Master2 Informatique
UE: Algorithmique Distribuées – 4TIN913U
Responsable : M. Gavoille
Date : 4 janvier 2023
Durée : 1h30
Documents (course and personnal notes) allowed

Answers can be written in French or in English. In this examen, all graphs are supposed to be connected, even if not said explicitly.

Bellman-Ford

In the course, we saw that the distributed Bellman-Ford algorithm computes, in the asynchronous case, a shortest-path spanning tree rooted at given vertex r_0. Remind that the principle of the algorithm is, for a vertex u, to send the length d of the current path to r_0 to all its neighbors as soon as this distance decreases while updating its parent.

Recall that a graph is said to be *weighted* if each edge uv has a certain weight $\omega(uv) \in \mathbb{R}^+$. Obviously, a shortest-path spanning tree in a weighted graph must take into account the weight of its edges.

Question 1. *Explain what is a "shortest-path spanning tree rooted at r_0" in a weighted graph G.*

Solution. [3 pts] C'est un arbre T couvrant les sommets de G tel que pour tout sommet u, $d_T(r_0, u) = d_G(r_0, u)$ où $d_H(x, y)$ représente la distance entre dans le sous-graphe H entre u et v.

Question 2. *Give a description of the distributed Bellman-Ford algorithm, i.e., the pseudo-code for any vertex u, when the graph is weighted. You will assume that the root r_0 as well as $\omega(uv)$ for each v-neighbor of u, are known from u.*

Solution. [3 pts]

Algorithme Bellman-Ford(r_0)
(code du sommet u)

Layer(u) := $+\infty$

Répéter:

d := Receive(), et soit v l'émetteur
(si $u = r_0$, poser $v := \perp$ et $d := \omega(u\perp) := 0$)

Si $d + \omega(uv) <$ Layer(u):

1. Layer(u) := $d + \omega(uv)$
2. Parent(u) := v
3. Send(Layer(u), w) pour tout $w \in N(u) \setminus \{v\}$

We also saw that it was possible that, in some asynchronous scenario, Bellman-Ford produces $2^{\Omega(n)}$ messages for a certain n-vertex weighted graph, i.e., at least 2^{cn} messages when n is large enough and for a certain constant $c > 0$. The objective of the exercise is to show that in all cases, the algorithm cannot produce more than $2^{O(n \log \Delta)}$ messages, where Δ is the maximum degree of the graph.

Given a graph G, let $p(G)$ be the maximum number of different paths of G that start from the same vertex. In this context, observe that: (1) a path cannot pass twice through the same vertex; (2) a path can consist of only one vertex and no edges; and (3) the number of paths may differ depending on the originating vertex. Note that for a cycle on three vertices u, v, w, the paths $u - v - w$ and $u - w - v$ are different.

Question 3. *Compute $p(G_0)$ in the case where G_0 is the particular graph composed of two cycles on three vertices having one vertex in common. Specify the originating vertex realizing $p(G_0)$.*

SOLUTION. [3 pts] Il y a $p(G_0) = 13$ chemins en partant d'un sommet de degré 2. Plus précisément, il y a 1,2,4,4,2 chemins de longueur respective 0,1,2,3 et 4 arêtes. Pour le sommet de degré 4, c'est moins, les chemins étant de longueur 0, 1 ou 2.

Question 4. *Prove that, for any weighted graph G with n vertices, m edges and maximum degree Δ, the distributed Bellman-Ford algorithm cannot produce more than $\Delta + (2m - n) \cdot p(G)$ messages.*

SOLUTION. [3 pts] D'après l'algorithme, il y a émission de messages par u si le chemin courant de u vers r_0 change. Cela ne peut se produire qu'au plus $p(G)$ fois, sinon on passerait plusieurs fois par le même chemin ce qui est impossible car la longueur diminue strictement (le "$<$" strict est du coup très important). L'émission de messages de mise à jour (vers tous les voisins sauf le père) peut se produire (au pire) sur tous les sommets, sauf la racine, soit $\sum_{u \neq r_0}(\deg(u) - 1) \leqslant \sum_u(\deg(u) - 1) = 2m - n$ messages par mise à jour. On note que la racine émet $\deg(r_0) \leqslant \Delta$ messages au plus, tous les autres étant des messages de mise à jour. Au total cela fait donc au plus $\Delta + (2m - n) \cdot p(G)$ comme requis.

