Algorithmique 27 octobre 2004

Correction TD 6- Composantes fortement connexes

1 Composantes fortement connexes d'un graphe orienté :

1.1 Parcours en profondeur

1.1.1 Complexité et relation entre les sommets

L'initialisation se fait en $\mathcal{O}(|S|)$. Grâce au système de coloriage, il est évident que l'on appelle la procédure **Visiter_PP** sur chaque sommet une fois et une seule, soit directement depuis **PP**, soit depuis le traitement d'un autre sommet. Le traitement d'un sommet se fait en $\mathcal{O}(1)$ plus une comparaison pour chaque arc qui part de ce sommet. Donc chaque arc est également visitée une fois et une seule car elle est visitée uniquement depuis son sommet de tête et chaque sommet n'est traité qu'une fois. La complexité du parcours est donc clairement en $\mathcal{O}(|S| + |A|)$.

Soient u et v deux sommets du graphe. On suppose $d[u] < d[v]$, dans le cas contraire il suffit d'inverser les notations. On distingue deux cas :

1. v est un descendant de u. On rencontre alors v avant d'avoir terminé u, et on termine de l'explorer avant également. Donc
$$d[u] < \quad d[v] < f[v] \quad < f[u]$$
2. v n'est pas un descendant de u, ce qui signifie que l'on termine l'exploration de u avant de commencer celle de v. D'où :
$$d[u] < \quad f[u] < d[v] \quad < f[v]$$

1.1.2 Lemme du chemin blanc

Lemme : v est un descendant de u si et seulement si il existe un *chemin blanc* de u à v à l'instant $d[u]$

preuve : Soit v un descendant de u. Il existe donc un chemin de u à v emprunté lors de l'exécution de l'algorithme. L'algorithme n'empruntant que des sommets blancs, tous les sommets de ce chemin étaient blanc à l'instant $d[u]$. Il existe donc un chemin blanc de u à v à l'instant $d[u]$.

Supposons la réciproque fausse : si il existe un chemin blanc de u à v, v n'est pas forcément descendant de u. Prenons alors v comme le sommet le plus proche de u tel qu'il existe un chemin blanc de u à v à $d[u]$ et tel que v ne soit pas un descendant de u. Prenons w le sommet précédent v sur ce chemin. On a donc w descendant de u (ou $w = u$) ce qui implique $f(w) \leq f(u)$. u est donc encore blanc à l'instant $d[w]$ Donc pendant l'exécution de w, soit v est blanc quand on considère l'arc $w \rightarrow v$, dans ce cas on le traite à ce moment là, et $d[w] < d[u] < f[w]$, soit il est noir à ce moment là et donc $d[w] < f[u] < f[w]$. Comme il n'y a que deux configurations possible pour les inégalités entre les débutsfins de traitement, on a dans les deux cas $d[w] < d[u] < f[u] < f[w]$. Autrement dit, v est un descendant de w, et donc aussi de u par transitivité. D'où contradiction avec l'hypothèse de départ.

1.2 Composantes fortement connexes

1.2.1 Relation d'équivalence

Il est facile de montrer que R est une relation d'équivalence. Par convention, un sommet x a un chemin de longueur nulle vers lui-même, d'où xRx. D'autre part si xRy et yRz alors il existe un chemin de x à y et un chemin de y à z. En composant les deux on a donc un chemin de x à z. De même, comme il existe un chemin de z à y et un chemin de y à x, il existe un chemin de z à x, donc xRz. Enfin comme xRy implique qu'il y a un chemin de x à y *et* un chemin de y à x, on a, par définition yRx. R est bien une relation d'équivalence.

1.2.2 Les composantes fortement connexes sont dans le même arbre

Soit C une classe composante fortement connexe. Prenons $u \in C$ tel que $d[u] = min_{x \in C}(d[x])$. A $d[u]$ tous les sommets de C sont blancs (sauf u qui est gris), donc $\forall x \in C$, il existe un chemin blanc de u à x. Donc d'après le lemme du chemin blanc, x est descendant de u. Donc tous les sommets de C (à part u) sont descendants de u. Donc tous les sommets de C sont dans le même arbre de profondeur.

