

# Trajet en voiture optimal en France

---

*Projet présenté par ReyVisions, MI OIVM.*



# Sommaire de la présentation

01

*Introduction du sujet*

04

*Résultats statistiques*

02

*Modélisation du problème*

05

*Difficultés et perspectives*

03

*Fonctionnalités du projet*

06

*Conclusion*

01

## *Introduction*

Léa veut voyager d'un point A à un point B.

Conditions:

- Visiter le maximum de grandes villes sur le chemin
- Être le plus rapide possible malgré la première condition



Problème de plannification d'itinéraire



Lille



Marseille

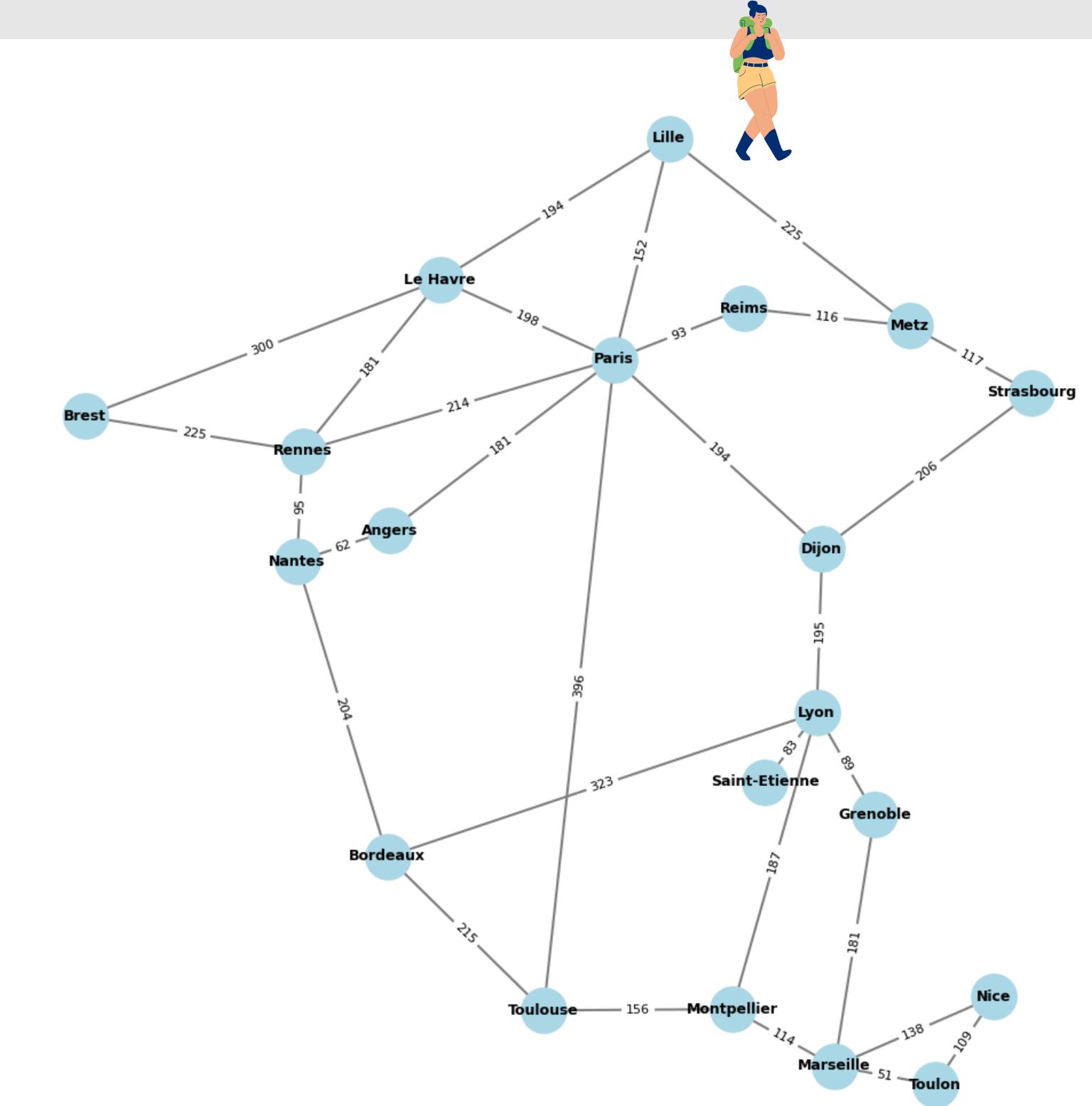
## Modélisation du problème

### Graphe

- Modélisation sous forme de graphe simple non orienté  $GI(V,E)$ .
- $V(GI) = \text{villes de France}$
- $E(GI) = \text{temps de trajet entre les deux villes}$

