Travaux de voirie

Samuel ARSAC

Niveau : première
Durée envisagée : 1h/1h30

Table des matières

Présentation de l'activité

Le problème

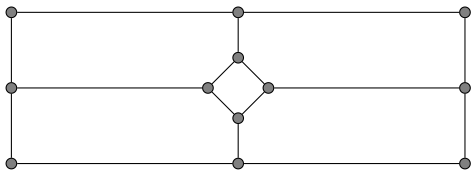
Il s'agit de résoudre le problème suivant : vous devez refaire les canalisations d'un quartier, mais par souci d'optimisation, vous souhaitez éviter à tout prix de devoir transporter votre matériel inutilement, notamment en repassant par une rue que vous avez déjà refaite. Difficulté supplémentaire : le quartier est entièrement piéton, vous ne pouvez charger et décharger votre matériel qu'en un point donné, qui sera donc votre point de départ et d'arrivée.

L'activité

Les élèves vont donc tenter de trouver de tels itinéraires sur des plans variés. Il existe une condition sans laquelle ces itinéraires ne peuvent exister, l'objectif est de les amener à déterminer cette condition. Enfin, ils pourront essayer de trouver un algorithme cherchant ces itinéraires, et appliquer les algorithmes des autres élèves. Ces recherches pourront se faire par groupes sur les tables ou, si le nombre d'élèves le permet, en se déplaçant sur un plan matérialisé au sol.

Notions d'informatique abordées

Cette activité est directement inspirée du problème des sept ponts de Königsberg, résolu par Leonhard Euler au XVIIIe siècle, donnant naissance à la théorie des graphes. Les graphes peuvent être utilisés pour représenter toute situation dans laquelle on trouve des éléments liés entre eux par une quelconque relation, comme c'est le cas dans notre problème, les intersections étant reliées par les rues. La dernière partie de l'activité est axée sur les algorithmes de recherche de cycles eulériens dans un graphe.



Graphe d'une partie des rues devant l'ENS à Cachan.
Existe-t-il une solution au problème pour ce cas ?

Fiche d'activité

Durée approximative	Étape	Détails	Matériel
2 min	Présentation de mon parcours		
5 min	Présentation de l'activité	Introduction au problème : trouver un itinéraire dans les rues d'un quartier empruntant chaque rue exactement une fois et retournant au point de départ. On peut prendre l'exemple d'un facteur voulant optimiser sa tournée, des éboueurs, ou une entreprise devant faire des travaux de voirie. Explication du déroulement de l'activité et des objectifs. — Trouver une propriété des graphes permettant de savoir si un tel itinéraire est possible ou non. — Dans le cas où un tel itinéraire existe, trouver un algorithme permettant de le calculer.	
15-20 min	Essais et recherche	Division en groupes de 2 ou 3. Les élèves essaient de trouver des solutions sur des graphes arbitraires pour se familiariser avec le problème. Ils ont donc à leur disposition des fiches sur lesquelles ils peuvent tracer un itinéraire, avec un point de départ indiqué. Ils tentent ensuite de trouver la condition d'existence d'une solution, avec plusieurs fiches différentes (s'assurer que chaque groupe possède au moins une fiche avec solution, et une fiche sans solution).	Fiches graphes et feutres
5-10 min	Discussion sur la condition	Si des élèves ont trouvé qu'il fallait qu'un nombre pair de rues parte de chaque intersection, ils peuvent l'expliquer aux autres, sinon des indications peuvent être données pour les y amener. — Leur conseiller de regarder ce qui se passe au niveau des nœuds lorsqu'ils ne trouvent pas d'itinéraire. — Que se passe-t-il si on a un nœud connecté à une seule arête non visitée ? Que peut-on en déduire pour un nœud à trois arêtes, après un premier passage par celui-ci ? Si aucun élève ne trouve la solution dans les dix minutes, elle peut être donnée pour passer à la suite. Au contraire, si la réponse est trouvée rapidement, on peut passer quelques minutes sur la même question dans le cas où l'on ne retourne pas au point de départ (voir compléments).	

10 min	Présentation de la deuxième partie	Explications pour la deuxième partie de l'activité : trouver un algorithme de recherche pour ce problème, dans un graphe vérifiant la propriété précédemment trouvée. Pour éviter de détailler les possibles représentations des graphes, on peut autoriser certaines opérations à utiliser comme des « boîtes noires ». — Obtension de la liste des voisins d'un nœud. — Itération sur les voisins (dans un ordre a priori inconnu). — Coloration : les arêtes peuvent être colorées pour conserver des informations les concernant. — Et autres, si celles-ci ne font pas d'opérations trop complexes.	
15-20 min	Réflexion	Temps de réflexion sur l'algorithme par groupes de 4 à 6 élèves, avec assistance individuelle. Indications pour aider les élèves à avancer : — Commencer par voir comment trouver un cycle ne passant qu'au plus une fois par chaque arête. — Une fois ce cycle trouvé, peut-on en trouver un autre ? — Comment faire se rejoindre les deux cycles trouvés ? Peut-on construire le second de façon à ce que ce soit facile à faire ? — Comment itérer ensuite ? — Quand arrêter l'algorithme ? Ces indications doivent être limitées aux groupes n'ayant pas d'idées, il est intéressant de voir quels algorithmes les élèves créent par eux-mêmes, même si ceux-ci ne sont pas corrects. Les élèves ayant une idée sont encouragés à rédiger leur algorithme en pseudo-code, ou du moins de détailler les étapes.	Fiches graphes et feutres
10 min	Discussion sur les algorithmes	L'algorithme n'étant pas facile à trouver, l'idée est surtout de rechercher des idées et de voir comment construire un algorithme sur un graphe. On peut enfin donner l'algorithme de Hierholzer pour ce problème, qui se trouve en annexe. Leur donner le temps d'essayer les algorithmes allongeant trop l'activité, on peut encourager les élèves à garder les graphes et à appliquer les algorithmes plus tard.	De même que précédemment avec éventuellement un tableau.

5-10 min	Conclusion	Conclusion de l'activité. Ce type de cycle se nomme un cycle eulérien. Quelques mots sur d'autres problèmes sur des graphes : le problème des ponts de Königsberg, le problème des chemins hamiltoniens, le problème des quatre couleurs, le problème du voyageur de commerce... Applications des problèmes sur les graphes, recherche d'itinéraires par exemple.	

Compléments

Sans retour au départ

Lorsque l'on enlève la condition de retour au départ, on passe d'un cycle à un chemin. Il existe également une condition nécessaire et suffisante à l'existence d'un tel chemin.
En effet, un chemin eulérien existe si et seulement si le graphe contient au plus deux nœuds connectés à un nombre impair d'arêtes.

— Si le graphe n'en contient aucun, alors ces chemins sont plus particulièrement des cycles.
— Il est impossible que le graphe ne contienne qu'un seul nœud de ce type.
— Si deux nœuds possèdent un nombre impair d'arêtes incidentes, alors ceux-ci doivent nécessairement être les extrêmités du chemin.

Algorithme de Hierholzer

— À partir du nœud de départ, parcourir les arêtes du graphes jusqu'à retourner au départ (en colorant les arêtes visitées pour ne pas les emprunter plusieurs fois).
  On peut noter qu'il est toujours possible de faire cela, puisque le nombre d'arêtes sortant d'un nœud est pair.
  On obtient ainsi un premier cycle.
— Si l'un des nœuds du cycle est connecté à des arêtes non encore visitées, on répète la première étape à partir de ce nœud, et on fusionne les deux cycles obtenus.
  Cette fusion se fait de la façon suivante : si on voit un cycle comme une suite d'arêtes, le nœud en question se situe entre deux arêtes consécutives de la suite correspondant au premier cycle. On peut alors intercaler le second cycle entre ces deux arêtes.
— On répète la deuxième étape tant que c'est possible (tant qu'il reste des arêtes non visitées).

Voici une version pseudo-code suivie une application sur un graphe.

Dans le pseudo-code on représente les arêtes comme des couples de nœuds, que l'on peut colorer (blanc pour non visitée, noir pour visitée). On a accès à Couleur, qui enregistre les couleurs des arêtes. L'opérateur :: permet d'ajouter un élément au début d'une liste.
Quelques fonctions purement techniques sont utilisées sans écrire leur code :

— ListeVoisins(s), qui renvoie l'ensemble des voisins de s dans le graphe.
— Voisinblanc(s), qui renvoie vrai si s est connecté à une arête blanche, et faux sinon.
— Fusion(c1, c2), qui fusionne les chemins c1 et c2.