1.2.3 L'aïeul est un ancêtre

On a donc $\phi(u) = w$ successeur de u et quelque soit v successeur de u, $f(v) \leq f(w)$.

Si à $d[u]$ tous les successeurs de u sont blancs, alors on a clairement $w = u$, car d'après le lemme du chemin blanc, tous les autre successeurs seront des fils de u. $\phi(u)$ est donc bien un ancêtre de u.

Sinon, à $d[u]$, w est gris ou noir. Si il est noir : $f[w] < d[u] < f[u]$, ce qui contredit la définition de w. Donc w est gris, ce qui implique la relation $d[w] < d[u] < f[w]$, c'est à dire que u est un descendant de w, donc $\phi(u)$ est bien un ancêtre de u.

$\phi(u)$ est donc à la fois un successeur et un ancêtre de u. Ces deux sommets font donc partis de la même composante fortement connexe. De manière plus général u et v font partie de la même CFC si et seulement si $\phi(u) = \phi(v)$.

1.3 Algorithme pour les composantes fortement connexe

Voyons de manière informelle ce que fait l'algorithme décrit. Tout d'abord, il fait un parcours en profondeur classique. Ainsi à tout sommet u, il associe l'instant $f(u)$ de fin d'exécution. A la fin de cet algorithme, on a donc plusieurs arbres de profondeurs. Tous

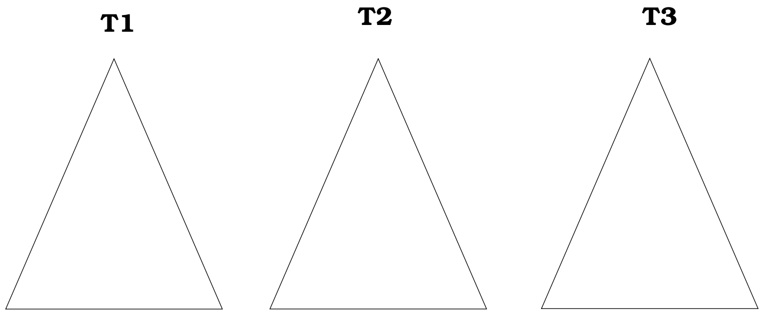


FIG. 1 – Après exécution du premier parcours

les sommets d'un même arbre sont donc atteignables depuis la racine. De même, si on prend un sous-arbre de l'un des arbres de profondeurs, tous les sommets de ce sous-arbre sont atteignable depuis la racine de ce sous-arbre. Ensuite, on change le sens des arcs G et on effectue un parcours en profondeur par f décroissant. On va forcément commencer par la racine du dernier arbre de profondeur (dans la figure 1, il s'agit du sommet T_3). Chaque arbres généré par ce second algorithme sera entièrement inclus dans un des arbres du premier algorithme. En effet, supposons qu'un sommet de T_2 est atteignable depuis T_3. Celà signifie que dans G, il y a un arc entre un sommet u de T_2 et un sommet v de T_3, ce qui est impossible, car à l'exécution du premier algorithme, à l'instant $d(u)$, v était blanc, et donc il aurait été visité et donc v aurait appartenu à T_2. Supposons maintenant qu'un sommet de T_2 est atteignable depuis T_3. Alors, ce sera lors du traitement du sommet de T_2 qu'on pourra atteindre celui de T_3. Mais les sommets étant traités par f décroissants, le sommet de T_3 aura déjà été traité, donc sera noir, et ne sera donc pas visité. Les nouveaux arbres sont donc inclus dans les anciens comme le montre la figure 2.

