Travaux pratiques

Recherche de flot de flux maximal en PHP

L'objectif de ce projet est de réaliser en PHP un programme affichant en détail l'algorithme de Ford-Fulkerson permettant de déterminer un flot de flux maximal.

Le travail est divisé en différente tache qu'il faudra réaliser pour capitaliser des points. Si certaines taches ne sont aboutis des points négatifs pourront être attribués. Référez-vous en annexe au *Tableau synthétique des taches attendues et optionnelles* pour plus de détail.

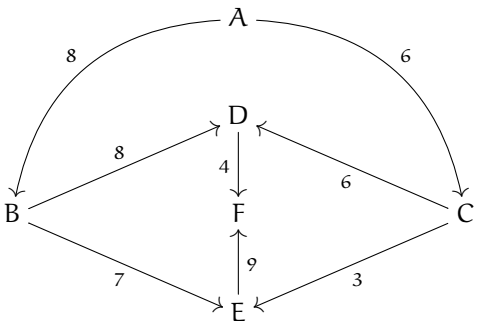
Variables & Co.

La saisie.

Un réseau sera considéré du point de vue matricielle.

Reprenons l'exemple du cours (on a changer l'ordre des sommet).

La représentation sagittale est



Nous allons considérer la matrice booléenne de ce réseau que nous allons augmenter par la valeur de la valuation. Nous luis faisons donc correspondre la matrice

	A	B	C	D	E	F
A	0	8	6	0	0	0
B	0	0	0	8	7	0
C	0	0	0	6	3	0
D	0	0	0	0	0	4
E	0	0	0	0	0	9
F	0	0	0	0	0	0

L'ordre des sommets n'importe pas.

Nous aurons donc en entrée cette matrice booléenne augmenté par la valuation représentant notre réseau. Cette donnée est saisie par l'utilisateur. Vous récupérer donc cette matrice que nous appellerons tout au long du programme `$res`.

Le flot.

Pour modéliser le flot nous utiliserons également une matrice que nous nommerons `$flot`

Le flot est initialisé à 0. On a donc la matrice

	A	B	C	D	E	F
A	0	0	0	0	0	0
B	0	0	0	0	0	0
C	0	0	0	0	0	0
D	0	0	0	0	0	0
E	0	0	0	0	0	0
F	0	0	0	0	0	0

A la fin de l'algorithme (*cf.* cours) le flot est

	A	B	C	D	E	F
A	0	7	6	0	0	0
B	0	0	0	1	6	0
C	0	0	0	3	3	0
D	0	0	0	0	0	4
E	0	0	0	0	0	9
F	0	0	0	0	0	0

Le chemins améliorants.

La plus grosse difficulté de cet algorithme réside dans le fait de chercher dans le réseau un chemin améliorant. Puisque nous proposons une solution informatique nous n'allons pas raisonner comme sur le papier. Il parait plus judicieux d'utiliser la puissance de calcul de l'ordinateur. Nous allons donc lister tous les chemins possible. Il suffira alors à chaque étape de choisir le meilleur. Les chemins que nous considérerons devront répondre à trois critères :

- Commencer à la source.
- Finir au puits.

– Ne pas passer deux fois par le même sommet.

Nous allons construire une variable contenant l'ensemble des chemins possibles que nous nommerons `$e_chem` qui sera un tableau à 2 colonnes.

1. La première colonne contiendra le chemin sous forme d'un tableau d'indice. Le chemin "EDA" correspondant au tableau $[4, 3, 0]$.
2. La seconde colonne contiendra la valeur d'amélioration du flux.

Dans notre exemple la variable `$e_chem` a la forme suivante :

$[0, 1, 3, 2, 4, 5]$	0
$[0, 1, 3, 5]$	4
$[0, 1, 4, 2, 3, 5]$	0
$[0, 1, 4, 5]$	7
$[0, 2, 3, 1, 4, 5]$	0
$[0, 2, 3, 5]$	4
$[0, 2, 4, 1, 3, 5]$	0
$[0, 2, 4, 5]$	3

"ABDCEF"	0
"ABDF"	4
"ABECDF"	0
"ABEF"	7
"ACDBEF"	0
"ACDF"	4
"ACEBDF"	0
"ACEF"	3

La grosse difficulté de la recherche de chemin améliorant est donc divisée en deux taches un peu plus facile :

1. Initialisé la variable `$e_chem`. Notamment la recherche de tous les chemins, qui est *un peu* technique.
2. Mettre à jour cette variable - uniquement la dernière colonne

L'algorithme de Ford-Fulkerson.