Je détaille la fonction Voisin, qui prend un nœud en argument, renvoie un de ses voisins connecté par une arête blanche, et noircit cette dernière.

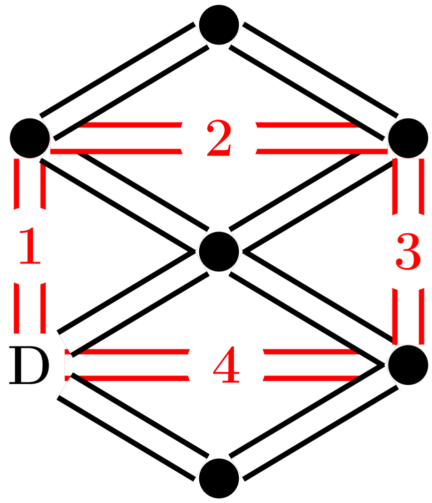
début
  pour u dans ListeVoisins(s) **faire**
    si Couleur.((s,u)) $= blanc$ **alors**
      Couleur.((s,u)) := noir;
      retourner (u)
    fin
  fin
fin

Algorithme 1 : Voisin(s)

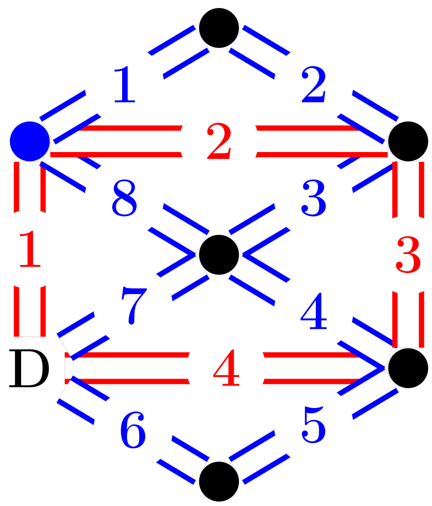
Données : (S,A) le graphe (sommets, arêtes), s le point de départ
début
  pour a dans A **faire**
    Couleur.(a) := blanc
  fin
  chemin := [s];
  tant que Il existe a dans A tel que Couleur.(a) := blanc **faire**
    Choisir d dans chemin tel que Voisinblanc(d);
    cheminlocal := [d];
    actif := Voisin(d);
    tant que actif $\neq d$ **faire**
      cheminlocal := actif :: cheminlocal;
      actif := Voisin(actif);
    fin
    chemin := Fusion(chemin, cheminlocal);
  fin
  retourner chemin
fin

Algorithme 2 : Algorithme de Hierholzer

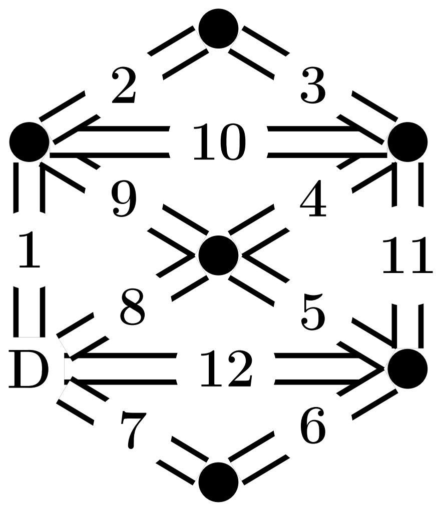
Application de l'algorithme de Hierholzer