### Données réelles

- Données récupérées sur Google Maps
- Les positions des sommets correspondent aux positions géographiques
- Les distances sont proportionnelles à la réalité
- + Quelques hypothèses



## Les algorithmes utilisés

### Dijkstra

- Calcule le chemin de plus court
- Pas d'heuristique
- Normalement plus lent que A\*

### A\*

- Fonctionne comme Dijkstra
- Mais prend en compte l'heuristique  $f = g + h$
- Normalement plus performant que Dijkstra

### Bellman-Ford

- Fonctionne pour les graphes à poids négatif
- Déetecte les cycles négatifs
- Permet de résoudre des cas plus complexes.

### Floyd-Warshall

- Calcule tous les chemins les plus courts
- Efficace pour un graphe de taille moyen.

## Algorithme de Dijkstra

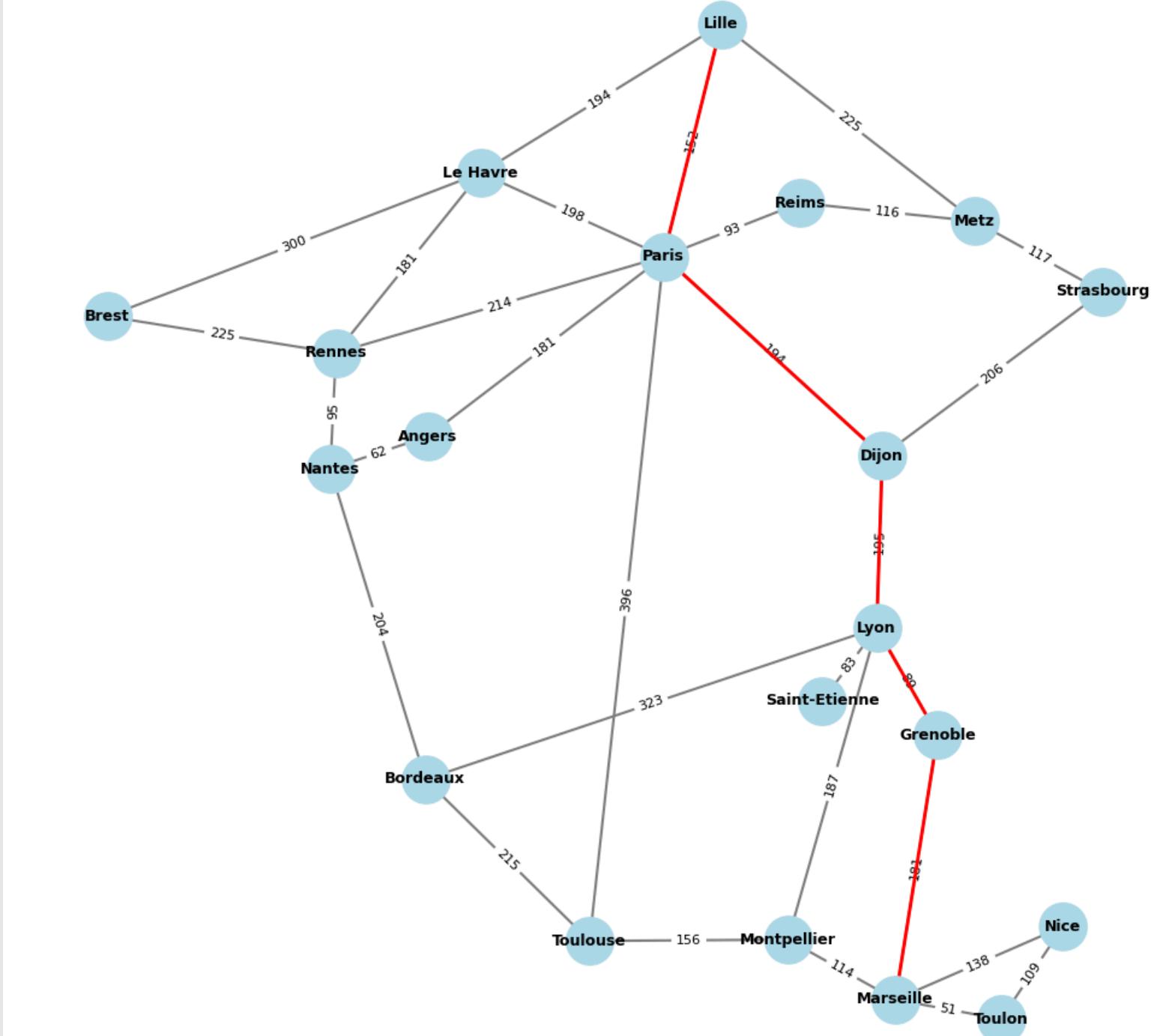
### Résultats

Résultats pour l'algorithme Dijkstra :

Chemin optimal de Lille à Marseille : ['Lille', 'Paris', 'Dijon', 'Lyon', 'Grenoble', 'Marseille']

Temps minimal : 811 minutes

Distance totale parcourue pour le plus court chemin (Dijkstra) : 1145.0 km

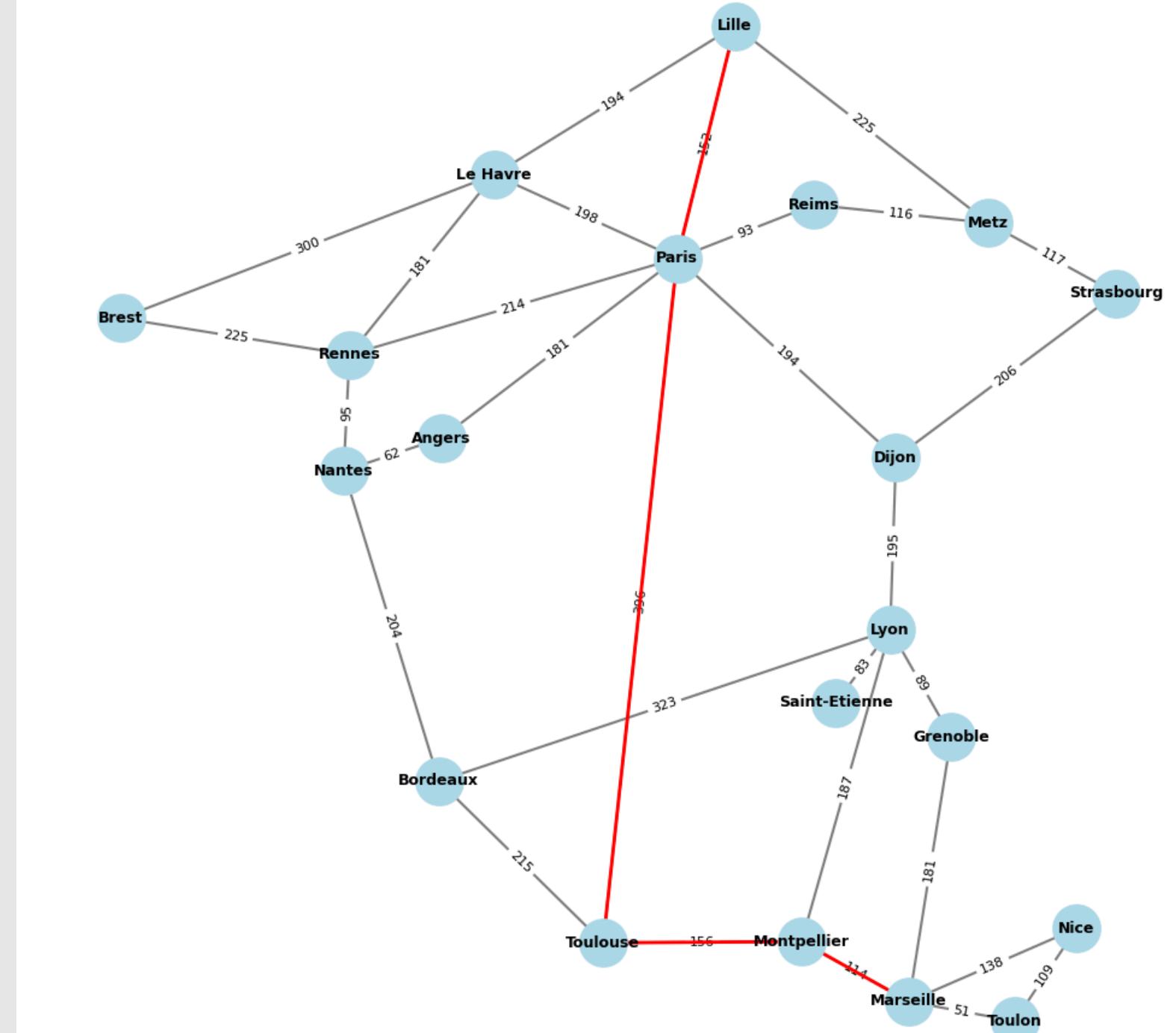


## Résultats

Resultats pour l'algorithme A\* :

