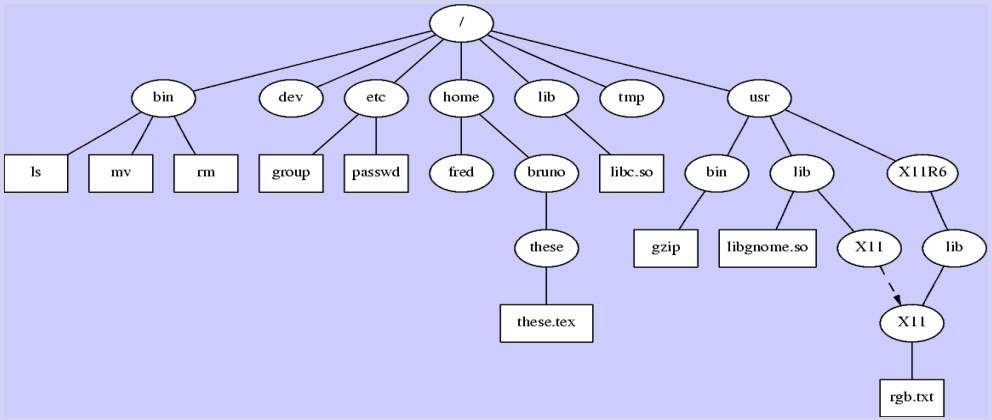
Dessin de Graphes

Objectifs

4) Contraintes sémantiques

Critères pour aider à l'interprétation des composantes du graphe

- ✔ Proximité entre les sommets
- ✔ Formes particulières des sommets et des arcs
- ✔ ...



Dessin de Graphes

Plan

- Introduction
- Dessin de graphes planaires
- Dessin hiérarchique
- Dessin par analogie physique

Dessin de Graphes

Plan

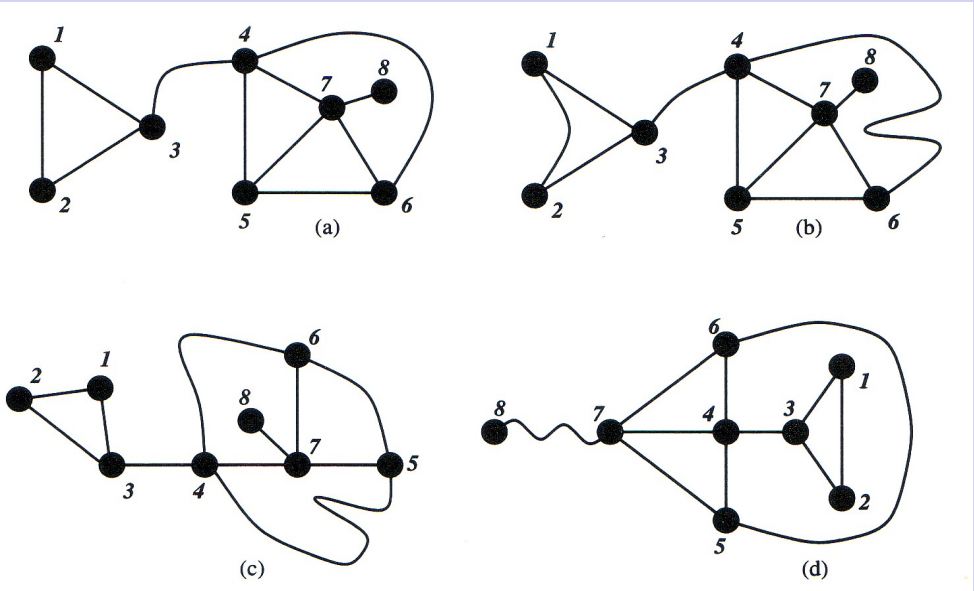
- Introduction
- Dessin de graphes planaires
- Dessin hiérarchique
- Dessin par analogie physique

Dessin de Graphes Planaires

Notions de base

- Graphe planaire: Graphe pouvant se représenter sans croisement dans le plan.
- Carte : Graphe dont l'ordre du voisinage de chaque sommet est fixé.
- Formule de Euler

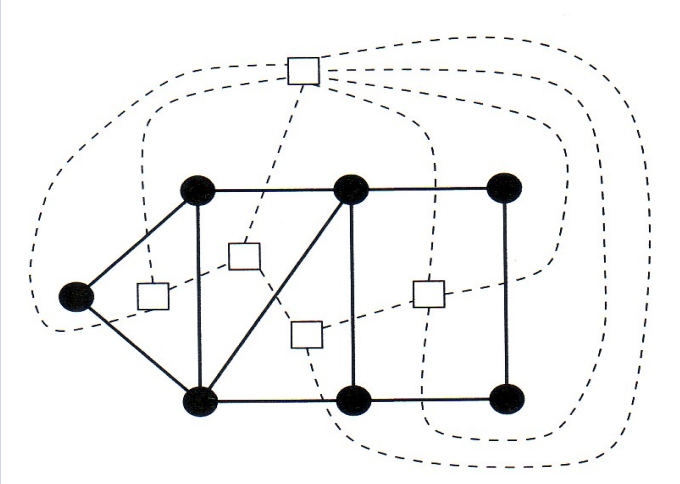
  |E| <= 3 * |V| - 6



Dessin de Graphes Planaires

Notions de base

- Graphe dual:
  - Graphe représentant les connections entre les faces d'un graphe.
- Graphe planaire-extérieur (outerplanar): Graphe qui peut être représenté de façon planaire avec tous ses sommets sur la face extérieure.



Dessin de Graphes Planaires

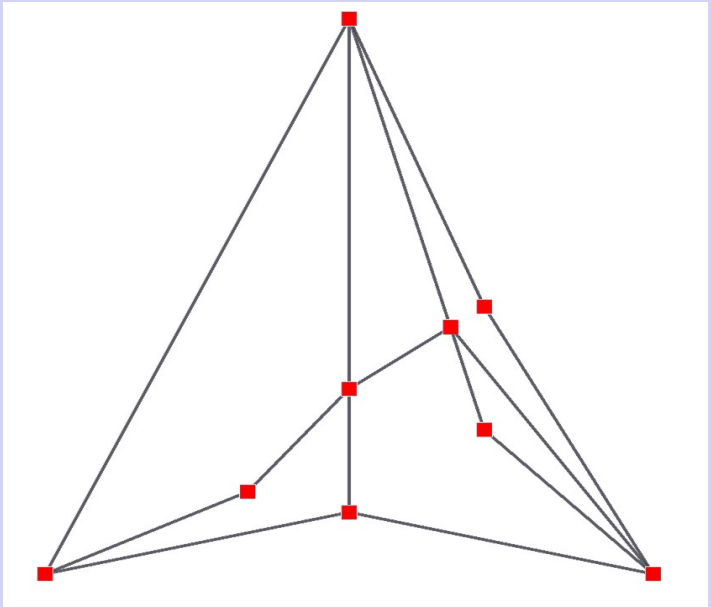
Différents styles

- Dessin en lignes droites

Dessin de Graphes Planaires

Différents styles

- Dessin en lignes droites



Dessin de Graphes Planaires

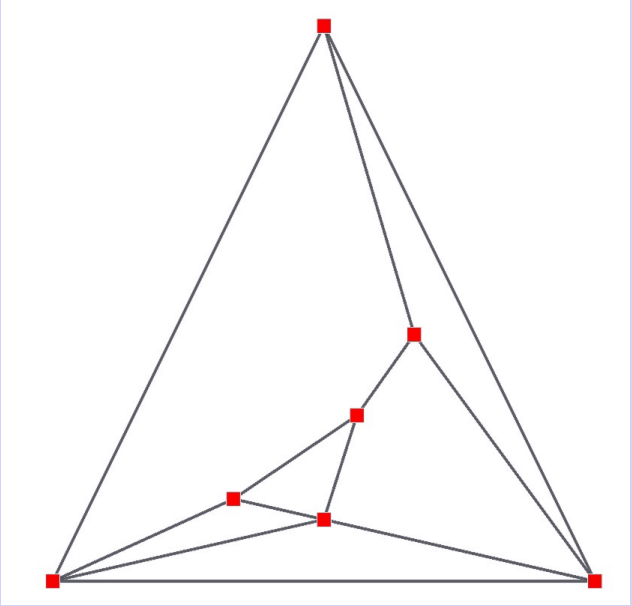
Différents styles

- Dessin en lignes droites
  - Dessin convexe

Dessin de Graphes Planaires

Différents styles

- Dessin en lignes droites
  - Dessin convexe



Dessin de Graphes Planaires

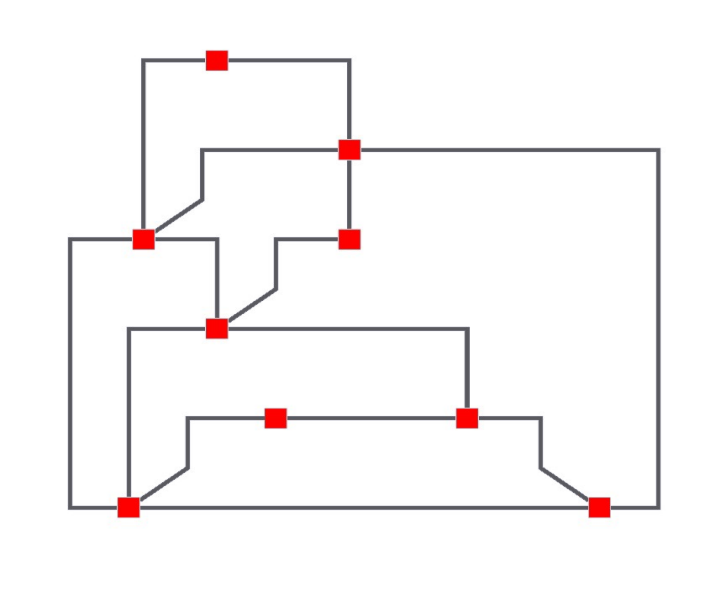
Différents styles

- Dessin en lignes droites
  - Dessin convexe
- Dessin en lignes brisées

Dessin de Graphes Planaires

Différents styles

- Dessin en lignes droites
  - Dessin convexe
- Dessin en lignes brisées



Dessin de Graphes Planaires

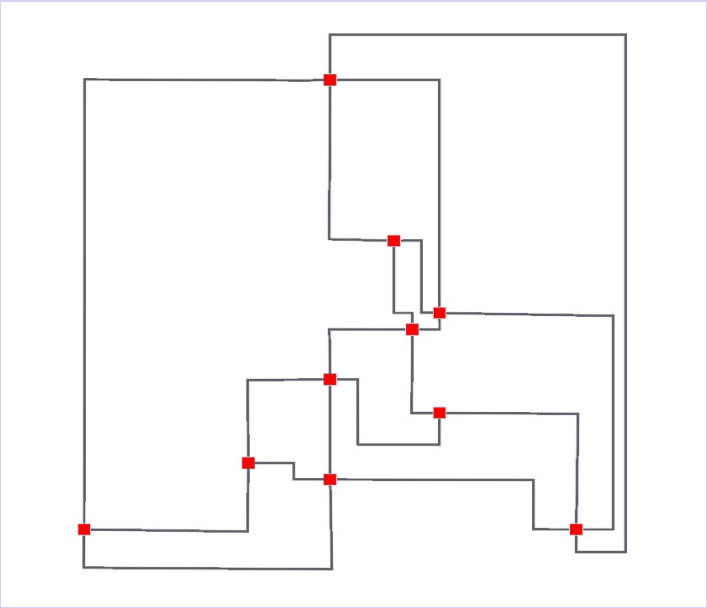
Différents styles

- Dessin en lignes droites
  - Dessin convexe
- Dessin en lignes brisées
  - Dessin orthogonal

Dessin de Graphes Planaires

Différents styles

- Dessin en lignes droites
  - Dessin convexe
- Dessin en lignes brisées
  - Dessin orthogonal



Dessin de Graphes Planaires

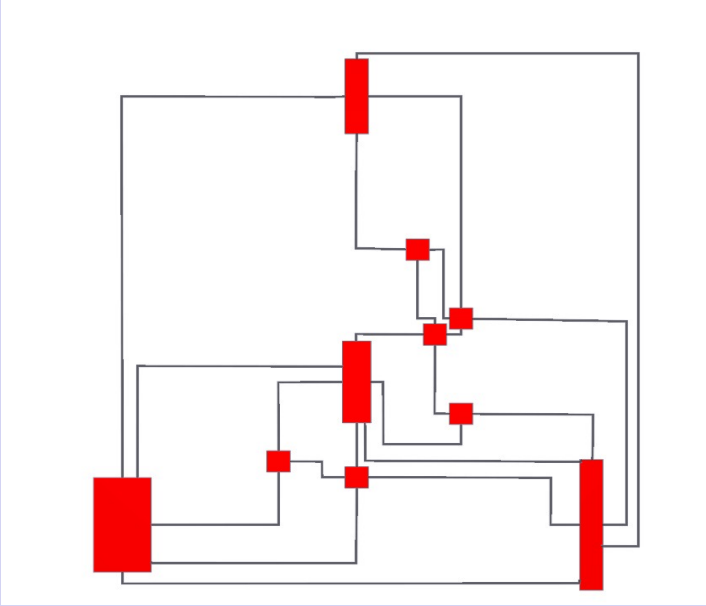
Différents styles

- Dessin en lignes droites
  - Dessin convexe
- Dessin en lignes brisées
  - Dessin orthogonal
  - Dessin « box » - orthogonal

