

Génération de graphes avec GraphViz

Table des matières

- I. Avant Propos
- II. Présentation de GraphViz
- III. Installation
 - III-A. Installation sous Windows XP
 - III-B. Installation sous Linux RedHat 9
- IV. Formats de sortie
- V. Fichier d'entrée
 - V-A. Graphe non orienté
 - V-B. V-B. Graphe orienté
 - V-C. Souplesse syntaxique
 - V-D. Identifiants
 - V-E. Attributs
 - V-E-1. V-E-1. Propriétés générales
 - V-E-2. Propriétés particulières
 - V-E-3. V-E-3. Principaux attributs
- VI. Interface de commande
 - VI-A. GUI Windows
 - VI-B. En ligne de commande
 - VI-C. Scripting PHP
 - VI-C-1. Bibliothèque PEAR Image_GraphViz
 - VI-C-2. Script PHP personnalisé
 - VI-D. Autres langages
- VII. Les différents générateurs
 - VII-A. Dot
 - VII-B. Neato
 - VII-C. Twopi
- VIII. Conclusion
- IX. Remerciements

Cet article présente la génération de graphes orientés à l'aide de l'outil open-source GraphViz.

Article lu 55952 fois.

L'auteur

Hugo ETIEVANT 

L'article

Publié le 23 janvier 2004 - Mis à jour le 5 janvier 2013

I. Avant Propos

La production de certaines applications exige de pouvoir générer des graphes au sens *recherche opérationnelle* du terme. C'est-à-dire des graphiques représentant des noeuds liés entre eux via des arcs orientés ou non.

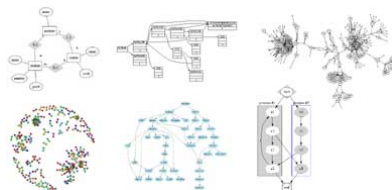
Or la représentation graphique des graphes est un problème algorithmique ardu. La conception d'un programme offrant une telle fonctionnalité est une tâche de longue haleine qui requiert de fortes compétences en mathématiques et algorithmique. Ainsi, il est utile de recourir à un programme externe à qui on délègue la génération des graphes.

II. Présentation de GraphViz

L'application  GraphViz permet de représenter graphiquement des graphes. Elle a été conçue par une équipe des laboratoires de recherche de  AT&T (American Telephone & Telegraph).

Cette application convient à la représentation de graphes très denses comprenant un très grand nombre de nœuds grâce des algorithmes très puissants. Elle est très rapide à l'exécution et le rendu est optimisé afin que les liens ne recouvrent pas les nœuds et qu'ils ne se croisent pas.

De plus, entièrement paramétrable, l'application permet de personnaliser le rendu des graphes par le choix des formes, couleurs et polices de caractères. Le format des fichiers d'entrée est simple et souple. Les formats de sortie sont très variés.



Aperçu de graphes produits par GraphViz

Agrandir cet aperçu...

III. Installation

GraphViz est open source, gratuite et libre de droits. Elle est disponible sous de nombreuses plate-formes dont Linux, Microsoft Windows et MacOS.

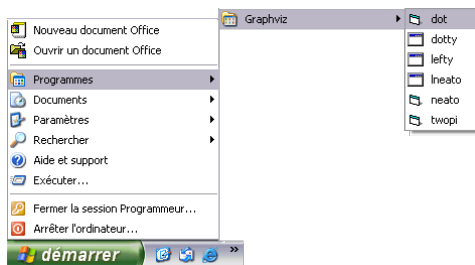
Vous pouvez télécharger les **sources** ou les **binaires** (setup Windows, rpm Redhat ou Debian) de GraphViz sur le site  <http://www.graphviz.org>.

La version utilisée pour cet article est la **1.9**.

III-A. Installation sous Windows XP

Nous téléchargeons le fichier *graphviz-1.9.exe* et exécutons le programme d'installation qui installe l'application GraphViz (bin/), la documentation (doc/) et de nombreux exemples (graphs/) de fichiers d'entrée définissant des graphes orientés (directed/) et non orientés (undirected/).