Chemin optimal de Lille à Marseille : ['Lille', 'Paris', 'Toulouse', 'Montpellier', 'Marseille']  
 Cout total estimé : 818 minutes  
 Distance totale parcourue pour le plus court chemin (Dijkstra) : 1285.0 km

- Resultat moins optimal (plus long) en raison des heuristiques

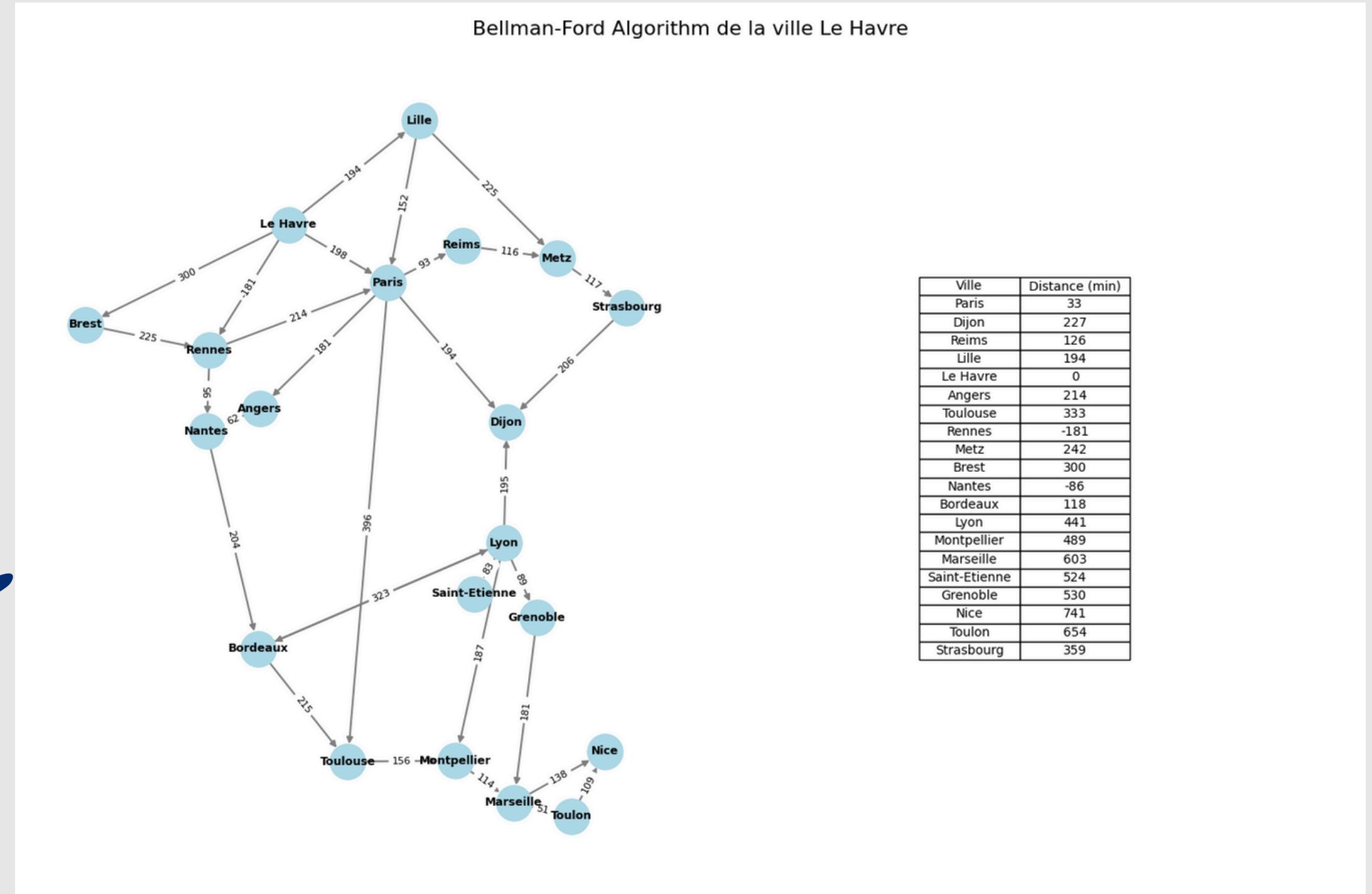


## Algorithme de Bellman-Ford

### Résultats

On change légèrement le problème de Léa :