Premier cycle



Second cycle



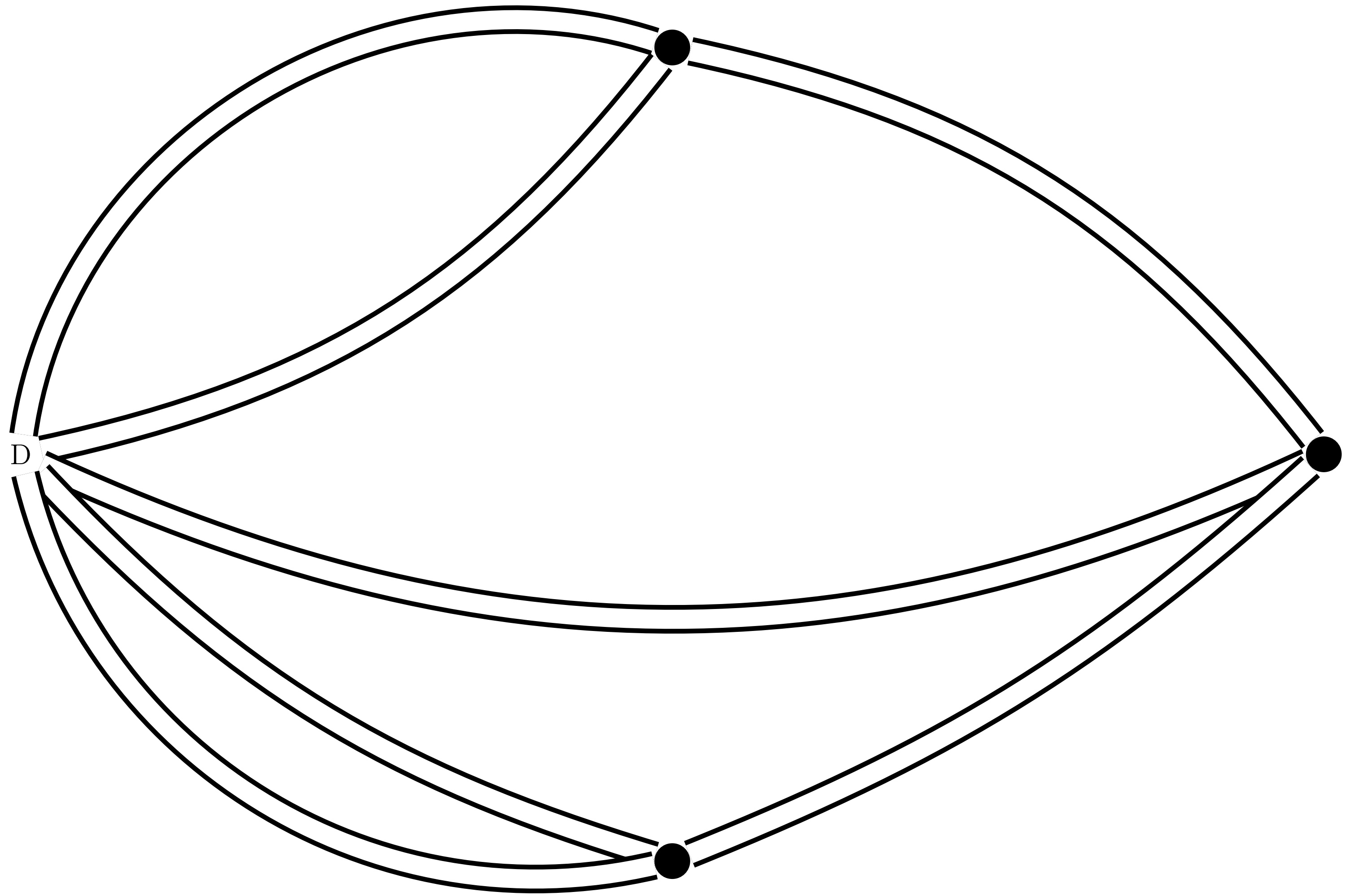
Fusion

Fiches graphes

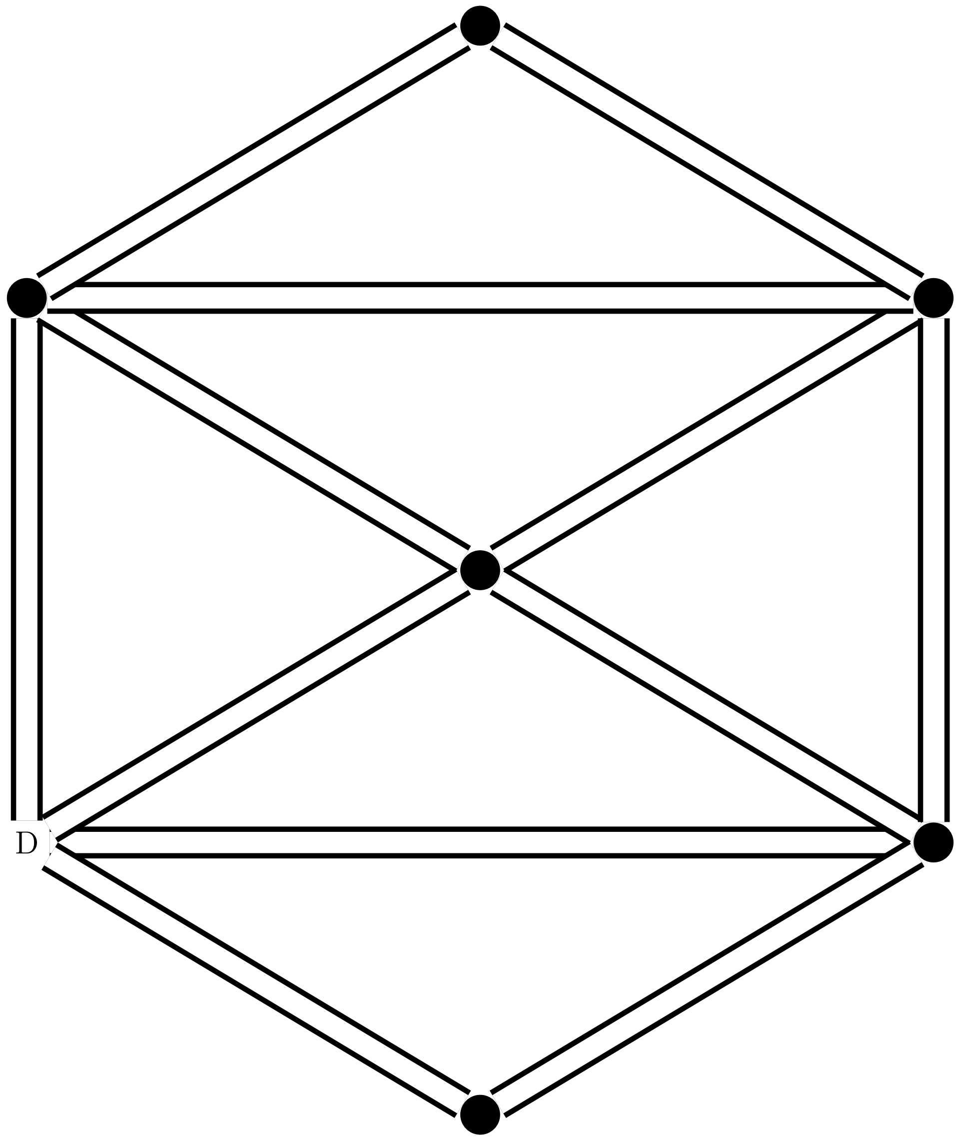
Voici dix fiches d'exemples de graphes variés. On y trouve le problème des ponts de Königsberg (Graphe 1), mais aussi des graphes plus classiques comme le 2, ou inspirés de plans de villes (9 et 10). J'ai également utilisé la deuxième itération du triangle de Sierpiński (Graphe 4), mais cette fractale n'a pas de lien avec ce problème.

Les graphes numérotés par des nombres pairs possèdent des solutions, et sont accompagnés d'une fiche présentant l'une d'entre elles.

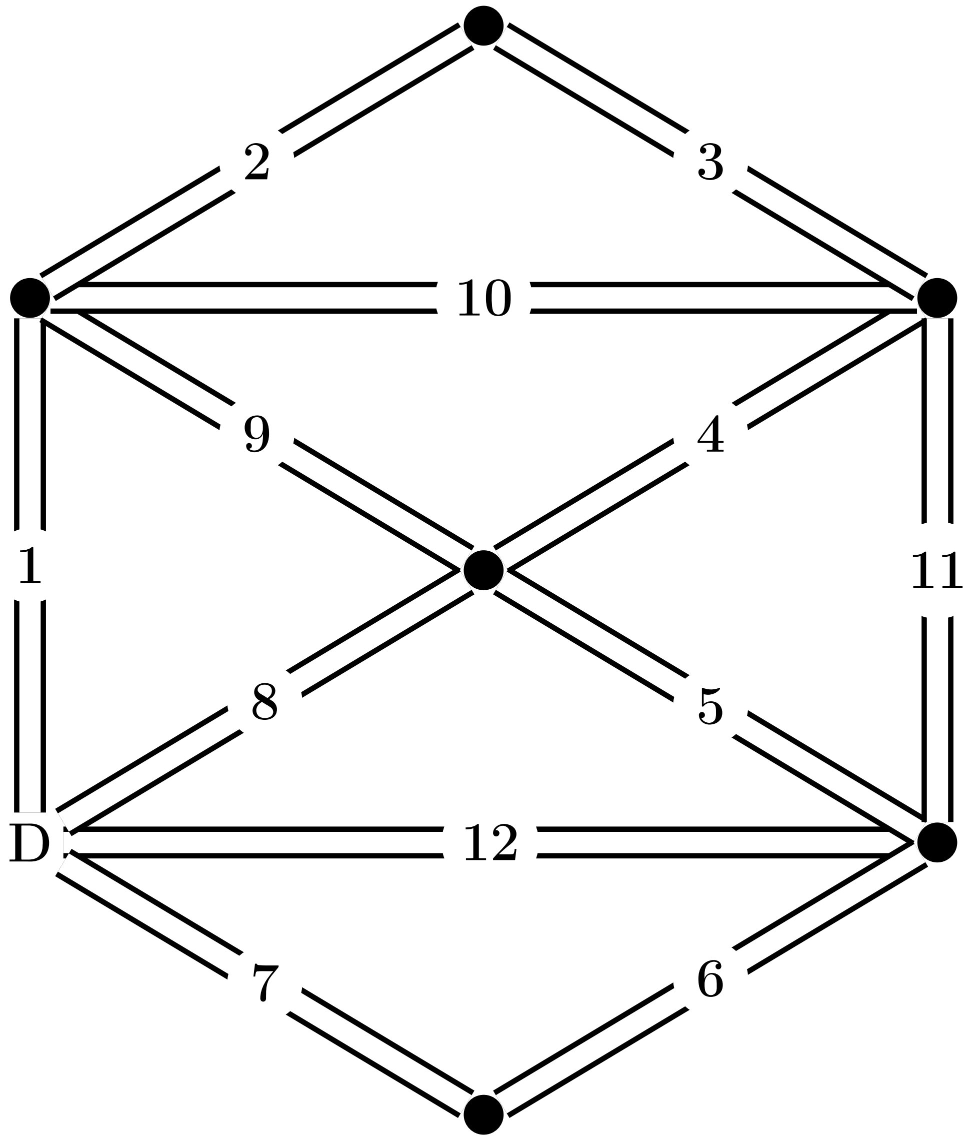
Ces graphes possèdent un nœud de départ, noté D, mais il s'agit d'un simple repère pour aider à commencer, mais il faut bien que les élèves comprennent après quelques essais que le nœud de départ n'importe pas, puisque l'on cherche un cycle couvrant tout le graphe.