Let $P(n, \Delta)$ be the maximum of $p(G)$ for a graph G with n vertices and maximum degree Δ.

Question 5. *Prove that $P(n, \Delta) \leqslant 1 + \Delta \sum_{i=0}^{n-2}(\Delta - 1)^i$. [Hint: Consider the maximum number of paths, starting at the same vertex, and composed of exactly i edges.]*

SOLUTION. [3 pts] Pour $i = 0$, il y a exactement un chemin. Pour $i = 1$, il y en a Δ au plus. Pour $i = 2$, il y en a $\Delta - 1$ pour chaque chemin de longueur 1, soit $\Delta \cdot (\Delta - 1)$. Plus généralement, dès que $i > 1$, il y a au plus $\Delta - 1$ fois plus de chemins de longueur i que de chemins de longueur $i - 1$, ce qui en fait $\Delta \cdot (\Delta - 1)^{i-1}$. On note aussi que $i \leqslant n - 1$. D'où la formule: $P(n, \Delta) \leqslant 1 + \Delta + \Delta \cdot (\Delta - 1) + \Delta \cdot (\Delta - 1)^2 + \cdots + \Delta \cdot (\Delta - 1)^{n-2} = 1 + \Delta\left((\Delta - 1)^0 + \cdots + (\Delta - 1)^{n-2}\right) = 1 + \Delta \sum_{i=0}^{n-2}(\Delta - 1)^i$.

Let C_n denote the cycle graph on n vertices.

Question 6. *Compute $p(C_n)$ and show that the upper bound on $P(n, 2)$ in Question 5 is optimal for $\Delta = 2$.*

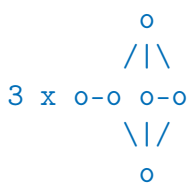
SOLUTION. [3 pts] Il est facile de voir que $p(C_n) = 1 + 2 \cdot (n - 1)$, chaque sommet étant identique et en considérant les chemins de longueurs $i = 0$ (il y en a 1) et les chemins de longueur $i = 1, ..., n - 1$ (il y en a deux à chaque fois). La borne supérieure pour $P(n, 2)$ donne $1 + 2 \cdot (1 + \cdots + 1) = 1 + 2 \cdot (n - 1)$, soit la même chose que $p(C_n)$.

Question 7. *Show that there exists a cubic graph G where $p(G) < 1 + 3 \sum_{i=0}^{|V(G)|-2} 2^i$. Recall that in a cubic graph, all the vertices have degree exactly three.*

SOLUTION. [3 pts] On peut considérer le graphe K_4 qui est cubic. Il faut considérer les chemins de longueur 3. La formule dit qu'il devrait en avoir $\Delta \cdot (\Delta - 1)^2 = 3 \times 2 \times 2 = 12$. Mais il y en a seulement $3 \times 2 \times 1$, car pour la dernière arête, il n'y a qu'un seul choix (et pas deux). D'ailleurs pour $K_{\Delta+1}$, il y a $\Delta \times (\Delta - 1) \times \cdots \times (\Delta - i)$ chemins de longueur $i + 1$ au lieu de $\Delta \times (\Delta - 1)^i$ comme dans la formule. Cela suggère une formule plus fine pour le cas général pour le chemin de longueur $n - i$ lorsqu'il reste moins de Δ sommets.

On peut aussi choisir un graphe cubic G qui n'est pas une clique et qui n'est pas hamiltonien, donc sans chemin de longueur $n - 1$. En effet, pas besoin de calculer le nombre exact de chemins. Si le dernier terme de la somme est nul, alors ce nombre sera strictement inférieur à la formule. On

peut choisir $n = 1 + 3 \times 5 = 16$ en prenant trois copies du graphe ci-dessous, qui sont fusionnés par le sommet pendant. Clairement ce graphe n'est pas hamiltonien à cause d'un sommet d'articulation.



