Université Sorbonne Paris Nord

PROJET STRUCTURE DE DONNÉES

SERIK MOHAMED

ANNÉE UNIVERSITAIRE 2020/2021

Fonctionnement algorithme

Préparation: le programme va charger les villes avec la population correspondante, les « ranger » dans des groupes différenst et enregistrer des connexions (liens) entre chaque ville dans un tas binaire

Algorithme:

> Tant que toutes les villes ne sont pas dans le même groupe, et qu'il reste des connexions non examinées, on continue :

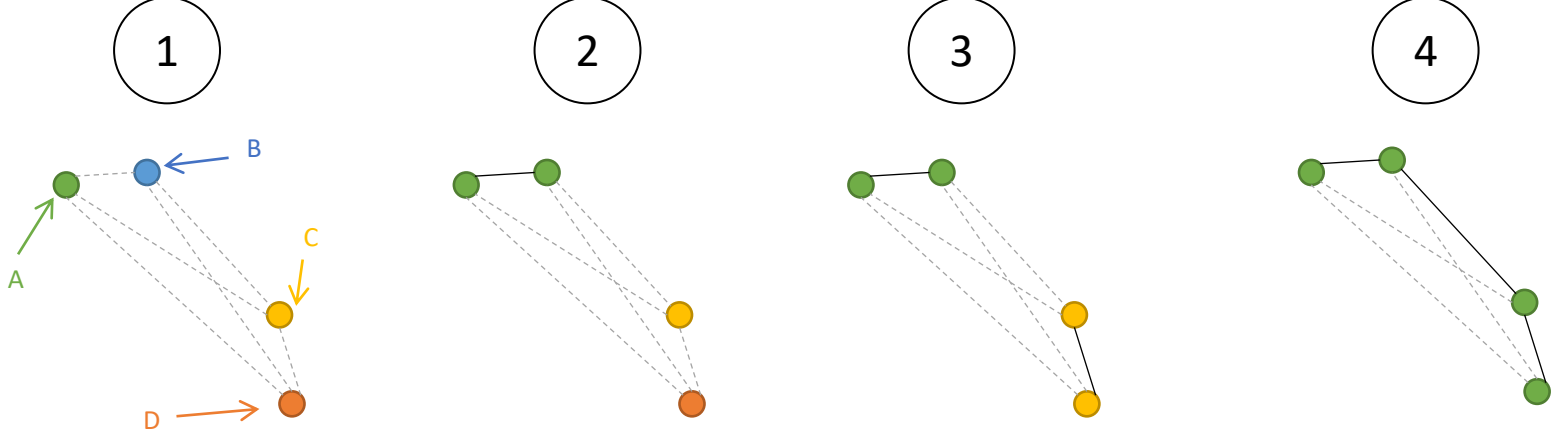
- On prend la connexion avec la distance la plus faible
- Si elle relie deux villes déjà dans un même groupe, la connexion est supprimée et on continue
- Sinon, on relie les deux villes dans un même groupe et on note ce lien comme « valide ». Ce lien est conservé pour être affiché plus tard

Fin:

On affiche les liens « valide » enregistrés

Un exemple schématisé à la page suivante

Représentation graphique de l'évolution



Connexions sélectionnées pour la solution finale

12km	A <-> B

12km	A <-> B
14km	C <-> D

12km	A <-> B
14km	C <-> D
23km	B <-> C

Connexions présentes dans le tas binaire, non sélectionné pour le moment

12km	A <-> B
14km	C <-> D
23km	B <-> C
25km	A <-> C
27km	B <-> D
31km	A <-> D

14km	C <-> D
23km	B <-> C
25km	A <-> C
27km	B <-> D
31km	A <-> D

23km	B <-> C
25km	A <-> C
27km	B <-> D
31km	A <-> D

25km	A <-> C
27km	B <-> D
31km	A <-> D

Légende graphique

Instance **pipe_t** reliant deux **city_t**. Validé et sélectionné pour la solution finale

Instance **pipe_t** reliant deux **city_t**. Existant mais non sélectionné

Structure city_t. Les différences de couleurs représentes l'appartenance à différent groupe

Légende tas binaire

12km	A <-> B

Instance **bin_node_t** contenant un **pipe_t .** Ce **pipe_t** est Validé et sélectionné pour la solution finale

12km	A <-> B

Instance **bin_node_t** contenant un **pipe_t .** Ce **pipe_t** est non sélectionné

12km	A <-> B

Distance, aussi utilisé comme poids dans le tas binaire

Villes (**city_t**) ciblées par le lien (**pipe_t**)