Dessin de Graphes Planaires

Différents styles

- Dessin en lignes droites
  - Dessin convexe
- Dessin en lignes brisées
  - Dessin orthogonal
  - Dessin « box » - orthogonal



Dessin de Graphes Planaires

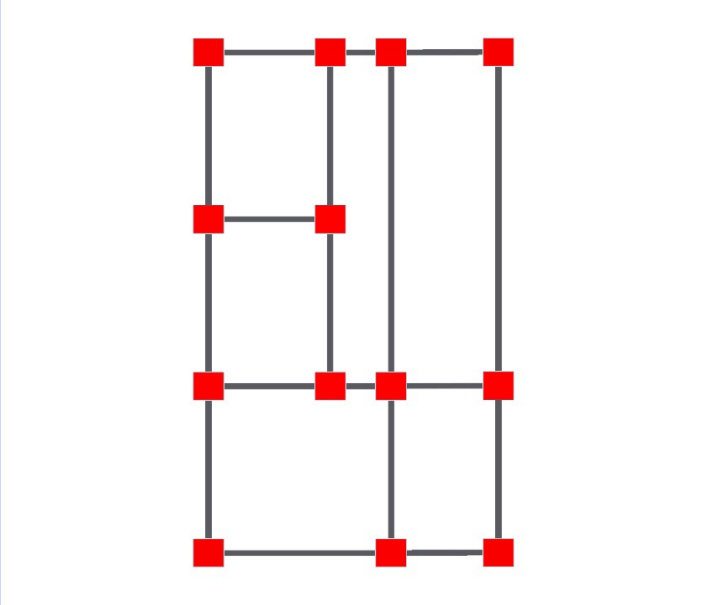
Différents styles

- Dessin en lignes droites
  - Dessin convexe
- Dessin en lignes brisées
  - Dessin orthogonal
  - Dessin « box » - orthogonal
  - Dessin rectangulaire

Dessin de Graphes Planaires

Différents styles

- Dessin en lignes droites
  - Dessin convexe
- Dessin en lignes brisées
  - Dessin orthogonal
  - Dessin « box » - orthogonal
  - Dessin rectangulaire



Dessin de Graphes Planaires

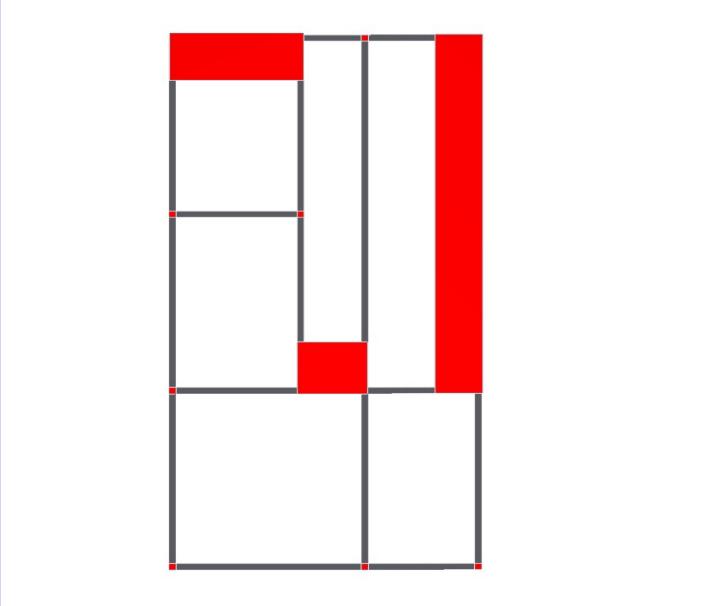
Différents styles

- Dessin en lignes droites
  - Dessin convexe
- Dessin en lignes brisées
  - Dessin orthogonal
  - Dessin « box » - orthogonal
  - Dessin rectangulaire
  - Dessin « box » - rectangulaire

Dessin de Graphes Planaires

Différents styles

- Dessin en lignes droites
  - Dessin convexe
- Dessin en lignes brisées
  - Dessin orthogonal
  - Dessin « box » - orthogonal
  - Dessin rectangulaire
  - Dessin « box » - rectangulaire



Dessin de Graphes Planaires

Différents styles

- Dessin en lignes droites
  - Dessin convexe
- Dessin en lignes brisées
  - Dessin orthogonal
  - Dessin « box » - orthogonal
  - Dessin rectangulaire
  - Dessin « box » - rectangulaire
- Dessin sur la grille

Dessin de Graphes Planaires

Exemple d'algorithme de dessin en ligne droite

Algorithme « shift » de De Fraisseix et al. **[FFP89]**

- Entrée
  - Un graphe G=(V,E) planaire triangulé, ainsi qu'une carte planaire associée.

Dessin de Graphes Planaires

Exemple d'algorithme de dessin en ligne droite

Algorithme « shift » de De Fraisseix et al. **[FFP89]**

- Entrée
  - Un graphe G=(V,E) planaire triangulé, ainsi qu'une carte planaire associée.
- Sortie
  - Un dessin en lignes droites sur la grille du graphe G
  - Taille du dessin: $O((2\ |V| - 4) \times (|V| - 2))$
  - Complexité temps/mémoire: $O(n \log(n))$ et $O(n)$

Dessin de Graphes Planaires

Exemple d'algorithme de dessin en ligne droite

Algorithme « shift » de De Fraisseix et al. [FFP89]

- Entrée
  - Un graphe G=(V,E) planaire triangulé, ainsi qu'une carte planaire associée.
- Sortie
  - Un dessin en lignes droites sur la grille du graphe G
  - Taille du dessin: $O((2\ |V| - 4) \times (|V| - 2))$
  - Complexité temps/mémoire: $O(n \log(n))$ et $O(n)$
- Principe
  - Calcul d'un ordre sur les sommets: l'ordre canonique
  - Placement des sommets utilisant cet ordre

Dessin de Graphes Planaires

Algorithme de De Fraissex et al.: Ordre Canonique

Soit $\Pi = \{v_1, v_2, ..., v_n\}$ un ordre sur les sommets de G.

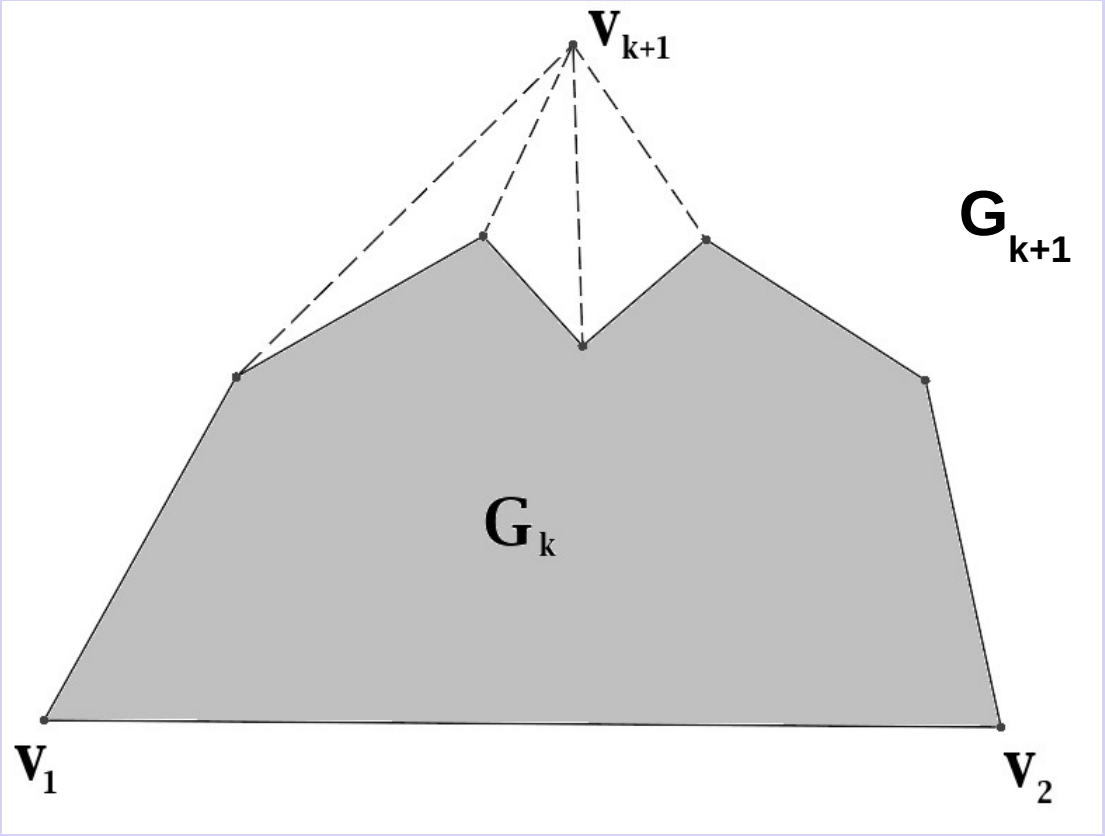
Pour tout $3 \leq k \leq n$, on note G_k le sous-graphe induit par $\{v_1, v_2, ..., v_k\}$ dans G.

On appelle Π **ordre canonique** ssi les conditions suivantes sont respectées:

- G_k est biconnexe et « intérieurement » triangulé
- (v_1, v_2) est une arête extérieure de G_k
- Si $k+1 \leq n$, alors le sommet v_{k+1} est positionné sur la face extérieure de G_k, et tous les voisins de v_{k+1} dans G_k apparaissent sur la face extérieure de G_k de manière consécutive.

Dessin de Graphes Planaires

Algorithme de De Fraissex et al.: Ordre Canonique



Dessin de Graphes Planaires

Algorithme de De Fraissex et al.: Ordre Canonique

Algorithm Canonical-Ordering(G)
begin

1 Let v_1, v_2 and v_n be the vertices appearing on the outer cycle counterclockwise in this order;

2 Set $chords(x) = 0$, $out(x) = false$, and $mark(x) = false$ for all vertices $x \in V$;

3 Set $out(v_1) = true$, $out(v_2) = true$, and $out(v_n) = true$;

4 **for** $k = n$ **down to** 3 **do**
begin

5 Choose any vertex x such that $mark(x) = false$, $out(x) = true$, $chords(x) = 0$, and $x \neq v_1, v_2$;

6 Set $v_k = x$ and $mark(x) = true$;

7 Let $C_o(G_{k-1}) = w_1, w_2, \cdots, w_t$, where $w_1 = v_1$ and $w_t = v_2$;

8 Let $w_p, w_{p+1}, \cdots, w_q$ be the neighbors of v_k which have $mark(w_i) = false$;

9 For each vertex w_i, $p < i < q$, set $out(w_i) = true$, and update the variable *chords* for w_i and its neighbors.

end

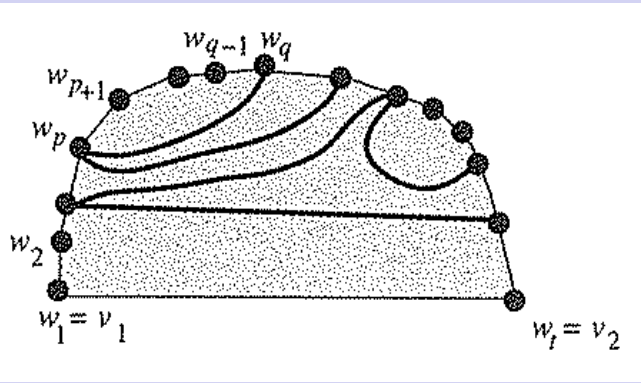
end.

Dessin de Graphes Planaires

Algorithme de De Fraissex et al.: Ordre Canonique

Algorithm Canonical-Ordering(G)
begin

1 Let v_1, v_2 and v_n be the vertices appearing on the outer cycle counterclockwise in this order;
2 Set $chords(x) = 0$, $out(x) = false$, and $mark(x) = false$ for all vertices $x \in V$;
3 Set $out(v_1) = true$, $out(v_2) = true$, and $out(v_n) = true$;
4 **for** $k = n$ **down to** 3 **do**
begin
5 Choose any vertex x such that $mark(x) = false$, $out(x) = true$, $chords(x) = 0$, and $x \neq v_1, v_2$;
6 Set $v_k = x$ and $mark(x) = true$;
7 Let $C_o(G_{k-1}) = w_1, w_2, \cdots, w_t$, where $w_1 = v_1$ and $w_t = v_2$;
8 Let $w_p, w_{p+1}, \cdots, w_q$ be the neighbors of v_k which have $mark(w_i) = false$;
9 For each vertex w_i, $p < i < q$, set $out(w_i) = true$, and update the variable *chords* for w_i and its neighbors.
end
end.

