Routage de véhicules (VRP/VRPTW).

Données.

- un dépot D,
- un ensemble de commandes $C = \{c_1, \ldots, c_n\}$,
- une flotte de véhicules identiques (au dépot).

Contraintes.

- capacité des véhicules,
- plages horaires pour la livraison.

Objectif : délivrer toutes les commandes en optimisant divers paramètres .

Commandes.

Chaque commande est définie par

- un *client*,
- un *poids* (voir capacité du véhicule),
- la durée (sur place) de livraison,
- une plage horaire dans laquelle s'effectue la livraison.

Carte, clients.

Carte : indique les positions des lieux géographiques et les liaisons routières entre eux.

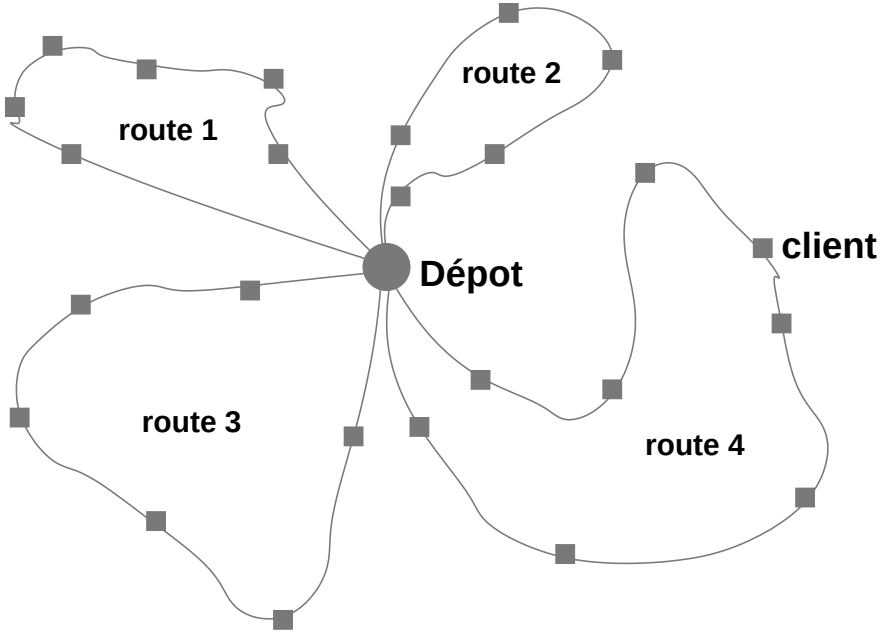
Clients : définis par leurs positions géographiques.

Solution du problème.

Solution : ensemble de tournées ou *routes*.

Tournée : suite de commandes.

- horaires de passages,
- temps de trajets,
- temps d'attente.



Solution du problème.

Tournée valide :

- respecte la capacité du véhicule,
- les horaires de livraisons sont dans les plages horaires.

Objectifs à optimiser.

Objectif minimal : satisfaire tous les clients.

On cherchera à optimiser les paramètres suivants par ordre de priorité :

- minimiser le nombre de véhicules utilisés,
- minimiser la durée totale d'utilisation des véhicules (cumul des durées des tournées),
- minimiser les distances parcourues par les véhicules.

Le projet.

Développement en JAVA.
Lecture des données dans des fichiers texte.
Le projet est composé de quatre parties (par ordre chronologique) :

- **structures de données : à rendre au plus tard le 23 février 2007** (5 points dans la note finale) ; représentation des données du problème, des solutions, d'objets temporairement utilisés pendant la recherche,
- **méthode glouton :** règles de choix (bon sens, intuition),
- **approche par recherche locale :** voir transparents suivants,
- **interface graphique :** représentation des solutions, de leurs caractéristiques.

Le projet.