Le répertoire d'installation est le suivant : C:\Programmes Files\ATT\GraphViz\.



Installation de GraphViz sous Windows XP

L'interface Windows **GVUI.exe** appelle le programme générateur **dot**, **neato** ou **twopi** selon votre choix. Les différences entre ces générateurs seront expliquées plus loin.

III-B. Installation sous Linux ReadHat 9▲

Nous téléchargeons le fichier *graphviz-1.9-0.i386.rpm* et installons le package comme suit :

Sélectionnez

```
1. rpm -i graphviz-1.9-0.i386.rpm
```

La génération de graphe nécessite que des polices de caractères TrueType soient présentes sur le système, pour cela il est nécessaire de copier les fichiers de police correspondants dans un répertoire du système et d'en spécifier le chemin à GraphViz. Nous téléchargeons alors l'archive **FreeType-compatible** dans le thème **Fonts** de la page de téléchargement.

Vérifiez également que les bibliothèques suivantes : **freetype2**, **libjpeg**, **libpng**, **libz** sont bien installées sur votre système.

IV. Formats de sortie▲

Les représentations graphiques produites par GraphViz sont des images stockées dans des fichiers qui peuvent être de formats très divers. Ici ne sont décrits que les principaux formats pouvant être utilisés dans un cadre assez général.

Format	Description
fig	Image vectorielle de l'utilitaire XFig pour Linux
gd	Image de la librairie GD pour PHP
gif	Graphics Interchange Format, format le plus populaire sur le Net
cmap	Description des zones réactives HTML
jpg	Image compressée standard du Net
plain	Description texte du graphe avec les coordonnées des nœuds et arcs
png	Image compressée standard du Net
ps2	PostScript pour PDF (Portable Document Format)
svg	Format vectoriel  SVG d'Adobe

Par exemple, pour l'intégration au sein d'une application web (J2EE, PHP, CGI...), les formats GIF, JPG, PNG et PDF peuvent être utilisés.

- Malheureusement, du fait du brevet sur le format GIF déposé par les sociétés CompuServe Inc. et Unisys Corporation, il n'est plus possible aux développeurs d'utiliser ce format sauf à payer des royalties.
- Le PDF qui est le format de référence pour la communication de documents à travers le Net, n'est pas généré directement par GraphViz mais doit résulter du traitement du format PS2 par un programme de conversion. Pour cela il existe un utilitaire Unix : *ps2pdf* qui s'utilise en ligne de commande suivant la syntaxe : *ps2pdf source.ps destination.pdf*.
- Le format texte PLAIN pourra aisément être utilisé pour générer en Flash via PHP des graphes dynamiques d'un design très évolué. Il peut aussi être utilisé avec les bibliothèques  EzPDF(1) ou  FPDF via PHP pour générer du PDF.
- Le format GD de la librairie  GD communément utilisée en C et en PHP pour la manipulation d'images permet de personnaliser le graphe par l'ajout de titres, légendes, mentions légales, etc. en PHP.
- Le format CMAP associé au GIF, PNG ou JPG permet de créer des zones réactives en HTML afin de rendre le graphe dynamique par l'ajout de liens hypertextes.

V. Fichier d'entrée▲

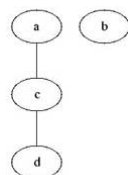
Le fichier d'entrée contient la définition du graphe. C'est un fichier texte qui liste les nœuds et les liens du graphe. Il permet aussi de personnaliser certaines propriétés des nœuds, liens, graphes et sous-graphes.

V-A. Graphe non orienté▲

Exemple :

Sélectionnez

```
1. graph G {
2.     a;
3.     b;
4.     c -- d;
5.     a -- c;
6. }
```



Dessin du graphe non orienté

Cet exemple dessine un graphe non orienté. Le fichier d'entrée commence par la déclaration du type de graphe : **graph** pour graphe non orienté. Puis on nomme notre graphe : **G**. Et enfin, on écrit la liste des nœuds et liens entre

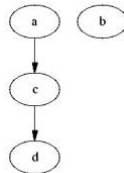
accolades **{ }**. Les liens non orientés sont représentés par deux tirets : **--**. Il n'est pas nécessaire de déclarer au préalable les nœuds avant de les lier entre eux, sauf à devoir en spécifier les attributs d'affichage (couleur, forme...).

