Algorithmique Distribuée : TD 5, arbre couvrant en largeur d'abord

Alain Cournier Stéphane Devismes

1 Le problème

Nous nous intéressons au calcul d'un arbre couvrant en largeur enraciné en un processus R (R est appelée *racine*) d'un réseau $G = (V, E)$. Un *graphe partiel* de G est un graphe avec le même ensemble de nœuds V et dont l'ensemble d'arête E' est un sous-ensemble de E ($E' \subseteq E$). Un *arbre couvrant* est un graphe partiel de G acyclique et connexe. Un *arbre couvrant en largeur d'abord* de G, $T = (V, E_T)$, enraciné en R est un arbre couvrant de G tel que, pour tout processus p, l'unique chemin élémentaire de p à R dans T est de longueur $\|p, R\|$, où $\|p, R\|$ est la distance de p à R dans G (*i.e.*, la longueur du plus court chemin de p à R dans G). La hauteur d'un arbre enraciné est la longueur maximale d'un chemin élémentaire reliant la racine à une feuille (une feuille est un nœud de degré 1).

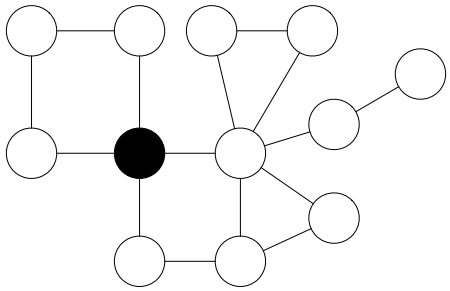

FIGURE 1 – Réseau quelconque.

Question 1. Proposez un arbre couvrant en largeur d'abord du graphe donné en figure 1. La racine étant le nœud noir. Quelle est la hauteur de votre arbre ?

Question 2. Combien y a-t-il de d'arbres couvrants en largeur d'abord possible avec comme racine le nœud noir ?

Nous rappelons que le *diamètre* d'un graphe est la distance maximale entre deux nœuds, c'est-à-dire, la longueur du plus long des plus court chemins.

Question 3. Quel est le diamètre du graphe proposé en figure 1 ? Quelle est la relation entre le diamètre d'un graphe connexe et la hauteur d'un arbre couvrant en largeur d'abord de ce graphe.

2 L'algorithme

Nous allons écrire un algorithme (autostabilisant) *silencieux* pour le calcul d'un arbre couvrant en largeur d'abord d'un réseau bidirectionnel connexe. Cet arbre sera enraciné au processus R. Nous considérons le modèle à états sous l'hypothèse d'un démon distribué inéquitable. Nous rappelons que pour tout processus p, l'ensemble des voisins de p est noté $\mathcal{N}_p$. Nous supposons également que les processus connaissent une borne supérieure D sur le diamètre du réseau. Ensuite, nous utilisons les variables suivantes :

— Chaque processus p détient une variable d_p, dont le domaine est $\{0, \ldots, D\}$.
Dans une configuration terminale, $d_p = \|p, R\|$.

— De plus, si p n'est pas la racine, alors p a en plus un pointeur $père_p$, dont le domaine est $\mathcal{N}_p$.
Dans une configuration terminale, $père_p$ désigne le « père » de p dans l'arbre, c'est-à-dire un voisin q tel que $\|q, R\| = \|p, R\| - 1$

Ainsi, le programme de la racine R contient uniquement la règle CD suivante :

$$CD :: d_R \neq 0 \mapsto d_R \leftarrow 0$$

En revanche, tout processus nonracine p dispose de deux règles : la première, CD, pour calculer d_p, la seconde, CP, pour calculer $père_p$. La seconde règle est activable seulement si la première ne l'est pas. La première règle consiste à affecter d_p à la valeur minimum entre les variables d de ses voisins plus 1 et D, si d_p ne vaut pas déjà cette valeur. La seconde règle consiste à désigner comme père un voisin q de distance d_q immédiatement inférieur (s'il existe).