Vous serez évalués sur

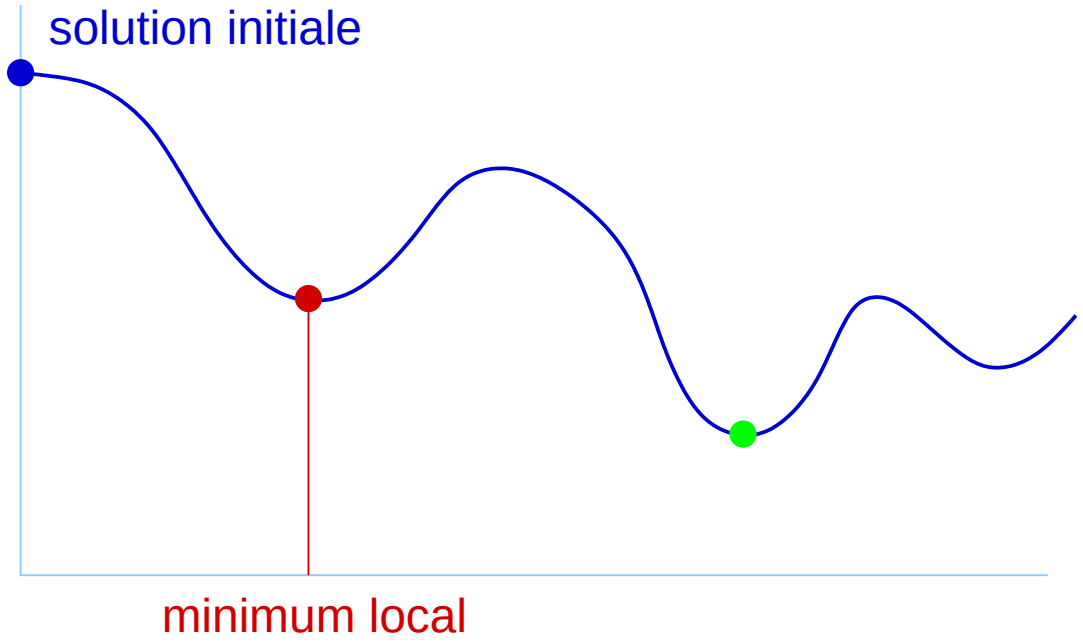
- l’originalité et l’efficacité des méthodes (recherche de stratégies, mise au point, tests expérimentaux),
- l’organisation du projet, la rigueur du développement, la clarté du code.

Recherche locale.

Problèmes d'optimisation *difficiles* : une méthode *complète* serait trop couteuse en temps.

Principe : explorer l'espace de recherche à partir d'une solution initiale par des modifications *locales*.

Problème du minimum local :



Recherche locale.

Stratégies :

intensification : amélioration de la solution courante

diversification : exploration d'autres régions de l'espace de recherche.

Choix de la modification

- la meilleure,
- la première améliorante,
- au hasard.

dans un ensemble de modifications dépendant de la stratégie.

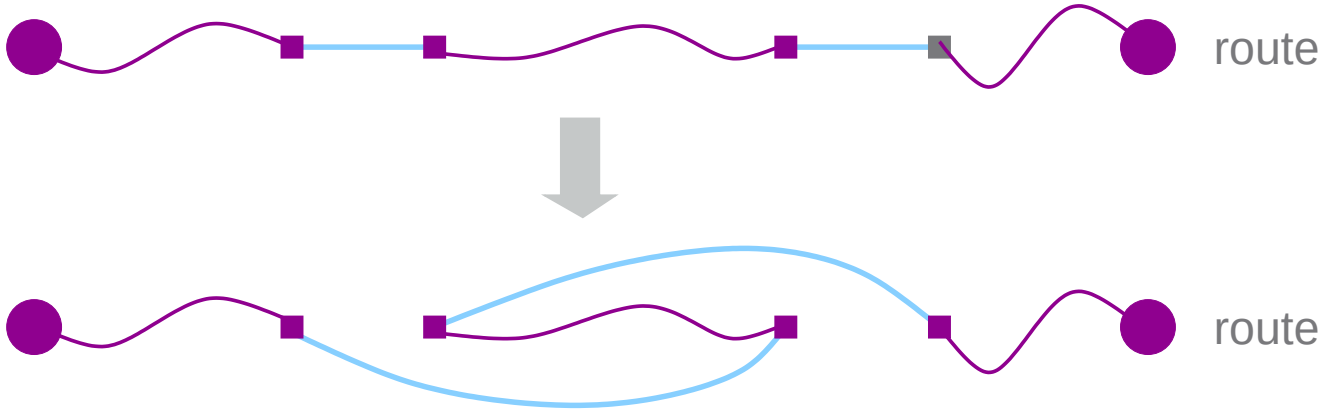
Recherche locale.

