# Analyse asymptotique théorique

## Glouton

L’algorithme glouton se compose des étapes suivantes, l’une à la suite de l’autre :

* Vérification que la taille de l’exemplaire est supérieure à 2 :
* Initialisation de la sonde temporelle :
* Initialisation de variables, notamment, de la ville courante :
* Sélection de la ville la plus proches : Pour chaque ville non visitée, on calcul la distance avec la ville courante () et si la distance est plus petite que la plus petite distance déterminée pour la ville courante jusqu’à maintenant on assigne la valeur de la distance () :
* Lorsque la ville la plus proche est déterminée, on l’ajoute au chemin et on ajouter la distance à la distance totale :
* Mettre fin à la mesure de temps de la sonde temporelle :

Au total, la complexité du temps de calcul globale est (règle du max).

## Programmation dynamique

L’algorithme glouton se compose des étapes suivantes, l’une à la suite de l’autre :

* Vérification que la taille de l’exemplaire est supérieure à 2 :
* Initialisation de la sonde temporelle :
* Initialisation de variables:
* Calcul de la distance entre la ville initiale et toutes les autres villes pour remplir la table passant par les ensembles vides {} :
* Remplissage de toutes les tables :
  + Il y a tables (nombre de lignes pour une grande table) :
  + Il y a sous essemble représentés dans le tableau (nombre de colonnes pour un grand tableau) :
  + L’effort pour remplir chaque case est de
* Retraçage du chemin parcouru, se faisant en parcourant les n traces indiquées dans le tableau des (distance, indice\_précédent) :
* Mettre fin à la mesure de temps de la sonde temporelle :

Au total, la complexité du temps de calcul globale est (règle du max).

## Approximatif

L’algorithme glouton se compose des étapes suivantes, l’une à la suite de l’autre :

* Vérification que la taille de l’exemplaire est supérieure à 2 :
* Initialisation de la sonde temporelle :
* Initialisation de variables, notamment, de la ville courante :
* Utilisation de l’algorithme de PRIM (non optimisé) pour concevoir un arbre minimum sous-tendant à partir d’un graphe complètement connecté : 🡺
  + Tant qu’il reste des villes non assignées dans l’exemplaire (donc )
    - Parcourir les m villes déjà parcourues ()
      * Calculer la distance entre les m villes parcourues est les p villes non parcourues
      * Connecter le couple ayant la plus petite distance et transférer cette ville de l’ensemble des villes non parcourues à l’ensemble des villes parcourues
* Parcours préfix de l’arbre minimal sous-tendant : En pire cas
* Retrait des doublures dans le parcours préfix (Il faut donc repasser au travers des points du parcours obtenus en pire cas)
* Mettre fin à la mesure de temps de la sonde temporelle :

Au total, la complexité du temps de calcul globale est (règle du max).