



Université Sultan Moulay Slimane Faculté Polydisciplinaire
Département de Mathématique et Informatique
- Béni Mellal

Licence Fondamentale : Sciences Mathématiques et Informatiques



Soutenance du projet de fin d'étude sous le thème :

Étude et implémentation de l'algorithme du plus court chemin A* en Python.

Présenté par :

Abdellah Karani

Encadré par:

M. Hicham MOUNCIF

Soutenu Le 20/06/2025 devant le jury:

- Pr. Hicham Mouncif: Encadrant
- Pr. Youness. Madani: Rapporteur
- Pr. Mohamed Zouina: Rapporteur
- Pr. Hicham Ouchitachene: Rapporteur

Plan:

01 Context du projet

02 Introduction a l'algorithme A*

03 Commet Algorithme A* Fonctionne ?

04 Analyse des besoins et conception de l'application

05 Résultats et démonstration de l'application

Plan:

01 Context du projet

02 Introduction a l'algorithme A*

03 Commet Algorithme A* Fonctionne ?

04 Analyse des besoins et conception de l'application

05 Résultats et démonstration de l'application

Plan:

01 Context du projet

02 Introduction a l'algorithme A*

03 Commet Algorithme A* Fonctionne ?

04 Analyse des besoins et conception de l'application

05 Résultats et démonstration de l'application

Plan:

01 Context du projet

02 Introduction a l'algorithme A*

03 Commet Algorithme A* Fonctionne ?

04 Analyse des besoins et conception de l'application

05 Résultats et démonstration de l'application

Plan:

01 Context du projet

02 Introduction a l'algorithme A*

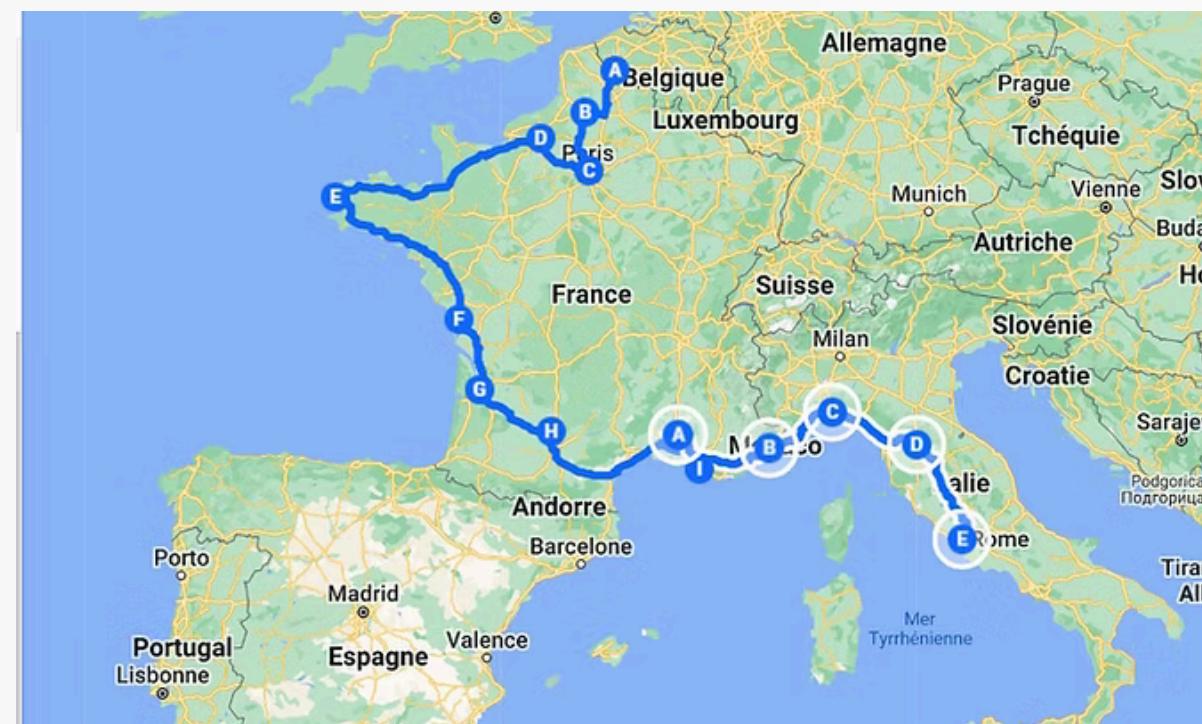
03 Commet Algorithme A* Fonctionne ?

04 Analyse des besoins et conception de l'application

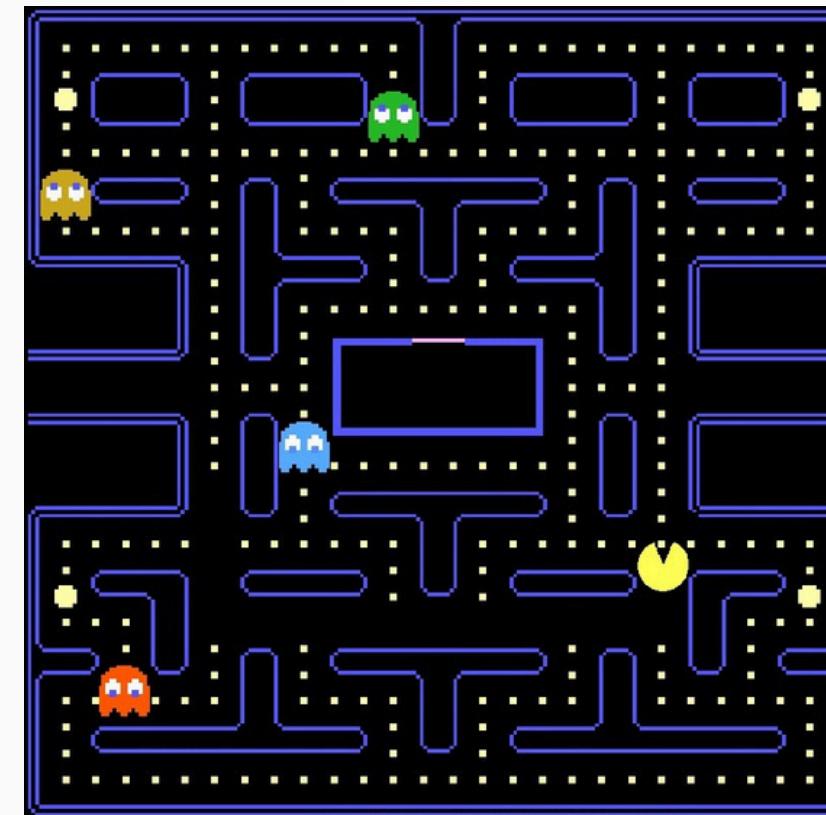
05 Résultats et démonstration de l'application

01 Context du projet

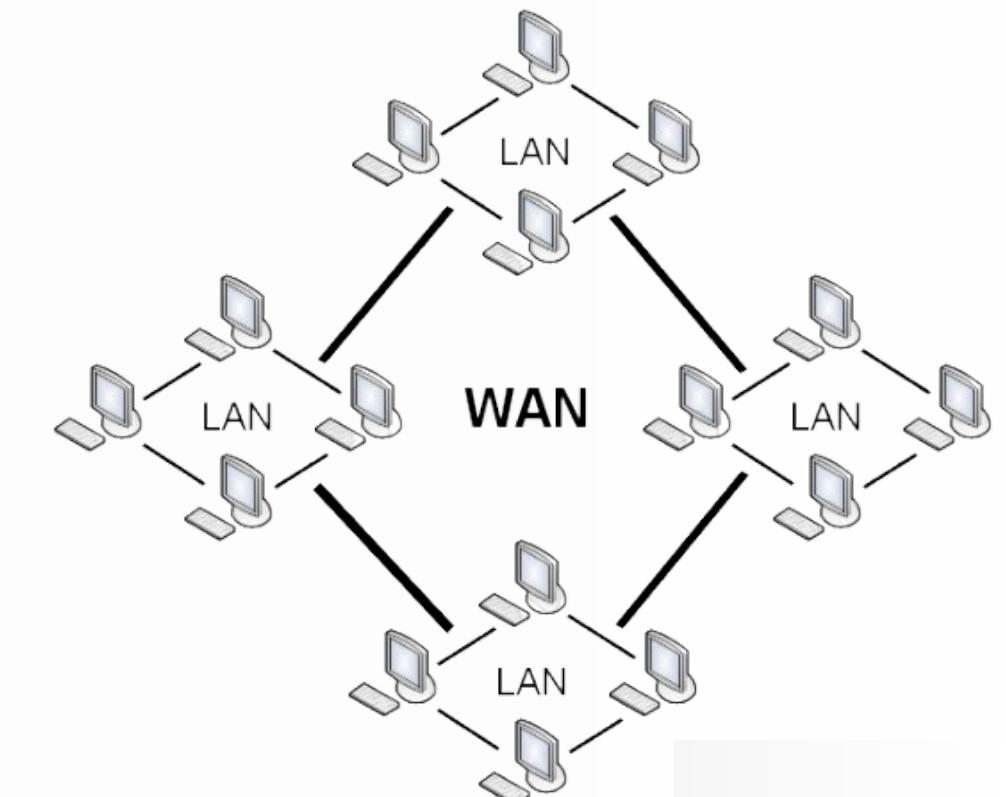
- Dans plusieurs domaines comme les transports, les réseaux ou les GPS, on utilise des systèmes faits de points reliés entre eux. Cela crée le besoin de trouver le chemin le plus court entre ces points.