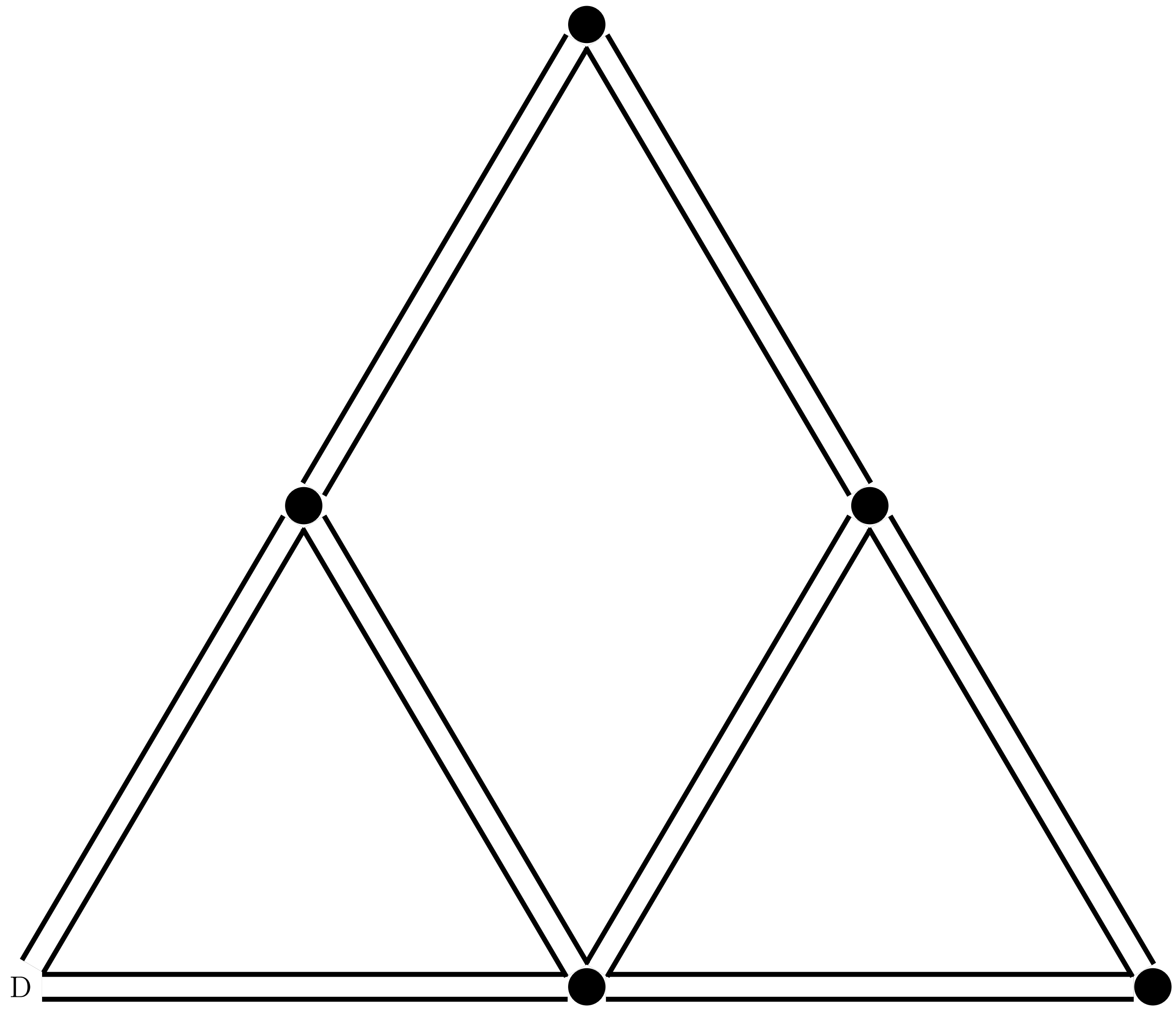
Graphe 1



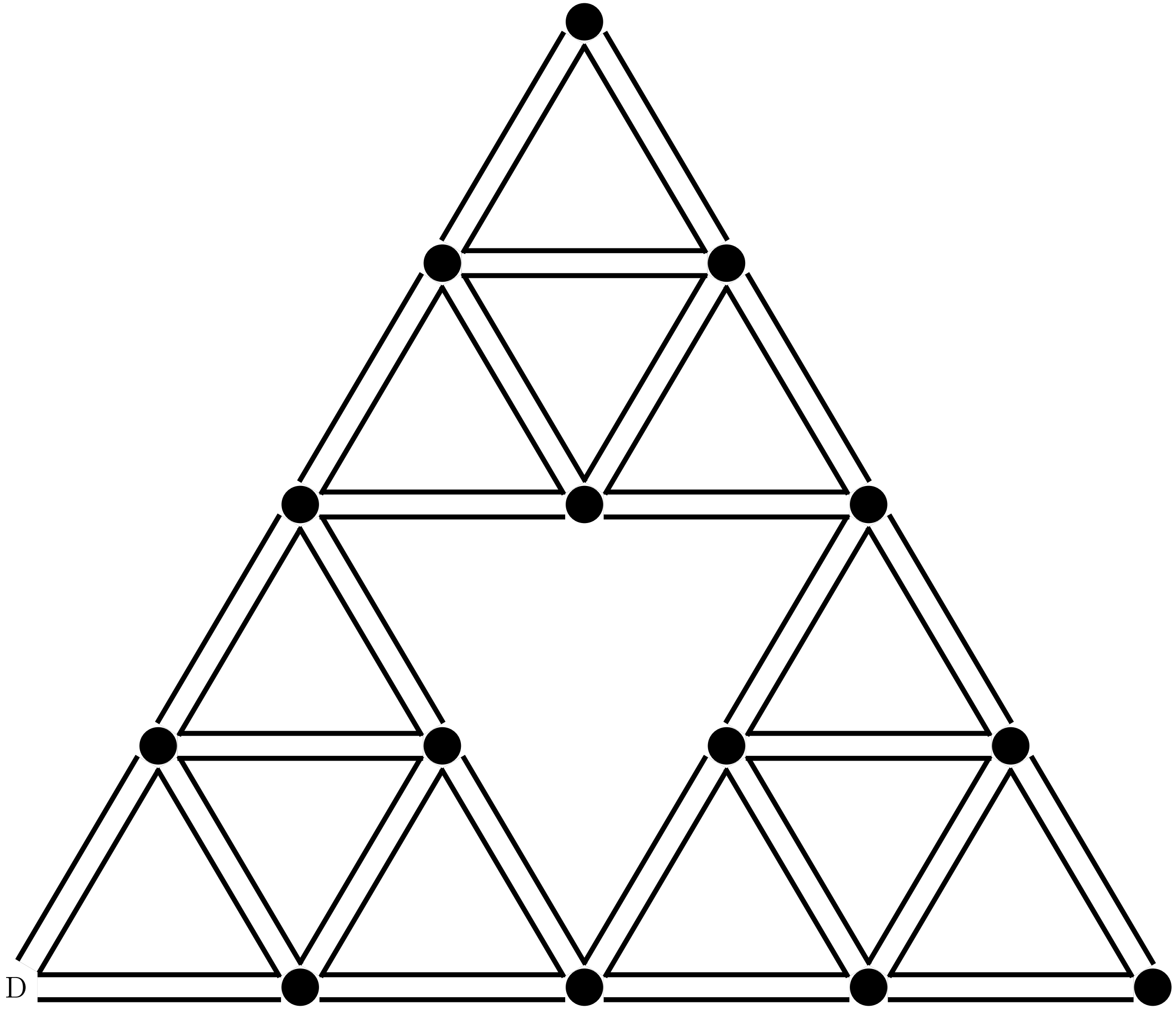
Graphe 2



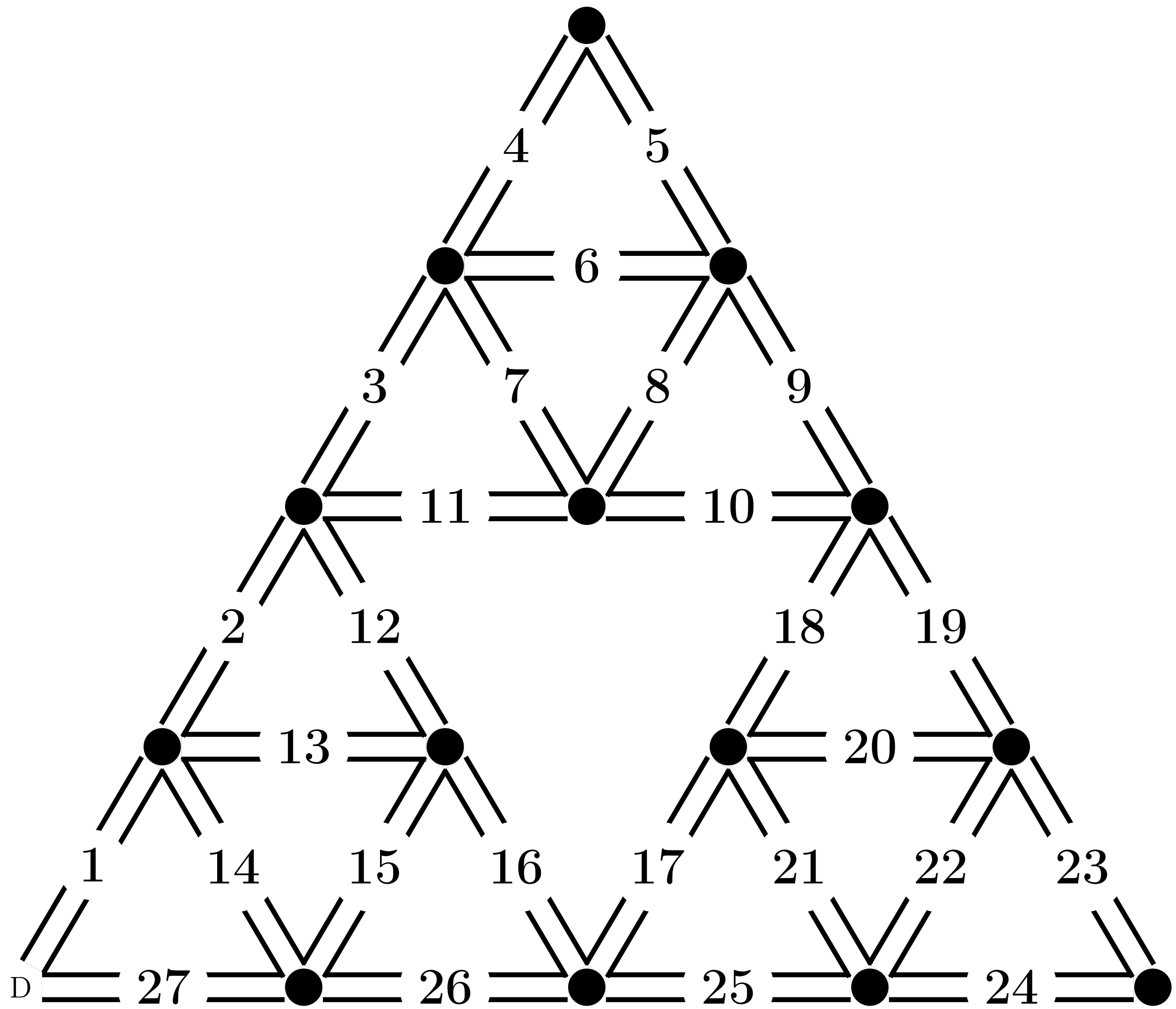
