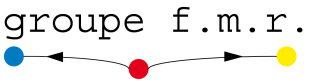
E.W. Dijkstra, 1959, A Note on Two Problems in Connexion with Graphs. Numerische Mathematik 1, p. 269271 Version bilingue et commentée

Edsger W. Dijkstra, Laurent Beauguitte, Marion Maisonobe

E.W. Dijkstra, 1959, A Note on Two Problems in Connexion with Graphs.

Numerische Mathematik 1, p. 269—271

Version bilingue et commentée



Février 2021

Version traduite et commentée par Laurent Beauguitte, CNRS, UMR Géographie-cités. Relecture : Marion Maisonobe, CNRS, UMR Géographie-cités.

Présentation

Le court article d'Edsger W. Dijkstra, A Note on Two Problems in Connexion with Graphs paraît dans *Numerische Mathematik* 1, p. 269—271 en 1959 [1]. Il s'inscrit dans le champ de la détection des plus courts chemins dans un réseau, jeu mathématique apparenté au dilemme du représentant de commerce formulé au XIX^{e} siècle [2], devenu une des applications de la théorie des graphes.

Dijkstra, le premier programmeur hollandais selon ses dires (Dijkstra, 1972), a dans une interview accordée en 2001 raconté les circonstances de la création de l'algorithme - le terme n'étant pas utilisé dans l'article - à Amsterdam en 1956. L'interview en question n'est pas en libre accès mais la page wikipédia anglophone donne suffisamment de précisions : "I designed [it] in about twenty minutes. One morning I was shopping in Amsterdam with my young fiancée, and tired, we sat down on the café terrace to drink a cup of coffee and I was just thinking about whether I could do this, and I then designed the algorithm for the shortest path. As I said, it was a twenty-minute invention [3]" (interview avec Philip L. Frana, *Communications of the ACM*, 2001, extrait de la page Dijkstra's algorithm) - à noter que la page est plus longue que l'article original.

Références

Dijkstra, Edsger W. "The humble programmer." *Communications of the ACM* 15.10 (1972) : 859-866. En ligne.

Les numéros entre crochets indiquent la pagination originale. Les notes de bas de page en bleu de la version française sont du traducteur.

1. L'article est librement accessible en ligne.
2. Soit un vendeur itinérant censé visiter x villes, comment minimiser la distance parcourue ? L'acronyme anglais pour ce problème algorithmique fameux est TSP - *travelling salesperson problem*. Tout comme le problème des ponts de Königsberg ou celui du théorème des 4 couleurs, il s'agit d'un jeu mathématique ; il n'existe pas avant les années 1930 - édition en allemand du premier manuel signé Denés König, *Theorie der endlichen und unendlichen Graphen*, Akademische Verlagsgesellschaft, Leipzig - voire 1950 - réédition de cet ouvrage à New York mais toujours en allemand - un domaine autonome des mathématiques appelé "théorie des graphes".
3. "Je l'ai élaboré en 20 minutes. Un matin, je faisais du shopping à Amsterdam avec ma jeune fiancée. Fatigués, nous nous sommes installés à la terrasse d'un café et j'ai réfléchi au moyen de résoudre ce problème et j'ai mis au point l'algorithme des plus courts chemins. C'était l'histoire de 20 minutes."

A Note on Two Problems in Connexion with Graphs

We consider n points (nodes), some or all pairs of which are connected by a branch ; the length of each branch is given. We restrict ourselves to the case where at least one path exists between any two nodes. We now consider two problems.

Problem 1. Construct the tree of minimum total length between the n nodes. (A tree is a graph with one and only one path between every two nodes.)

In the course of the construction that we present here, the branches are subdivided into three sets :

I. the branches definitely assigned to the tree under construction (they will form a subtree) ;

II. the branches from which the next branch to be added to set I, will be selected ;

III. the remaining branches (rejected or not yet considered).

The nodes are subdivided into two sets :

A. the nodes connected by the branches of set I,

B. the remaining nodes (one and only one branch of set II will lead to each of these nodes).

We start the construction by choosing an arbitrary node as the only member of set A, and by placing all branches that end in this node in set II. To start with, set I is empty. From then onwards we perform the following two steps repeatedly.