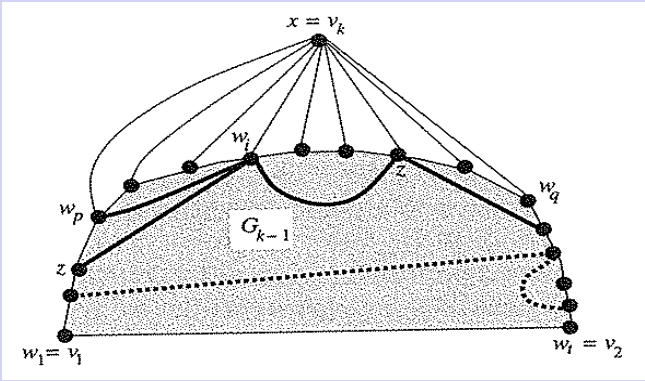


Dessin de Graphes Planaires

Algorithme de De Fraissex et al.: Ordre Canonique

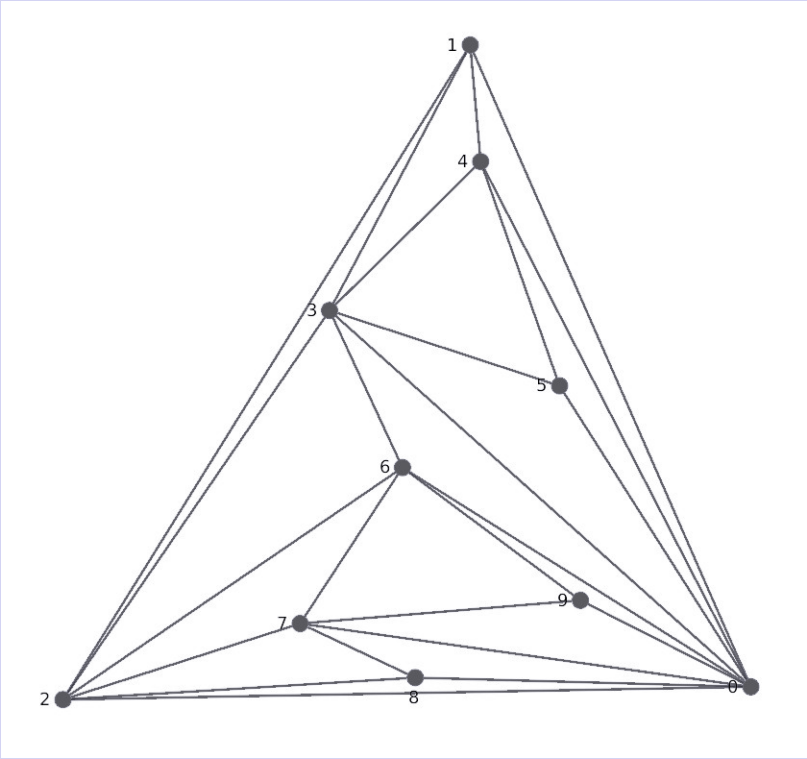
Algorithm Canonical-Ordering(G)
begin

1 Let v_1, v_2 and v_n be the vertices appearing on the outer cycle counterclockwise in this order;
2 Set $chords(x) = 0$, $out(x) = false$, and $mark(x) = false$ for all vertices $x \in V$;
3 Set $out(v_1) = true$, $out(v_2) = true$, and $out(v_n) = true$;
4 **for** $k = n$ **down to** 3 **do**
begin
5 Choose any vertex x such that $mark(x) = false$, $out(x) = true$, $chords(x) = 0$, and $x \neq v_1, v_2$;
6 Set $v_k = x$ and $mark(x) = true$;
7 Let $C_o(G_{k-1}) = w_1, w_2, \cdots, w_t$, where $w_1 = v_1$ and $w_t = v_2$;
8 Let $w_p, w_{p+1}, \cdots, w_q$ be the neighbors of v_k which have $mark(w_i) = false$;
9 For each vertex w_i, $p < i < q$, set $out(w_i) = true$, and update the variable *chords* for w_i and its neighbors.
end
end.