V-B. V-B. Graphe orienté▲

Exemple :

Sélectionnez

```
1. digraph G {
2.     a;
3.     b;
4.     c -> d;
5.     a -> c;
6. }
```



Dessin du graphe orienté

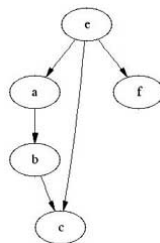
Cet exemple dessine un graphe orienté. La déclaration du type de graphe est différente : **digraph**. Quant aux liens, ils sont représentés par des flèches : **->**.

V-C. Souplesse syntaxique▲

La syntaxe du fichier d'entrée est assez souple, un même graphe peut être écrit de multiples manières. D'autant que certains procédés syntaxiques permettent de gagner beaucoup de temps et de place en terme de quantité de caractères à écrire.

Sélectionnez

```
1. digraph G {
2.     a -> b -> c;
3.     e -> {f; c; a;};
4. }
```



Dessin de graphe orienté

Dans cet exemple, on utilise deux nouvelles syntaxes :

1. liens en **bus**, les nœuds se suivent et chacun est lié à son suivant
2. liens en **étoile**, un nœud est lié à chacun des autres déclarés entre accolades et séparés par un point virgule

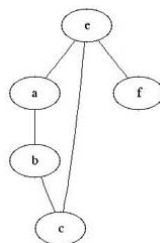
Cette souplesse syntaxique permet de n'écrire en une seule ligne du fichier d'entrée :

1. les longues branches
2. les étoiles denses

La syntaxe est la même pour les graphes non orientés :

Sélectionnez

```
1. graph G {
2.     a -- b -- c ;
3.     e -- {f; c; a;};
4. }
```



Dessin de graphe non orienté

Il suffit de prendre garde à bien définir le bon type de graphe et de liens. Et c'est vraiment très pratique, et surtout plus lisible.

V-D. Identifiants▲

Les identifiants des nœuds sont alphanumériques. Ils peuvent contenir des chiffres, des lettres, des caractères de soulignement. Mais le premier caractère ne doit pas être un chiffre s'il ne constitue pas à lui seul l'identifiant. Les identifiants peuvent contenir des caractères accentués à la condition que l'identifiant soit protégé par des doubles quotes :

```
Sélectionnez

1. digraph G {
2.     y -> 5 -> aa;
3.     foo -> bar;
4.     toto -> "roméo" -> aa;
5. }
```

V-E. Attributs▲

Comme le montrent les exemples précédents, le dessin d'un graphe est par défaut d'aspect plutôt austère. Sachez, que le dessin est entièrement paramétrable : les formes et couleurs peuvent être paramétrées à loisir.

V-E-1. V-E-1. Propriétés générales▲

Les graphes, nœuds et liens ont des propriétés par défaut qui peuvent être redéfinies par des directives placées directement dans le fichier d'entrée (ou bien passées en ligne de commande au programme).

Ces directives affectent les nœuds et liens déclarés après elles. Pour qu'elles s'appliquent à tout le graphe, il faut les placer en début de graphe.

```
Sélectionnez

1. digraph G {
2.     bgcolor=azure;
3.     node [shape=box, color=lightblue2, style=filled];
4.     edge [arrowsize=2, color=gold];
5.     "zero" -> "dix";
6.     "un" -> "dix";
7.     "zero" -> "vingt";
8.     "deux" -> "vingt";
9. }
```

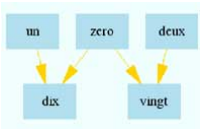


Illustration des propriétés générales

Syntaxe : **objet [attribut=valeur, ...] ;**

Pour les propriétés du graphe, il est inutile de spécifier l'objet puisque l'objet c'est le graphe or le graphe en cours est aussi le contexte en cours. Pour les propriétés des nœuds ou des liens, il faut préciser l'objet "**node**" pour nœud, "**edge**" pour lien (ou arc) et "**graph**" pour le graphe entier (et ses éventuels sous-graphes).