Step I. The shortest branch of set II is removed from this set and added to set I. As a result one node is transferred from set B to set A.

Step 2. Consider the branches leading from the node, that has just been transferred to set A, to the nodes that are still in set B. If the branch under consideration is longer than the corresponding branch in set II, it is rejected ; if it is shorter, it replaces the corresponding branch in set II, and the latter is rejected.

We then return to step 1 and repeat the process until sets II and B are empty. The branches in set I form the tree required.

The solution given here is to be preferred to the solution given by J. B. Kruskal [1] and those given by H. Loberman and A. Weinberger [2]. In their solutions all the - possibly $\frac{1}{2}n(n-1)$ - branches are first of all sorted according to length. Even if the length of the branches is a computable function of the node coordinates, their methods demand that data for all branches are stored simultaneously. Our method only requires the simultaneous storing of [270] the data for at most n branches, viz. the branches in sets I and II and the branch under consideration in step 2.

Problem 2. Find the path of minimum total length between two given nodes P and Q.

We use the fact that, if R is a node on the minimal path from P to Q, knowledge of the latter implies the knowledge of the minimal path from P to R. In the solution presented, the minimal paths from P to the other nodes are constructed in order of increasing length until Q is reached.

In the course of the solution the nodes are subdivided into three sets :

A. the nodes for which the path of minimum length from P is known ; nodes will be added to this set in order of increasing mimimum path length from node P ;

B. the nodes from which the next node to be added to set A will be selected ; this set comprises all those nodes that are connected to at least one node of set A but do not yet belong to A themselves ;

C. the remaining nodes.

The branches are also subdivided into three sets :

I. the branches occurring in the minimal paths from node P to the nodes in set A ;

II. the branches from which the next branch to be placed in set I will be selected ; one and only one branch of this set will lead to each node in set B ;

III. the remaining branches (rejected or not yet considered).

To start with, all nodes are in set C and all branches are in set III. We now transfer node P to set A and from then onwards repeatedly perform the following steps.

Step 1. Consider all branches r connecting the node just transferred to set A with nodes R in sets B or C. If node R belongs to set B, we investigate whether the use of branch r gives rise to a shorter path from P to R than the known path that uses the corresponding branch in set II. If this is not so, branch r is rejected ; if, however, use of branch r results in a shorter connexion between P and R than hitherto obtained, it replaces the corresponding branch in set II and the latter is rejected. If the node R belongs to set C, it is added to set B and branch r is added to set II.

Step 2. Every node in set B can be connected to node P in only one way if we restrict ourselves to branches from set I and one from set II. In this sense each node in set B has a distance from node P : the node with minimum distance from P is transferred from set B to set A, and the corresponding branch is transferred from set II to set I. We then return to step 1 and repeat the process until node Q is transferred to set A. Then the solution has been found.

Remark 1. The above process can also be applied in the case where the length of a branch depends on the direction in which it is traversed.

Remark 2. For each branch in sets I and II it is advisable to record its two nodes (in order of increasing distance from P), and the distance between P and that node of the branch that is furthest from P. For the branches of set I this [271] is the actual minimum distance, for the branches of set II it is only the minimum thus far obtained.

The solution given above is to be preferred to the solution by L. R. FORD [3] as described by C. BERGE [4], for, irrespective of the number of branches, we need not store the data for all branches simultaneously but only those for the branches in sets I and II, and this number is always less than n. Furthermore the amount of work to be done seems to be considerably less.

References

[1] Kruskal jr., J. B. : On the Shortest Spanning Subtree of a Graph and the Travelling Salesman Problem. Proc. Amer. Math. Soc. 7, 48—50 (1956).

[2] Loberman, H., and A. Weinberger : Formal Procedures for Connecting Terminals with a Minimum Total Wire Length. J. Ass. Comp. Mach. 4, 428—437 (1957).

[3] Ford, L. R. : Network flow theory. Rand Corp. Paper, P-923, 1956.

[4] Berge, C. : Théorie des graphes et ses applications, pp. 68—69. Paris : Dunod 1958.

Mathematisch Centrum
2e Boerhaavestraat 49
Amsterdam-O

(Received June 11, 1959)

Remarques sur deux problèmes de connexion dans les graphes

Nous considérons n points (nœuds), dont certaines ou toutes les paires sont connectées par une branche ; la longueur de chaque branche est donnée. Nous nous limitons au cas où au moins un chemin existe entre deux nœuds [4]. Nous considérons maintenant deux problèmes.