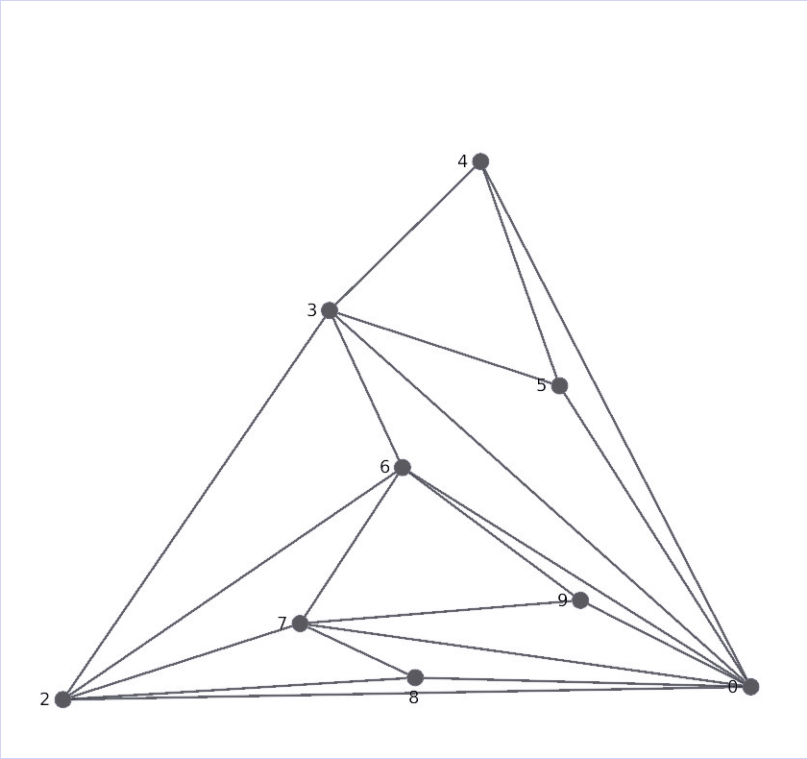
Dessin de Graphes Planaires

Algorithme de De Fraissex et al.: Ordre Canonique



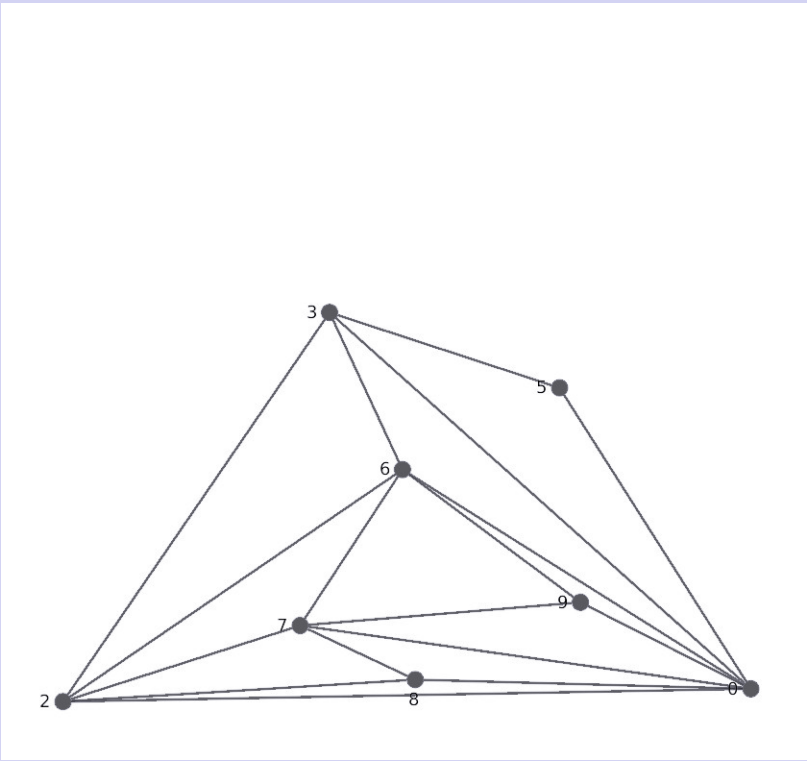
Dessin de Graphes Planaires

Algorithme de De Fraissex et al.: Ordre Canonique



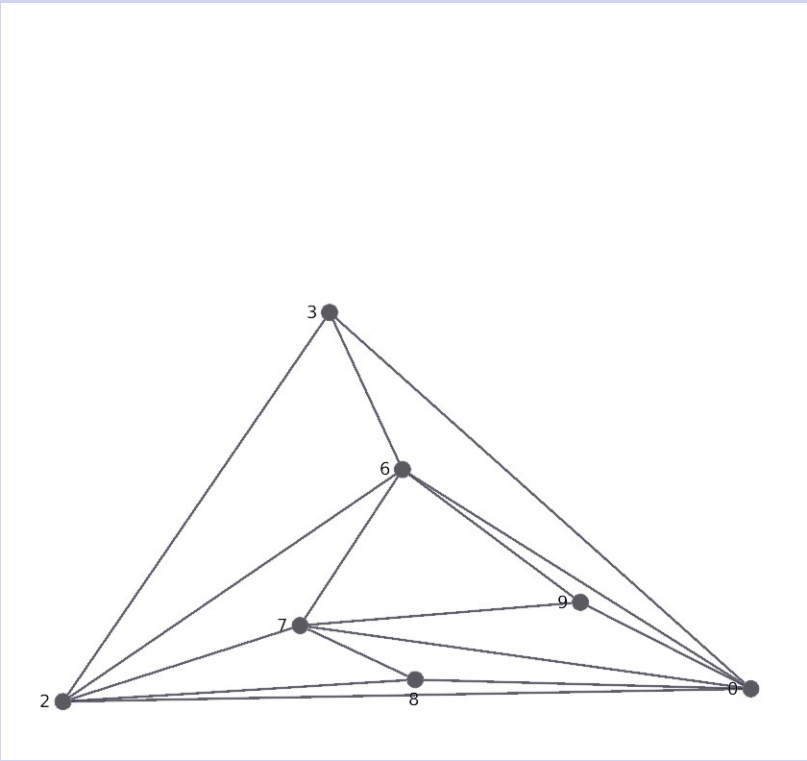
Dessin de Graphes Planaires

Algorithme de De Fraissex et al.: Ordre Canonique



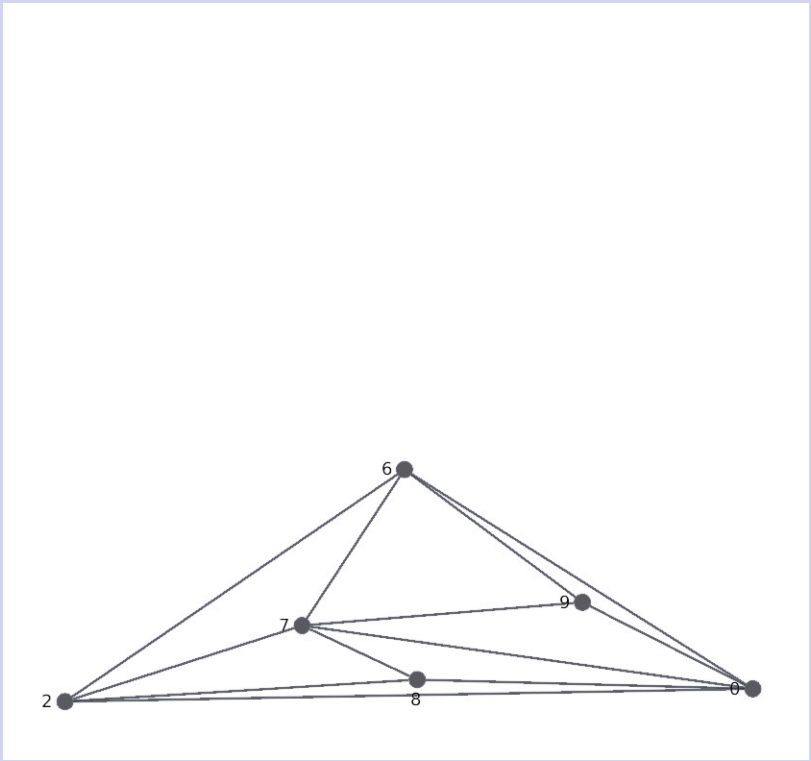
Dessin de Graphes Planaires

Algorithme de De Fraissex et al.: Ordre Canonique



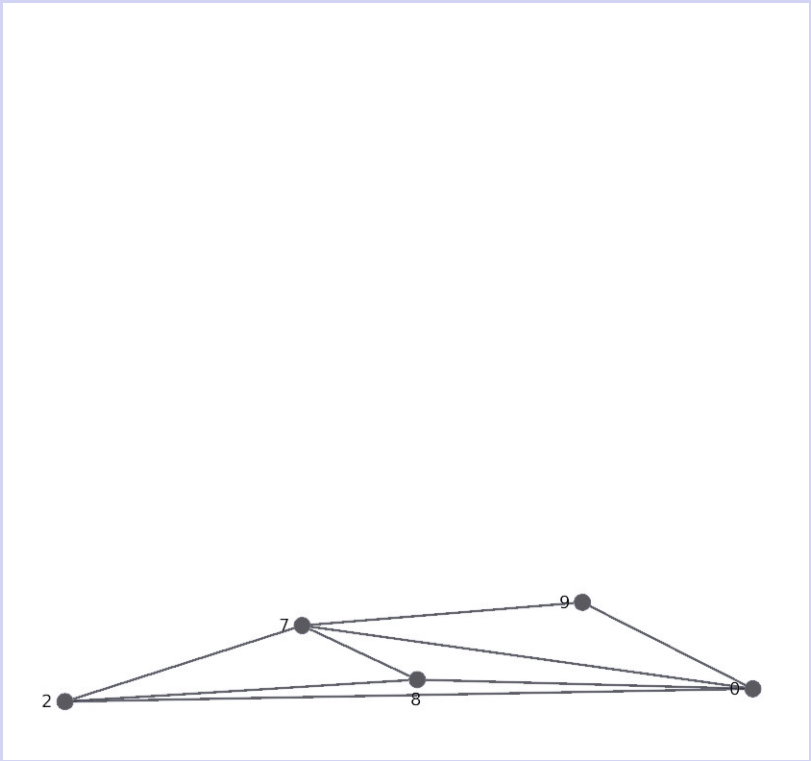
Dessin de Graphes Planaires

Algorithme de De Fraissex et al.: Ordre Canonique



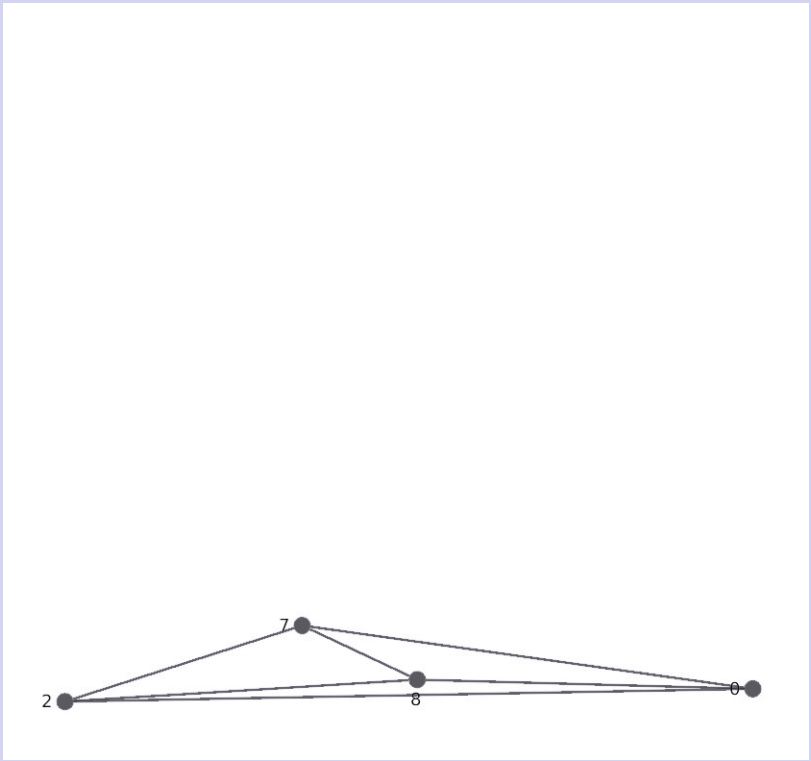
Dessin de Graphes Planaires

Algorithme de De Fraissex et al.: Ordre Canonique



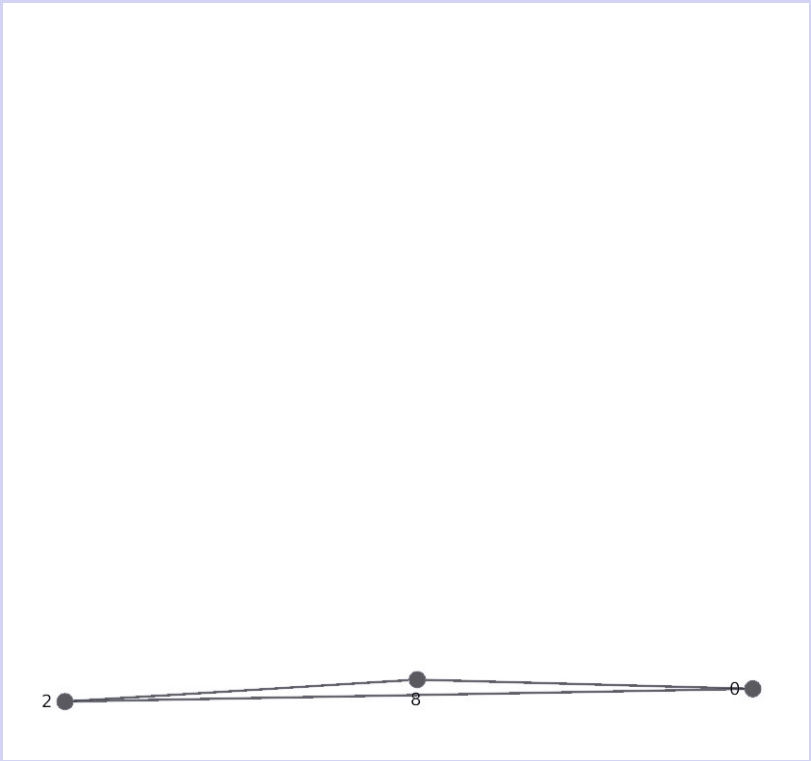
Dessin de Graphes Planaires

Algorithme de De Fraissex et al.: Ordre Canonique



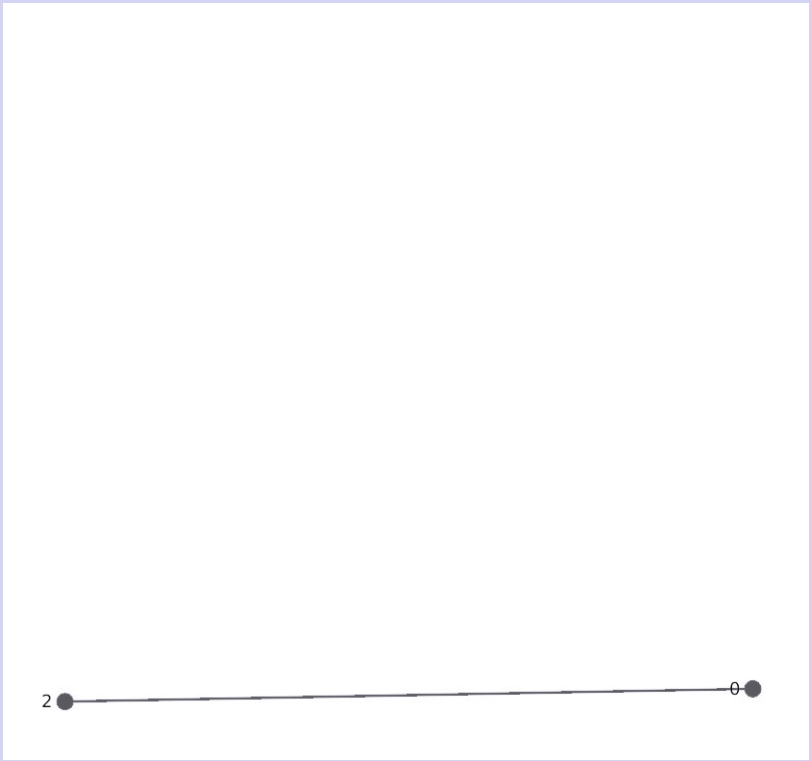
Dessin de Graphes Planaires

Algorithme de De Fraissex et al.: Ordre Canonique



Dessin de Graphes Planaires

Algorithme de De Fraissex et al.: Ordre Canonique



Dessin de Graphes Planaires

Algorithme de De Fraissex et al.: Ordre Canonique

Algorithm Canonical-Ordering(G)
begin

1 Let v_1, v_2 and v_n be the vertices appearing on the outer cycle counterclockwise in this order;
2 Set $chords(x) = 0$, $out(x) = false$, and $mark(x) = false$ for all vertices $x \in V$;
3 Set $out(v_1) = true$, $out(v_2) = true$, and $out(v_n) = true$;
4 **for** $k = n$ **down to** 3 **do**
begin
5 Choose any vertex x such that $mark(x) = false$, $out(x) = true$, $chords(x) = 0$, and $x \neq v_1, v_2$;
6 Set $v_k = x$ and $mark(x) = true$;
7 Let $C_o(G_{k-1}) = w_1, w_2, \cdots, w_t$, where $w_1 = v_1$ and $w_t = v_2$;
8 Let $w_p, w_{p+1}, \cdots, w_q$ be the neighbors of v_k which have $mark(w_i) = false$;
9 For each vertex w_i, $p < i < q$, set $out(w_i) = true$, and update the variable *chords* for w_i and its neighbors.
end
end.

- **Complexité linéaire !**

Dessin de Graphes Planaires

Algorithme de De Fraissex et al.: Ordre Canonique

Algorithm Canonical-Ordering(G)
begin

1 Let v_1, v_2 and v_n be the vertices appearing on the outer cycle counterclockwise in this order;
2 Set $chords(x) = 0$, $out(x) = false$, and $mark(x) = false$ for all vertices $x \in V$;
3 Set $out(v_1) = true$, $out(v_2) = true$, and $out(v_n) = true$;
4 **for** $k = n$ **down to** 3 **do**
begin
5 Choose any vertex x such that $mark(x) = false$, $out(x) = true$, $chords(x) = 0$, and $x \neq v_1, v_2$;
6 Set $v_k = x$ and $mark(x) = true$;
7 Let $C_o(G_{k-1}) = w_1, w_2, \cdots, w_t$, where $w_1 = v_1$ and $w_t = v_2$;
8 Let $w_p, w_{p+1}, \cdots, w_q$ be the neighbors of v_k which have $mark(w_i) = false$;
9 For each vertex w_i, $p < i < q$, set $out(w_i) = true$, and update the variable *chords* for w_i and its neighbors.
end
end.

- **Complexité linéaire !**
  - Ligne 5: liste maj
  - Ligne 8: deg(v_k)
  - Ligne 9: deg(w_i)

Dessin de Graphes Planaires

Algorithme de De Fraissex et al.: Placement

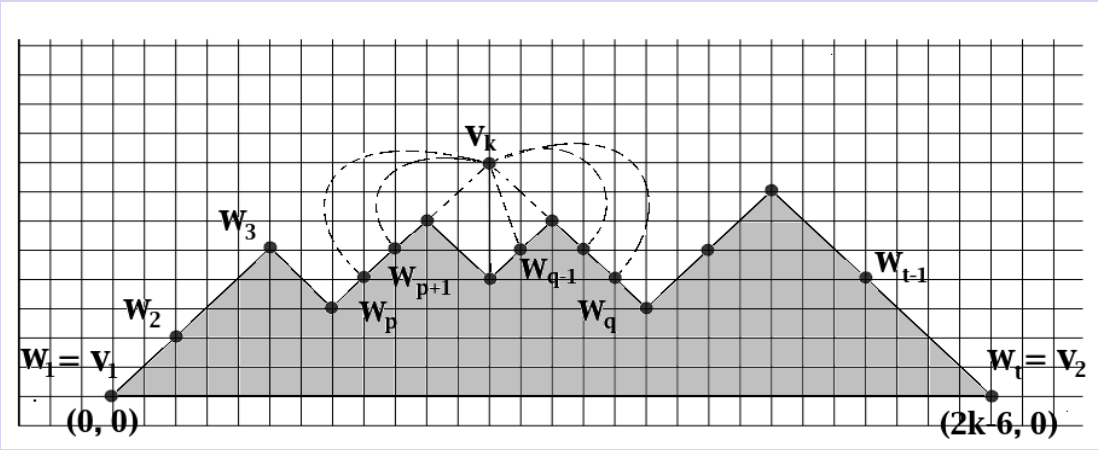
- Placement de G_3: $P(v_1) = (0,0)$, $P(v_2) = (2, 0)$ et $P(v_3) = (1, 1)$

