Recherche d'itinéraire

Par **RATIARIVELO Nekena Rayane**

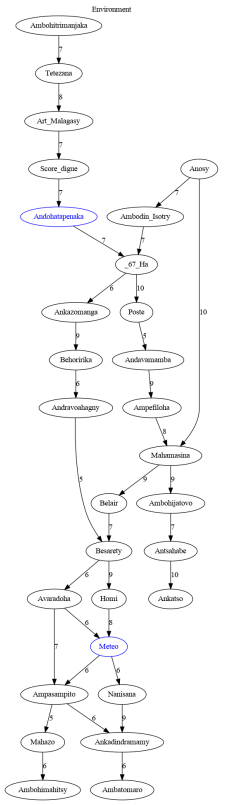
Contexte du problème

Le problème consiste à prendre des décisions concernant les étapes spécifiques de l'itinéraire et à choisir le bus optimal pour chaque étape, en se basant sur les données du trafic.

Prenons l'exemple d'un trajet de l'arrêt **Andohatapenaka** à l'arrêt **Meteo**. Nous avons établi une durée maximale de trajet de `120 minutes`, avec la possibilité de changer de bus `au maximum 3 fois`.

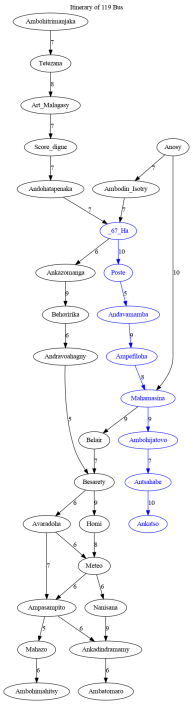
Route

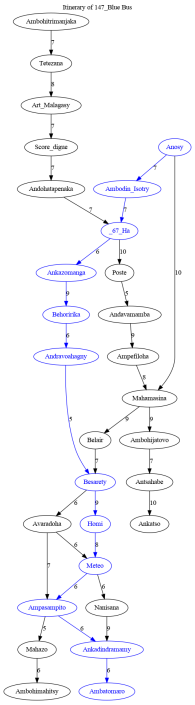
Une liaison entre deux arrêts constitue une route, et la durée qu'un bus passe sur une route est préétablie. De surcroît, une route peut se trouver dans un état `en bon état`, `en mauvais état`, ou `en construction`, avec des implications potentielles sur la durée du trajet.

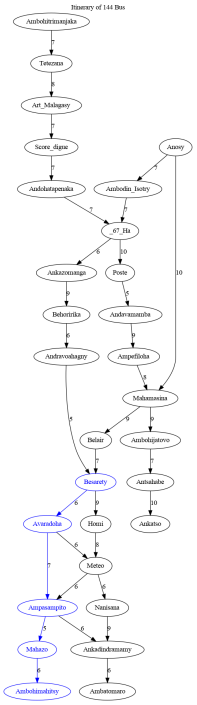


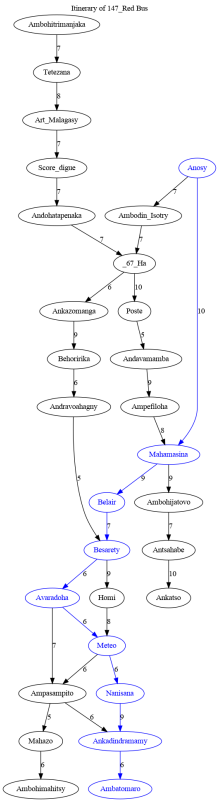
Bus

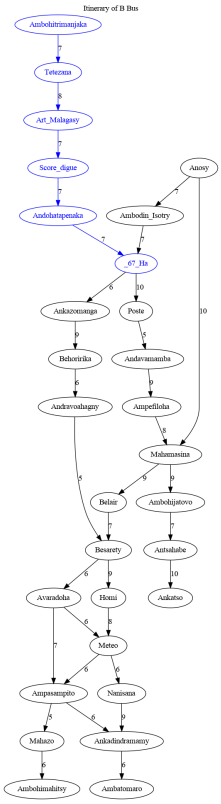
Chaque bus suit un itinéraire spécifique à Madagascar, où un itinéraire est défini comme une séquence d'arrêts de bus. Pour ce problème, considérons les 5 bus : `119`, `144`, `147 Bleu`, `147 Rouge`, `B`.











Modélisation du problème

Soient
- $\text{BusStops} = \{bs_i; i \in \{1, \dots, n\}\}$ l'ensemble des arrêts,
- $\text{Buses} = \{b_i; i \in \{1, \dots, m\}\}$ l'ensemble des bus,
- $\text{Edge} = \{(bs_i, bs_j); bs_i, bs_j \in \text{BusStops}\}$ l'ensemble des routes,
- $\text{Node} = \{(bs, b); bs \in \text{BusStops}, b \in \text{Buses}\}$ l'ensemble des couples arrêt bus et bus.