Google Maps



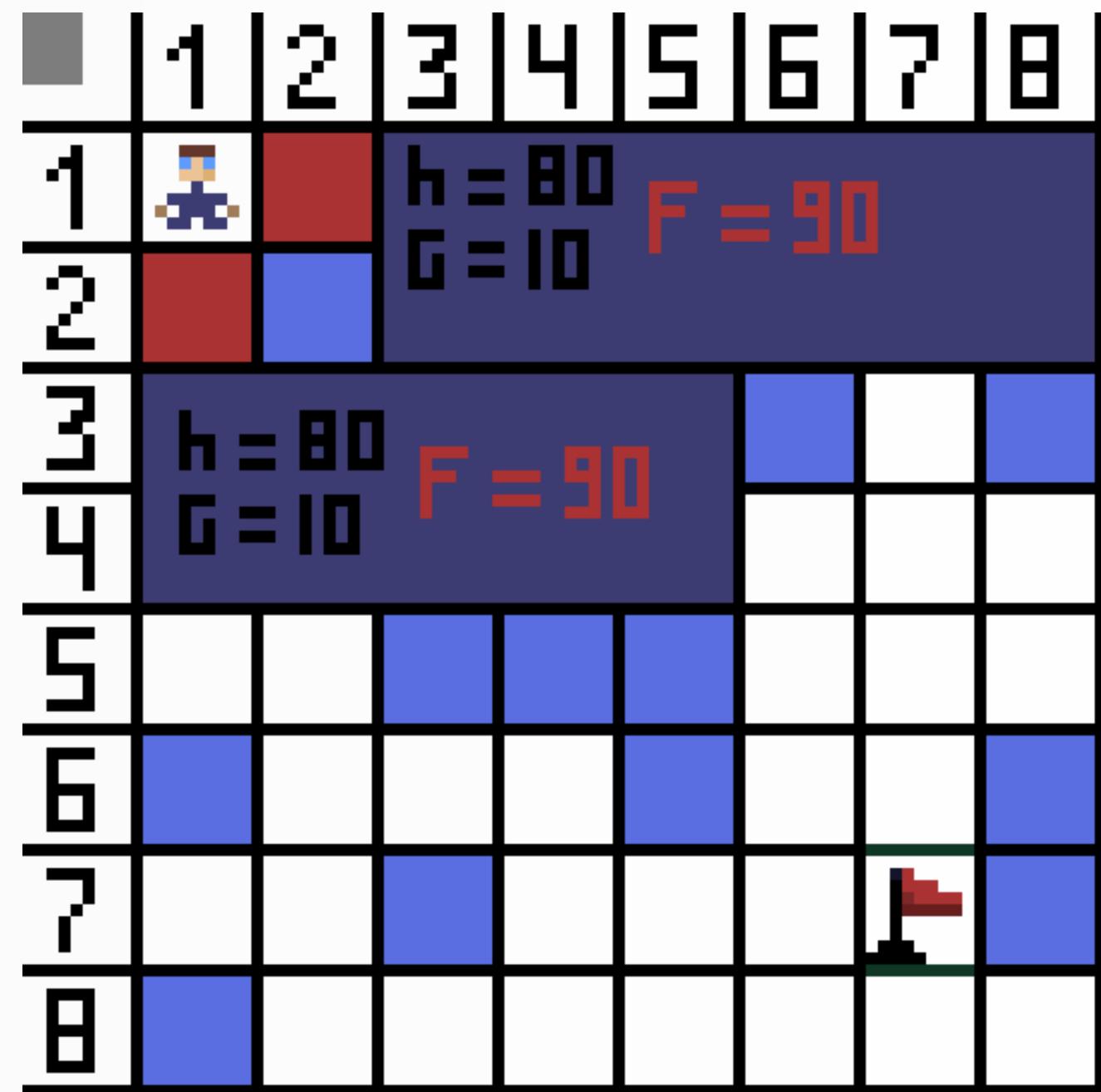
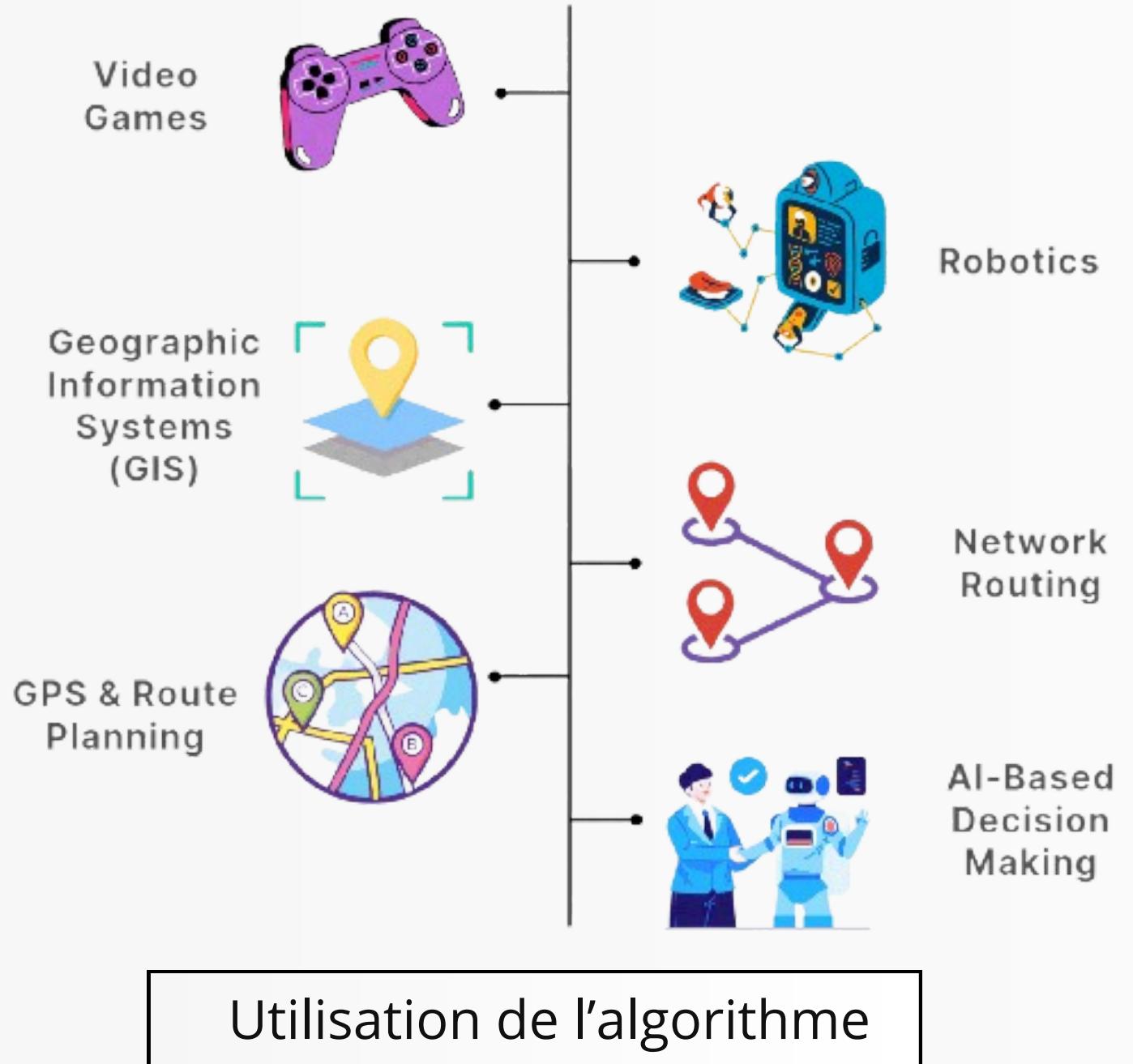
Jeux Video