Solution graphe 2



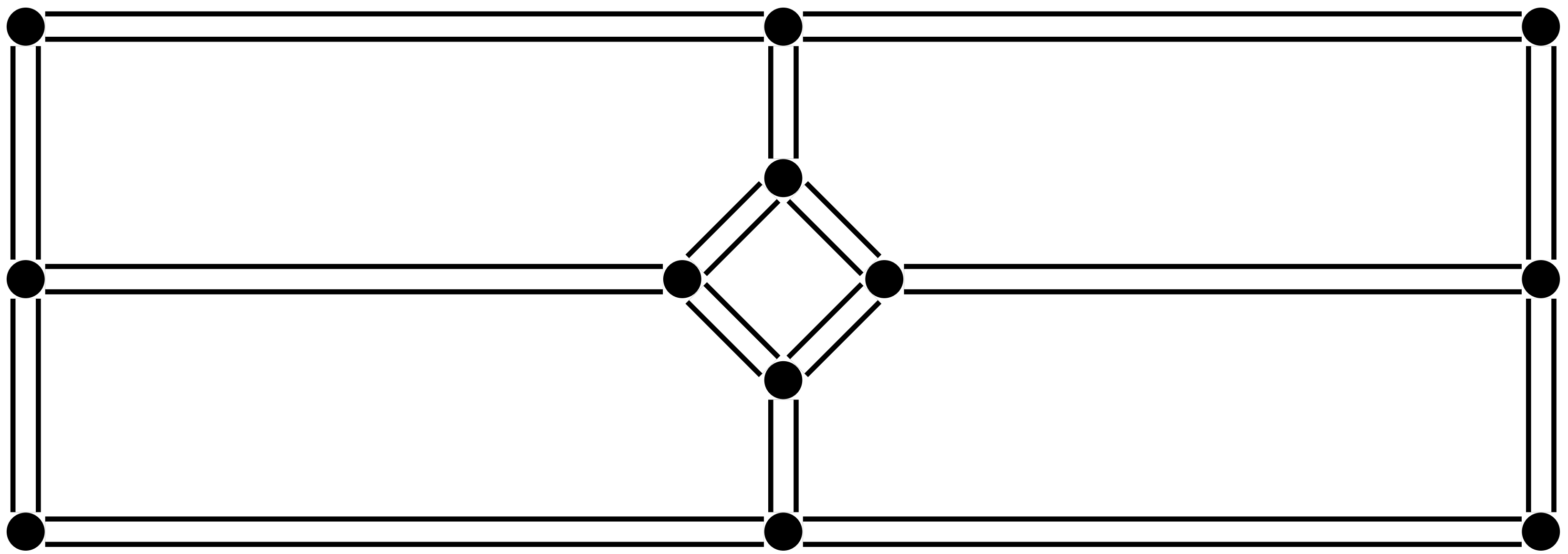
Graphe 3



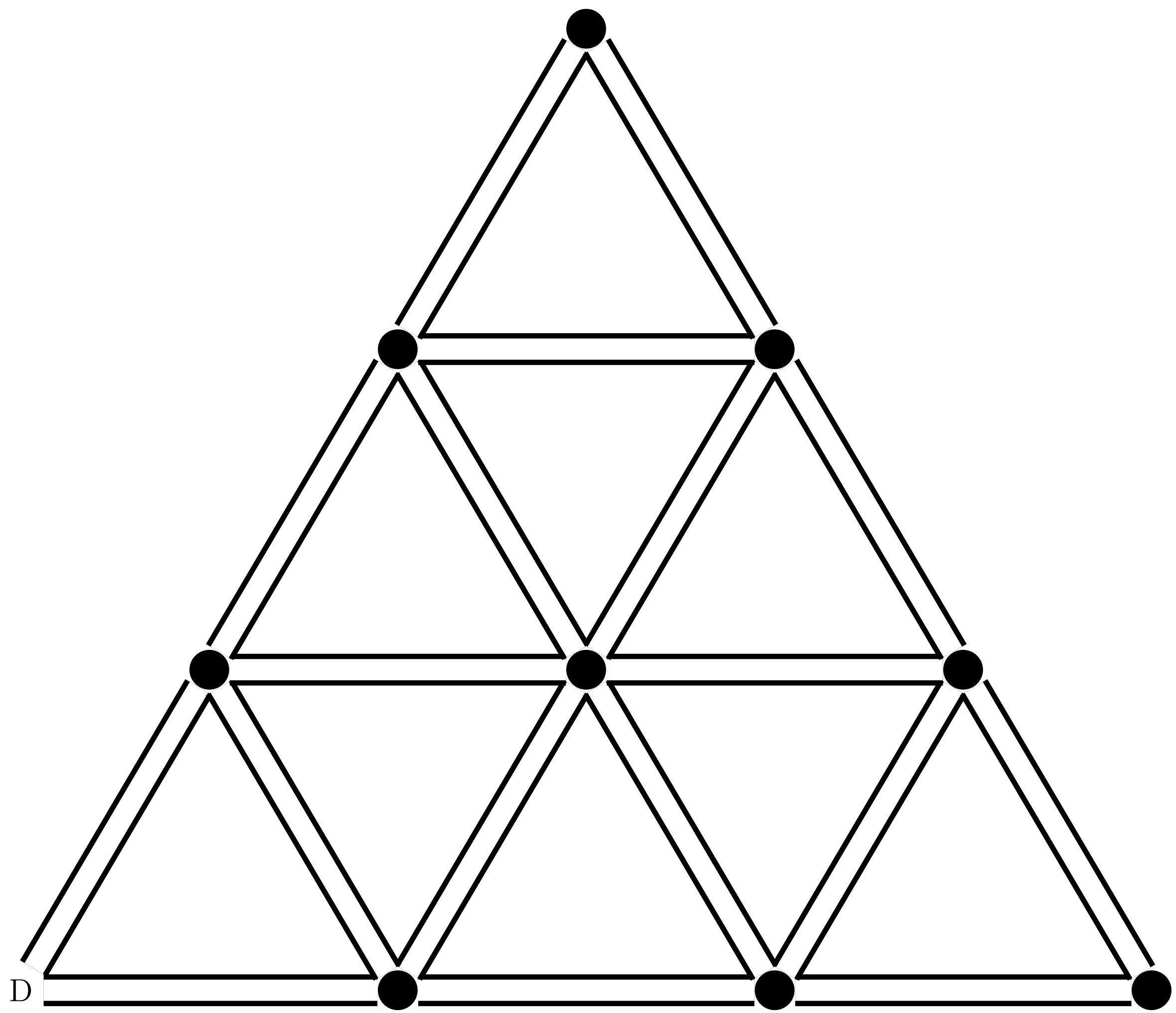
Graphe 4