Problème 1. Construire l'arbre de longueur totale minimale [5] entre les n nœuds. (Un arbre est un graphe avec un et un seul chemin entre toute paire de nœuds. [6])

Au cours de la construction que nous présentons ici, les branches sont subdivisées en trois ensembles :

I. les branches définitivement assignées à l'arbre en construction (elles formeront un sous-arbre) ;

II. les branches à partir desquelles la prochaine branche à ajouter à l'ensemble I sera sélectionnée ;

III. les branches restantes (rejetées ou non encore considérées).

Les nœuds sont subdivisés en deux ensembles :

A. les nœuds reliés par les branches de l'ensemble I,

B. les nœuds restants (une branche et une seule de l'ensemble II mènera à chacun de ces nœuds).

Nous commençons en choisissant arbitrairement un nœud comme seul membre de l'ensemble A, et en plaçant toutes les branches qui se terminent par ce nœud dans l'ensemble II. Au départ, l'ensemble I est vide. Puis nous répétons les deux étapes suivantes.

Étape 1. La branche la plus courte de l'ensemble II est supprimée de cet ensemble et ajoutée à l'ensemble I. En conséquence, un nœud est transféré de l'ensemble B à l'ensemble A.

Étape 2. Considérer les branches provenant du nœud qui vient d'être transféré à l'ensemble A vers les nœuds qui sont toujours dans l'ensemble B. Si la branche considérée est plus longue que la branche correspondante dans l'ensemble II, elle est rejetée ; si elle est plus courte, elle remplace la branche correspondante de l'ensemble II, et cette dernière est rejetée.

Nous retournons ensuite à l'étape 1 et répétons le processus jusqu'à ce que les ensembles II et B soient vides. Les branches de l'ensemble I forment l'arbre requis.

La solution donnée ici doit être préférée à la solution donnée par J. B. KRUSKAL [1] et celles données par H. Loberman et A. Weinberger [2]. Dans leurs solutions, toutes les branches - éventuellement $\frac{1}{2}n(n-1)$ [7] - sont

4. L'auteur considère donc des graphes connexes et valués : chaque lien - nommé ici branche - étant qualifié par une longueur. Implicitement, le graphe considéré est non orienté.

5. On parlerait aujourd'hui d'arbre couvrant minimum - *minimum spanning tree.*

6. Utiliser cette formulation peu courante aujourd'hui permet de définir un arbre sans avoir à définir la notion de cycle et de graphe acyclique.

7. Dijkstra considère donc un graphe connexe non planaire.

d'abord triées en fonction de leur longueur. Même s'il est possible de calculer la longueur des branches à partir des coordonnées des nœuds, leurs méthodes exigent que les données de toutes les branches soient stockées simultanément. Notre méthode ne nécessite que le stockage simultané des données pour au plus n branches, à savoir les branches des ensembles I et II et la branche à l'étude à l'étape 2.

Problème 2. Trouver le chemin de longueur totale minimale entre deux nœuds P et Q.

Nous utilisons le fait que, si R est un nœud sur le chemin minimal de P à Q, la connaissance de ce dernier implique la connaissance du chemin minimal de P à R. Dans la solution présentée, les chemins minimaux de P vers les autres nœuds sont construits dans l'ordre de longueur croissante jusqu'à ce que Q soit atteint.

Pour parvenir à la solution, les nœuds sont subdivisés en trois ensembles :

A. les nœuds pour lesquels le chemin de longueur minimale à partir de P est connu ; les nœuds seront ajoutés à cet ensemble afin d'augmenter le moins possible la longueur du chemin depuis le nœud P ;

B. les nœuds à partir desquels le prochain nœud à ajouter à A sera sélectionné ; cet ensemble comprend tous les nœuds qui sont connectés à au moins un nœud de A mais qui n'appartiennent pas encore à A ;

C. les nœuds restants.

Les branches sont également subdivisées en trois ensembles :

I. les branches apparaissant dans les chemins minimaux du nœud P vers les nœuds dans l'ensemble A ;

II. les branches à partir desquelles la prochaine branche à placer dans l'ensemble I sera sélectionnée ; une seule branche de cet ensemble mènera à chaque nœud de l'ensemble B ;

III. les branches restantes (rejetées ou non encore considérées).

Pour commencer, tous les nœuds sont dans l'ensemble C et toutes les branches sont dans l'ensemble III. Nous transférons maintenant le nœud P pour définir A et à partir de là répétons les étapes suivantes.