Elle doit récupérer un ami à Rennes,  
(Le Havre, Rennes) = -181



# 03

# Algorithme de Floyd-Warshall

## Résultats

Matrice des distances (Floyd-Warshall)

	Paris	Dijon	Reims	Lille	Le Havre	Angers	Toulouse	Rennes	Metz	Brest	Nantes	Bordeaux	Lyon	Montpellier	Marseille	Saint-Etienne	Grenoble	Nice	Toulon	Strasbourg
Paris	0.0	inf	inf	152.0	33.0	inf	inf	214.0	inf	439.0	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf
Dijon	194.0	0.0	439.0	346.0	227.0	784.0	inf	408.0	323.0	633.0	722.0	518.0	195.0	inf	inf	inf	inf	inf	inf	206.0
Reims	93.0	inf	0.0	245.0	126.0	inf	inf	307.0	inf	532.0	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf
Angers	181.0	inf	inf	333.0	214.0	0.0	inf	395.0	inf	620.0	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf
Toulouse	396.0	inf	inf	548.0	333.0	481.0	0.0	514.0	inf	739.0	419.0	215.0	538.0	inf	inf	inf	inf	inf	inf	inf
Lille	inf	inf	inf	0.0	194.0	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf
Le Havre	inf	inf	inf	inf	0.0	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf
Rennes	inf	inf	inf	inf	-181.0	inf	inf	0.0	inf	225.0	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf
Metz	209.0	inf	116.0	225.0	242.0	inf	inf	423.0	0.0	648.0	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf
Brest	inf	inf	inf	inf	300.0	inf	inf	inf	0.0	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf
Nantes	243.0	inf	inf	395.0	-86.0	62.0	inf	95.0	inf	320.0	0.0	inf	inf	inf	inf	inf	inf	inf	inf	inf
Bordeaux	447.0	inf	inf	599.0	118.0	266.0	inf	299.0	inf	524.0	204.0	0.0	323.0	inf	inf	inf	inf	inf	inf	inf
Lyon	770.0	inf	inf	922.0	441.0	589.0	inf	622.0	inf	847.0	527.0	323.0	0.0	inf	inf	inf	inf	inf	inf	inf
Montpellier	552.0	inf	inf	704.0	489.0	637.0	156.0	670.0	inf	895.0	575.0	371.0	187.0	0.0	inf	inf	inf	inf	inf	inf
Marseille	666.0	inf	inf	818.0	603.0	751.0	270.0	784.0	inf	1009.0	689.0	485.0	270.0	114.0	0.0	inf	181.0	inf	inf	inf
Saint-Etienne	853.0	inf	inf	1005.0	524.0	672.0	inf	705.0	inf	930.0	610.0	406.0	83.0	inf	0.0	inf	inf	inf	inf	inf
Grenoble	859.0	inf	inf	1011.0	530.0	678.0	inf	711.0	inf	936.0	616.0	412.0	89.0	inf	inf	0.0	inf	inf	inf	inf
Nice	804.0	inf	inf	956.0	741.0	889.0	408.0	922.0	inf	1147.0	827.0	623.0	408.0	252.0	138.0	inf	319.0	0.0	109.0	inf
Toulon	717.0	inf	inf	869.0	654.0	802.0	321.0	835.0	inf	1060.0	740.0	536.0	321.0	165.0	51.0	inf	232.0	inf	0.0	inf
Strasbourg	326.0	inf	233.0	342.0	359.0	inf	inf	540.0	117.0	765.0	inf	inf	inf	inf	inf	inf	inf	inf	inf	0.0

Matrice des prédecesseurs (Floyd-Warshall)