- Si $k\text{-}1 \geq 3$, G_{k-1} doit être positionné de la manière suivante:
  - $P(v_1) = (0,0)$ et $P(v_2) = (2k\text{-}6, 0)$
  - $x(w_1) < x(w_2) < \ldots < x(w_t)$, où $C_o(G_{k-1}) = w_1, w_2, \ldots, w_t$, $w_1 = v_1$ et $w_t = v_2$
  - Chaque arête (w_i, w_{i+1}) sur $C_o(G_{k-1})$ est dessiné par une ligne droite de pente +1 ou -1

Dessin de Graphes Planaires

Algorithme de De Fraissex et al.: Placement de v_k dans G_{k-1}

- Soient w_p, w_{p+1},..., w_q les voisins de v_k dans G_{k-1}
- Idée: placer v_k à l'intersection des droites de pentes +1 et -1 passant respectivement par w_p et w_q

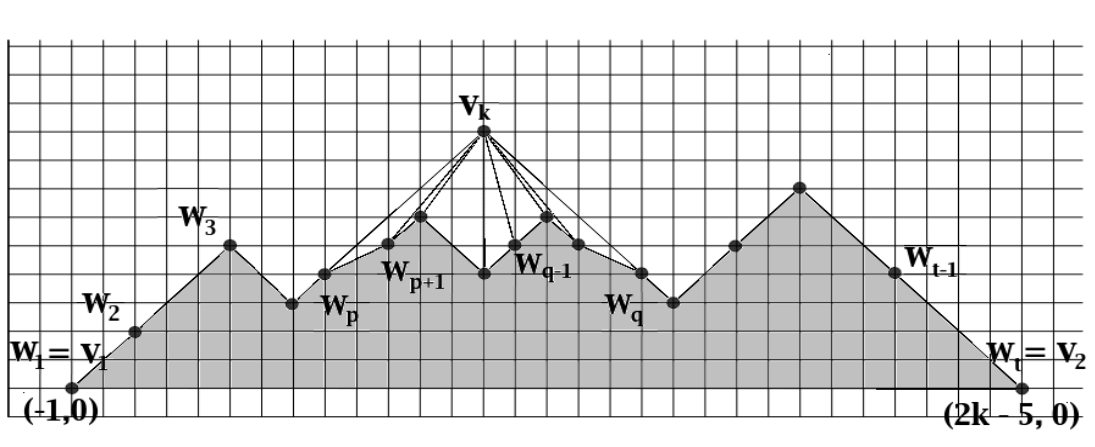


- **Pb : Création de recouvrements arête-sommet/arête-arête**

Dessin de Graphes Planaires

Algorithme de De Fraissex et al.: Placement de v_k dans G_{k-1}

- Suppression des recouvrements par une technique de « Shift »
  - Les sommets $w_1, w_2, \ldots, w_p$ sont décalés de 1 vers la gauche
  - Les sommets $w_q, w_{q+1}, \ldots, w_t$ sont décalés de 1 vers la droite



- Il existe une implémentation linéaire utilisant un arbre comme structure de données pour calculer des coordonnées relatives [CP95]

Dessin de Graphes

Plan

- Introduction
- Dessin de graphes planaires
- Dessin hiérarchique
- Dessin par analogie physique

Dessin de Graphes

Plan

- Introduction
- Dessin de graphes planaires
- Dessin hiérarchique
- Dessin par analogie physique

Dessin Hiérarchique

Principe

Algorithme hiérarchique

- Entrée
  - Un graphe G=(V,E).
- Sortie
  - Un dessin de G où les sommets ont été assignés à différentes niveaux
  - Sans recouvrement sommet-arête

Dessin Hiérarchique

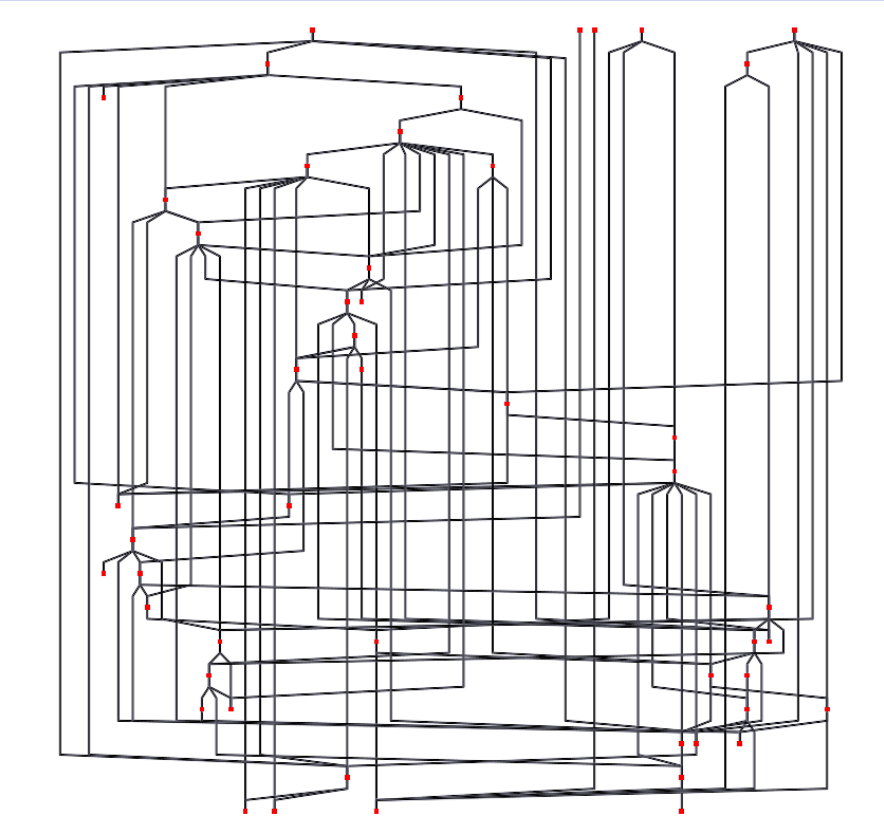
Principe

Algorithme hiérarchique

- Entrée
  - Un graphe G=(V,E).
- Sortie
  - Un dessin de G où les sommets ont été assignés à différents niveaux
  - Sans recouvrement sommet-arête
- Principe
  - Rendre acyclique le graphe (fabrication d'un DAG)
  - Assigner un niveau à chaque sommet
  - Réduction du nombre de croisements d'arêtes
  - Attribuer les abscisses à chaque sommet

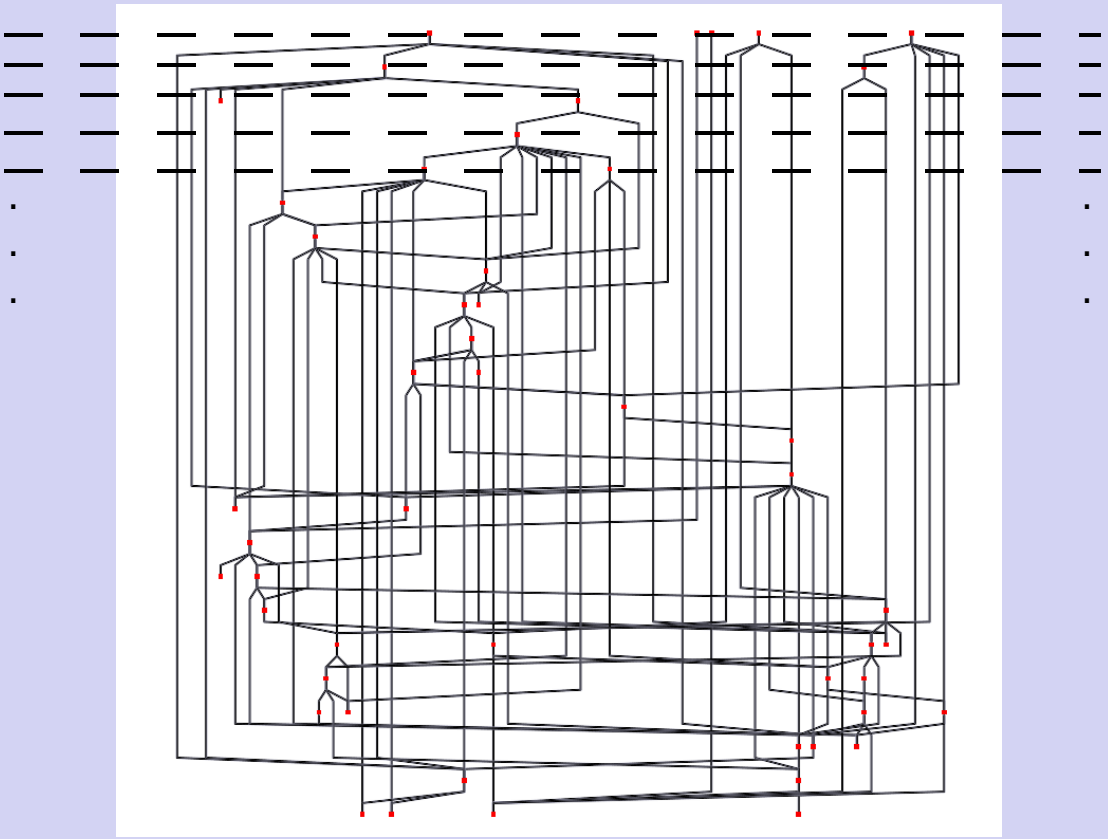
Dessin Hiérarchique

Exemple de dessin hiérarchique



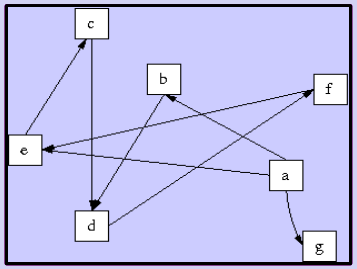
Dessin Hiérarchique

Exemple de dessin hiérarchique



Dessin Hiérarchique

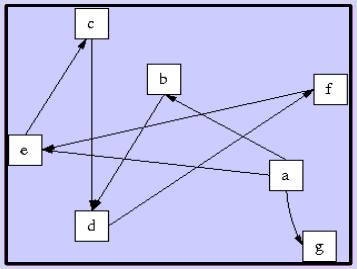
Exemple: Heuristique de Sugiyama et al. [STT81]



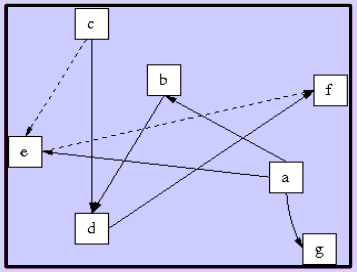
Un graphe à dessiner

Dessin Hiérarchique

Exemple: Heuristique de Sugiyama et al. [STT81]



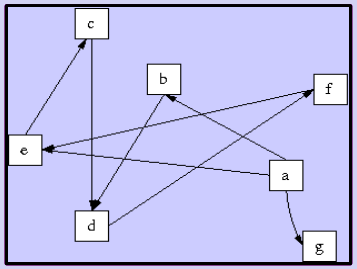
Un graphe à dessiner



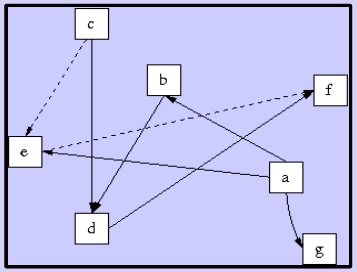
Suppression des cycles

Dessin Hiérarchique

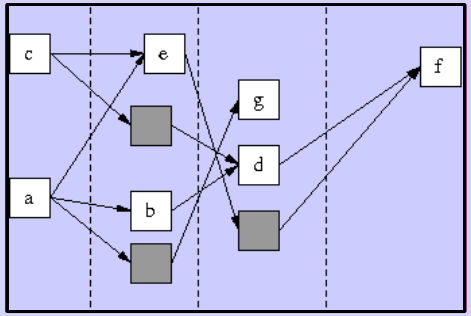
Exemple: Heuristique de Sugiyama et al. [STT81]



Un graphe à dessiner



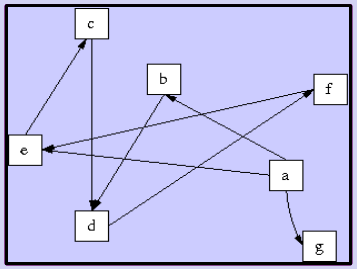
Suppression des cycles



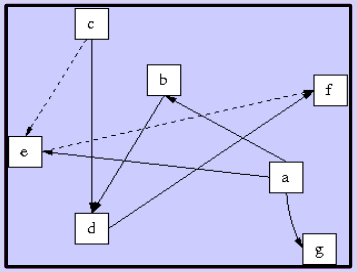
Ordonnancement dans les niveaux
et ajout des sommets virtuels