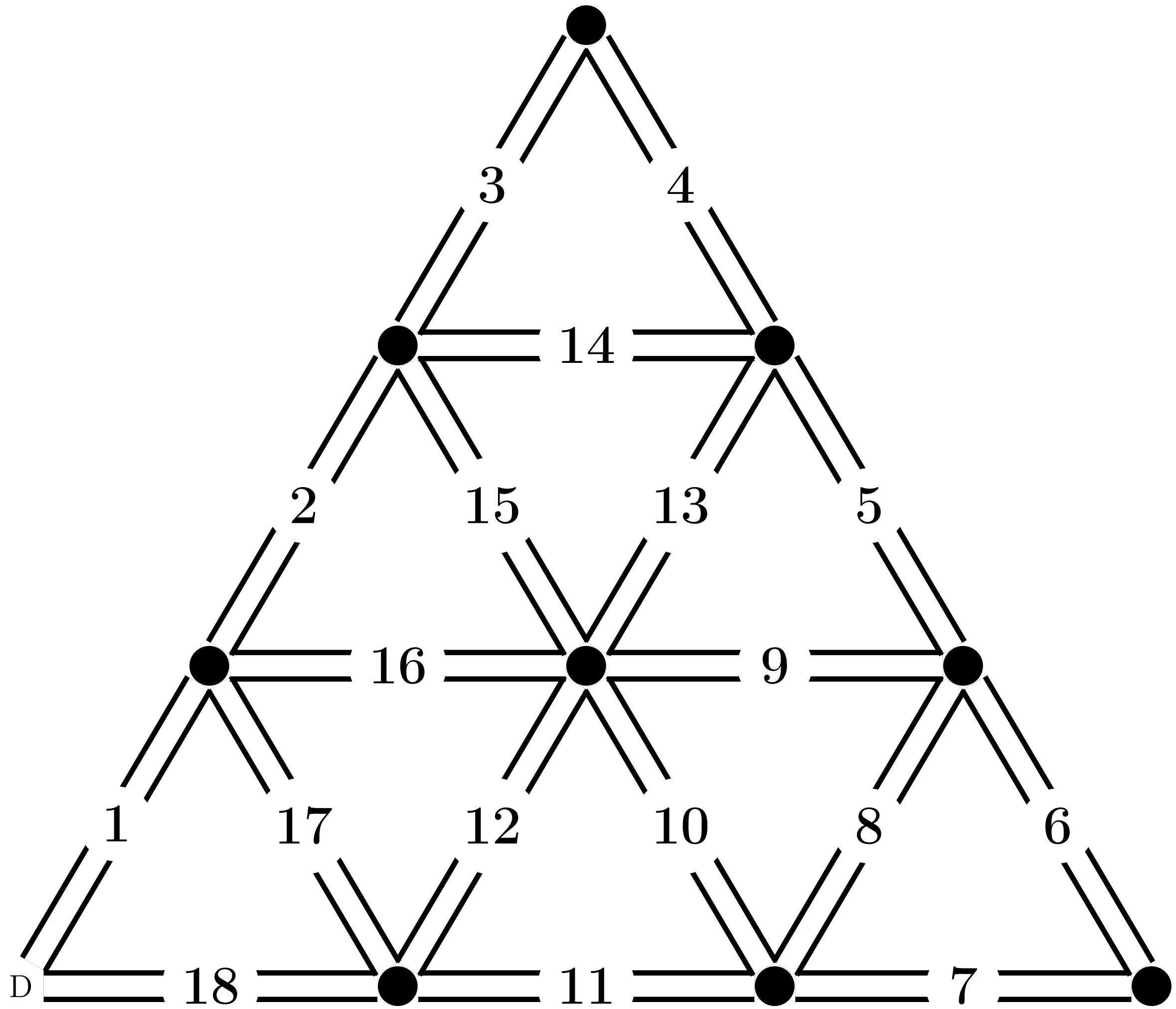
Solution graphe 4



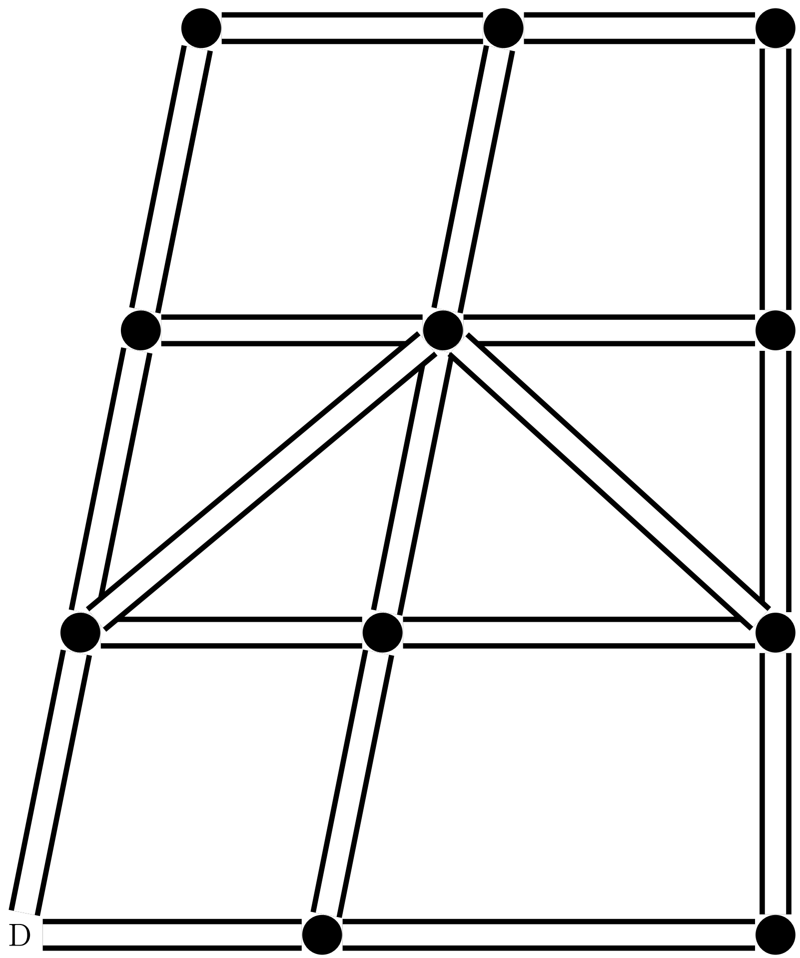
Graphe 5



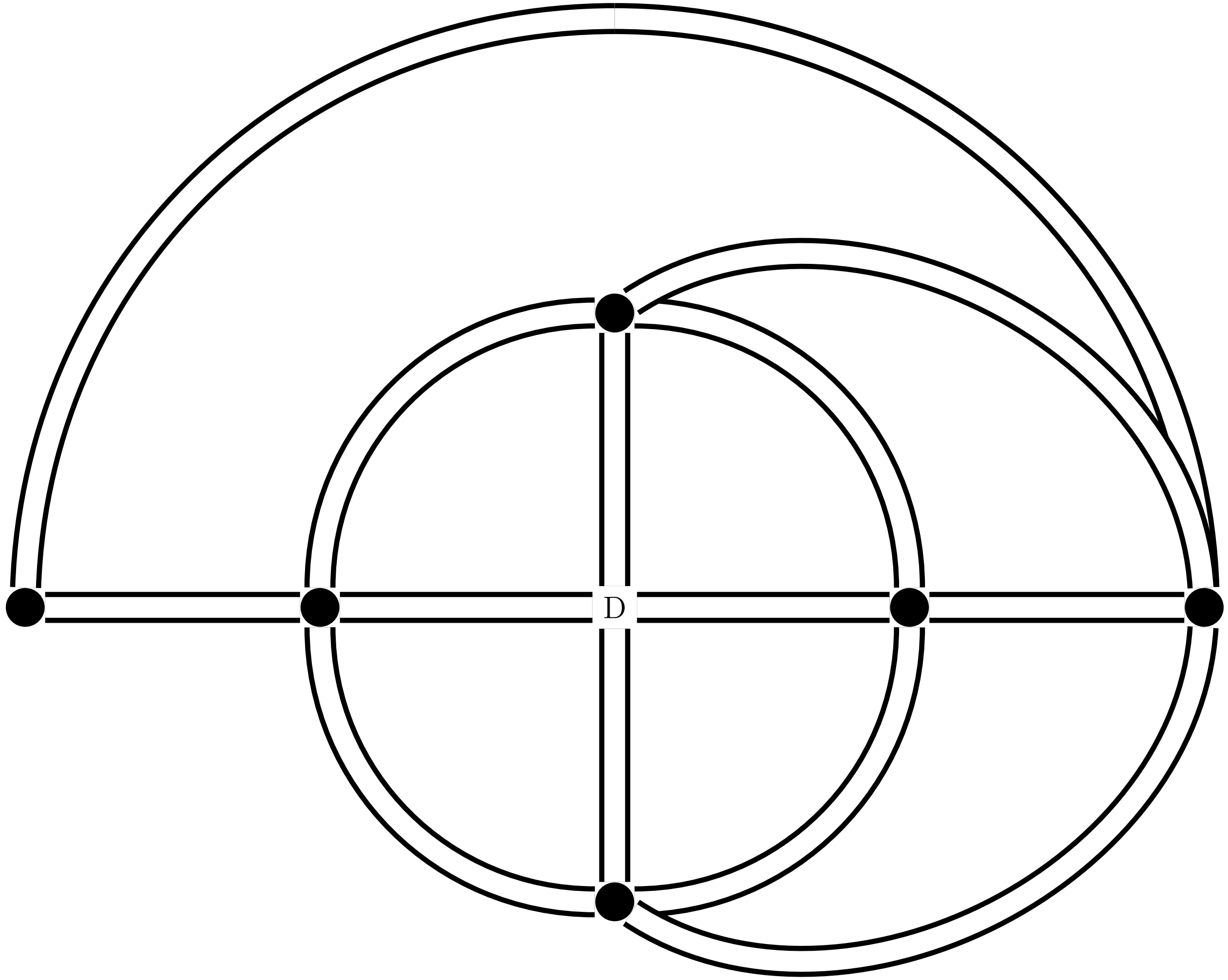