Remarque. Dans ce type de problème on peut accepter des solutions qui violent certaintes contraintes, les fenêtres de temps par exemple.

Dans ce cas le passage en dehors de la plage horaire apporte une pénalité.

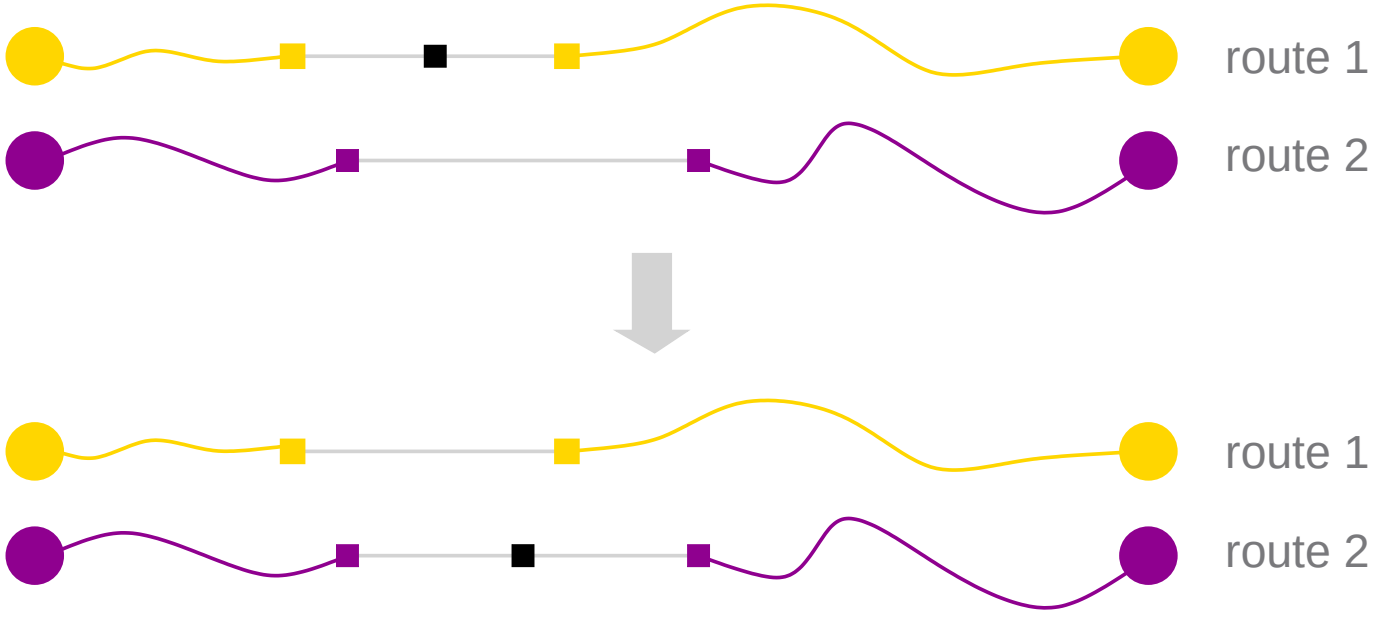
Modifications élémentaires : suggestions.