Question 4. Écrivez les deux règles CD et CP pour un processus nonracine p.

Question 5. CD et CP sont-elles mutuellement exclusive pour un processus nonracine donné ? Justifiez.

3 Preuve de l'algorithme

L'algorithme étant silencieux, la preuve de correction consiste d'abord à montrer que les variables locales définissent un arbre couvrant en largeur dans toute configuration terminale (correction partielle). Puis, nous montrons que toute exécution atteint une configuration terminale en un nombre fini de pas de calcul (terminaison).

3.1 Correction partielle

Soit γ_t une configuration terminale.

Question 6. Démontrez, par contradiction, le lemme suivant.

Lemme 1. *Pour tout processus p, on a* $d_p \geq \|p, R\|$ *dans* γ_t.

Question 7. Démontrez, par récurrence sur la distance des processus à la racine R, le lemme suivant.

Lemme 2. *Pour tout processus p, on a* $d_p = \|p, R\|$ *dans* γ_t.

Corollaire 1. *Pour tout processus* $p \neq R$*, on a* $\min\{d_q, q \in \mathcal{N}_p\} = d_p - 1$ *dans* γ_t.

Preuve. Soit p un processus différent de R. Par définition, il existe un voisin q de p tel que $\|q, R\| = \|p, R\| - 1$. De plus, dans γ_t, $d_p = \|p, R\|$ et pour tout voisin q' de p, on a $d_{q'} = \|q', R\| \in \{\|p, R\| - 1, \|p, R\|, \|p, R\| + 1\}$ avec en particulier $d_q = \|p, R\| - 1$, d'après le lemme 2. Donc, $d_p - 1 = \|p, R\| - 1 = \min\{d_q, q \in \mathcal{N}_p\}$ dans γ_t. □

D'après le corollaire 1 et le fait que la garde de la règle CP qui est inactivable pour tout processus nonracine dans γ_t, on a :

Corollaire 2. *Pour tout processus* $p \neq R$*, on a* $d_{père_p} = d_p - 1$ *dans* γ_t.

Soit $T = (V, E_T)$, où $E_T = \{\{p, q\} \in E, q \neq R \wedge père_q = p \text{ dans } \gamma_t\}$.

Lemme 3 (Fermeture et correction)**.** T *est un arbre couvrant en largeur d'abord.*

Preuve. Nous démontrons tout d'abord que T un arbre couvrant en utilisant l'équivalence : « Un graphe de n nœuds est un arbre si et seulement s'il contient $n - 1$ arêtes et est acyclique ».

- T **est sans cycle :** Supposons, par contradiction, que T contient un cycle $p_0, \ldots, p_k, p_0$. Tout d'abord, par définition, R ne fait pas partie de ce cycle. Ensuite, supposons, sans perte de généralité, que pour tout $i > 0$, $père_{p_i} = p_{i-1}$ et $père_{p_0} = p_k$. D'après le corollaire 2, on a $d_{p_{i-1}} < d_{p_i}$ et par transitivité, $d_{p_0} < d_{p_k}$. Or, puisque $père_{p_0} = p_k$, on a $d_{p_k} < d_{p_0}$ d'après le corollaire 2, contradiction.
- E_T **contient** $n - 1$ **arêtes :** Suivant le même raisonnement que précédement, chaque arête de E_T est pointée par le pointeur $père$ d'un unique processus nonracine (deux processus adjacents ne peuvent désigner la même arête) : Supposons le contraire, soit deux voisins p et q tels que (1) $père_p = q$ et (2) $père_q = p$ D'après le corollaire 2, (1) implique $d_q < d_p$ et (2) implique $d_p < d_q$, contradiction.

  Ainsi, puisqu'il y a $n - 1$ processus nonracines, on a $|E_T| = n - 1$.