Un bus a son propre trajet. Notons $\text{G}_i = (\text{Node}_{b_i}, \text{Edge}_{b_i})$ le trajet du bus b_i avec :

- $\text{BusStops}_{b_i} = \{bs_k; bs_k \in \text{BusStops}\}_{k \in N_i} \subset \text{BusStops}$
- $\text{Node}_{b_i} = \{(bs_k, b_i); bs_k \in \text{BusStops}\}_{k \in N_i} \subset \text{Node}$
- $\text{Edge}_{b_i} = \{(bs_k, bs_{k+1}); bs_k, bs_{k+1} \in \text{BusStops}\}_{k \in N_i} \subset \text{Edge}$

Notons $bs_{start} \in \text{BusStops}$ l'arrêt de départ et $bs_{end} \in \text{BusStops}$ l'arrêt d'arrivé.

Posons change_bus la fonction qui retourne le nombre de bus, étant donné une itinéraire

$$\begin{array}{rcccc} \text{change\_bus} & : & \text{Node}^p & \to & \mathbf{N} \\ & & n = ((bs_1, b_1), \dots, (bs_p, b_p)) & \mapsto & \text{change\_bus}(n) \end{array}$$

Posons duration la fonction qui retourne la durée du trajet, étant donné une itinéraire

$$\begin{array}{rcccc} \text{duration} & : & \text{Node}^p & \to & \mathbf{N} \\ & & n = ((bs_1, b_1), \dots, (bs_p, b_p)) & \mapsto & \text{duration}(n) = d \end{array}$$

Définition du problème

Ce problème peut être représenter comme étant un problème CSP avec :

- Variables : (p étant le nombre d'arrêts pour arriver à destination)
  - $X = ((bs_1, b_1), \dots, (bs_p, b_p))$ est une **itinéraire** : la liste des arrêts et bus entre l'arrêt de départ et l'arrêt d'arrivé
- Domaines :
  - Node^p : l'ensemble des itinéraires possibles
- Contraintes : Pour $X = ((bs_1, b_1), \dots, (bs_p, b_p))$
  - contraintes de l'environnement
    - $bs_1 = bs_{start}$ et $bs_2 = bs_{end}$
    - $\forall i, bs_i \in \text{BusStops}_{b_i}$
    - $\forall i, (bs_i, bs_{i+1}) \in \text{Edge}$
    - $\forall i \neq k, bs_i \neq bs_k$
  - contraintes sur le temps
    - $\text{duration}(X) \leq 120$
  - contraintes de changement de bus
    - $\text{change\_bus}(X) \leq 3$

Résolution du problème

Comme la taille de la variable X n'est pas fixe, il serait difficile d'itérer sur tous les combinaisons possibles de solutions. Pour parcourir l'arbre des cas possibles, les algorithmes de parcours d'un graphe serait le mieux approprié. De ce fait, l'algorithme **Breadth-First Search** (BFS) sera utilisé pour ce problème.

1. Backtracking

Le parcours se fera donc tout en respectant automatiquement les contraintes de l'environnement. Dès que les contraintes de temps et de changement de bus ne sont pas respectés, on arrête de continuer sur cette voie, et on passe au cas possible suivant.

Après résolution, nous obtenons **16 itinéraires possibles**.

```
change_bus_time= 3
duration= 50min
 B (Andohatapenaka, _67_Ha)
 147_Blue (Ankazomanga, Homi)
 147_Red (Meteo, Meteo)

change_bus_time= 2
duration= 50min
 B (Andohatapenaka, _67_Ha)
 147_Blue (Ankazomanga, Meteo)

change_bus_time= 3
duration= 50min
 B (Andohatapenaka, Andohatapenaka)
 119 (_67_Ha, _67_Ha)
 147_Blue (Ankazomanga, Meteo)

change_bus_time= 3
duration= 50min
 B (Andohatapenaka, Andohatapenaka)
 147_Blue (_67_Ha, Homi)
 147_Red (Meteo, Meteo)

change_bus_time= 2
duration= 50min
 B (Andohatapenaka, Andohatapenaka)
 147_Blue (_67_Ha, Meteo)

change_bus_time= 3
duration= 50min
 119 (Andohatapenaka, Andohatapenaka)
 B (_67_Ha, _67_Ha)
 147_Blue (Ankazomanga, Meteo)
```

```
change_bus_time= 3
duration= 50min
 119 (Andohatapenaka, _67_Ha)
 147_Blue (Ankazomanga, Homi)
 147_Red (Meteo, Meteo)

change_bus_time= 2
duration= 50min
 119 (Andohatapenaka, _67_Ha)
 147_Blue (Ankazomanga, Meteo)

change_bus_time= 3
duration= 50min
 119 (Andohatapenaka, Andohatapenaka)
 147_Blue (_67_Ha, Homi)
 147_Red (Meteo, Meteo)

change_bus_time= 2
duration= 50min
 119 (Andohatapenaka, Andohatapenaka)
 147_Blue (_67_Ha, Meteo)

change_bus_time= 3
duration= 50min
 147_Blue (Andohatapenaka, Andohatapenaka)
 B (_67_Ha, _67_Ha)
 147_Blue (Ankazomanga, Meteo)

change_bus_time= 3
duration= 50min
 147_Blue (Andohatapenaka, Andohatapenaka)
 119 (_67_Ha, _67_Ha)
 147_Blue (Ankazomanga, Meteo)

change_bus_time= 3
duration= 50min
 147_Blue (Andohatapenaka, Andravoahagny)
 147_Red (Besarety, Besarety)
 147_Blue (Homi, Meteo)

change_bus_time= 2
duration= 50min
 147_Blue (Andohatapenaka, Homi)
 147_Red (Meteo, Meteo)

change_bus_time= 1
duration= 50min
 147_Blue (Andohatapenaka, Meteo)
```