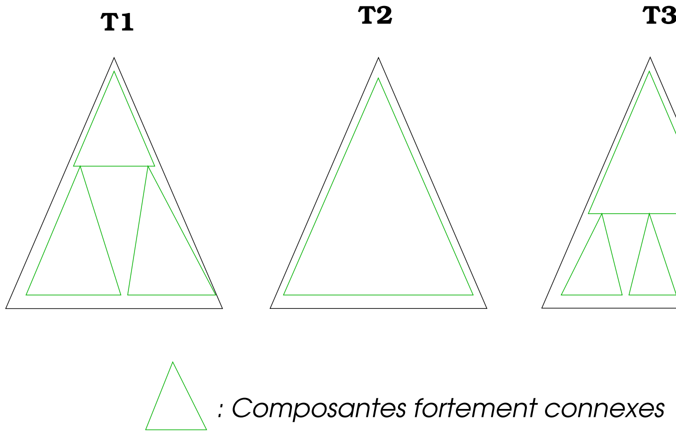


FIG. 2 – Après exécution du second parcours

Les arbres générés par ce deuxième algorithme, au moins aussi nombreux que ceux générés par le premier algorithme, regroupent un ensemble de sommets qui sont tous atteignables depuis la racine, et qui peuvent tous atteindre la racine. Ce sont donc effectivement les composantes fortement connexe de G (et aussi de ${}^tG$). On va le démontrer plus formellement par récurrence sur les arbres de profondeur de ${}^tG$.

Supposons que tous les arbres obtenus jusqu'à présent sont des composantes fortement connexes de G. Notons r la racine du prochain arbre T. Définissons l'ensemble $C(r) = \{v \in S \mid \phi(v) = r\}$. Il s'agit donc de la composante fortement connexe de r.

Montrons que $u \in T \Leftrightarrow u \in C(r)$.

$(\Leftarrow)$: r étant la racine de T, si $\phi(u) = r$ alors r est un successeurs de u et surtout, il n'existe pas de successeurs v de u tels que $f[v] \leq f[r]$. Donc u ne peut pas appartenir à un arbre créé avant T, donc $u \in T$.

$(\Rightarrow)$: si $u \in T$, supposons $\phi(u) \neq r$. Alors, soit $f(\phi(u)) > r$ soit $f(\phi(u)) < r$.

- si $f(\phi(u)) > r$, alors $\phi(u)$ a déjà été visité avant le traitement de r. Et comme u et $\phi(u)$ sont dans la même CFC, alors par hypothèse de récurrence, ils appartiennent au même arbre de profondeur. Donc $u \notin T$, absurde.
- sinon, on a $f(\phi(u)) < r$, et r descendant de u car $u \in T$, ce qui est impossible par définition de ϕ.

Il ne reste plus qu'à démontrer que le premier arbre de profondeur de ${}^tG$ est bien une composante fortement connexe. Notons T ce premier arbre et r sa racine. Tous les sommets de T peuvent atteindre r dans G par définition de ${}^tG$, et comme $f[v]$ est le plus grand f, alors quelque soit $u \in T$, $\phi(u) = r$. Donc tous les sommets de T font partie de la même CFC.

Maintenant, supposons un sommet v appartenant à cette CFC. Alors on a $\phi(v) = r$ donc, il existe un chemin de v à r dans G, donc un chemin de r à v dans ${}^tG$. Ce chemin est forcément blanc au début de l'exécution de l'algorithme, donc v est un fils de r. Ainsi $v \in T$.

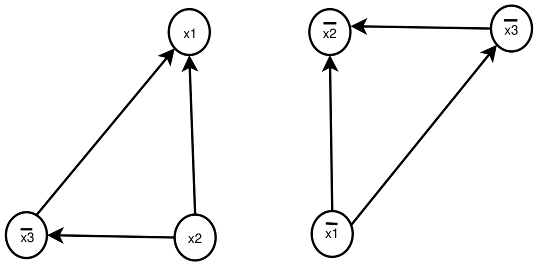
On a donc démontré la validité de l'algorithme. Calculons rapidement sa complexité. La complexité du parcours en profondeur est $\mathcal{O}(|S| + |A|)$, l'inversion des arcs se fait en $\mathcal{O}(|A|)$. Comme G et ${}^tG$ ont le même nombre d'arcs et de sommets, on a donc :

$$\begin{aligned}\mathcal{C}(G(S,A)) &= \mathcal{O}(|S| + |A|) + \mathcal{O}(|A|) + \mathcal{O}(|S| + |A|) \\ &= \mathcal{O}(|S| + |A|)\end{aligned}$$