réarrangement :



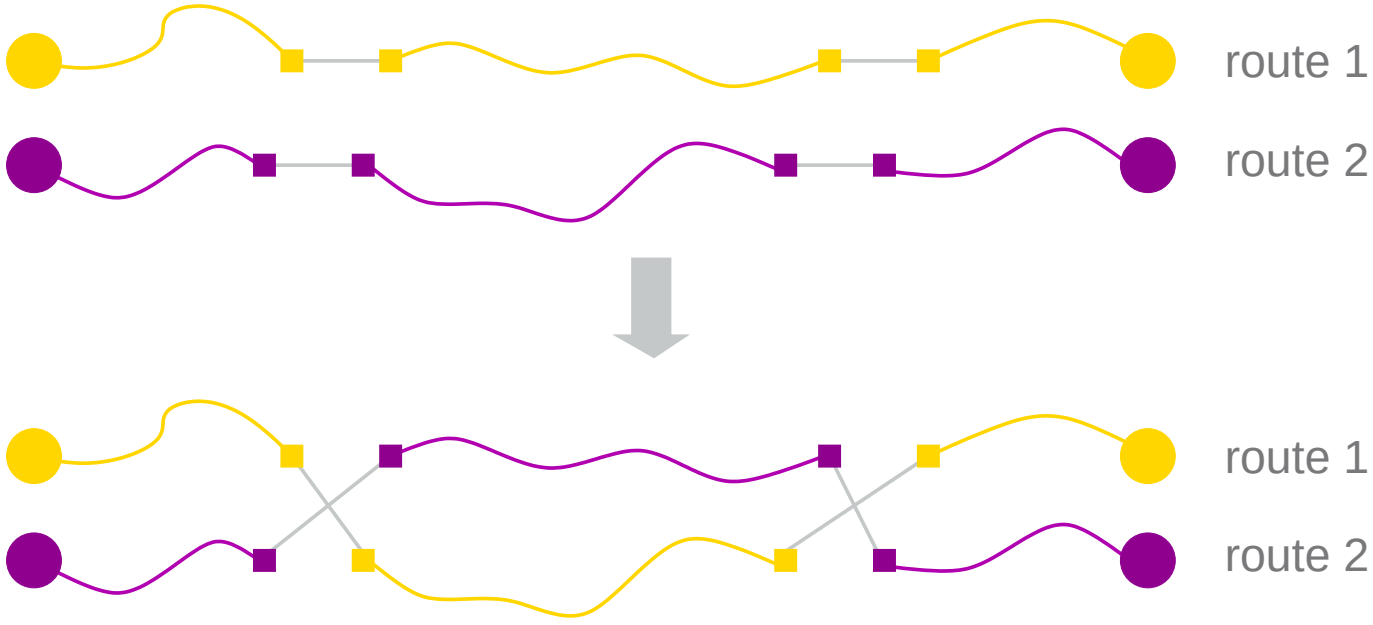
Modifications élémentaires : suggestions.

éjection :



Modifications élémentaires : suggestions.

échange :



Syntaxe des fichiers de données.

Vous suivrez la syntaxe des problèmes de Solomon.

```
<nom du probleme>
<nombre de vehicules> <capacite des vehicules>
<num client> <abcisse> <ordonnee> <poids> <debut plage> <fi
  ...
<num client> <abcisse> <ordonnee> <poids> <debut plage> <fi
```

Les distances représentent implicitement des durées.

Vous trouverez les instances de Solomon sur le web (25/50/100 clients).

Modalités pratiques.

Trois créneaux par semaines :

- le jeudi matin de 8h00 à 12h00
- le vendredi de 14h00 à 16h00 (je serai présent à ce créneau)

Structures de données : avant le 23 février 2007

Projet final : avant le 13 avril 2007