Réseau WAN

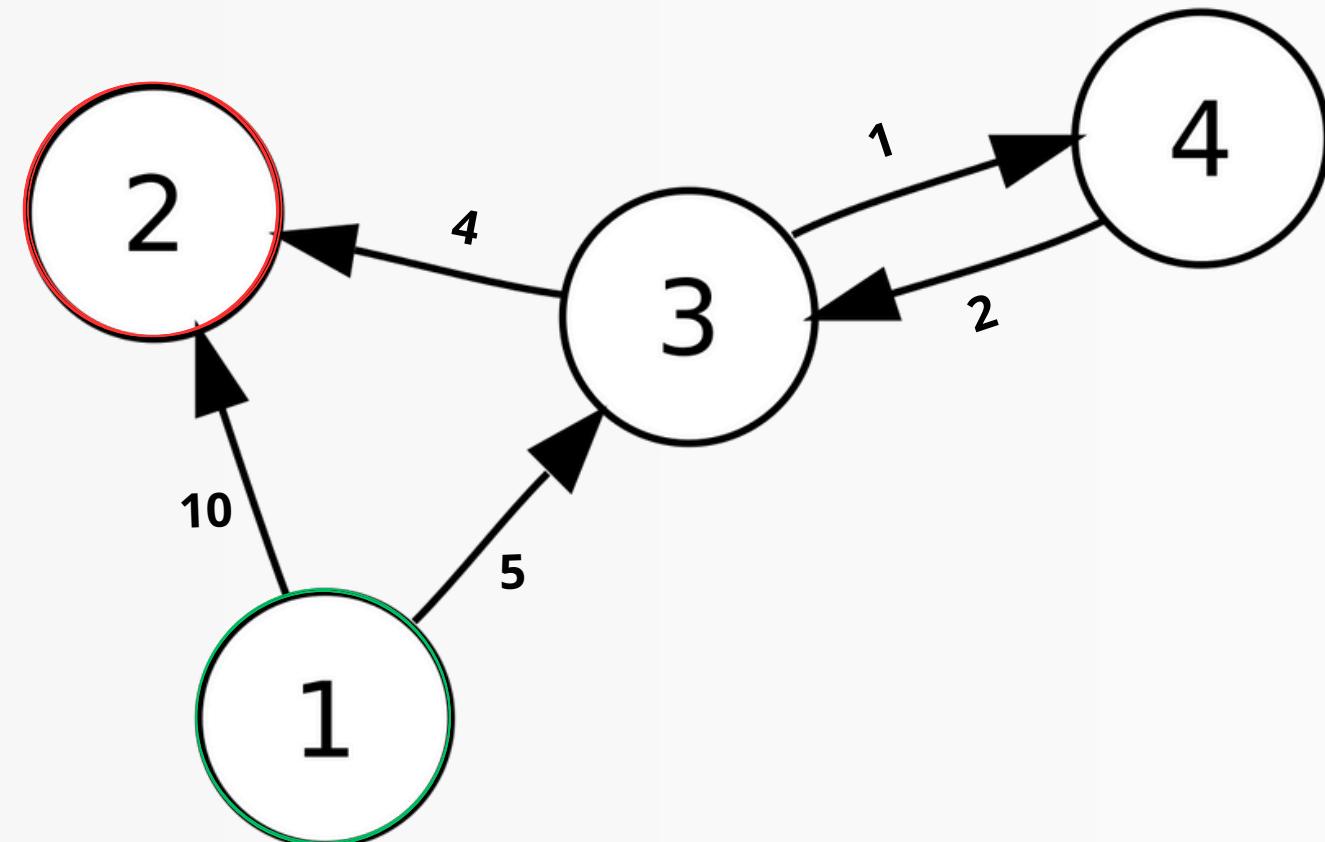
02 Introduction à l'algorithme A*

- L'algorithme A* sert à trouver le chemin le plus court en allant plus vite que les autres, grâce à une estimation du meilleur chemin possible.



03 Comment Algorithme A* Fonctionne ?

- Comment peut-on déterminer le chemin le plus court dans ce graphe ?

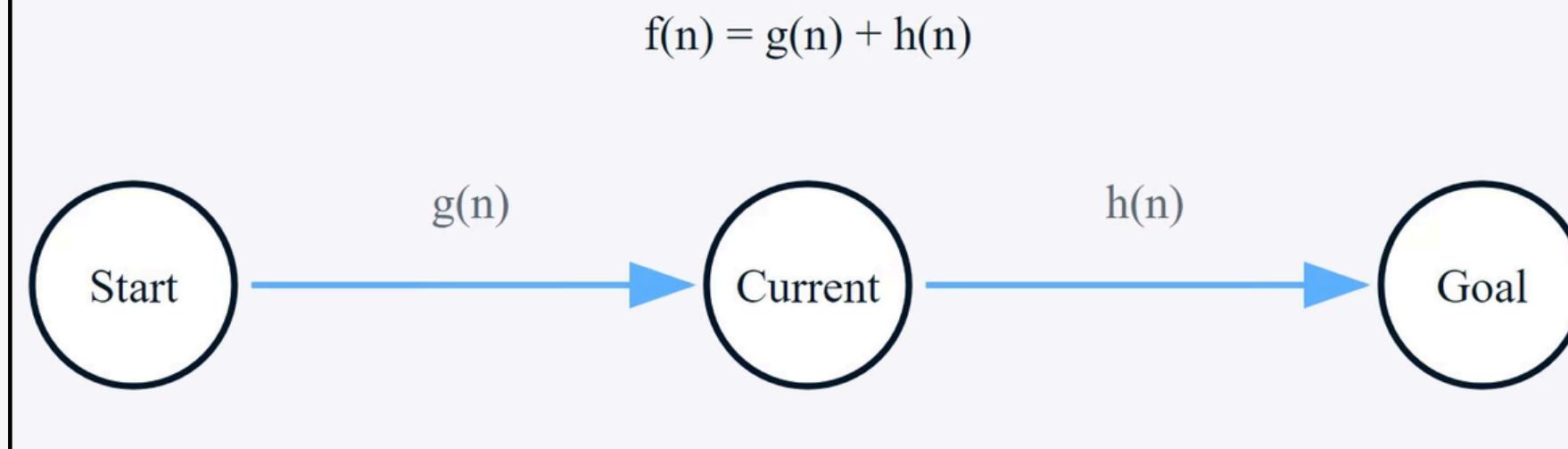


Son Principe:

A* base sur la fonction: $f(n) = g(n) + h(n)$

- $g(n)$: le coût réel du chemin depuis le sommet de départ jusqu'au sommet n.
- $h(n)$: une estimation du coût pour aller du sommet n au sommet d'arrivée.
- $f(n)$: le coût total estimé pour aller du sommet de départ au sommet d'arrivée en passant par le sommet n