Une fois les variables initialisées, l'algorithme se résume à

```
Tant qu'il y a des cases val non nul dans la variable $e_chem faire
     Prendre le chemin qui a le plus grand val
          - mettre à jour la variable $flot
          - incrementer le $flux de la valeur val
          - mettre à jour la variable $e_chem
          - garder cette opération en mémoire pour afficher le détail.
             A cette fin on pourra créer une variable $etape
             qui sera un tableau où chaque case contient elle
             même un tableau ou le premier champ est le chemin
             améliorant, le second champs le nouveau flux et
             le troisième la variable de $flot.
Fin Tant que
```

Voir en annexe le *déroulé de l'exemple.*

Le théorème de Ford-Fulkerson.

Pour prouver que le flot trouvé est de flux maximal, on applique le théorème de Ford-Fulkerson sur les coupes. Le programme justifiera que le flot trouvé est bien le plus grand en présentant une coupe dont la valeur vaut le flux trouvé.

Tableau synthétique des taches attendues et optionnelles

La colonne '+' indique le nombre de point gagné en réalisant la mission demandée.
Le colonne '-' indique le nombre de point perdu si la mission n'est pas réalisée/achevée/opérationnelle.

+	-	Nom	Description
0	2	`RecupRes($_POST)`	Prend en paramètre une variable qui est un retour d'envoi de formulaire par la méthode `$_POST`. On prendra soin de s'assurer que l'utilisateur n'a envoyé que des nombres. La valeur de retour est la matrice (tableau de tableaux) `$res`.
1	1	`IsReseau($res)`	Renvoie `true` si le graphe représenté par la matrice `$res` est un réseau, `false` sinon.
1	1	`InitFlot($res)`	Renvoie la variable `$flot`, initialisée à 0.
3	1	`ValChem($chem, $res, $flot)`	Prend en paramètre un chemin (tableau d'indice), le réseau et le flot actuel et détermine la valeur du chemin. La fonction renvoie ce nombre.
3	10	`InitE_chem($res)`	Prend en paramètre le réseau et renvoie la variable `$e_chem` initialisée.
3	1	`MajFlot($chem, $res, $flot, $x)`	Cette fonction renvoie la variable `$flot` mise à jour par l'augmentation ou la diminution de `$x` le long du chemin `$chem`
1	1	`MajE_chem($e_chem, $res,$flot)`	Cette fonction renvoie la variable `$e_chem` mise à jour.
0	10	`AlgoFordFulkerson($res)`	Cette fonction renvoie le tableau `$etape` détaillé dans le descriptif de ce projet.
2	1	`ValCoupe($res, $x, $xc)`	Renvoie le nombre correspondant à la valeur de la coupe entre les deux ensembles passées en paramètre.
2	1	`ListeCoupe($res)`	Renvoie la liste de toutes les coupes possible dans un tableau à deux lignes. La première est l'ensemble X le second l'ensemble $\overline{X}$ (*cf.* cours).
5	5	IHM	Proposer une interface de travail. Vous pourrez intégrer du LaTeX via *MathJaX*
3	3	D3JS	Cette bibliothèque de javascript propose des outils très riche pour faire des graphes - il devrait être possible d'afficher sagittalement le réseau.
1	0	Logo	Proposer un logo à votre projet. L'image doit être libre de droit et au format svg (image vectorielle).

Si ces missions sont réalisées correctement, l'algorithme de Ford-Fulkerson se résume à ces 30 lignes de code.

```
1  function AlgoFordFulkerson($res){
2
3     $etape=array();
4     if(!IsReseau($res)) return $etape;
5
6     //Initialisation des variables
7     $flot=InitFlot($res);
8     $flux=0;
9     $e_chem=InitE_chem($res);
10    $etape[]=array(array(), $flux, $flot);
11
12    while(true){
13       //On récupère le chemin améliorant
14       $m=0;
15       foreach($e_chem as $chem){
16          if($chem[1]>$m) {
17                $chem_a=$chem[0];
18                $m=$chem[1];
19          }
20       }
21       if($m==0) break;
22
23       //Maj des variables
24       $flot=MajFlot($chem_a, $res, $flot, $m);
25       $flux+=$m;
26       $e_chem=MajE_chem($e_chem, $res, $flot);
27       $etape[]=array($chem_a, $flux, $flot);
28    }
29    return $etape;
30 }
```

Déroulé de l'exemple

On présente le déroulé de l'exemple.