Lorsqu'on veut définir plusieurs propriétés à la fois, on les liste entre crochets, et le séparateur est la virgule.

V-E-2. Propriétés particulières▲

On peut choisir de modifier les propriétés locales d'un nœud ou d'un lien en particulier.

A l'exemple précédent, on a rajouté la couleur "purple" à l'arc entre les nœuds "zéro" et "dix". Et on a également changé la forme, la couleur de fond et la couleur de police du nœud "zéro".

```
Sélectionnez

1. digraph G {
2.     bgcolor=azure;
3.     node [shape=box, color=lightblue2, style=filled];
4.     edge [arrowsize=2, color=gold];
5.     "zero" -> "dix" [color=purple];
6.     "un" -> "dix";
7.     "zero" -> "vingt";
8.     "deux" -> "vingt";
9.     "zero" [shape=circle, color=thistle1, fontcolor=purple];
10. }
```



Illustration des propriétés particulières

Les objets décrits dans le graphe héritent des propriétés générales mais les propriétés particulières sont prioritaires.

V-E-3. V-E-3. Principaux attributs▲

Ces attributs peuvent s'appliquer aux entités suivantes :

- au graphe dans son entier
- aux noeuds sélectionnés
- aux liens (orientés ou non) sélectionnés

Propriété	Description	Type de valeur	Graphe	Noeud	Lien
color	Couleur de tracé	Composantes RVB ou HSV ou nom de la couleur		X	X

arrowsize	Taille de la flèche	entier	X		X
style	Méthode de remplissage	filled		X	
bgcolor	Couleur de fond	Composantes RVB ou HSV ou nom de la couleur	X		
decorate	Trait liant le texte d'un lien au lien	Booléen true ou false			X
dir	Direction d'un lien entre deux noeuds	L'une des 4 valeurs suivantes : forward (du premier vers le deuxième), back (lien inverse), both (double flèche), none (lien non orienté)			X
fixedsize	Force le non dimensionnement automatique d'un noeud en fonction du texte qu'il contient. Ainsi les attributs width et height spécifiés seront obligatoirement respectés.	Booléen		X	
fontcolor	Couleur du texte	Couleur RVB, HSV ou nom	X	X	X
fontname	Nom de la police TrueType	Chaîne de caractères	X	X	X
fontsize	Taille du texte	entier	X	X	X
label	Texte à afficher	Chaîne de caractères	X	X	X
rotate	Rotation en degrés, si vaut 90, alors orientation paysage	entier	X		
shape	Forme du noeud	L'une des valeurs prédéfinies : polygon, box, ellipse, triangle, circle, trapezium, pentagon, hexagon, doublecircle, none...		X	
sides	Nombre de segments, si la forme choisie est : polygon	entier		X	
arrowhead	Forme de la tête d'une flèche	Valeur parmi celles prédéfinies, dont : normal, dot, none, empty, diamond, box, open			X
arrowtail	Forme de la queue d'une flèche	Valeur parmi celles prédéfinies, voir ci-dessus			X
arrowsize	Epaisseur d'une flèche	réel			X
size	Taille du dessin généré, en pouces	réel	X		

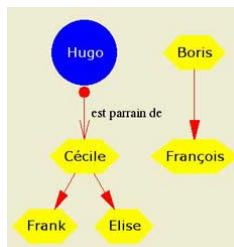
Exemple :

Sélectionnez

```

1. digraph G {
2.   graph [bgcolor=lightyellow2, splines=true];
3.   edge [color=red, arrowsize=2];
4.   node [color=yellow, style=filled, shape=polygon, sides=6, fontname="Verdana"];
5.   A1 -> A3 [label="est parrain de", arrowtail=dot, arrowhead=open];
6.   A3 -> A5;
7.   A3 -> A6;
8.   A2 -> A4;
9.   A1 [label="Hugo", shape=circle, color=blue, fontcolor=white];
10.  A2 [label="Boris"];
11.  A3 [label="Cécile"];
12.  A4 [label="François"];
13.  A5 [label="Frank"];
14.  A6 [label="Elise"];
15. }

```



Dessin du graphe avec quelques attributs de mise en forme

Pour la liste exhaustive des attributs et valeurs d'attributs possibles, reportez-vous à la documentation en ligne de GraphViz sur le site officiel.

