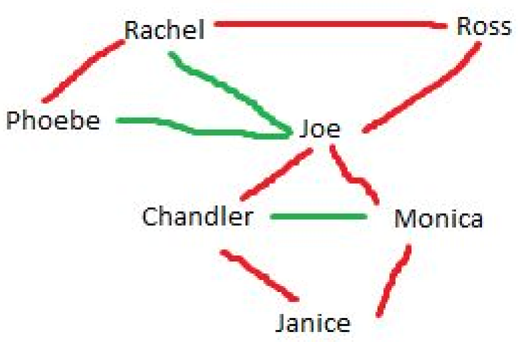
2) Covoiturage compliqué

Voici un graphe représentant le problème :



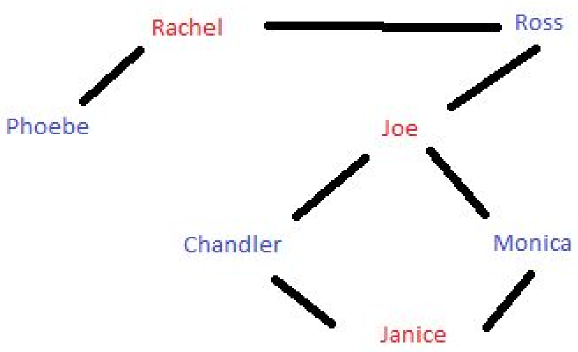
Les arcs en rouge déterminent de mauvaises relations tandis que les arcs en vert traduisent de bonnes relations.

Compte tenu du problème, il semble intéressant de commencer par remplir les 2 voitures avec deux personnes s'entendant bien.

Chandler et Monica ne s'entendant bien qu'entre eux, on les met d'office dans la même voiture (voiture n°1). Janice et Joe s'entendant mal avec eux, mais n'ayant aucun différend entre eux on les met dans l'autre voiture (voiture n°2). Joe s'entendant mal avec Ross, ils ne doivent pas partager la même voiture, Ross va donc dans la voiture 1. On peut donc mettre Rachel dans la voiture 2, lui faisant éviter Ross et rejoindre Joe. Ne reste que Phoebe, qui doit éviter Rachel; elle va donc dans la voiture 1.

Pour résumer : Voiture n°1 : Chandler, Monica, Phoebe et Ross
Voiture n°2 : Janice, Joe et Rachel.

Le graphe Biparti suivant permet de synthétiser la situation :



3) Bipartite Graphs

Algo Bipartition (Données : (*E,Delta), x∈E; Résultat : v_sommets_X[], un tableau représentant les valeurs des sommets qui prend pour valeurs les “couleurs” de la bipartition)

```
X = E ; Y=Symétrique(E);
v_sommets_X[]; v_sommets_Y = [0,0,...,0];

Tant que l’ième sommet de X n’a pas d’arc sortant de ce sommet :
        Faire : i+=1;
v_sommets_X[i];

Pour chaque x sommet de X :
        Pour chaque y ∈ Delta(x) :
                Si ( v_sommets_X(x)==1 et v_sommets_X(y) !=1)
                        Alors v_sommets_X(y)=2;
                Et Si  ( v_sommets_X(x)==2 et v_sommets_X(y) !=2)
                        Alors  v_sommets_X(y)=1;

Tant que l’ième sommet de Y n’a pas d’arc sortant de ce sommet :
        Faire : i+=1;
v_sommets_Y[i];

Pour chaque x sommet de Y :
        Pour chaque y ∈ Delta(x) :
                Si ( v_sommets_Y(x)==1 et v_sommets_Y(y) !=1)
                        Alors v_sommets_Y(y)=2;
                Et Si  ( v_sommets_Y(x)==2 et v_sommets_Y(y) !=2)
                        Alors  v_sommets_Y(y)=1;

Pour i allant de 0 au nombre de sommets de X:
        Si (v_sommets_X[i] == 0):
                Alors (v_sommets_X[i] = v_sommets_Y[i]);

Afficher v_sommets_X;
Si un 0 est présent, le Graphe n’est pas Biparti
Sinon, il l’est.
```

Complexité :

Soit n le nombre de sommets et m le nombre d'arcs.
L'algorithme de Symétrie a une complexité en m.
On initialise deux tableaux de taille n avec des 0 => +2n de complexité.
Les boucles pour initialiser un élément des tableaux à 1 ont elle aussi une complexité en n => +2n à la complexité.
Les boucles effectuant la bipartition sur le graphe et son symétrique ont une complexité en 2m => +4m à la complexité
Enfin le test final pour vérifier la Bipartition a une complexité en n => +n de complexité.

Au total, on a donc une complexité en 5n+5m => Complexité linéaire.