Question 8. *From Question 4 and 5, show that for every weighted graph with n vertices and maximum degree Δ, the distributed Bellman-Ford algorithm produces at most $2^{O(n \log \Delta)}$ messages. [Hint: $\Delta^n = 2^{n \log_2 \Delta}$.]*

SOLUTION. [3 pts] D'après les questions précédentes, le nombre de messages M est au plus $\Delta + (2m - n) \cdot P(n, \Delta)$. On peut facilement majorer $P(n, \Delta) = O(\Delta^n)$, chaque terme de la somme étant au plus Δ et il y a au plus n termes. Puisque $m = O(n^2)$, cela fait au total $M = O(n^2 \cdot \Delta^n) = O(2^{2 \log n} \cdot 2^{n \log \Delta}) = O(2^{n \log \Delta + 2 \log n}) = 2^{O(n \log \Delta)}$.

Question 9. *Why can we say that the counterexample of the course, showing a scenario and a weighted graph producing $2^{\Omega(n)}$ messages is, in some way optimal? Discuss with pro and con arguments.*

SOLUTION. [3 pts] C'est parce que $\Delta = 4$ dans le contre-exemple. Du coup, $2^{\Omega(n)}$ et la borne supérieure en $2^{O(n \log \Delta)} = 2^{O(n)}$ sont relativement proches (argument pour). D'un autre coté, 2^n et 2^{cn} sont (polynomialement) différents, comme par exemple $c = 2$, car si $N = 2^n$, alors $2^{2n} = N^2$. De plus (argument contre), il aurait fallut trouver un contre-exemple valable pour chaque Δ, et pas seulement $\Delta = 4$ voir $\Delta = O(1)$.

Subgraph Detection

Question 10. *Give an example of distributed task (i.e., a distributed problem) that is feasible in t rounds in the LOCAL model (for some $t \in \mathbb{N}$), and not in the CONGEST model.*

SOLUTION. [3 pts] La racine d'un arbre de hauteur deux doit récupérer la liste des identifiants de toutes ces feuilles. Dans le modèle LOCAL $t = 2$, mais c'est $\Omega(n/b)$ rondes dans le modèle b-CONGEST. Il y a aussi le problème de détecter un C_4, où l'on a montré qu'un temps $\Omega(\sqrt{n})$ est nécessaire pour b-CONGEST, alors que deux rondes suffisent dans le modèle LOCAL.

In the course, we studied distributed algorithms to detect cycles of length 3 or 4, namely C_3 or C_4, in a graph G.

Question 11. *More generally, for a fixed small graph H, explain what does mean "to detect a H in G" in the distributed setting. Specify the inputs and the outputs.*

SOLUTION. [3 pts] Le problème est de déterminer si oui ou non G contient une copie de H comme sous-graphe. Chaque sommet u possède une variable DETECT_u initialisée à FALSE. À la fin de l'algorithme, chaque variable $\text{DETECT}_u \in \{\text{TRUE}, \text{FALSE}\}$ telle que G possède un H comme sous-graphe si et seulement si il existe un sommet u avec $\text{DETECT}_u = \text{TRUE}$.

Let us define the *radius* of a graph G, denoted by $\text{rad}(G)$, as the minimum depth of a spanning tree of G. In other words, it is the minimum eccentricity of G, that is $\text{rad}(G) = \min_{u \in V(G)} \{\text{ecc}_G(u)\}$. Here the edges have no weights, their cost is one.

Question 12. *Give the radius of a 3×3-grid, the graph with 9 vertices obtained from a mesh of 3 rows and 3 columns. In addition to the value of the radius, draw the graph as well as a minimum depth spanning tree. What is the diameter of the graph? Same question for a 4×4-grid.*

SOLUTION. [3 pts] 3×3-grid: le rayon est 2 et son diamètre est 4
4×4-grid: le rayon est 4 et son diamètre est 6

Question 13. *Give a distributed algorithm, in the LOCAL model, to detect a C_5 and specify its number of rounds (the smallest possible one).*

SOLUTION. [3 pts] Il faut trois rondes, à chaque ronde on transmet tout ce qu'on connaît. On apprend ainsi tous les identifiants des sommets à distance 1,2 et 3, ainsi que les arêtes les connectant, sauf les arêtes connectant deux sommets à distance trois. Deux rondes ne permettent pas de détecter l'arête à distance deux du C_5, si elle existe. NB. Initialement, avant la première ronde, on ne connaît que son identifiant.