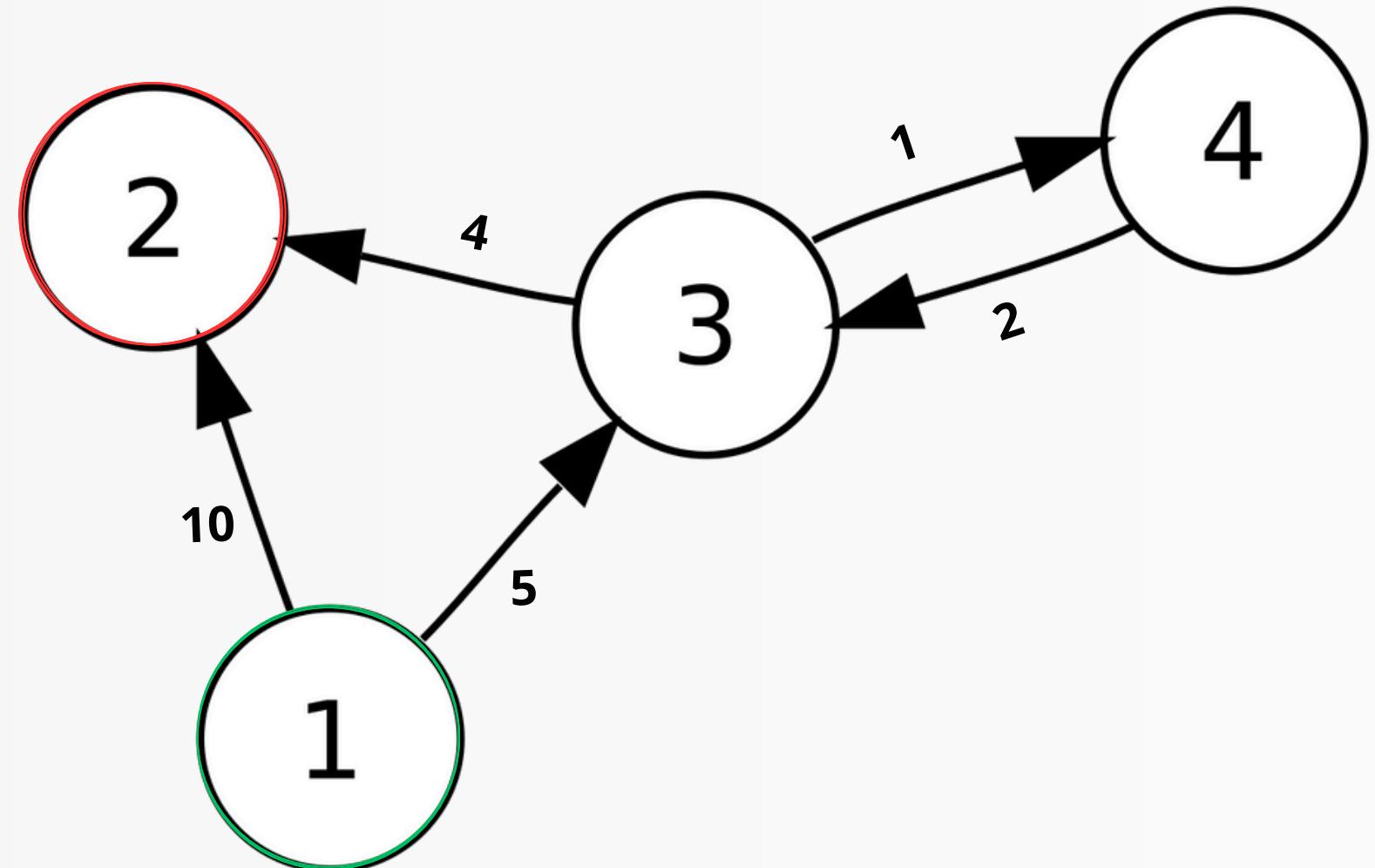
$$f(n) = g(n) + h(n)$$



À noter que l'heuristique est calculée à partir des coordonnées x et y du sommet.

03 Comment Algorithme A* Fonctionne ?

- Étape 1 : Supposer une heuristique pour chaque sommet

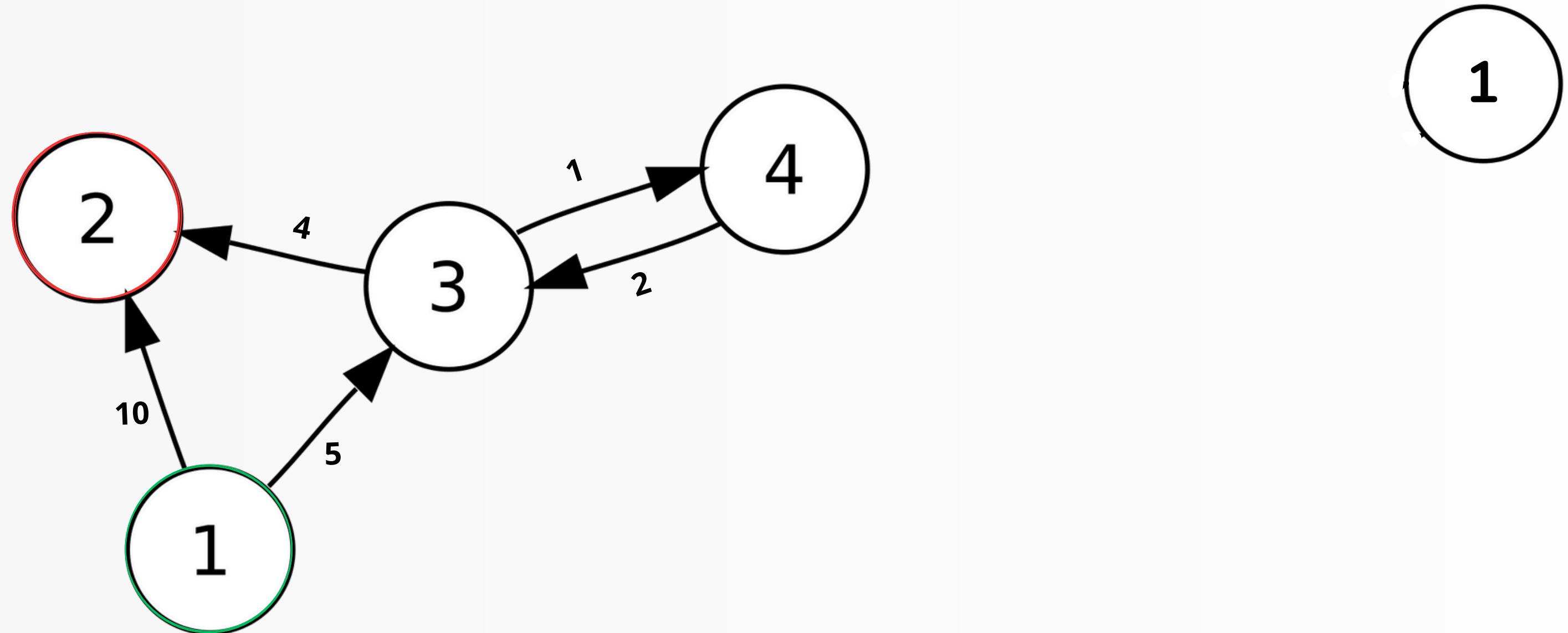


Sommet	Heuristique $h(n)$
1	3
2	0
3	2
4	5

Table d'heuristique

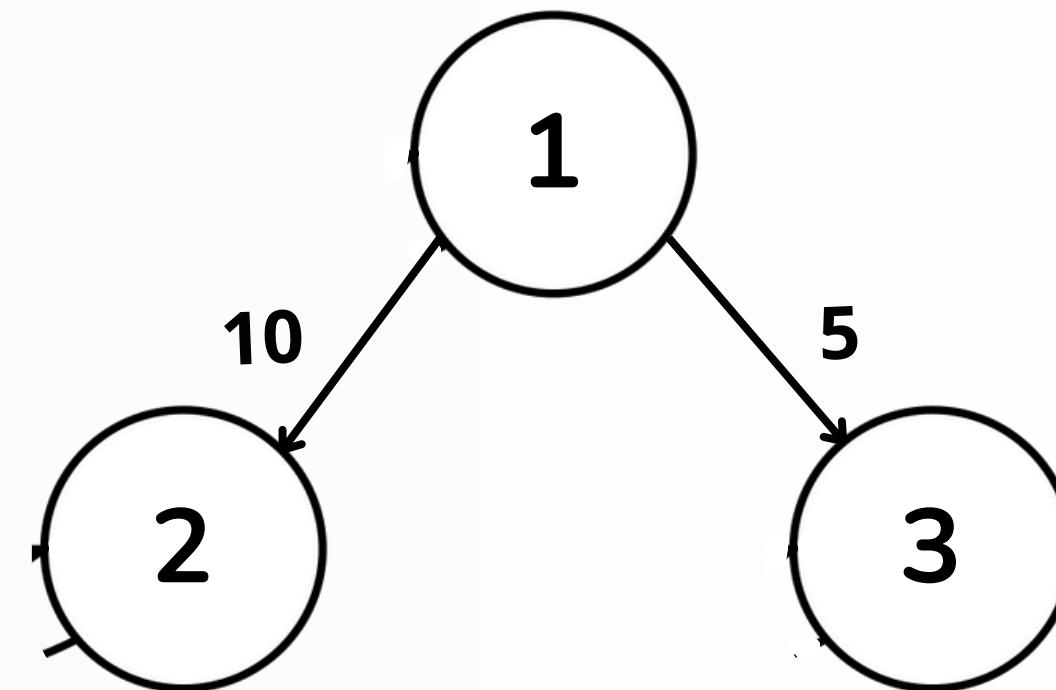
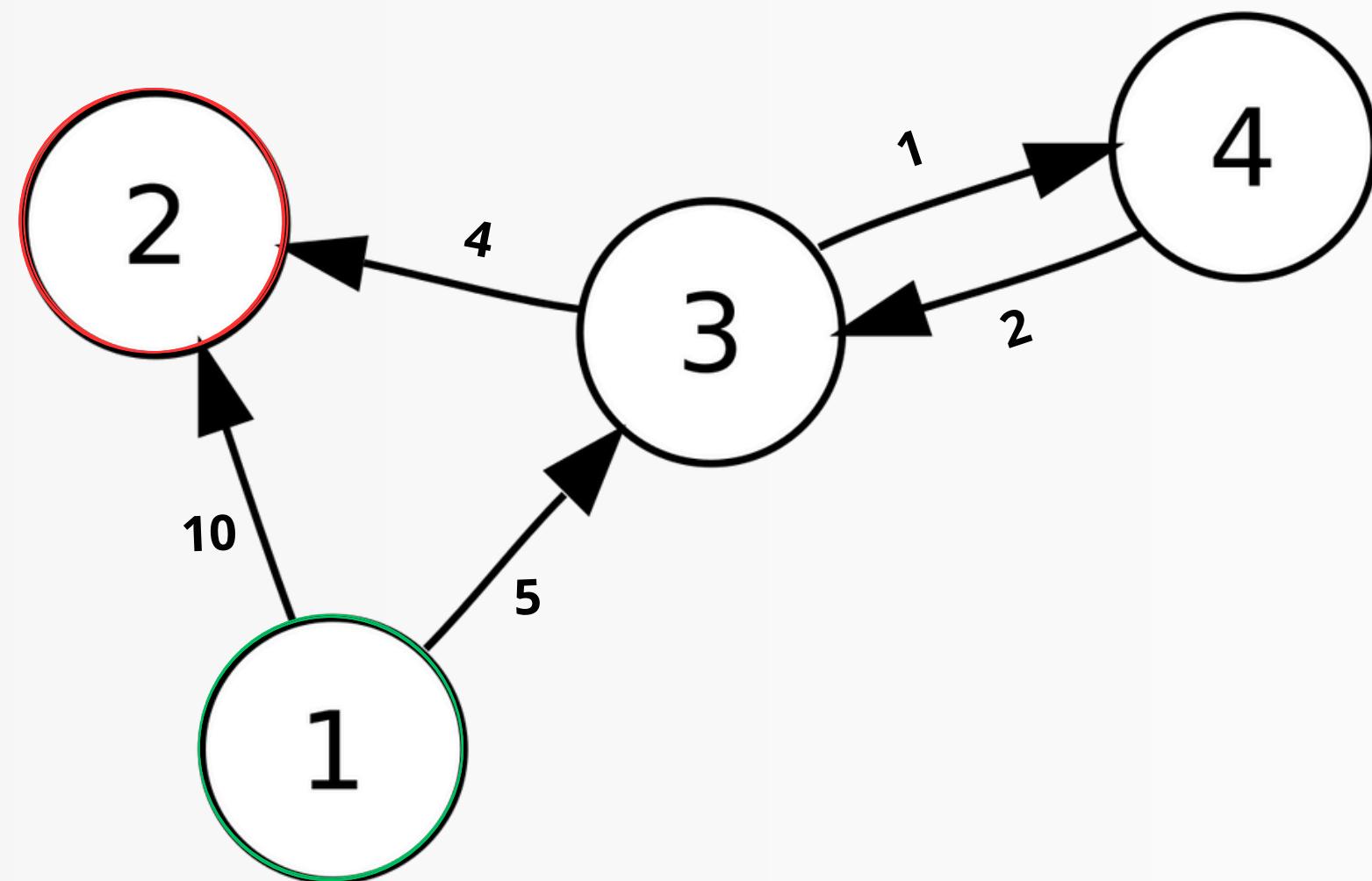
03 Comment Algorithme A* Fonctionne ?