2 2-SAT

2.1 Une condition nécessaire :

- On représente la clause $l \vee l'$ par deux liaisons :
  $\neg l' \to l$
  $\neg l \to l'$
  Ainsi, l'exemple $(x_1 \vee \neg x_2) \wedge (x_1 \vee x_3) \wedge (\neg x_2 \vee \neg x_3)$ sera représenté par le graphe :



  Une condition nécessaire pour que le graphe soit satisfiable est :
  y et $\neg y$ sont dans deux composantes connexes distinctes
- Montrons qu'une valuation v qui satisfait la formule F est constante sur une composante fortement connexe du graphe associé G.
  Soit v qui satisfait F, et supposons x et y sur la même composante connexe. Alors il existe un chemin de x à y, i.e. une série d'implications de x à y, et pout tout w de ce chemin, on a $v(x) = v(w)$. En particulier, $v(x) = v(y)$.
  Donc v est constante sur chaque composante connexe.

2.2 Graphe des composantes fortement connexes :

- On construit H de sorte que les sommets de H sont les composantes fortement connexes de G, et qu'il existe un arc de X à Y dans H s'il existe un arc d'un sommet de X vers un sommet de Y dans G.

– Montrons que H n'a pas de circuit.
Par l'absurde, s'il existe un circuit dans H, alors les composantes fortement connexes rencontrées sur ce circuit ne forment qu'une seule composante.
En effet, soit X et Y deux de ces composantes. Il existe un chemin de X à Y dans H, donc il existe un chemin de tout sommet de X à tout sommet de Y dans G, i.e X et Y forment même composante connexe.
Donc H est sans circuit.

2.3 Tri topologique :

– Chaque composante connexe X de G est représentée par son aïeul x. L'ordre partiel impose que si $X \to Y$ dans H, alors $f(x) < f(y)$. Il suffit donc de montrer que pour tous $u, v \in G$, on a :
$u \to v \Rightarrow f(u) > f(v)$
On se place au moment où on explore l'arc $u \to v$:
  – si v est blanc : u antécédent de $v \Rightarrow f(v) < f(u)$
  – si v est noir : $f(v) < f(u)$ car v a déjà été visité
  – si v est gris : u sera noir avant v donc $f(u) < f(v)$, donc on a :
  $d(v) < d(u) < f(u) < f(v) \Rightarrow u$ descendant de v
  Il existe donc un cycle orienté entre u et v. Soit X la composante connexe contenant u et Y celle contenant v. Alors il existe un cycle entre X et Y, ce qui est impossible car H est sans cycle.
Donc l'ordre total étend bien l'ordre partiel.
– Complexité de la construction de H :
  – On calcule les composantes fortement connexes de G en $O(|S|+|A|)$
  A chaque $u \in S$, on associe $\phi(u)$.
  – H est défini sur $\{\phi(u)\}$, et il faut parcourir tous les adjacents de chaque sommet u de G en ajoutant dans H :
  $\phi(u) \to \phi(v)$ pout tout $u \to v$
Donc H se construit en $O(|S|+|A|)$.
– On définit la valuation v par : si $x \in X$ et $\neg x \in Y$, alors $v(x) = faux$ si $X < Y$ et $v(x) = vrai$ sinon.
Cette définition est cohérente :
  – si $X < Y$, alors $v(x) = faux$ (et $vrai \Rightarrow Y$ toujours vrai)
  – si $Y < X$, alors $v(x) = vrai$ (et $Y \Rightarrow vrai$ toujours vrai)
– Ainsi, la complexité de $2 - SAT$ est de l'ordre de $O(|S|+|A|)$.