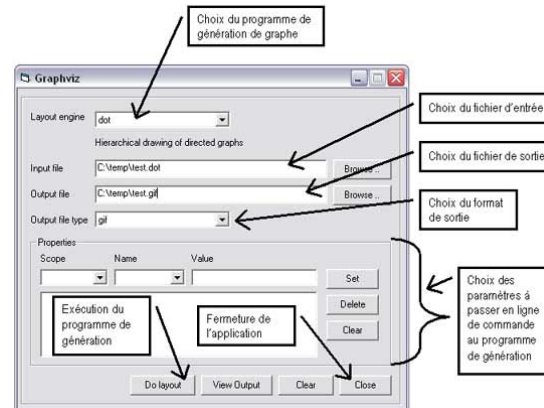
VI. Interface de commande▲

L'application GraphViz peut être utilisée de multiple manières : via une interface utilisateur ergonomique, en ligne de commande depuis le shell ou d'une autre application. Il suffit de lui passer en paramètre le fichier de description textuelle du graphe ainsi que le choix du format de sortie.

VI-A. GUI Windows▲

L'interface Windows est d'utilisation intuitive :

1. on choisit le générateur parmi les 3 disponibles
2. on sélectionne le fichier de description du graphe
3. on saisit le nom du fichier de sortie à générer
4. ainsi que le type de sortie
5. on peut éventuellement spécifier une liste d'arguments à passer au programme
6. et enfin on génère le graphe (*Do Layout*) qu'on peut par la suite visualiser (*View Output*)
7. on pourra remettre à zéro les paramètres de l'interface (*Clear*) ou fermer (*Close*)



GUI Windows

VI-B. En ligne de commande▲

GraphViz ne possède pas d'interface graphique sous Linux, il est utilisé en ligne de commande en passant en paramètre le nom du fichier d'entrée contenant la déclaration du graphe ou bien par l'écriture dans le flux d'entrée de cette déclaration.

Syntaxe (où **cmd** correspond à l'un des générateurs : **dot**, **neato**, **twopi**) :

Sélectionnez

- ```
1.cmd [flags] [input files]
```

Exemple produisant une image JPEG à partir du fichier d'entrée **monGraphe.dot** :

Sélectionnez

- ```
1. dot -Tjpg -omonImage.jpg monGraphe.dot
```

VI-C. Scripting PHP▲

VI-C-1. Bibliothèque PEAR Image_GraphViz▲

L'interface PHP du PEAR Project : la classe Image_GraphViz de Sebastian Bergmann permet de faire le lien entre un script PHP et le programme GraphViz installé sur le serveur web.

Il existe des méthodes pour déclarer itérativement les éléments (noeuds et arcs) du graphe : `addNode()`, `addEdge()`... ou encore pour charger un fichier d'entrée : `load()` contenant la description du graphe au format natif de GraphViz.

Sélectionnez

- ```

1.Exemple d'utilisation de la classe Image_GraphViz
2.// inclusion de la classe Image_GraphViz
3.require_once 'Image/GraphViz.php';
4.// création de l'objet graphe
5.$graph = new Image_GraphViz();
6.// ajout des noeuds du graphe
7.$graph->addNode('Node1', array('URL' => 'http://link1',
8. 'label' => 'This is a label',
9. 'shape' => 'box'
10.));
11.);
12.$graph->addNode('Node2', array('URL' => 'http://link2',
13. 'fontsize' => '14'
14.));
15.);
16.$graph->addNode('Node3', array('URL' => 'http://link3',
17. 'fontsize' => '20'
18.));
19.);
20.// ajout des arcs entre les noeuds et spécification de quelques attributs
21.$graph->addEdge(array('Node1' => 'Node2'), array('label' => 'Edge Label'));
22.$graph->addEdge(array('Node1' => 'Node3'), array('color' => 'red'));
23.// création de l'image de sortie représentant le graphe

```

```
24. $graph->image();
```

### VI-C-2. Script PHP personnalisé▲

Si vous désirez ne pas utiliser le package **Image\_GraphViz** de PEAR, vous pouvez créer très simplement un script réalisant la même tâche. Il suffit pour cela d'écrire le fichier d'entrée (*fopen()*, *fputs()*) que vous passez en paramètre au programme GraphViz préalablement installé sur le serveur web (*exec()*), puis vous affichez l'image ainsi générée.