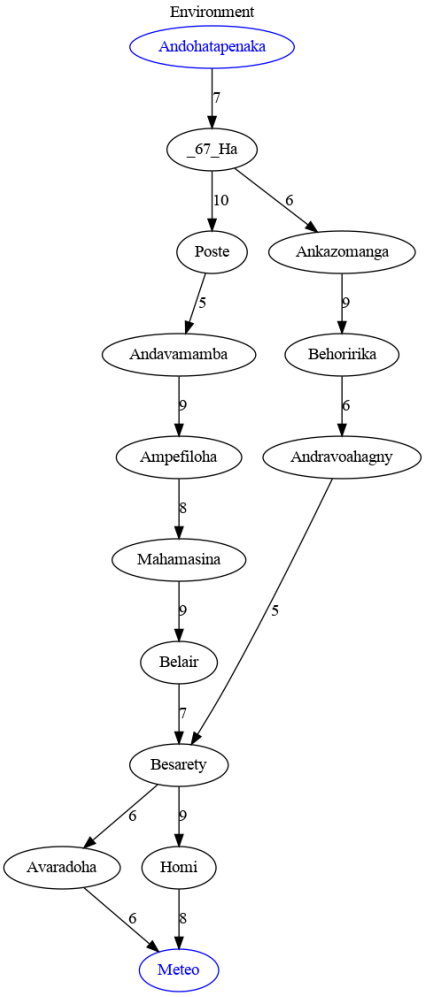
```
change_bus_time= 3
duration= 50min
 147_Blue (Andohatapenaka, Andravoahagny)
 144 (Besarety, Besarety)
 147_Blue (Homi, Meteo)
```

2. Backtracking + Arc Consistency

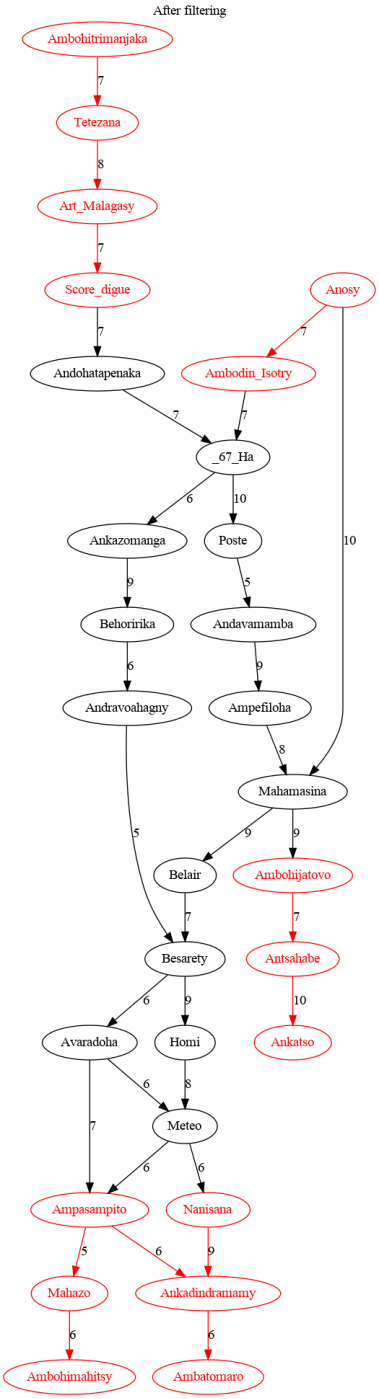
Pour optimiser la recherche, le domaine peut être filtrer en

- enlevant des nœuds qui ne sont pas dans l'itinéraire
- enlevant les graphes qui ne sont pas liés à l'arbre des solutions

Après filtrage du domaine, notre graphe devient :



Les nœuds enlevés sont coloriés en rouge :



3. Evaluation des performances des 2 résolutions

L'évaluation se fera sur un problème avec plus de solution possible. En effet, ainsi, sur un jeu de donnée de petite taille, la différence entre les 2 résolutions peut ne pas être significatif.

Après évaluation des 2 résolution, sur un problème avec **49 solutions possibles**, le filtrage du domaine rend plus rapide la recherche de solution. Sur **1000 itérations**, sur une résolution sans filtrage, on note un temps de `0.002345 seconde` alors qu'après filtrage, on obtient un temps de `0.000891 seconde`.