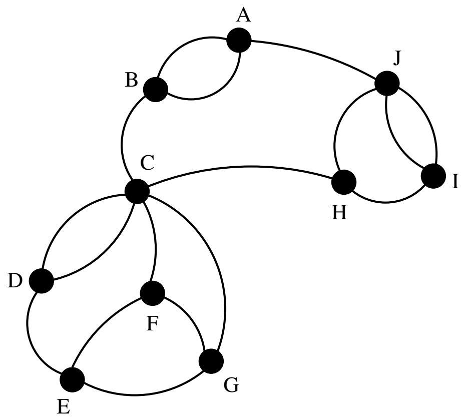
IREM de Clermont-Ferrand
Formation ISN
Algorithmique

20 octobre 2011
Malika More
`malika.more@u-clermont1.fr`

Forte connexité

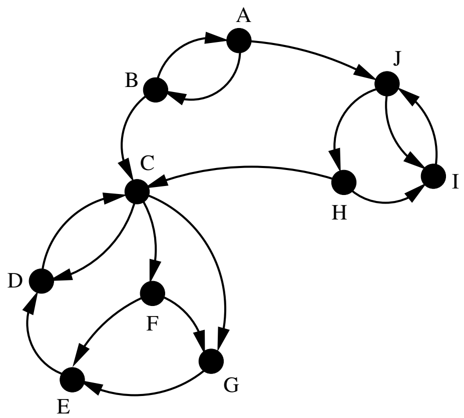
1 Introduction

Graphe non orienté
On peut parcourir les arêtes dans les deux sens



- Un graphe (non orienté) est **connexe** lorsqu'il n'est composé que d'un seul « morceau »
- S'il y a plusieurs morceaux, on parle des **composantes connexes** du graphe
- Les sommets d'une composante connexe sont mutuellement accessibles
- Un parcours permet de déterminer la composante connexe du sommet de départ

Graphe orienté
- Chaque arc a un sens (flèche)
- On ne peut suivre un arc que dans le sens de la flèche



La notion de connexité est différente

- On peut aller de J à C, mais pas de C à J
- Si tous les couples de sommets d'un graphe orienté sont mutuellement accessibles, on dit qu'il est **fortement connexe**
- Le graphe orienté ci-dessus comprend plusieurs **composantes fortement connexes**
- Comment calculer la composante fortement connexe d'un sommet ?

Un algorithme simple mais long

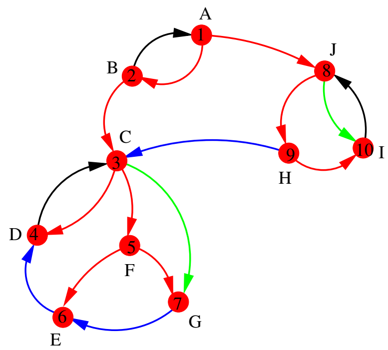
- Trouver tous les sommets accessibles à partir de X en utilisant un parcours
- Pour chaque sommet Y accessible à partir de X, utiliser un parcours pour vérifier si X est aussi accessible à partir de Y

2 Algorithme de Tarjan

Idée générale

- On effectue un parcours en profondeur à partir d'un sommet du graphe à l'aide d'une pile `Parcours` en mémorisant le rang de `Découverte` de chaque sommet visité
- Pendant le parcours, on calcule les composantes fortement connexes rencontrées à l'aide d'une deuxième pile `Tarjan` et d'un `Index` associé à chaque sommet visité

Préliminaire

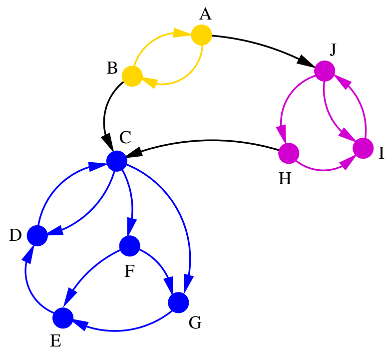


- Parcours en profondeur à partir de A et rang de `Découverte`
- Arc de liaison : rouge
- Arc avant : CG, JI
- Arc arrière : BA, DC, IJ
- Arc transverse : GE, ED, HC

Les 5 règles de calcul

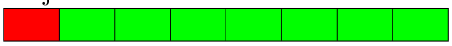
1. La première fois qu'on rencontre un sommet X, on l'empile sur `Tarjan` et on initialise son `Index` avec la valeur `Découverte`(X)
2. Quand on visite à partir de X un arc arrière (i.e. qui revient vers un sommet Y déjà visité et stocké dans la pile `Parcours`), on met à jour `Index`(X)=`Min`(`Index`(X), `Index`(Y))
3. Quand on visite à partir de X un arc transverse (i.e. qui revient vers un sommet Y déjà visité et tel que `Index`(Y)<`Index`(X)), à condition que Y soit stocké dans la pile `Tarjan`, on met à jour `Index`(X)=`Min`(`Index`(X), `Index`(Y))
4. Quand on revient à X après avoir dépilé de la pile `Parcours` son successeur Y, on met à jour `Index`(X)=`Min`(`Index`(X), `Index`(Y)) pour que X hérite éventuellement de l'index calculé pour Y
5. Au moment de supprimer un sommet X de la pile `Parcours`, si on a `Index`(X)=`Découverte`(X), alors X et tous les sommets empilés au-dessus de lui dans la pile `Tarjan` forment une composante fortement connexe de graphe : on les dépile tous de `Tarjan` et bien sûr on dépile seulement X de `Parcours`

Exemple



Parcours :

Tarjan :



Sommet	A	B	C	D	E	F	G	H	I	J
Découverte	1	2	3	4	6	5	7	9	10	8
Index	1	1	3	3	3	3	3	8	8	8

Nécessité d'une preuve

- On se convainc assez facilement que l'algoritmhe de Tarjan permet de calculer les composantes fortement connexes d'un graphe orienté
- Mais la preuve que c'est bien le cas est loin d'être évidente...

Autre exemple