- Étape 2: Choisir un sommet de départ.



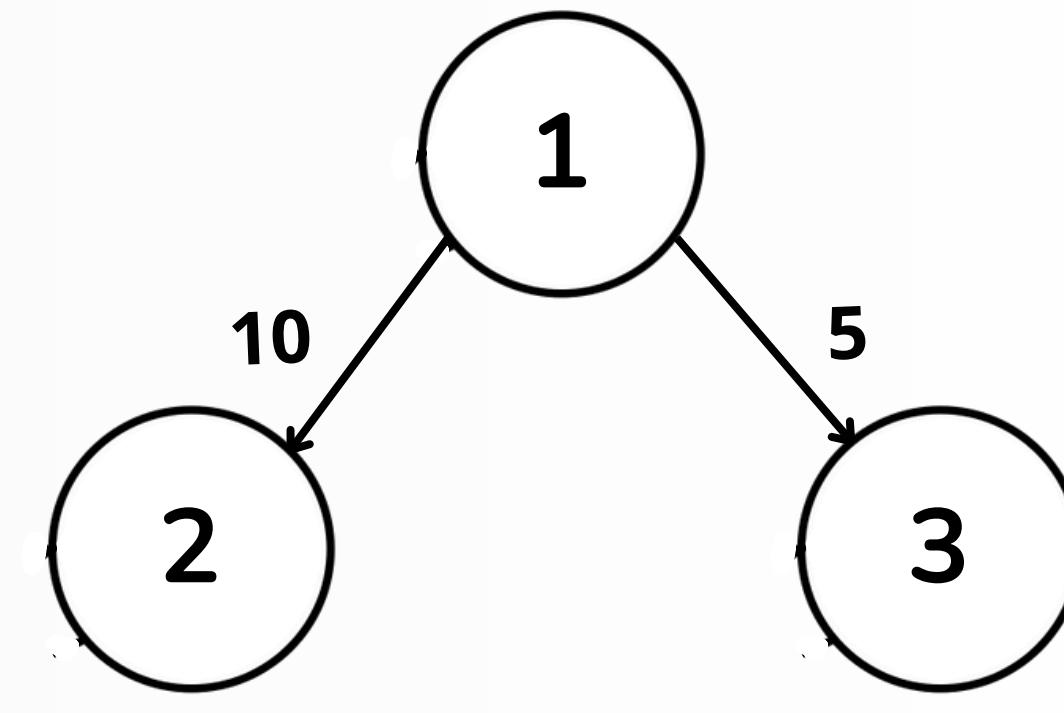
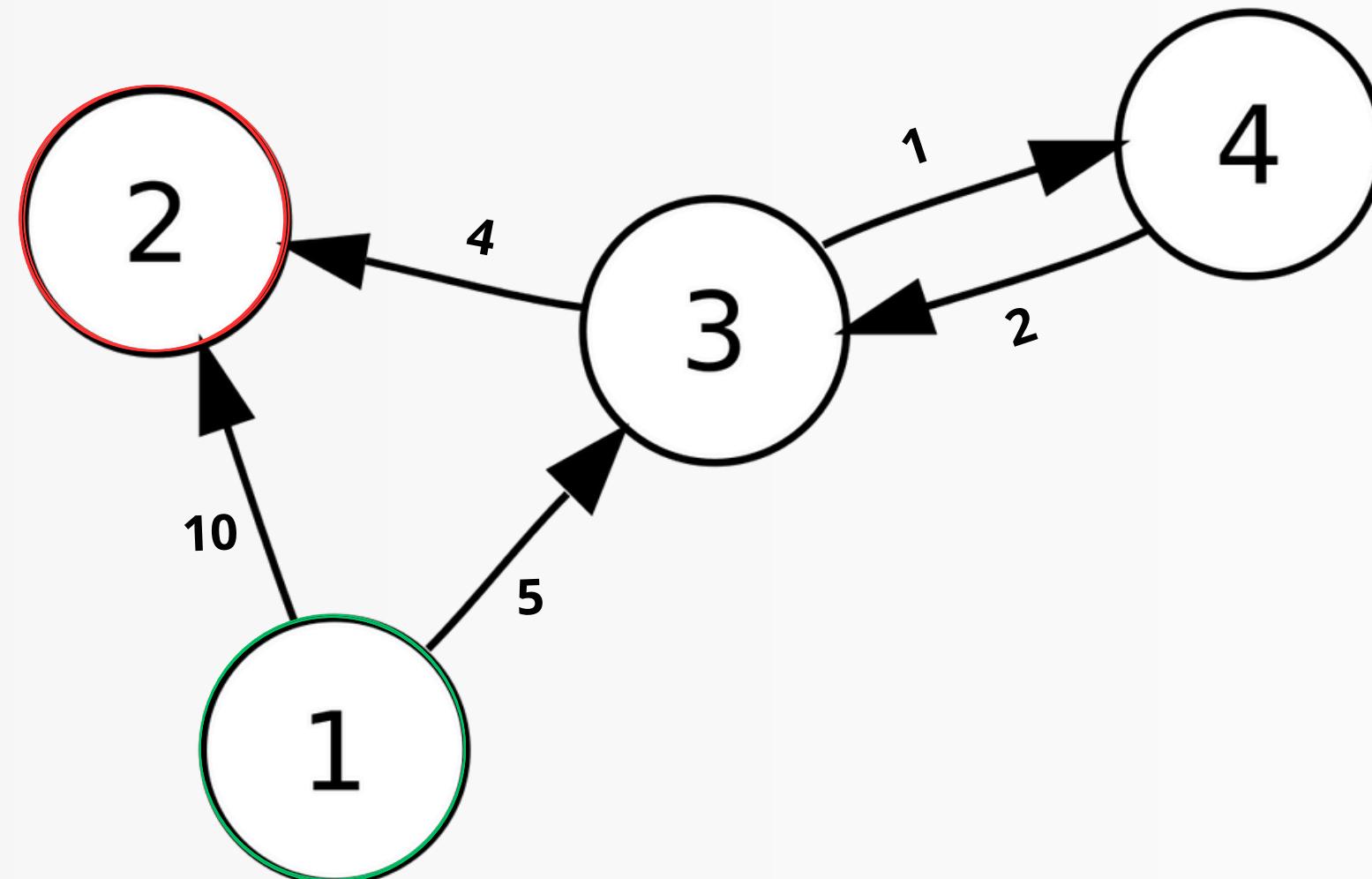
03 Comment Algorithme A* Fonctionne ?