Prenons l'exemple d'un graphe représentant les liens de parrainage entre les membres de la rédaction de Developpez.com : une table d'une base de données MySQL liste ces liens, à un parrain est associé son filleul. Un parrain pouvant avoir plusieurs filleuls, et un filleul n'ayant qu'un seul parrain.

Exemple de génération d'un graphe

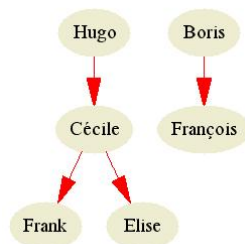
Sélectionnez

```
1. <?php
2. $type = "jpg";
3. $file = "monGraphe";
4. if($fp = fopen($file.'.dot', "w")) {
5. fputs("digraph G {");
6. fputs("\t edge [color=red, arrowsize=2];");
7. fputs("\t node [color=lightyellow2, style=filled];");
8. if($result = mysql_query("SELECT * FROM parrainage")) {
9. while($ligne = mysql_fetch_array($result)) {
10. fputs("P" . $ligne['parrain_id'] . " -> F" . $ligne['filleul_id'] . " ");
11. fputs("P" . $ligne['parrain_id'] . " [label=\"" . $ligne['parrain_nom'] . "\"];");
12. fputs("F" . $ligne['filleul_id'] . " [label=\"" . $ligne['filleul_nom'] . "\"];");
13. }
14. }
15. fputs("}");
16. fclose($fp);
17. exec("dot -T$type $file.dot");
18. echo "";
19. }
20. ?>
```

Exemple de fichier d'entrée généré

Sélectionnez

```
1. digraph G {
2. edge [color=red, arrowsize=2];
3. node [color=lightyellow2, style=filled];
4. A1 -> A3;
5. A3 -> A5;
6. A3 -> A6;
7. A2 -> A4;
8. A1 [label="Hugo"];
9. A2 [label="Boris"];
10. A3 [label="Cécile"];
11. A4 [label="François"];
12. A5 [label="Frank"];
13. A6 [label="Elise"];
14. }
```



Exemple de graphe de parrainage généré

### VI-D. Autres langages▲

De la même manière, avec tout autre langage (Java, Perl, ASP.NET, etc.) nous pouvons réaliser un fichier d'entrée, le soumettre au générateur de GraphViz puis en exploiter le fichier de sortie pour l'afficher directement ou l'analyser pour en générer une représentation dans un autre format (Flash ou autre...).

### VII. Les différents générateurs▲

Selon le programme sous-jacent de génération du graphe, on obtient des dessins différents à partir du même fichier d'entrée de description du graphe.

Exemple :

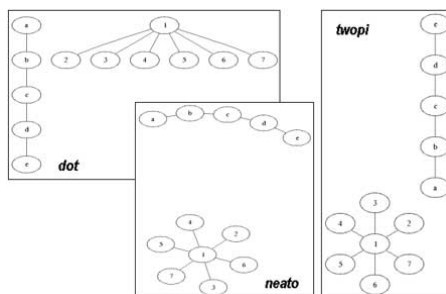
Sélectionnez

```
1. graph G {
2. a -- b -- c -- d -- e;
```

```

3. 1 -- {2; 3; 4; 5; 6; 7;}
4. }

```



Différences de dessin entre générateurs

### VII-A. Dot▲

Le programme **dot** dessine les graphes orientés comme des hiérarchies en les représentant par niveaux en fonction du nombre de prédécesseurs de chacun des nœuds.