	Paris	Reims	Lille	Le Havre	Angers	Toulouse	Rennes	Metz	Brest	Nantes	Bordeaux	Lyon	Montpellier	Marseille	Grenoble	Toulon	Strasbourg
Dijon	Paris	Strasbourg	Paris	Paris	Lyon	-	Paris	Strasbourg	Paris	Lyon	lyon	lyon	-	-	-	-	Strasbourg
Reims	Paris	-	Paris	Paris	-	-	Paris	-	Paris	-	-	-	-	-	-	-	-
Angers	Paris	-	Paris	Paris	-	-	Paris	-	Paris	-	-	-	-	-	-	-	-
Toulouse	Paris	-	Paris	Bordeaux	Bordeaux	-	Bordeaux	-	Bordeaux	Bordeaux	Bordeaux	Bordeaux	-	-	-	-	-
Metz	Reims	Reims	Lille	Reims	-	-	Reims	-	Reims	-	-	-	-	-	-	-	-
Nantes	Angers	-	Angers	Pennes	Angers	-	Pennes	-	Pennes	-	-	-	-	-	-	-	-
Montpellier	Toulouse	-	Toulouse	Toulouse	Toulouse	-	Toulouse	-	Toulouse	Toulouse	Toulouse	Lyon	-	-	-	-	-
Strasbourg	Metz	Metz	Metz	Metz	-	-	Metz	Metz	Metz	-	-	-	-	-	-	-	-
Bordeaux	Nantes	-	Nantes	Nantes	-	-	Nantes	-	Nantes	-	-	lyon	-	-	-	-	-
Saint-Etienne	lyon	Bordeaux	-	Bordeaux	Bordeaux	Bordeaux	-	Bordeaux	-	Bordeaux	Bordeaux	Bordeaux	-	-	-	-	-
Grenoble	lyon	-	lyon	lyon	lyon	-	lyon	-	lyon	lyon	lyon	lyon	-	-	-	-	-
Marseille	Montpellier	-	Montpellier	Montpellier	Montpellier	Montpellier	-	Montpellier	Montpellier	Montpellier	Montpellier	Grenoble	-	-	-	-	-
Nice	Marseille	-	Marseille	Marseille	Marseille	Marseille	-	Marseille	Marseille	Marseille	Marseille	Marseille	Marseille	Marseille	Toulon	-	-
Toulon	Marseille	-	Marseille	Marseille	Marseille	Marseille	-	Marseille	Marseille	Marseille	Marseille	Marseille	Marseille	Marseille	Marseille	-	-
Paris	-	-	Lille	Rennes	-	-	Pennes	-	Pennes	-	-	-	-	-	-	-	-
Lille	-	-	-	Le Havre	-	-	-	-	-	-	-	-	-	-	-	-	-
Brest	-	-	-	Le Havre	-	-	-	-	-	-	-	-	-	-	-	-	-
Rennes	-	-	-	Le Havre	-	-	-	-	-	-	-	Brest	-	-	-	-	-

## Résultats sur une exécution

```
Test de temps d'exécution :
```

```
Temps d'exécution de l'algorithme Dijkstra : 4.363059997558594e-05 secondes
Temps d'exécution de l'algorithme A* : 4.506111145019531e-05 secondes
Temps d'exécution de l'algorithme Bellman-Ford : 0.0001537799835205078 secondes
```

## Résultats moyen sur toutes les arêtes

```
Évaluation Dijkstra:
```

```
Temps moyen d'exécution de Dijkstra : 0.000036 secondes
```

```
Évaluation A*:
```

```
Temps moyen d'execution d'A* : 0.000018 secondes
```

```
Évaluation Bellman-Ford pour le graphe non orienté sans aretes negatives:
```

```
Temps moyen d'exécution de Bellman-Ford : 0.000164 secondes
```

```
Évaluation Bellman-Ford pour le graphe oreinté avec aretes negatives:
```

```
Temps moyen d'exécution de Bellman-Ford : 0.000120 secondes
```

```
Temps d'exécution de l'algorithme Floyd Marshall : 0.0008649826049804688 secondes
```

Algorithme	Complexité Temporelle	Complexité Spatiale
A*	$O(E + V \log V)$	$O(V)$
Dijkstra	$O(E + V \log V)$	$O(V)$
Bellman-Ford	$O(VE)$	$O(V)$
Floyd-Warshall	$O(V^3)$	$O(V)$