- Étape 3: Examiner tous les voisins du sommet sélectionné



03 Comment Algorithme A* Fonctionne ?

- Étape 4: Calculer $f(n) = g(n) + h(n)$ pour chaque voisin.



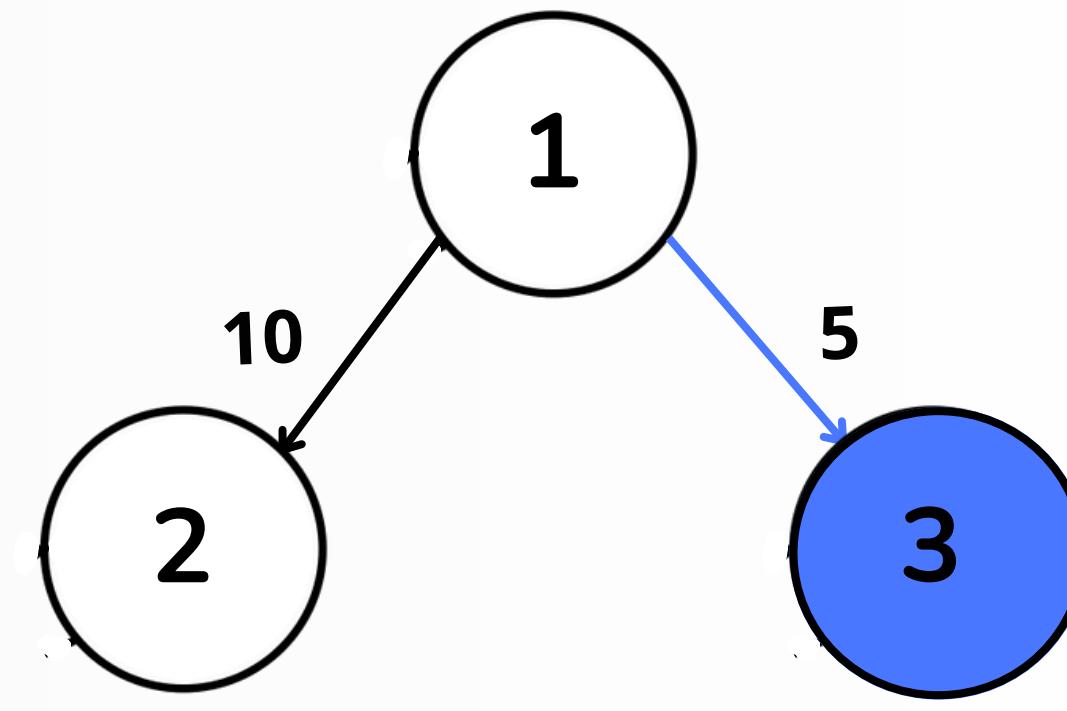
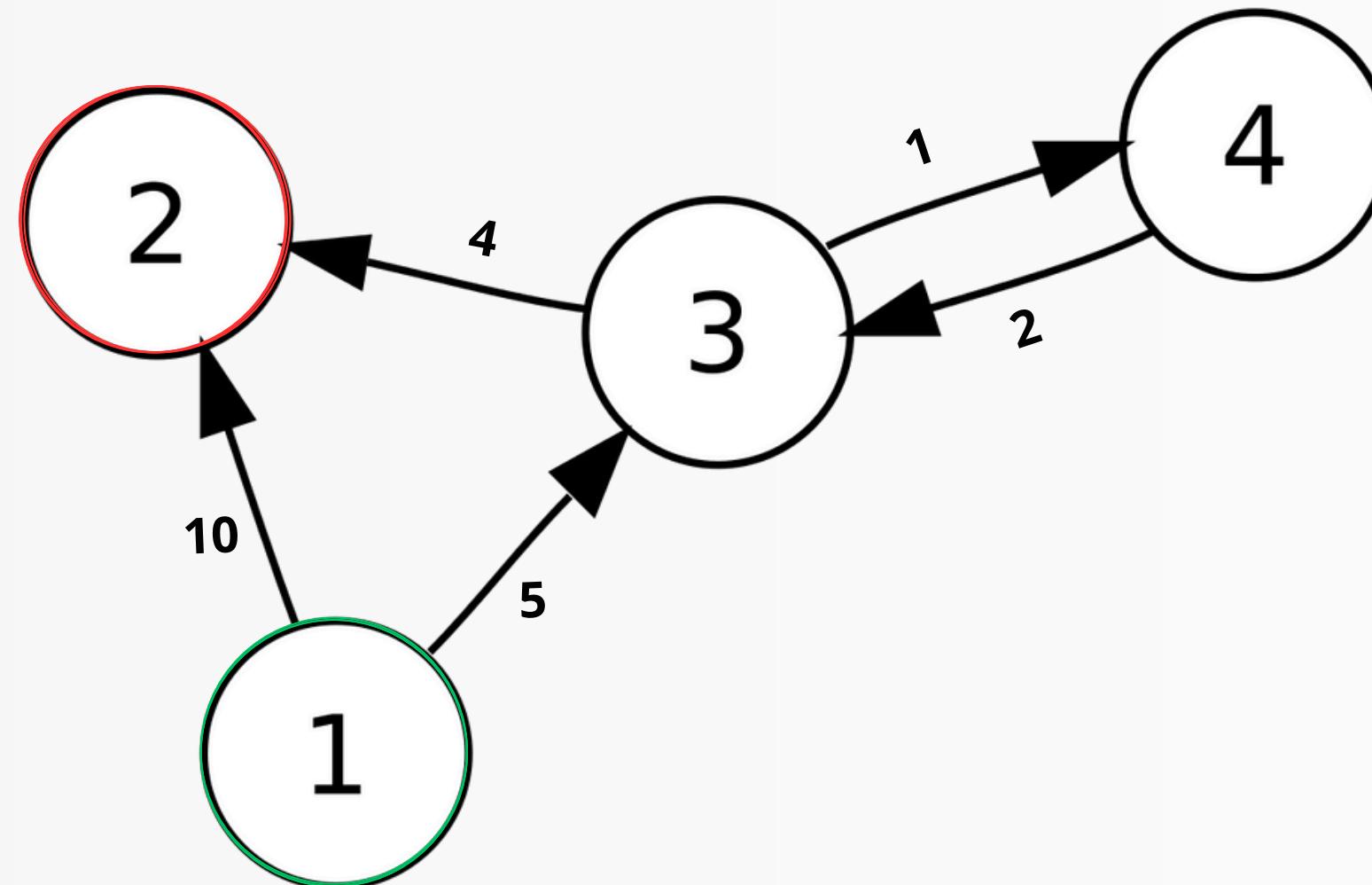
$$f(2) = 10$$

$$f(3) = 7$$

- $f(2) = h(2) + g(2) = 0 + 10 = 10$
- $f(3) = h(3) + g(3) = 2 + 5 = 7$

03 Comment Algorithme A* Fonctionne ?

- Étape 5: Choisir le sommet avec le plus petit $f(n)$.