Dessin Hiérarchique

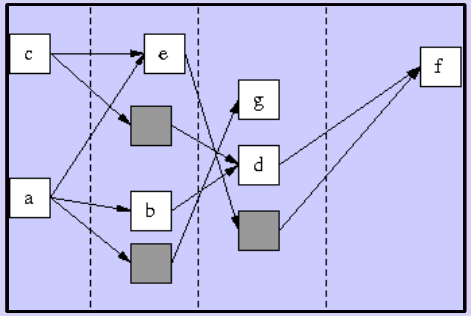
Exemple: Heuristique de Sugiyama et al. [STT81]



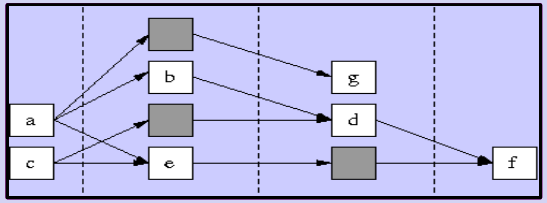
Un graphe à dessiner



Suppression des cycles



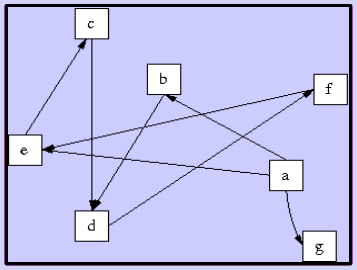
Ordonnancement dans les niveaux et ajout des sommets virtuels



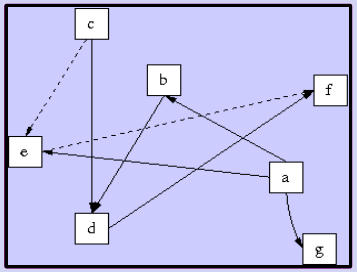
Réduction du nombre de croisements

Dessin Hiérarchique

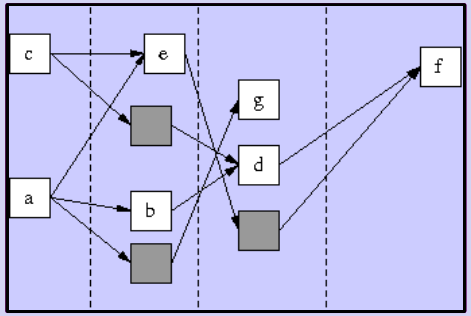
Exemple: Heuristique de Sugiyama et al. [STT81]



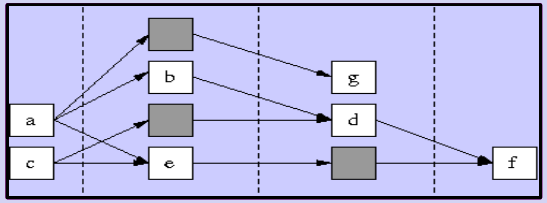
Un graphe à dessiner



Suppression des cycles

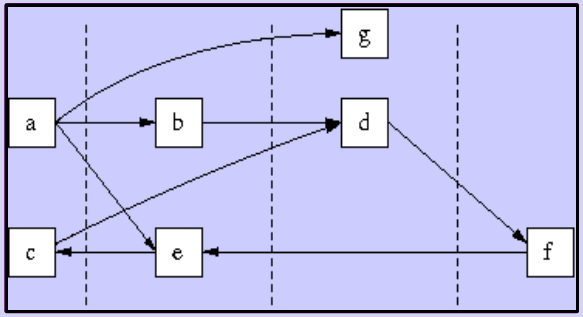


Ordonnancement dans les niveaux et ajout des sommets virtuels



Réduction du nombre de croisements

dessin final : suppression des sommets virtuels, remise des arcs dans le bon sens, optimisation critères esthétiques secondaires



Dessin Hiérarchique

Framework

- Rendre le graphe acyclique (Cycle Removal)
- Décomposer le graphe par niveaux
- Réduire le nombre de croisements
- Affecter les coordonnées horizontale

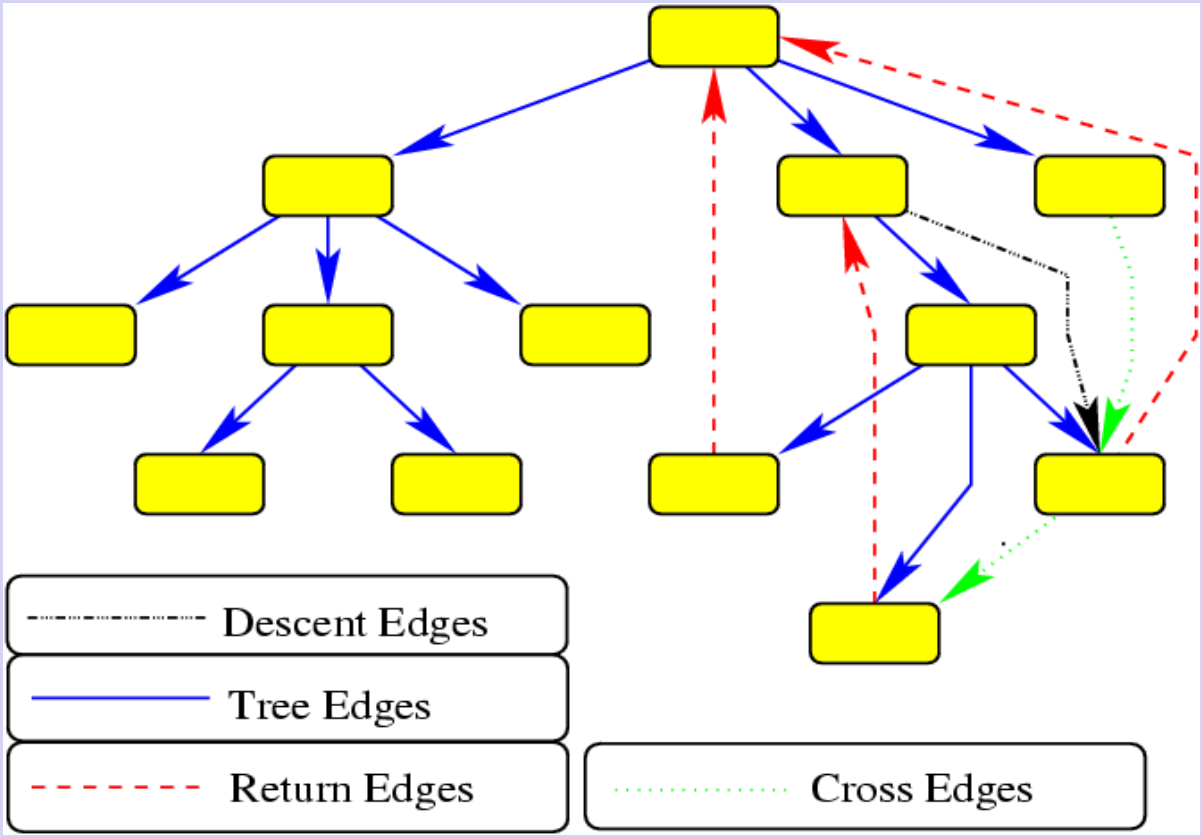
Dessin Hiérarchique

Rendre le graphe acyclique

- Minimiser le nombre d'arêtes à retourner pour rendre le graphe acyclique.
  - Feedback arc set problem (NP-Hard, Karp 72)
- Utilisation d'heuristique pour résoudre le problème
  - Quadri-partition Tarjan (depth first search)
  - Greedy Algorithm
  - Enhanced Greedy Algorithm

Dessin Hiérarchique

Rendre le graphe acyclique: quadri-partition



Dessin Hiérarchique

Rendre le graphe acyclique: greedy algorithm

- Ea = {}
- for all v in V
  - if ($deg^+(v) \geq deg^-(v)$)
    - add $inc^+(v)$ to Ea
  - else
    - add $inc^-(v)$ to Ea
  - remove v from G
- $|Ea| \geq |E|/2$ (Berger and Shor 1990)

Dessin Hiérarchique

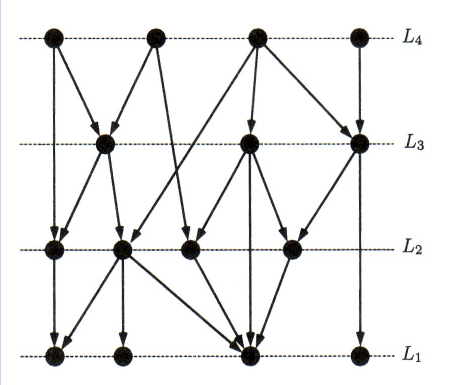
Rendre le graphe acyclique: enhanced greedy algorithm

- Ea={}
- Tant que G est pas vide
  - Tant que G contient un puit v
    - ajoute $inc^-(v)$ à Ea, détruire v
  - Détruire tous les sommets isolés
  - Tant que G contient une source v
    - ajoute $inc^+(v)$ à Ea, détruire v
  - Si G n'est pas vide
    - Soit le sommet v avec la valeur max de $(deg^+(v)-deg^-(v))$
      - Ajoute $inc^+(v)$ à Ea, détruire v
- $|Ea| \geq |E|/2 + |V|/6$ (Eades 1993)
- Amélioration possible en utilisant des composantes fortement connexes

Dessin Hiérarchique

Décomposition par niveaux

- On appel décomposition par niveau toute partition ordonnée P ={p_1,…,p_n} vérifiant la propriété suivante :
  - Pour tout sommets u,v tel que u∈p_i et v∈p_j
    - Si il existe un arc (u,v) alors i<j



Dessin Hiérarchique

Décomposition par niveaux

- Minimisation de la hauteur
  - Placer toutes les sources dans le niveau 1
  - Pour chaque sommet v
    - Calculer le niveau maximum M de ses ancêtres et placer v dans le niveau M+1
- Algorithme linéaire (Mehlhorn 1984)

Dessin Hiérarchique

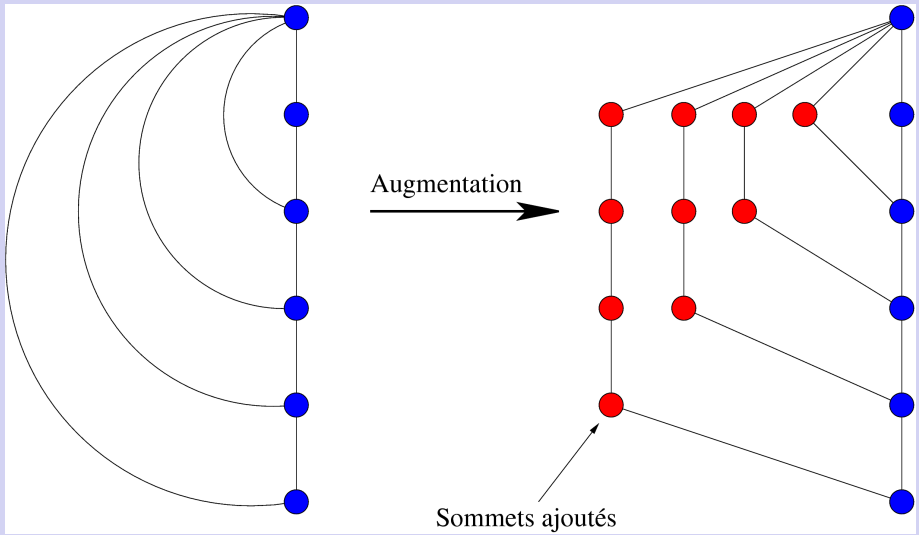
Décomposition par niveaux

- Si l'on veut fixer la taille maximale d'une partition et trouver la hauteur minimal le problème est NP-Hard (Karp, 1972)
- Coffman et Graham 72 ont proposé une heuristique pour résoudre ce problème.
  - Sachant que pour fonctionner l'algorithme à besoin d'un DAG « propre » on ne peut pas réellement fixer la largeur des partitions.