Étape 1. Considérer toutes les branches r connectant le nœud transféré dans l'ensemble A aux nœuds R des ensembles B ou C. Si le nœud R appartient à l'ensemble B, nous examinons si l'utilisation de la branche r permet d'obtenir un chemin plus court de P à R que le chemin connu qui utilise la branche correspondante dans l'ensemble II. Si ce n'est pas le cas, la branche r est rejetée ; si l'utilisation de la branche r entraîne une connexion plus courte entre P et R que celle obtenue jusqu'à présent, elle remplace la branche correspondante dans l'ensemble II et cette dernière est rejetée. Si le nœud R appartient à l'ensemble C, il est ajouté à l'ensemble B et la branche r est ajouté à l'ensemble II.

Étape 2. Tout nœud de l'ensemble B peut être connecté au nœud P d'une seule façon si nous nous limitons aux branches de l'ensemble I et à un nœud de l'ensemble II. Une distance existe entre tout nœud de l'ensemble B et le

nœud P : celui à une distance minimale de P est transféré de l'ensemble B à l'ensemble A, et la branche correspondante est transférée de l'ensemble II à l'ensemble I. Nous retournons ensuite à l'étape 1 et répétons le processus jusqu'à ce que le nœud Q soit transféré vers l'ensemble A. La solution est alors trouvée.

Remarque 1. Le processus ci-dessus peut également être appliqué dans le cas où la longueur d'une branche dépend de la direction dans laquelle elle est parcourue [8].

Remarque 2. Pour chaque branche des ensembles I et II, il est conseillé de conserver ses deux nœuds (afin d'augmenter la distance à partir de P), et la distance entre P et le nœud de la branche qui est le plus éloigné de P. Pour les branches de la série I, c'est la distance minimale réelle, pour les branches de la série II, c'est seulement le minimum obtenu jusqu'à présent.

La solution donnée ci-dessus doit être préférée à la solution de L. R. Ford [3] décrite par C. Berge [4], car, quel que soit le nombre de branches, nous n'avons pas besoin de stocker les données pour toutes les branches simultanément, mais seulement celles des branches des ensembles I et II, et ce nombre est toujours inférieur à n. En outre, la quantité de travail à accomplir semble être considérablement moindre.

Références

[1] Kruskal jr., J. B. : On the Shortest Spanning Subtree of a Graph and the Travelling Salesman Problem. Proc. Amer. Math. Soc. 7, 48—50 (1956).

[2] Loberman, H., and A. Weinberger : Formal Procedures for Connecting Terminals with a Minimum Total Wire Length. J. Ass. Comp. Mach. 4, 428—437 (1957).

[3] Ford, L. R. : Network flow theory. Rand Corp. Paper, P-923, 1956.

[4] Berge, C. : Théorie des graphes et ses applications, pp. 68—69. Paris : Dunod 1958.

Mathematisch Centrum
2e Boerhaavestraat 49
Amsterdam-O

(Reçu le 11 juin 1959)

8. Remarque rapide signalant que l'algorithme peut fonctionner sur un graphe orienté où la distance d_{ij} entre deux sommets i et j peut être différente de la distance d_{ji}.

groupe f.m.r.

La collection « textes » du groupe fmr (flux, matrices, réseaux) propose des rééditions bilingues d'articles consacrés à l'analyse de réseaux.

Parus

— L. Beauguitte, P. Beauguitte et P. Gourdon, 2021, « William L. Garrison, 1960, Connectivity of the Interstate Highway System ».
— L. Beauguitte et M. Maisonobe, 2021, « Joseph B. Kruskal Jr., 1956, On the Shortest Spanning Subtree of a Graph and the Traveling Salesman Problem ».

À paraître

— F. Bahoken et L. Beauguitte, 2021, « John D. Nystuen et F. Dacey, 1961, A graph theory interpretation of nodal regions ».