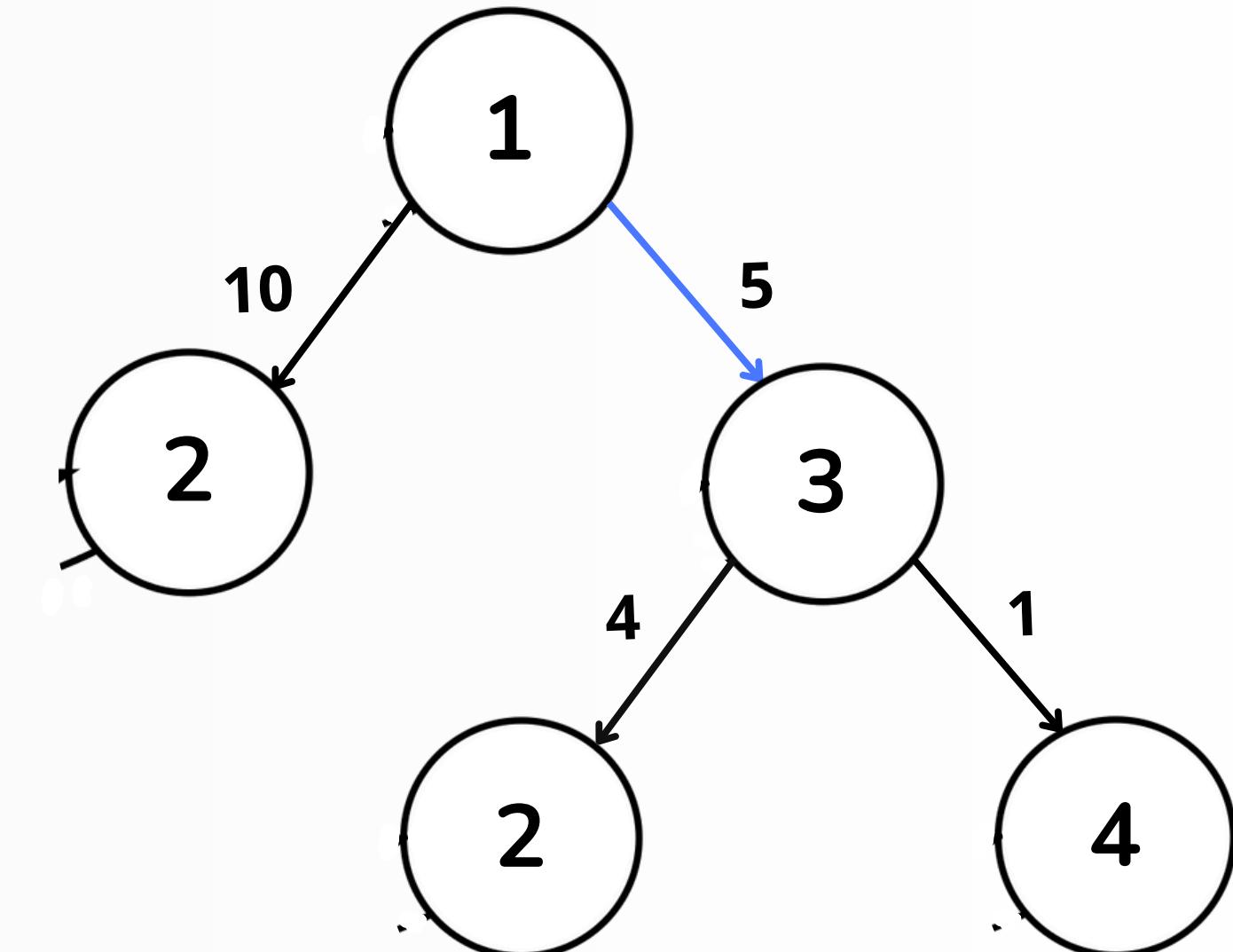
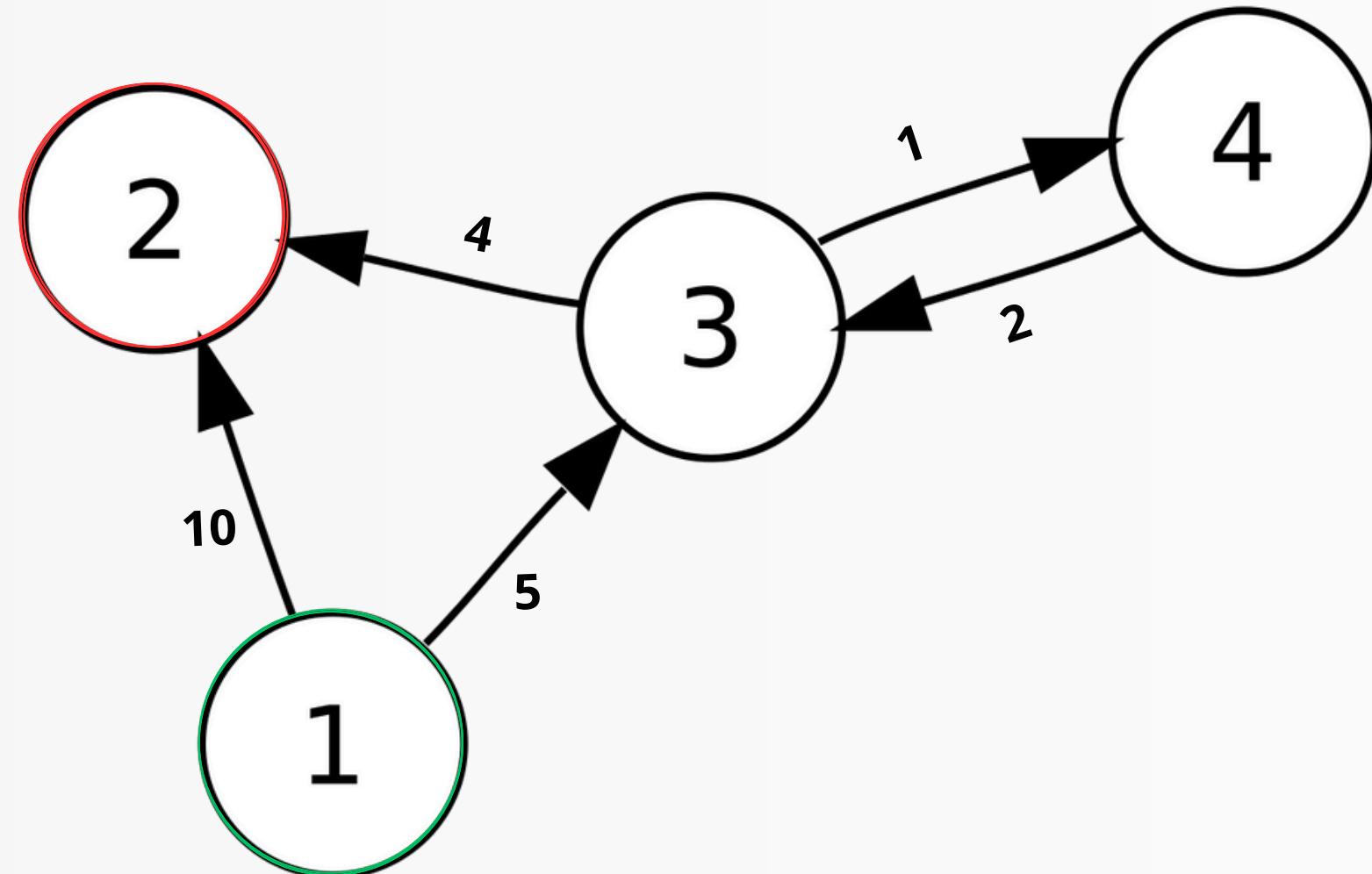
$$f(2) = 10$$

$$\mathbf{f(3) = 7}$$

$f(3) < f(2)$: donc le sommet sélectionné est 3.