## Difficultés et perspectives

### Difficultés

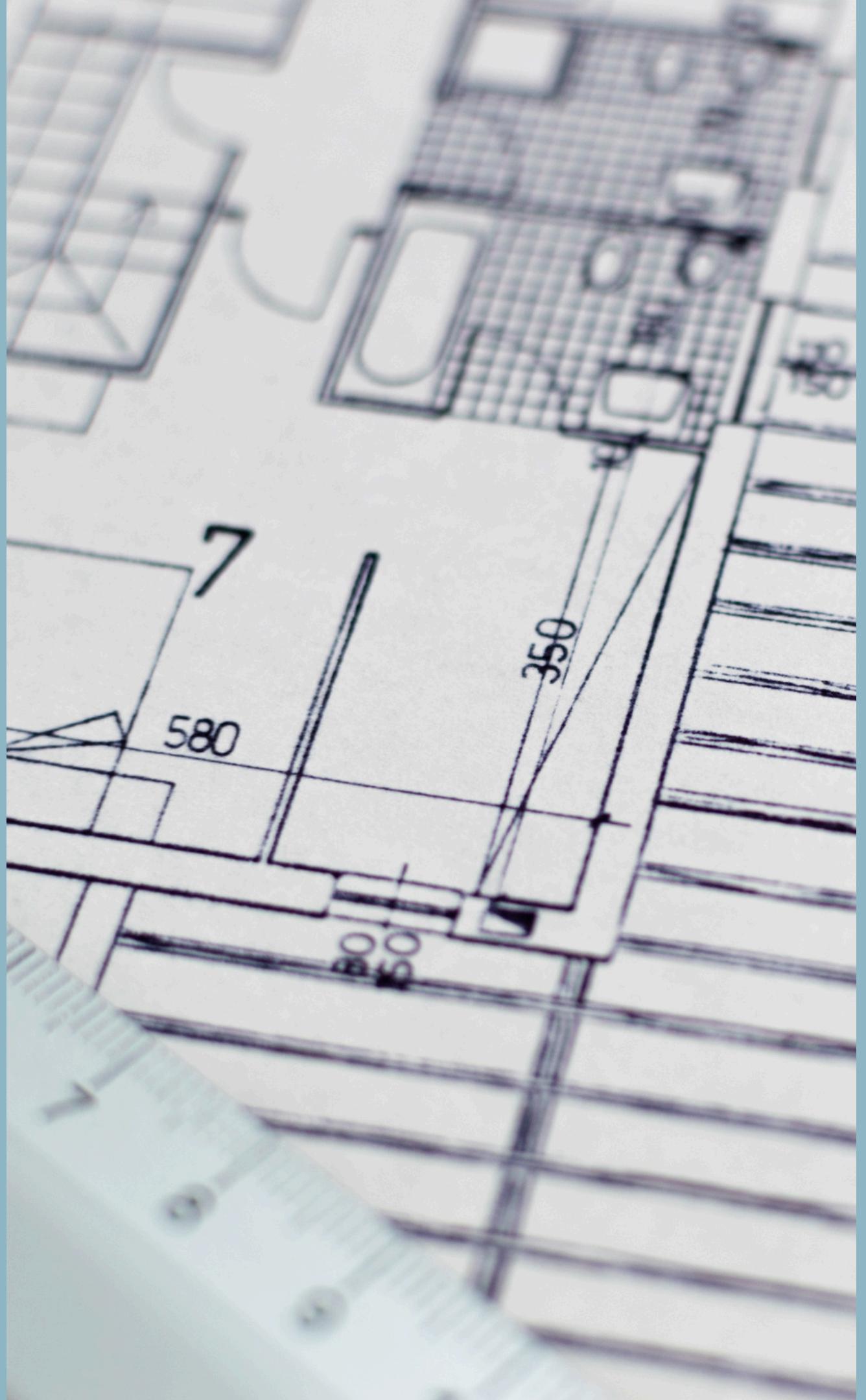
Valeurs d'heuristique difficile à trouver, et difficile de trouver une meilleure heuristique.

Temps d'execution peut augmenter fortement avec un graphe bien plus grand

### Perspectives

Meilleure heuristique : diviser la distance par la vitesse moyenne

Integrer une API de google :  
Pour l'aspect financie



## *Conclusion & remerciements*

### *Bilan final*

Le plus performant : A\*

Le plus complexe: Floyd Warshall

Les approximations peuvent changer l'ordre d'efficacité des algorithmes.

### *Réponse à notre problématique*

Grâce à la théorie des graphes, Léa va pouvoir optimiser son voyage en france!

Entreprise Concordia

# Merci pour votre attention !

*Avez-vous des questions ?*