`$etape[0].` (initialisation)

Chemin améliorant : []

`$flux`=0

`$flot`=

	A	B	C	D	E	F
A	0	0	0	0	0	0
B	0	0	0	0	0	0
C	0	0	0	0	0	0
D	0	0	0	0	0	0
E	0	0	0	0	0	0
F	0	0	0	0	0	0

`$e_chem`=

[0, 1, 3, 2, 4, 5]	0	"ABDCEF"	0
[0, 1, 3, 5]	4	"ABDF"	4
[0, 1, 4, 2, 3, 5]	0	"ABECDF"	0
[0, 1, 4, 5]	7	"ABEF"	7
[0, 2, 3, 1, 4, 5]	0	"ACDBEF"	0
[0, 2, 3, 5]	4	"ACDF"	4
[0, 2, 4, 1, 3, 5]	0	"ACEBDF"	0
[0, 2, 4, 5]	3	"ACEF"	3

`$etape[1]`

Chemin améliorant : [0,1,4,5]

`$flux`=7

`$flot`=

	A	B	C	D	E	F
A	0	7	0	0	0	0
B	0	0	0	0	7	0
C	0	0	0	0	0	0
D	0	0	0	0	0	0
E	0	0	0	0	0	7
F	0	0	0	0	0	0

`$e_chem`=

[0, 1, 3, 2, 4, 5]	0	"ABDCEF"	0
[0, 1, 3, 5]	1	"ABDF"	1
[0, 1, 4, 2, 3, 5]	0	"ABECDF"	0
[0, 1, 4, 5]	0	"ABEF"	0
[0, 2, 3, 1, 4, 5]	0	"ACDBEF"	0
[0, 2, 3, 5]	4	"ACDF"	4
[0, 2, 4, 1, 3, 5]	3	"ACEBDF"	3
[0, 2, 4, 5]	1	"ACEF"	1

`$etape[2]`

Chemin améliorant : [0,2,3,5]

`$flux`=11

`$flot`=

	A	B	C	D	E	F
A	0	7	4	0	0	0
B	0	0	0	0	7	0
C	0	0	0	4	0	0
D	0	0	0	0	0	4
E	0	0	0	0	0	7
F	0	0	0	0	0	0

`$e_chem`=

[0, 1, 3, 2, 4, 5]	1	"ABDCEF"	1
[0, 1, 3, 5]	0	"ABDF"	0
[0, 1, 4, 2, 3, 5]	0	"ABECDF"	0
[0, 1, 4, 5]	0	"ABEF"	0
[0, 2, 3, 1, 4, 5]	0	"ACDBEF"	0
[0, 2, 3, 5]	0	"ACDF"	0
[0, 2, 4, 1, 3, 5]	0	"ACEBDF"	0
[0, 2, 4, 5]	2	"ACEF"	2

`$etape[3]`

Chemin améliorant : [0,2,4,5]

`$flux`=13

`$flot`=

	A	B	C	D	E	F
A	0	7	2	0	0	0
B	0	0	0	0	7	0
C	0	0	0	6	0	0
D	0	0	0	0	0	4
E	0	0	0	0	0	9
F	0	0	0	0	0	0

`$e_chem`=

[0, 1, 3, 2, 4, 5]	0	"ABDCEF"	0
[0, 1, 3, 5]	0	"ABDF"	0
[0, 1, 4, 2, 3, 5]	0	"ABECDF"	0
[0, 1, 4, 5]	0	"ABEF"	0
[0, 2, 3, 1, 4, 5]	0	"ACDBEF"	0
[0, 2, 3, 5]	0	"ACDF"	0
[0, 2, 4, 1, 3, 5]	0	"ACEBDF"	0
[0, 2, 4, 5]	0	"ACEF"	0