Dessin Hiérarchique

Construction du DAG « propre »

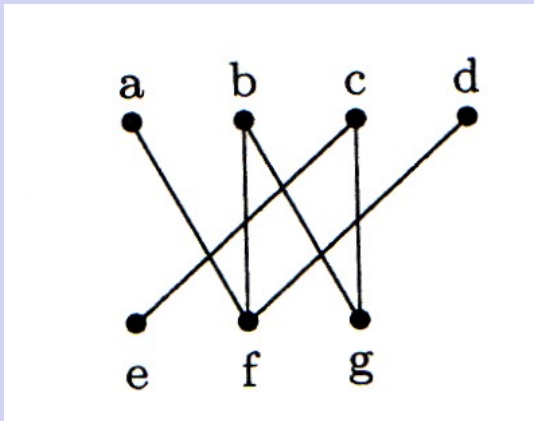
- Une arête ne doit connecter que deux niveaux consécutifs
  - Ajout de sommet et d'arête pour vérifier la propriété



Dessin Hiérarchique

Réduction des croisements

- Minimiser le nombre de croisements revient à déterminer une permutation des sommets de chaque niveau.
- Garey et Johnson ont prouvé que même dans le cas de graphe bipartie (2 niveaux) le problème est NP-HARD.
- Si l'on fixe l'ordre d'un niveau, le problème reste NP-Hard (Eades et Whitesides 1994)



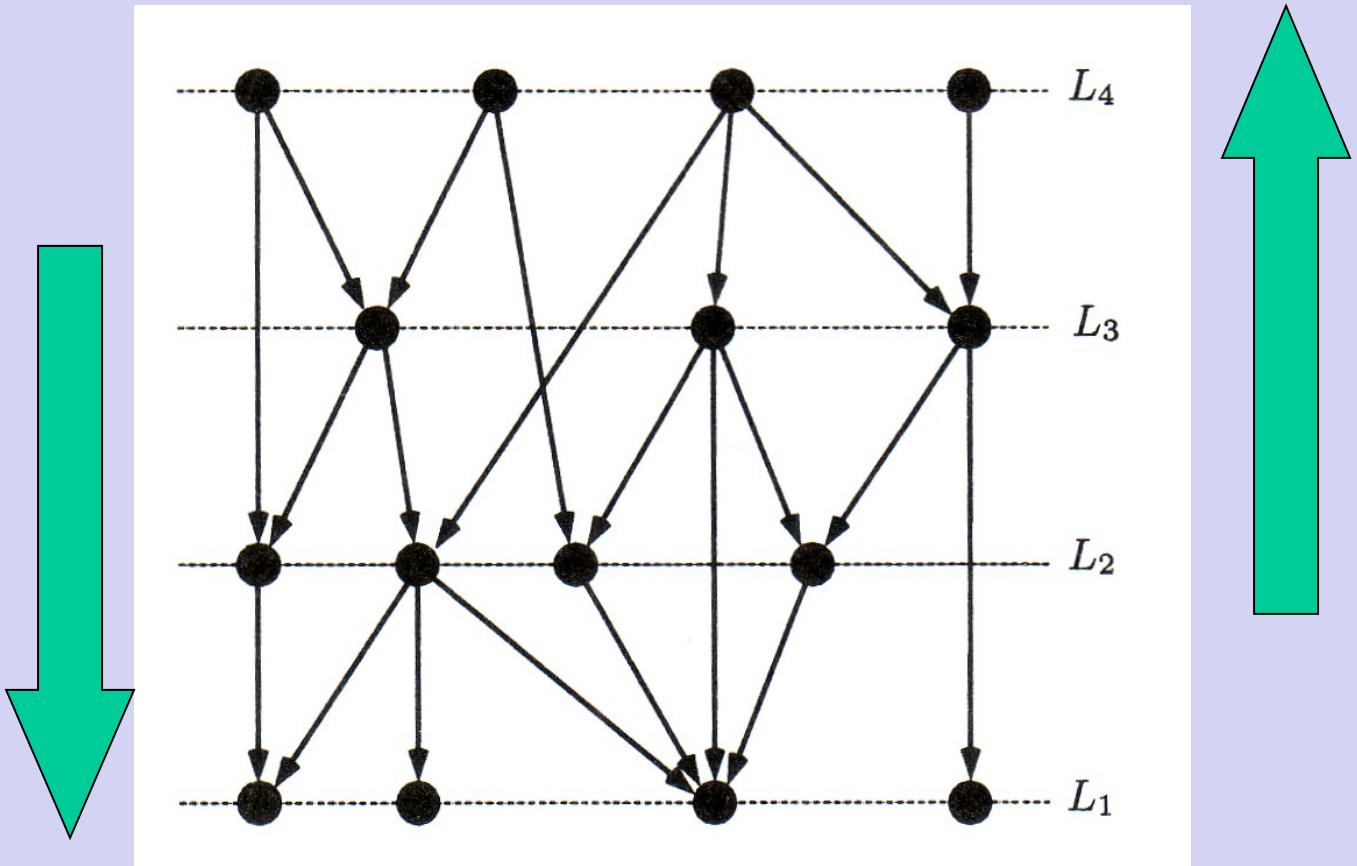
Dessin Hiérarchique

Réduction des croisements: niveau par niveau

- On choisi un ordre initial pour chaque niveau.
- Faire (n fois) (3, 4 itérations sont suffisantes)
  - Pour i allant de 2 à niveau_max
    - Calculer la permutation de p_i qui minimise les croisements entre p_i et p_{i-1}
  - Pour i allant de niveau_max-1 à 1
    - Calculer la permutation de p_i qui minimise les croisements entre p_i et p_{i+1}

Dessin Hiérarchique

Réduction des croisements: niveau par niveau



Dessin Hiérarchique

Réduction des croisements: niveau par niveau

- Algorithme du barycentre (Sugiyama et al.):
  - On place les sommets au barycentre de leurs prédécesseurs.
    - Bary(v)= 1/deg(v) Σ pos(x)
  - L'algorithme est optimal si il existe une solution sans croisement.
  - O(|V|log|V|)

Dessin Hiérarchique

Réduction des croisements: niveau par niveau

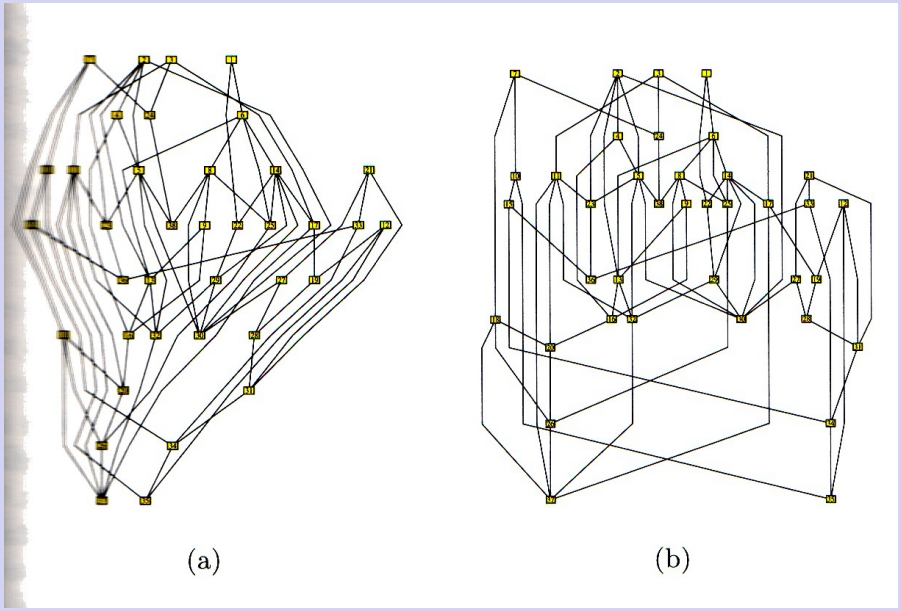
- Algorithme du médian (Eades 94):
  - On place les sommets au médian de leurs prédécesseurs.
    - On ordonne les k sommets en fonction de leur position et on prend la position du k/2 ème sommet
  - L'algorithme est optimal si il existe une solution sans croisement.
  - O(|V|log|V|)

Dessin Hiérarchique

Choix des coordonnées

- On cherche à rendre les arêtes les plus droit

Possible pour éviter l'effet spaghetti.



Dessin Hiérarchique

Choix des coordonnées

- Utilise les méthodes de programmation linéaire
- Trouver une solution
  - Pour minimiser la somme des $\Theta(u,v)|x_u-x_v|$
  - Avec comme contrainte :
    - Ne pas changer l'ordre fixé dans l'étape de réduction de croisement.
    - Les coordonnées doivent être positives.
  - Avec $\Theta(u,v)$:
    - 1 si u et v ne sont pas des sommets ajoutés
    - 2 si u ou v ne sont pas des sommets ajoutés
    - 8 si u et v sont des sommets ajoutés

Dessin Hiérarchique

Choix des coordonnées: Heuristique

- Tant que ?
  - Positionner (Avec barycentre ou médian)
  - Aligner (les sommets ajoutés doivent être alignés)
  - « Compacter »

Dessin Hiérarchique

Complexités

- Sugiyama *et al.* *[STT81]*
  - Temps : $O(|V|.|E|.log|E|)$
  - Mémoire: $O(|V|.|E|)$
- Auber *[A03]*
  - Temps: $O(|V|.|E|.log|E|)$
  - Mémoire: $O(|V|+|E|)$
- Eiglsperger et al. *[ESK04]*:
  - Temps: $O((|V|+|E|)\ log|E|)$
  - Mémoire: $O(|V|+|E|)$

Dessin de Graphes

Plan

- Introduction
- Dessin de graphes planaires
- Dessin hiérarchique
- Dessin par analogie physique

Dessin de Graphes

Plan

- Introduction
- Dessin de graphes planaires
- Dessin hiérarchique
- Dessin par analogie physique

Dessin par analogie physique

Introduction

- Basé sur la simulation de systèmes physiques
- Deux approches principales:
  - Modèle de force
  - Par minimisation du niveau d'énergie
- ≈ duales:
  - Modèle de force:

    déplacement => minimisation niveau d'énergie
  - Minimisation niveau d'énergie:

    minimisation => placement « optimal »

Dessin par analogie physique

Introduction

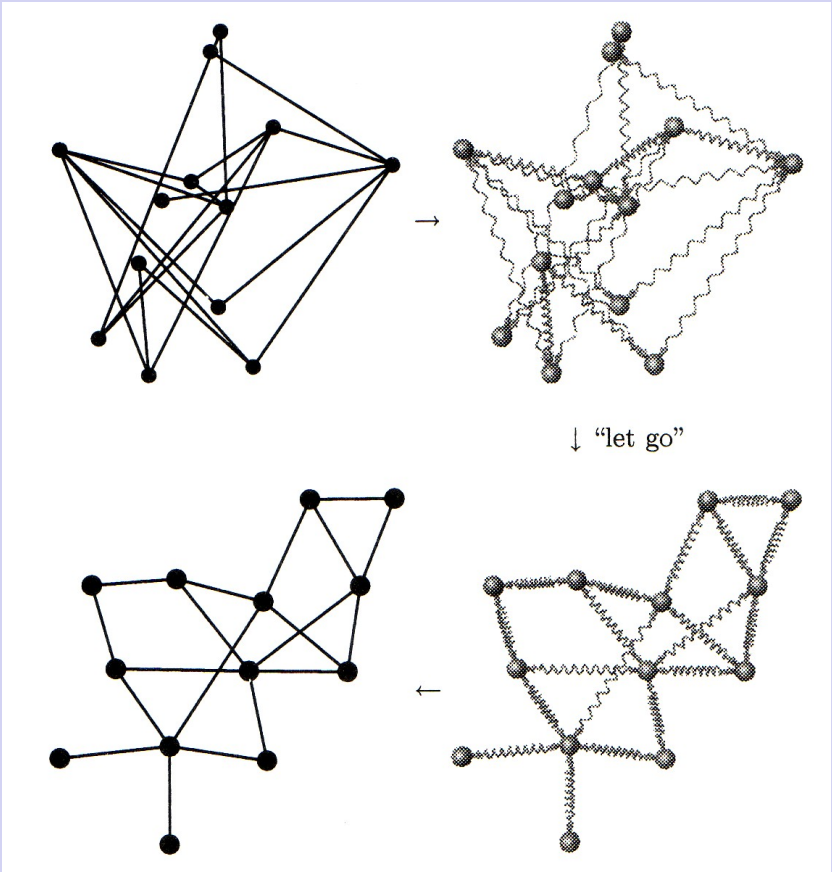
- Basé sur la simulation de systèmes physiques
- Deux approches principales:
  - **Modèle de force**
  - Par minimisation du niveau d'énergie
- ≈ duales:
  - **Modèle de force:**

    déplacement => minimisation niveau d'énergie
  - Minimisation niveau d'énergie:

    minimisation => placement « optimal »

Dessin par analogie physique

Algorithme par modèle de force: principe



Dessin par analogie physique

Algorithme par modèle de force: algorithme général

- Pour chaque itérations :
  - Calculer la force de répulsion (F_{rep}) appliquée à chaque sommet.
  - Calculer la force d'attraction (F_{attr}) appliquée à chaque sommet.
  - Déplacer chaque sommet :
    - $Pos(v) = Pos(v) + k * (F_{rep}(v) + F_{attr}(v))$.