Graphe 6



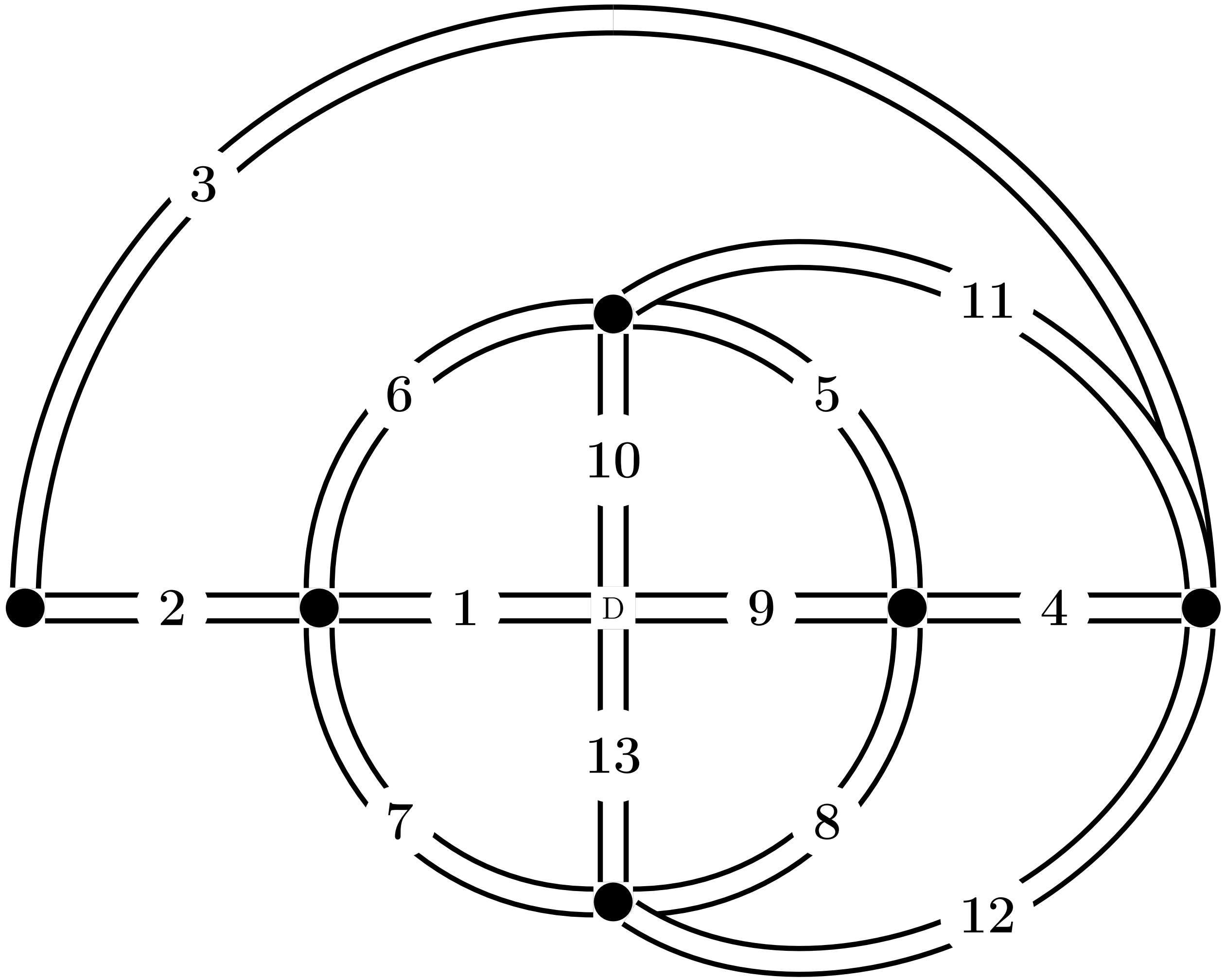
Solution graphe 6



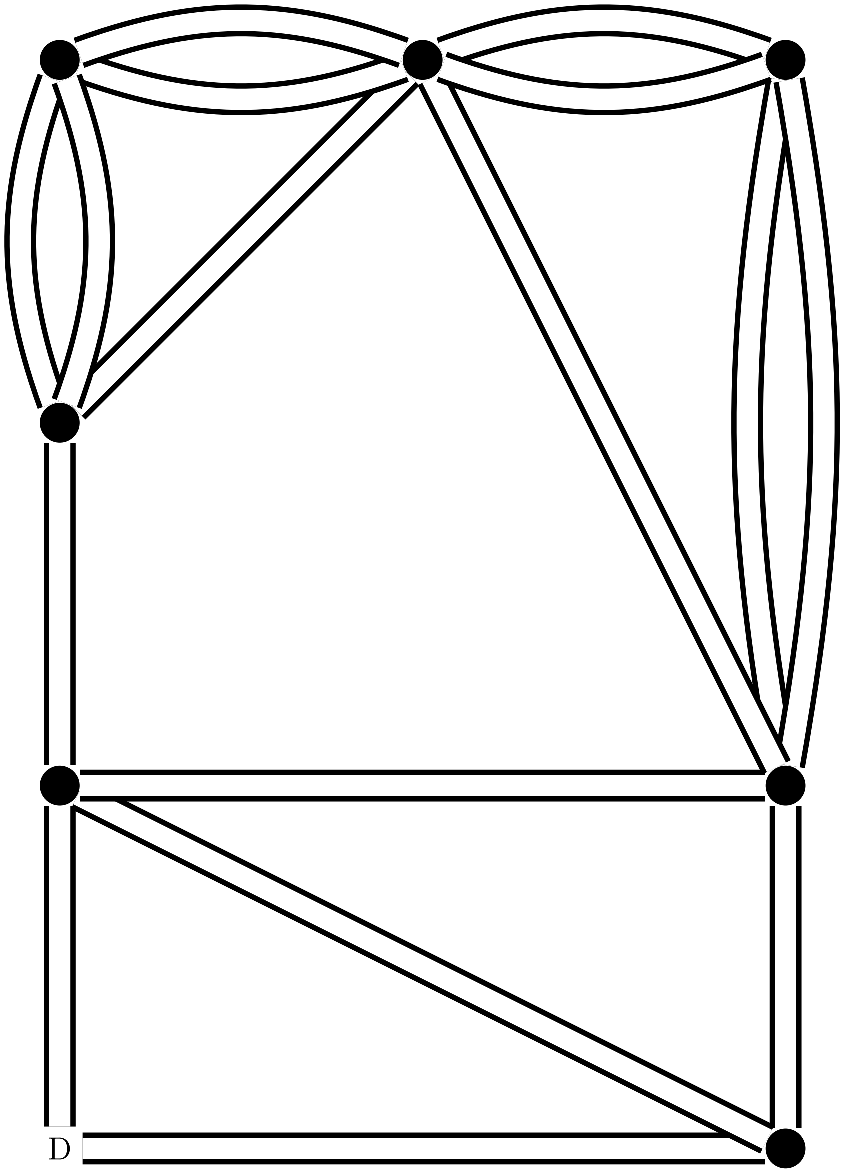
Graphe 7