Nous montrons maintenant que T est en largeur d'abord. Soit $p_0 = R, \ldots, p_k$ l'unique chemin de R à p_k dans l'arbre. Par définition, pour tout $i > 0$, $père_{p_i} = p_{i-1}$. D'après le corollaire 2, on a $d_{p_{i-1}} = d_{p_i} - 1$ et par transitivité, on a $d_{p_0} = d_{p_k} - k$. De plus, $d_{p_0} = d_R = 0$. Donc, $d_{p_k} - k = 0$: d_{p_k} est égale à la longueur k du chemin entre R et p_k. Or, d'après le lemme 2, d_{p_k} est aussi égale à $\|R, p_k\|$. D'où la longueur k de l'unique chemin de p_k vers R dans T est égale à la distance de R vers p_k dans G. □

3.2 Terminaison

Soit $D_i = \{p \in V, d_p \leq i\}$, pour tout $i \in [0..D]$.

Question 8. Combien de fois D_0 peut être modifié ?

Question 9. Supposons que le contenu de D_k ne peut être modifié qu'un nombre fini de fois. Qu'en est-il alors de l'ensemble D_{k+1}, s'il existe ? (Justifiez votre réponse par une preuve)

Question 10. D'après la réponse à la question précédente, que concluez-vous au sujet de la règle CD ?

Question 11. Si le nombre d'exécution de la règle CD est finie dans toute l'exécution, que peut-on conclure sur le nombre d'exécution de la règle CP dans l'exécution ? (Justifiez)

D'après les réponses aux questions 7 à 10, nous pouvons conclure :

Lemme 4. *À partir d'une configuration quelconque, le système atteint une configuration terminale en un nombre fini de pas de calcul.*

À partir d'une configuration quelconque, le système atteint une configuration terminale en un nombre fini de pas de calcul d'après le lemme 4, cette dernière étant légitime d'après le lemme 3. Ainsi, nous pouvons conclure :

Théorème 1. *L'algorithme constitué des règles CD et CP est autostabilisant et silencieux pour le calcul d'un arbre couvrant en largeur sous l'hypothèse d'un démon distribué inéquitable.*

4 Temps de stabilisation en rondes de l'algorithme

Question 12. Considérons le réseau de $\mathcal{D} + 2$ processus $p_0, \ldots p_{\mathcal{D}+1}$ donné dans la figure 3. Initialement :
- La variable d de tous les processus nonracines p_i ($i \in [0..\mathcal{D} + 1]$) est égale à D.
- La variable $père_p$ de chaque processus nonracine p_i ($i \in [1..\mathcal{D} + 1]$) pointe sur p_{i-1}.

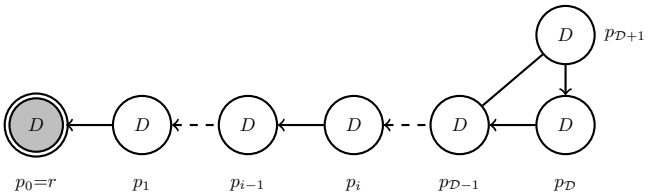

FIGURE 3 – Configuration initiale.

En supposant que $D > \mathcal{D}$, donnez la trace de l'exécution synchrone de l'algorithme à partir de cette configuration initiale pour $n = 7$ processus. Quel est le temps de stabilisation en rondes sur cet exemple ? Généralisez à tout réseau de cette forme.

Question 13. Reprenez l'exemple précédent avec $D = \mathcal{D}$. Que concluez-vous ?

Pour les questions suivantes, nous rappelons que nous considérons un démon dustribué inéquitable.

Question 14. Après une ronde, que peut-on dire de l'état de la racine ? de l'état des processus nonracine ?

Question 15. Que peut-on déduire de l'état des processus après $k > 0$ rondes ? après $\mathcal{D} + 1$ rondes ?

Question 16. Quel est le temps de stabilisation en rondes de l'algorithme (justifiez).