Question 14. *Give the principle of a distributed algorithm, in the LOCAL model, to detect a graph H that is supposed to be known from all the vertices. Specify the smallest possible number of rounds expressed as a fonction of* $\mathrm{rad}(H)$.

SOLUTION. [3 pts] Le nombre de rondes est $\mathrm{rad}(H) + c(H)$ où $c(H) \in \{0, 1\}$ et $c(H) = 1$ ssi H contient un cycle de $2\mathrm{rad}(H) + 1$ sommets. Pour $H = C_5$, on a vu que deux rondes ne suffisent pas. On a en effet $\mathrm{rad}(C_5)$ et $c(C_5) = 1$. Le principe est une généralisation de l'algorithme pour C_5. Chaque sommet u calcule localement un sommet racine r de H d'excentricité minimum $\mathrm{rad}(H)$, ainsi que $c(H)$. Puis, pendant $t = \mathrm{rad}(H) + c(H)$ rondes, le sommet u recolte son voisinage à distance t (identifiants et arêtes) ce qui représente un certain sous graphe G_u. Après ces t rondes, il vérifie si G_u contient H comme sous-graphe ayant $r = u$ comme racine. Si H apparaît dans G, il doit y avoir un sommet u qui va prendre la position de r dans H et observer une copie de H dans G_u. Et bien sûr, si H n'apparaît pas, aucun sommet u ne pourra obterver H dans G_u.

We have seen in the course that every distributed algorithm in the b-CONGEST model that detects a C_4 in a graph with n vertices requires $\Omega(\sqrt{n}/b)$ rounds. Recall that it was partially based on the fact there are graphs without any C_4 and with $m \geqslant n^{1+1/2}$ edges. For the next question, whose goal is to generalize this result, we will assume that, for each integer $k \geqslant 1$, there is a graph G_k with n vertices and $m \geqslant n^{1+1/k}$ edges without cycles of length $2k$ or less.

Question 15. *Using an Alice-Bob simulation communication complexity argument based on the* SET-DISJOINTNESS *problem, explain why every distributed algorithm that detects a C_{2k} requires $\Omega(n^{1/k}/b)$ rounds in the b-CONGEST model.*

SOLUTION. [3 pts] On prend deux copies G_A, G_B de G_k dans lesquels on ne garde que des arêtes d'un ensemble d'arêtes A pour G_A et B pour G_B. Les graphes G_A et G_B sont alors connectés par des chemins de $k-1$ arêtes (on connecte les sommets homologues). Le graphe G obtenu possède $2n + n(k-2)$ sommets et au plus $2m + n(k-1)$ arêtes. Il est facile de voir que G possède un C_{2k} si et seulement si une arête $u - v$ est présente dans G_A et G_B, c'est-à-dire dans $A \cap B \neq \varnothing$. À partir de la simulation d'un algorithme de détection de C_{2k} dans G, Alice (qui contrôle G_A et les arêtes des chemins de connexions) et Bob (qui contrôle G_A et les arêtes des chemins de connexions) pourront en déduire un protocole de communication pour le problème SET-DISJOINTNESS entre A et B qui sont des sous-ensembles de $m \geqslant n^{1+1/k}$ arêtes. D'après le cours, on a vu qu'il faut échanger m bits pour ce dernier problème, quel que soit le protocole. Si l'algorithme distribué utilise t rondes de b bits sur chacun des n chemins connectant G_A à G_B, alors le protocole qu'Alice et Bob déduiront de cette simulation utilisera tbn bits. On doit avoir $tbn \geqslant m \geqslant n^{1+1/k}$, ce qui implique bien $t \geqslant n^{1/k}/b$. Notons qu'en terme du nombre de sommets de G, on $|V(G)| = N = \Theta(nk)$. La borne est donc $t = \Omega((N/k)^{1/k}/b) = \Omega(N^{1/k}/b)$ pour chaque k fixé.

FIN.