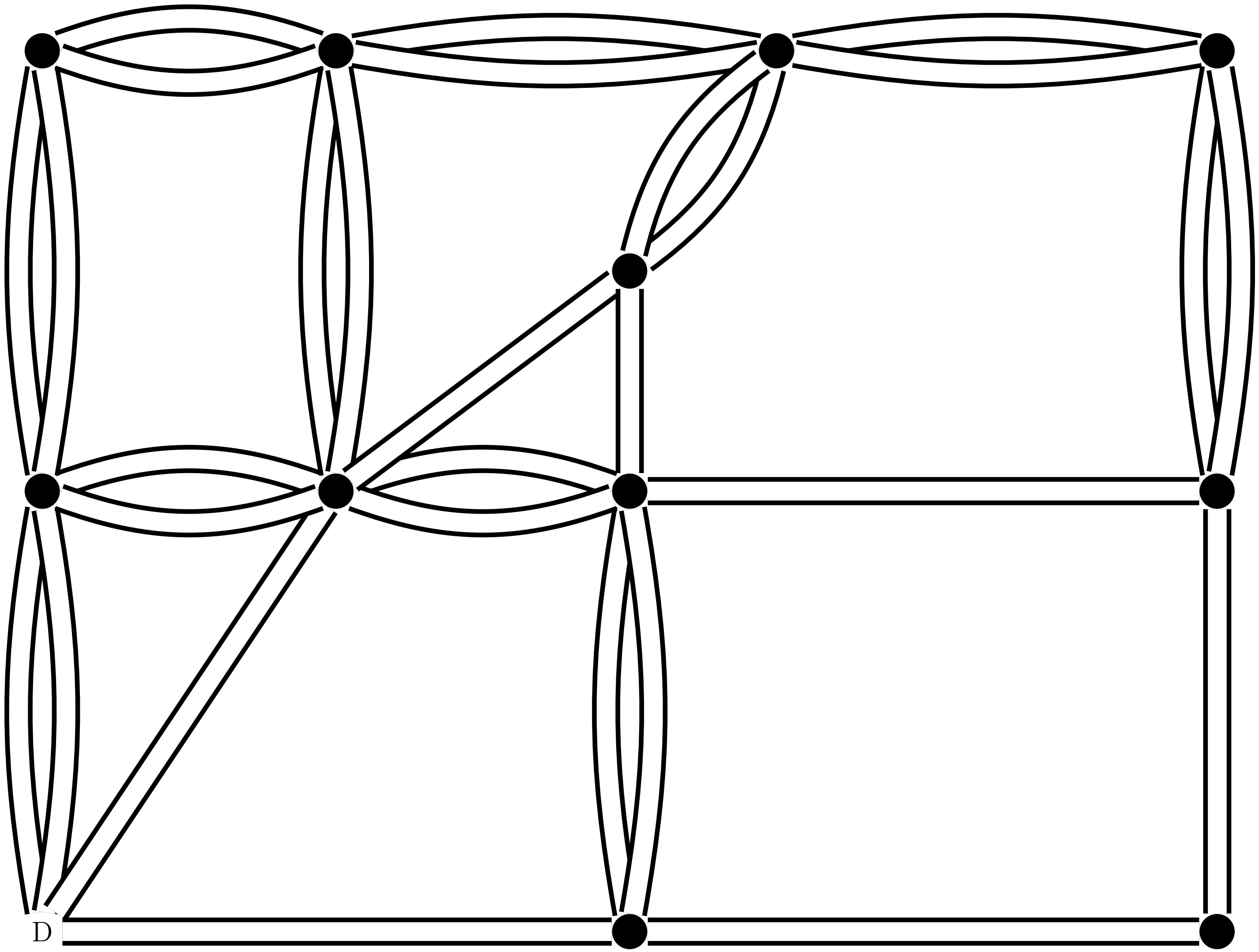
Graphe 8



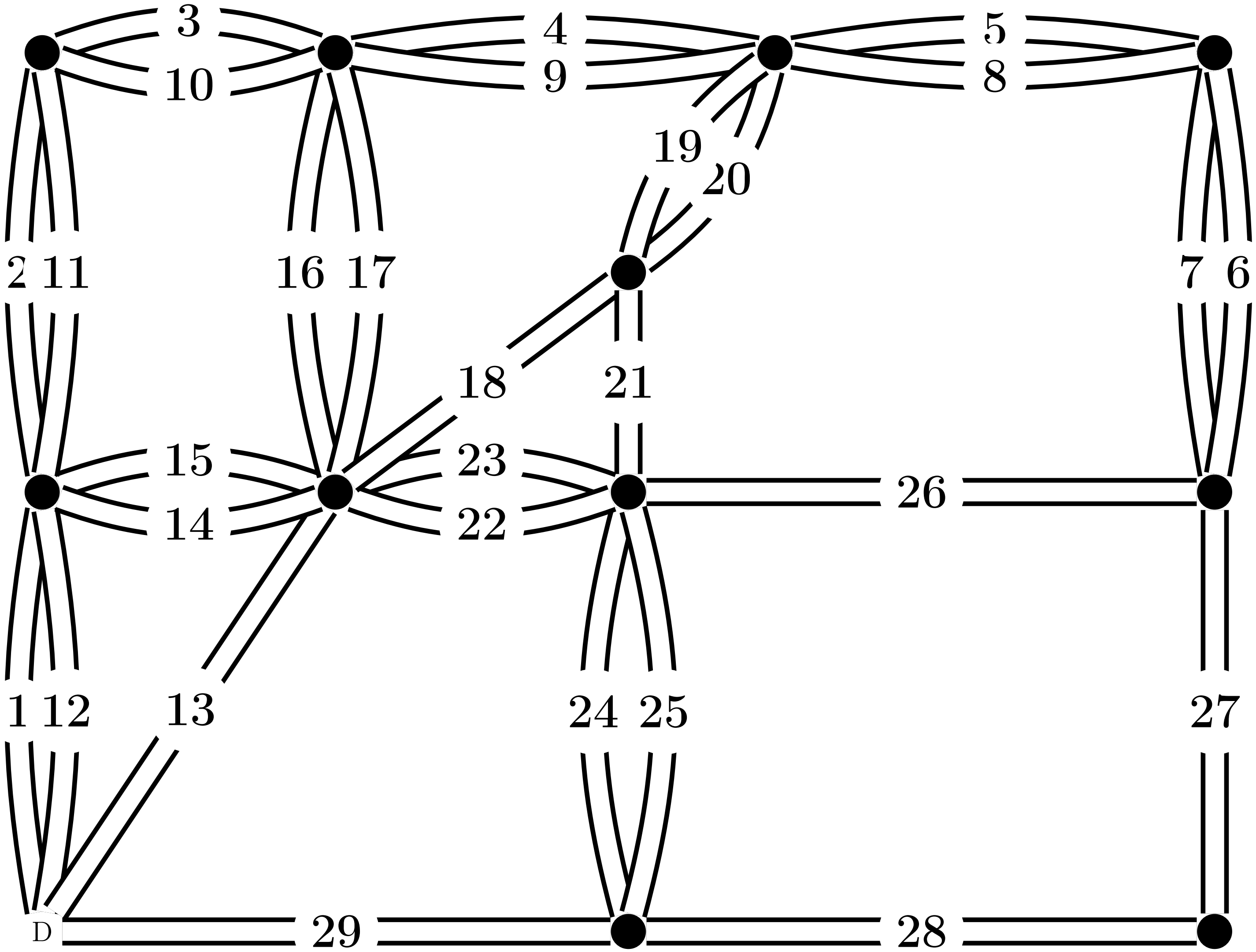
Solution graphe 8



Graphe 9



Graphe 10



Solution graphe 10