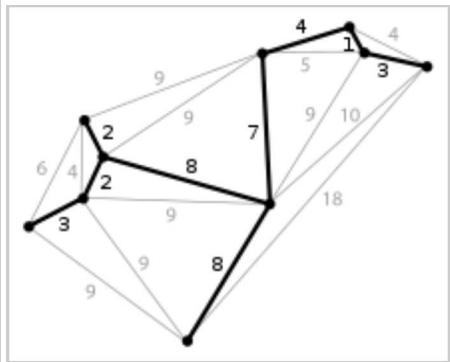
Algorithme de Prim

L'**algorithme de Prim** est un algorithme glouton qui calcule un arbre couvrant minimal dans un graphe connexe valué et non orienté. En d'autres termes, cet algorithme trouve un sous-ensemble d'arêtes formant un arbre sur l'ensemble des sommets du graphe initial, et tel que la somme des poids de ces arêtes soit minimale. Si le graphe n'est pas connexe, alors l'algorithme détermine un arbre couvrant minimal d'une composante connexe du graphe.



Arbre couvrant de poids minimum

Sommaire

Historique

L'algorithme a été développé en 1930 par le mathématicien tchèque Vojtech Jarnik [1] puis a été redécouvert et republié par Robert C. Prim[2] et Edsger W. Dijkstra en 1959. Ainsi, il est aussi parfois appelé **DJP algorithm**[3], **Jarník's algorithm**[4], the **Prim–Jarník algorithm**[5], or the **Prim–Dijkstra algorithm**[6].

Principe

L'algorithme[7] consiste à faire croître un arbre depuis un sommet. On commence avec un seul sommet puis à chaque étape, on ajoute une arête de poids minimum adjacente à l'arbre en construction.

Exemple

À droite, on donne un exemple d'exécution de l'algorithme de Prim.

Algorithme

Pseudo-code

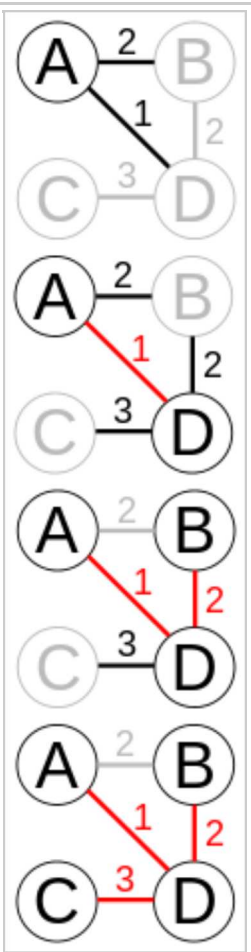
Le pseudo-code[7] de l'algorithme de Prim est similaire à celui de l'algorithme de Dijkstra et utilise le type abstrait file de priorité.

```
fonction prim(G, s)
```

```
pour tout sommet t
       cout[t] := +∞
       pred[t] := nil
cout[s] := 0
F := file de priorité contenant les sommets de G avec cout[.] comme priorité
tant que F ≠ vide
     t := F.defiler
     pour toute arête t--u
          si cout[u] > poids(t--u)
                 cout[u] := poids(t--u)
                 pred[u] := t
                 F.notifierDiminution(u)

retourner pred
```

Au début tous les sommets sont dans la file de priorité. La priorité est donnée par cout[.]. On retire un à un les sommets de la file de priorité. Le tableau pred[.] contient le prédécesseur d'un sommet dans l'arbre en construction. L'algorithme retourne le tableau pred qui représente l'arbre couvrant minimum.



Exécution de l'algorithme de Prim depuis le sommet A.

Complexité

Si on représente le graphe par une matrice d'adjacence alors la complexité est en $O(|S|^2)$ où $|S|$ est le nombre de sommets dans le graphe. Sinon, si on représente le graphe par une liste d'adjacence, nous donnons ici les complexités en fonction de l'implémentation de la file de priorité.

Implémentation de la file de priorité	Complexité temporelle
Liste ou tableau	$O(\|A\| \times \|S\|)$
tas min binaire	$O(\|A\| \times log\|S\|)$
tas de Fibonacci	$O(\|A\| + \|S\| \times log\|S\|)$

où $|S|$ est le nombre de sommets dans le graphe et $|A|$ est le nombre d'arcs dans le graphe.

Notes et références

1. « O jistém problému minimálním. (Z dopisu panu O. Borůvkovi) » (http://dml.cz/dmlcz/500726), sur *dml.cz* (consulté le 26 décembre 2015)
2. R.C. Prim, « Shortest connection networks and some generalizations », *Bell System Technical Journal, The*, vol. 36, 1er novembre 1957, p. 1389-1401 (ISSN 0005-8580 (http://worldcat.org/issn/0005-8580&lang=fr), DOI 10.1002/j.1538-7305.1957.tb01515.x (http://dx.doi.org/10.1002%2Fj.1538-7305.1957.tb01515.x), lire en ligne (http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6773228))
3. Seth Pettie et Vijaya Ramachandran, « An Optimal Minimum Spanning Tree Algorithm », *J. ACM*, vol. 49, 1er janvier 2002, p. 16–34 (ISSN 0004-5411 (http://worldcat.org/issn/0004-5411&lang=fr), DOI 10.1145/505241.505243 (http://dx.doi.org/10.1145%2F505241.505243), lire en ligne (http://doi.acm.org/10.1145/505241.505243))
4. **(en)** Robert Sedgewick et Kevin Daniel Wayne, *Algorithms*, Addison-Wesley Professional, 1er janvier 2011 (ISBN 9780321573513, lire en ligne (https://books.google.com/books?id=MTpsAQAAQBAJ))
5. **(en)** Kenneth Rosen, *Discrete Mathematics and Its Applications 7th edition*, McGraw-Hill Science, 14 juin 2011 (lire en ligne (https://books.google.com/books?id=6EJOCAAAQBAJ))
6. D. Cheriton et R. Tarjan, « Finding Minimum Spanning Trees », *SIAM Journal on Computing*, vol. 5, 1er décembre 1976, p. 724-742 (ISSN 0097-5397 (http://worldcat.org/issn/0097-5397&lang=fr), DOI 10.1137/0205051 (http://dx.doi.org/10.1137%2F0205051), lire en ligne (http://epubs.siam.org/doi/abs/10.1137/0205051))
7. **(en)** S. Dasgupta, C. H. Papadimitriou, and U. V. Vazirani, *Algorithms*

Bibliographie

- J.-F. Hêche, ROSO-EPFL, Cours SC de recherche opérationnelle : *Quelques problèmes classiques en théorie des graphes*
- Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest et Clifford Stein, *Introduction à l'algorithmique*, Dunod, 2002 [détail de l'édition]

Liens internes

- Algorithme de Kruskal
- Algorithme de Borůvka