Il procède en quatre étapes :

1. suppression des cycles
2. assignation d'un rang à chacun des nœuds en fonction de la longueur du plus court chemin le séparant du nœud de départ, en vue de déterminer leur position sur l'axe des Y
3. ordonnancement des nœuds en fonction de leur rang
4. détermination de leur position sur l'axe des X afin de minimiser la longueur des arcs dans le dessin

Les dessins ont ainsi l'aspect particulier des arbres. Il en découle pour les graphes denses des arcs particulièrement "tordus" calculés par des splines aux angles parfois aigus. Il est adapté à la représentation d'organigrammes, des dépendances dans un programme...

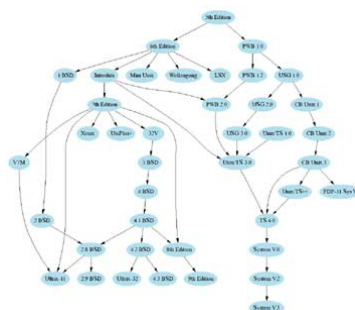


Image générée avec Dot

### VII-B. Neato▲

Le programme **neato** trace des graphes non orientés. Il se base sur le modèle de dot mais utilise des heuristiques particulières afin de tracer des graphes dont la configuration est de moindre énergie. On retrouvera donc des tracés en forme de cercle avec un centre de gravité central et une répartition homogène des nœuds tout autour de ce centre. Il s'attache à ce que les distances des arcs dans le dessin soit le plus proche de celui dans la représentation théorique. Ainsi, un arbre très dense sera représenté comme une fractale dont le centre est le sommet de l'arbre et les branches se déploient tout autour.

Il est particulièrement adapté à la représentation de systèmes de télécommunication réseaux, à l'ingénierie logicielle...

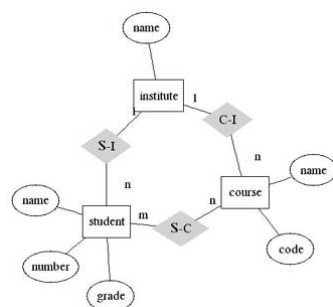


Image générée avec Neato

### VII-C. Twopi▲

Le programme **twopi** trace des graphes en affichant les étoiles sous forme de cercle contrairement à dot. De plus il gère autant les non orientés que les orientés contrairement à neato.

### VIII. Conclusion▲

Nous avons pu constater la souplesse et l'efficacité de l'outil GraphViz. Sa documentation utilisateur est assez dense, et les articles techniques sur les algorithmes employés sont également en nombre sur le site officiel. Cet outil fait l'objet de mises à jour régulières (on est passé de la 1.3 à la 1.9 en 10 mois) et est développé par une équipe d'experts. Il a même été adopté par l'outil de documentation de sources [Doxygen\(2\)](#) pour générer les graphes de dépendance. En outre il s'interface à merveille avec tout logiciel. Et la diversité des formats de sortie est un gage de souplesse et de portabilité.

C'est donc l'outil externe idéal à utiliser pour le dessin de graphes.

### IX. Remerciements▲

Je remercie vivement Bestiol pour sa relecture et ses critiques constructives qui m'ont permis d'améliorer la qualité

de cet article.

---

(1)  
La librairie EzPDF fait l'objet d'un tutorial complet sur Developpez.com :  Création de documents PDF avec la classe EZPDF de R&OS Ltd de Hugo Etiévant

(2)  
L'outil Doxygen fait l'objet d'un tutorial sur Developpez.com :  Documenter un projet avec Doxygen 1.3.8 par Hugo Etiévant

---

Copyright © 2004 Hugo ETIEVANT. Aucune reproduction, même partielle, ne peut être faite de ce site et de l'ensemble de son contenu : textes, documents, images, etc. sans l'autorisation expresse de l'auteur. Sinon vous encourez selon la loi jusqu'à trois ans de prison et jusqu'à 300 000 € de dommages et intérêts.

---

Contactez le responsable de la rubrique PHP

---

[Nous contacter](#)   [Participez](#)   [Hébergement](#)   [Informations légales](#)  
Copyright © 2000-2017 - [www.developpez.com](http://www.developpez.com)