Structures principales

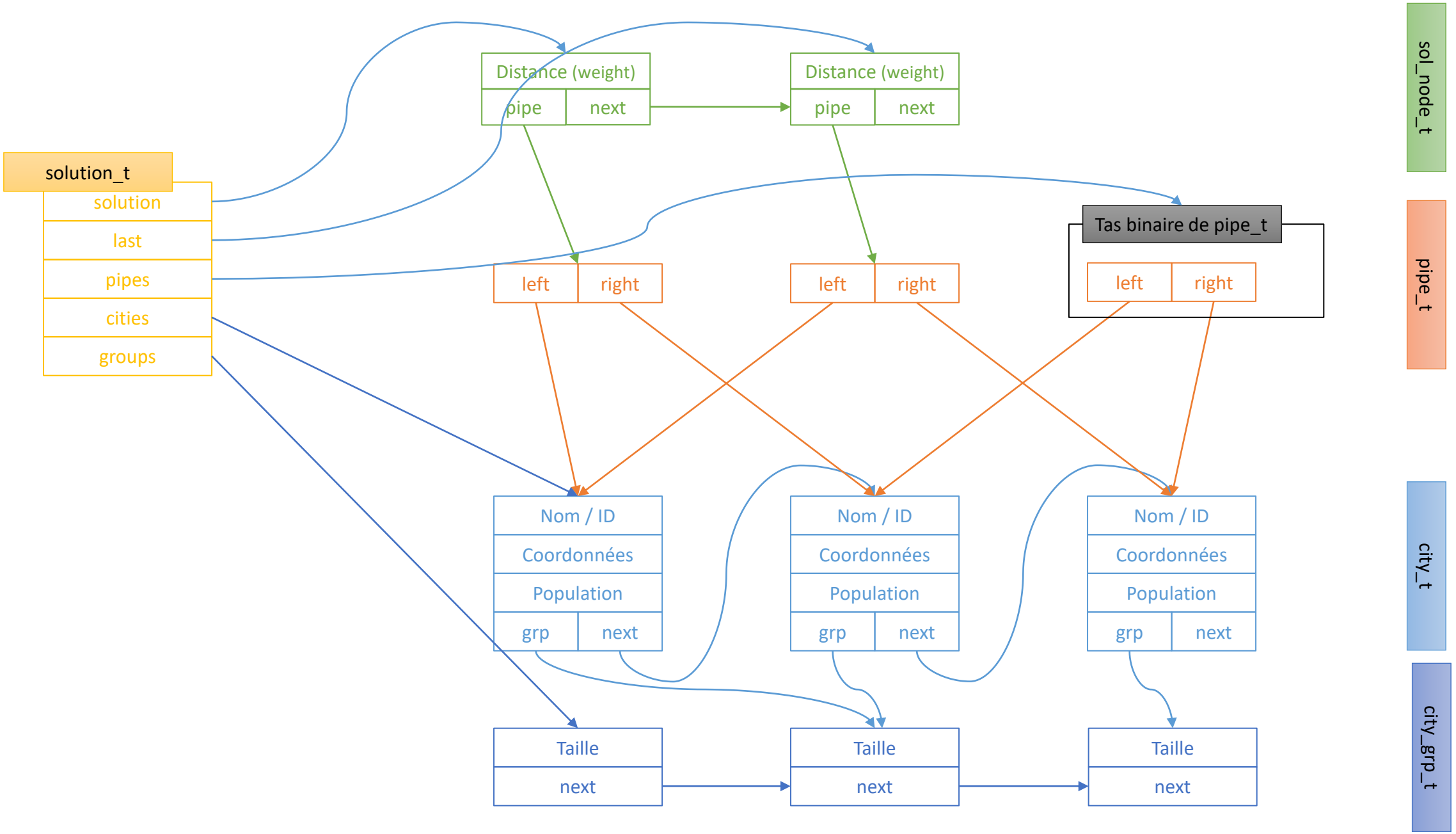
Les villes sont des **city_t**, contenant un nom, un ID, des coordonnées satellites, une taille de population, un group et un pointeur vers la ville suivante. Le pointeur suivant sert juste d'outil pour traité les villes comme une liste chaînée.

Les groupes sont des **city_grp_t**, ne contenant qu'une taille et un pointeur vers le groupe suivant. Ces structures ne servent que d'outils pour grouper les villes. Comme pour les villes, le pointeur suivant n'est la que pour gérer les groupes comme une liste chaînée.

Les connexions sont des **pipe_t**, ne contenant que deux pointers vers deux villes. Ils représentent les liens possibles entre deux villes.

La structure **solution_t**, est simplement là pour contenir et transporter les informations utiles à l'algorithme. Cette structure comprend, la liste chaînée de ville (*cities*), la liste chaînée de groupe (*groups*), le nombre de lien manquant avant d'avoir relié toute les villes (*coutdown*), un tas binaire contenant toutes les connexions (*pipes*), une liste chaînée de connexions (*solution* & *last*). La liste chaînée de connexion est gérer par une structure intermédiaire (**sol_node_t**) pour garder l'information sur la distance des solutions validées.

Un exemple schématisé à la page suivante


solution_t
solution
last
pipes
cities
groups
Distance (weight)
pipe
next
Distance (weight)
pipe
next
left
right
left
right
Tas binaire de pipe_t
left
right
Nom / ID
Coordonnées
Population
grp
next
Nom / ID
Coordonnées
Population
grp
next
Nom / ID
Coordonnées
Population
grp
next
Taille
next
Taille
next
Taille
next
sol_node_t
pipe_t
city_t
city_grp_t