Dessin par analogie physique

Algorithme par modèle de force: Critiques

- **Avantages:**
  - Favorise le critère d'esthétisme de longueur des arêtes uniforme.
  - L'analogie du modèle physique est très facile à comprendre.
  - Implémentation est simple.
  - Pour de petits graphes, les dessins produits sont généralement agréable visuellement.
- **Inconvénients:**
  - Temps de calcul
  - Dépend de la position initiale

Dessin par analogie physique

Algorithme par modèle de force

- Force: ressort électrique vs ressort mécanique
- Approximation de Eades (84)
- Optimisation de Fruchterman and Reingold (91)
- Optimisation de Frick et al (95)

Dessin par analogie physique

Heuristique de Eades 84

- Force d'attraction: $f_{att}(u,v) = c_{att} \cdot \log \frac{dist_{\mathbb{R}}(u,v)}{length} \cdot \overrightarrow{p_u p_v}$
- Force de répulsion: $f_{rep}(u,v) = \frac{c_{rep}}{dist_{\mathbb{R}}(u,v)^2} \cdot \overrightarrow{p_v p_u}$
- Force totale: $f_{totale}(u) = \sum_{e=\{u,v\} \in E} f_{att}(u,v) + \sum_{e=\{u,v\} \notin E} f_{rep}(u,v)$

Dessin par analogie physique

Heuristique de Eades 84

- Force d'attraction: $f_{att}(u,v) = c_{att} \cdot \log \frac{dist_{\mathbb{R}}(u,v)}{length} \cdot \overrightarrow{p_u p_v}$
- Force de répulsion: $f_{rep}(u,v) = \frac{c_{rep}}{dist_{\mathbb{R}}(u,v)^2} \cdot \overrightarrow{p_v p_u}$
- Force totale: $f_{totale}(u) = \sum_{e=\{u,v\} \in E} f_{att}(u,v) + \sum_{e=\{u,v\} \notin E} f_{rep}(u,v)$
- **Pb:** les sommets sont déplacés de manière synchrone

Dessin par analogie physique

Heuristique de Eades 84

- Force d'attraction: $f_{att}(u,v) = c_{att} \cdot \log \frac{dist_{\mathbb{R}}(u,v)}{length} \cdot \overrightarrow{p_u p_v}$
- Force de répulsion: $f_{rep}(u,v) = \frac{c_{rep}}{dist_{\mathbb{R}}(u,v)^2} \cdot \overrightarrow{p_v p_u}$
- Force totale: $f_{totale}(u) = \sum_{e=\{u,v\} \in E} f_{att}(u,v) + \sum_{e=\{u,v\} \notin E} f_{rep}(u,v)$
- **Pb:** les sommets sont déplacés de manière synchrone

=> Limitation du déplacement:

$disp(u) = \delta \cdot f_{totale}(u)$ où $\delta \in [0,1[$ est une constante

Dessin par analogie physique

Heuristique Fruchterman et Reingold 91

- Heuristique pour accélérer la convergence
- Force d'attraction: $$f_{att}(u,v) = \frac{dist_{\mathbb{R}^2}(u,v)^2}{length} \cdot \overrightarrow{p_u p_v}$$
- Force de répulsion: $$f_{rep}(u,v) = s \cdot \frac{length^2}{dist_{\mathbb{R}^2}(u,v)} \cdot \overrightarrow{p_v p_u}$$
- Force totale: $$f_{totale}(u) = \sum_{v \in N(u)} f_{att}(u,v) + \sum_{v \in V, v \neq u} f_{rep}(u,v)$$

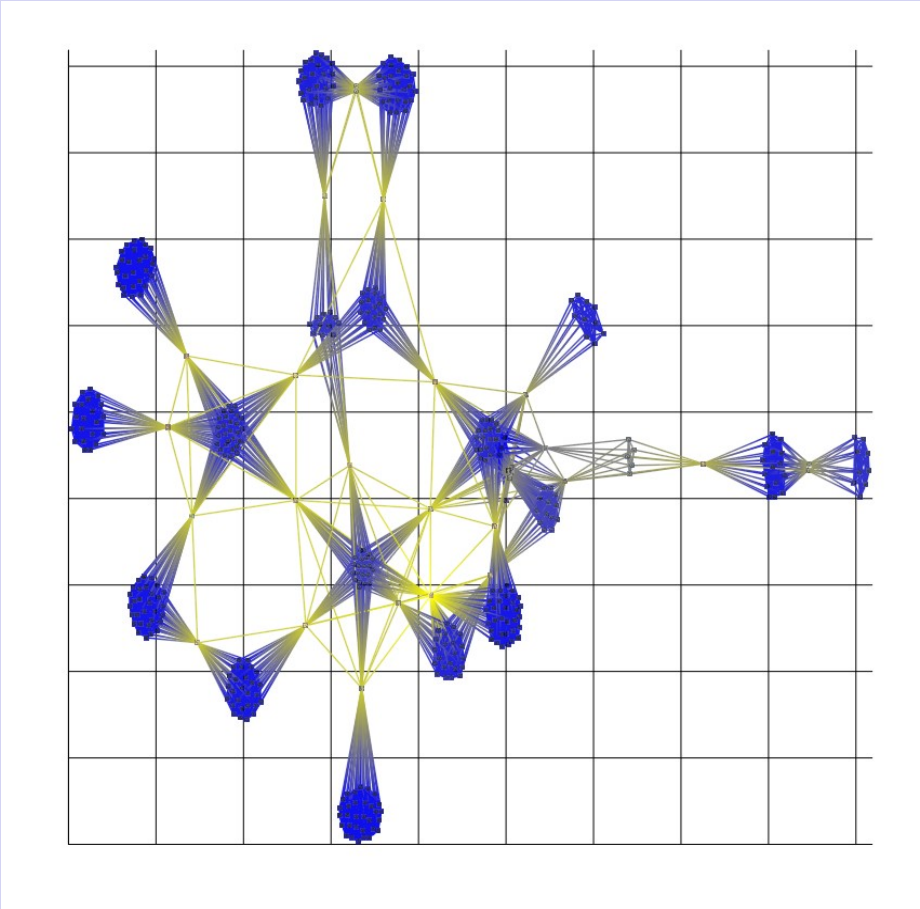
Dessin par analogie physique

Heuristique Fruchterman et Reingold 91

- Heuristique pour accélérer la convergence
- Force d'attraction: $f_{att}(u,v) = \frac{dist_{\mathbb{R}^2}(u,v)^2}{length} \cdot \overrightarrow{p_u p_v}$
- Force de répulsion: $f_{rep}(u,v) = s \cdot \frac{length^2}{dist_{\mathbb{R}^2}(u,v)} \cdot \overrightarrow{p_v p_u}$
- Force totale: $f_{totale}(u) = \sum_{v \in N(u)} f_{att}(u,v) + \sum_{v \in V, v \neq u} f_{rep}(u,v)$
- **Optimisation de la convergence:**
  - δ n'est plus une constante (fct de l'itération)
  - par grille

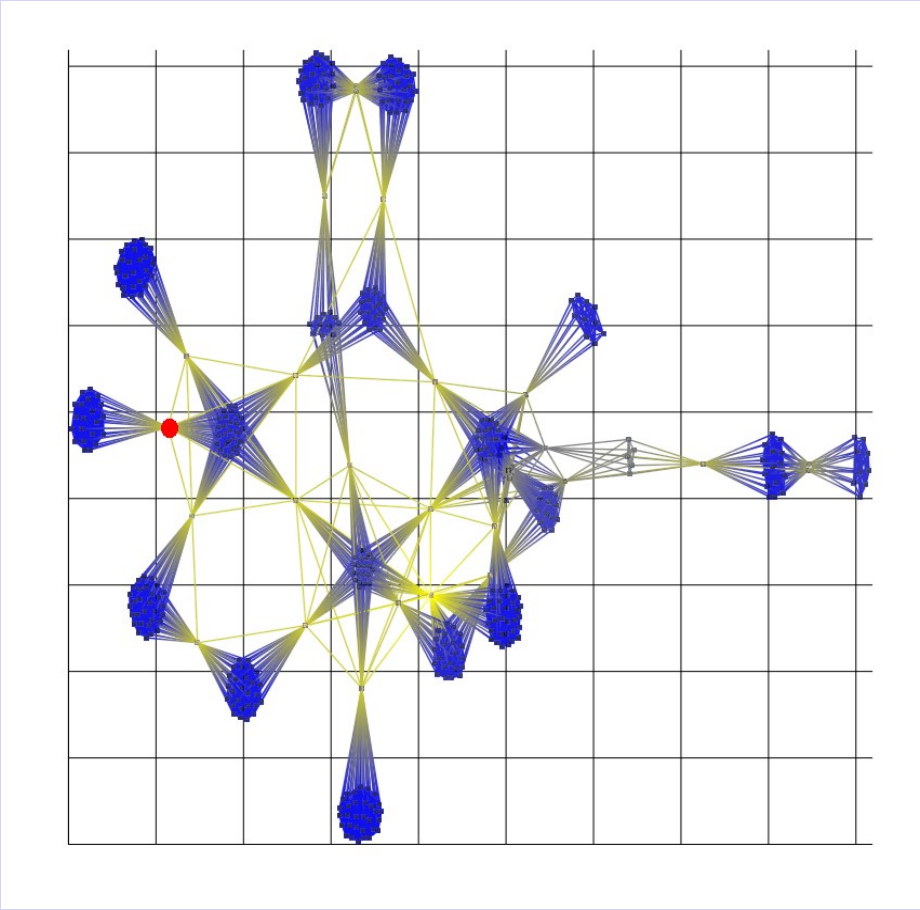
Dessin par analogie physique

Heuristique Fruchterman et Reingold 91



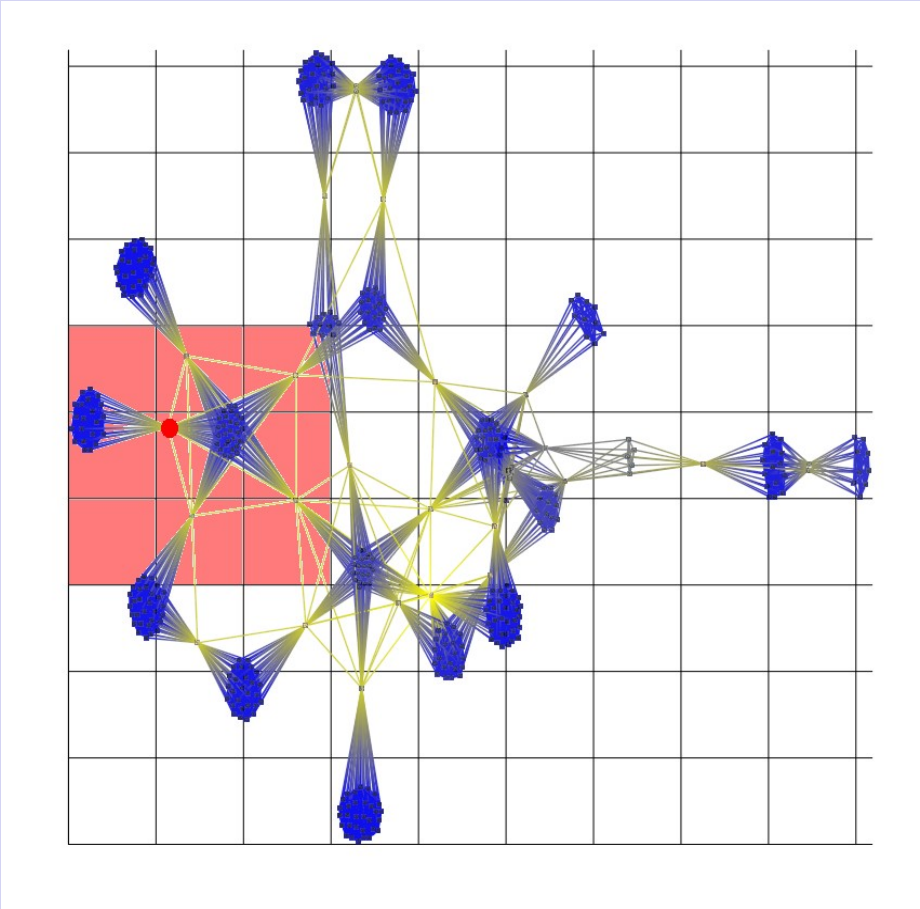
Dessin par analogie physique

Heuristique Fruchterman et Reingold 91



Dessin par analogie physique

Heuristique Fruchterman et Reingold 91



Dessin par analogie physique

Heurisitque de Frick et al. 94

- Heuristique pour accélérer la convergence
- Force d'attraction: $f_{att}(u,v) = \frac{dist_{\mathbb{R}^2}(u,v)^2}{length \cdot \Phi(u)} \cdot \overrightarrow{p_u p_v}$ $\Phi(u) = 1 + \frac{deg_G(u)}{2}$
- Force de répulsion: $f_{rep}(u,v) = \frac{length^2}{dist_{\mathbb{R}^2}(u,v)} \cdot \overrightarrow{p_v p_u}$
- Force totale:

$$f_{totale}(u) = \sum_{v \in N(u)} f_{att}(u,v) + \sum_{v \in V, v \neq u} f_{rep}(u,v) + f_{grav}(u) + f_{rand}(u)$$

- **Autre optimisation**: δ est propre à chaque sommet