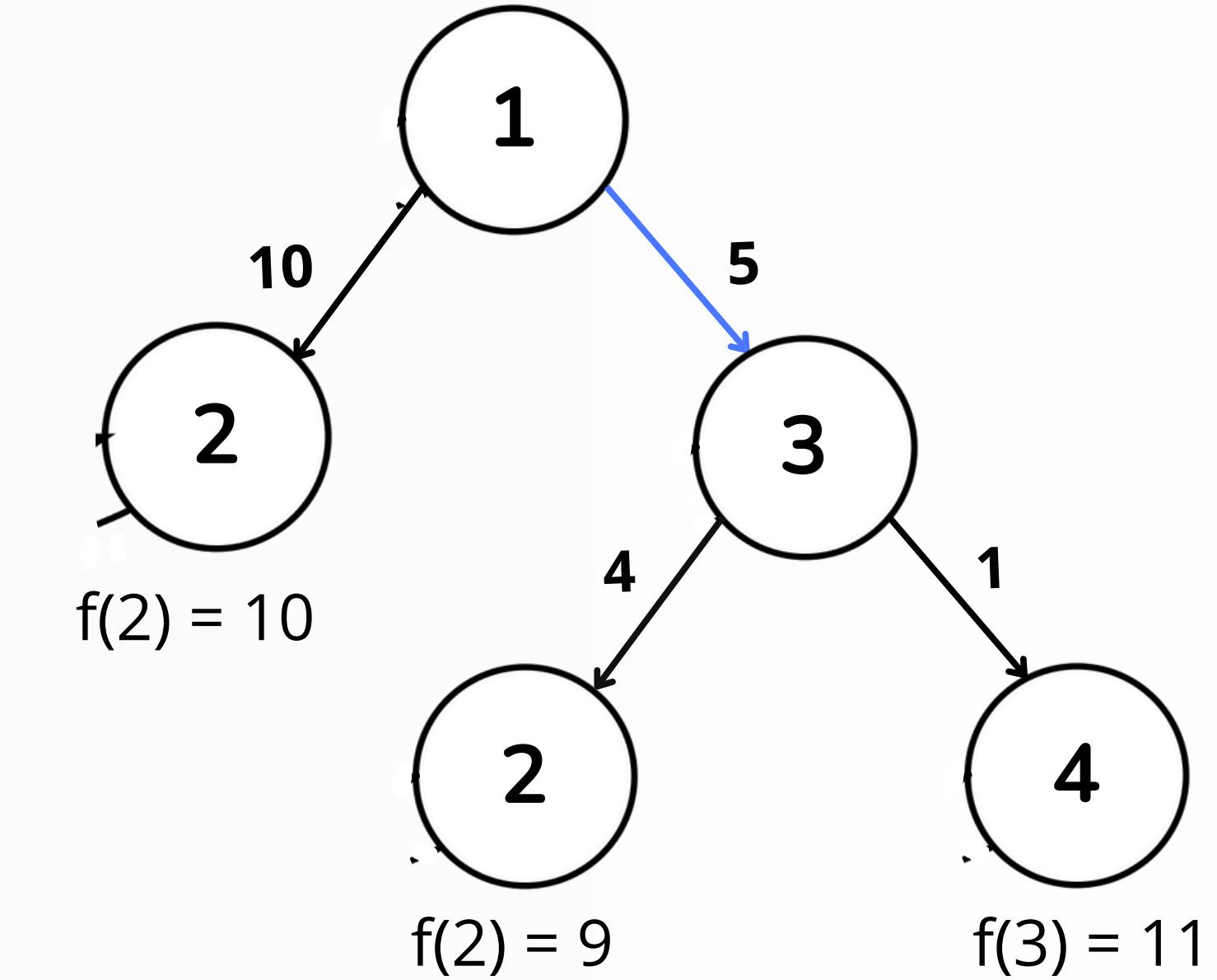
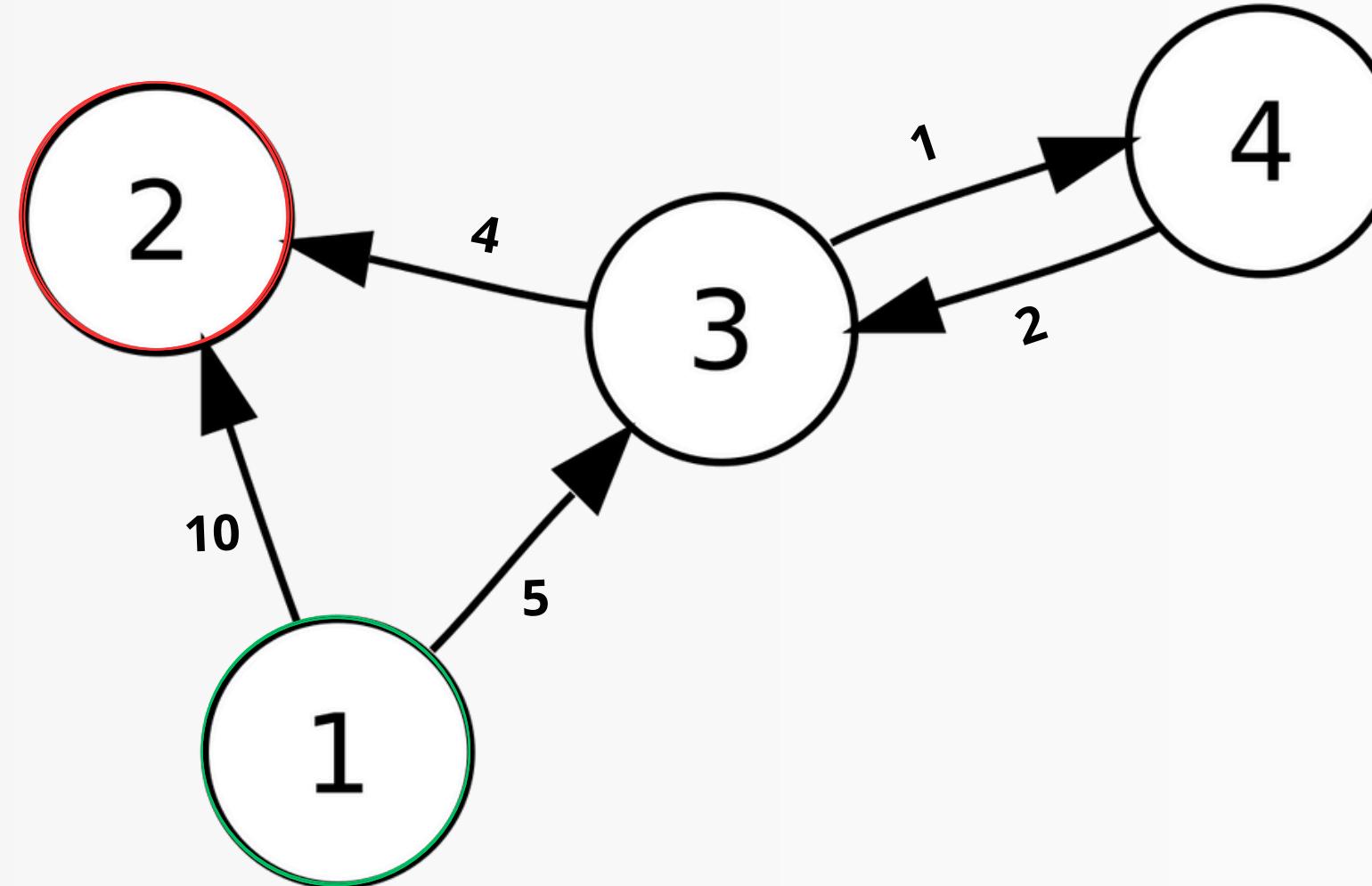
03 Comment Algorithme A* Fonctionne ?

- Étape 3.1: Examiner tous les voisins du sommet sélectionné



03 Comment Algorithme A* Fonctionne ?

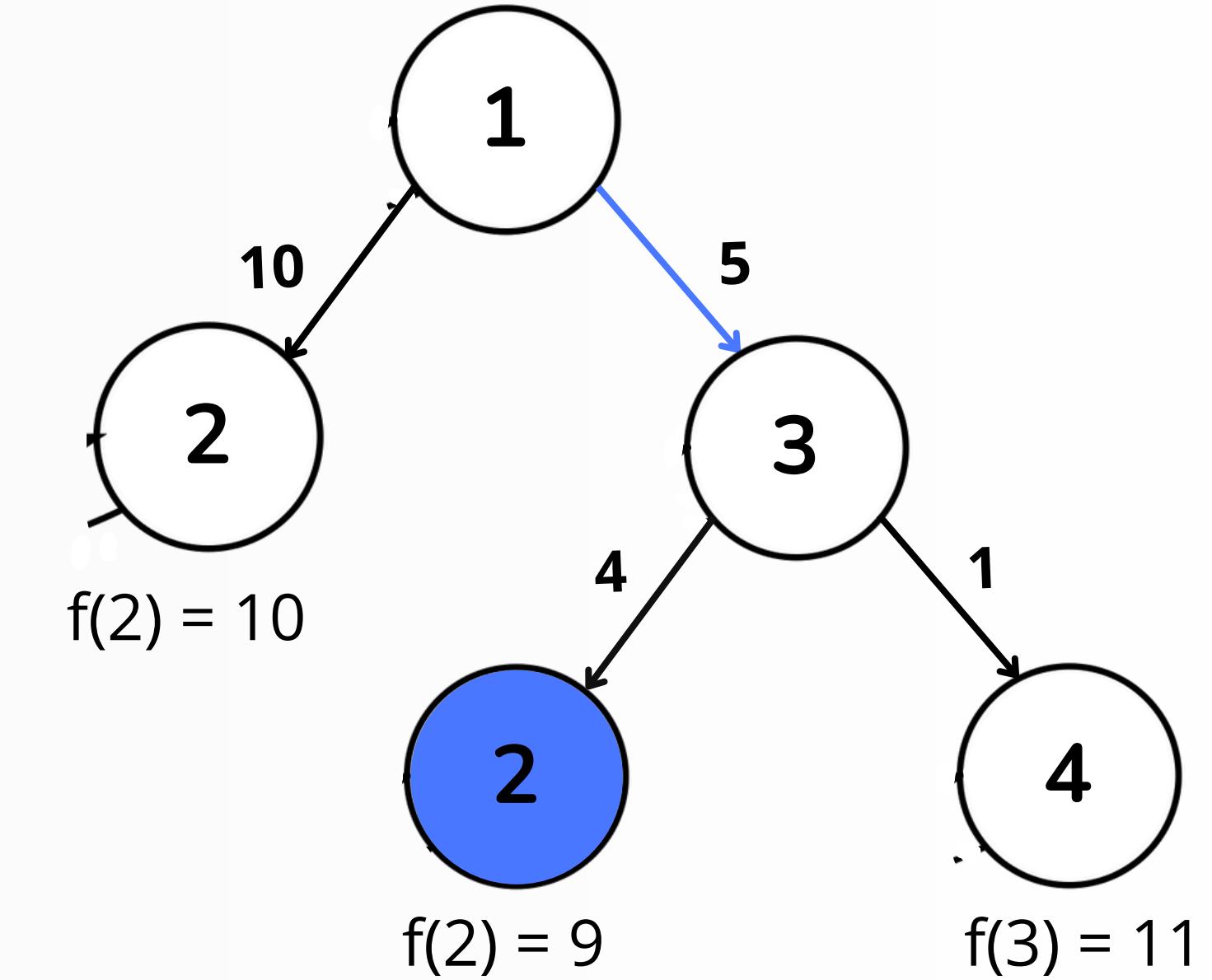
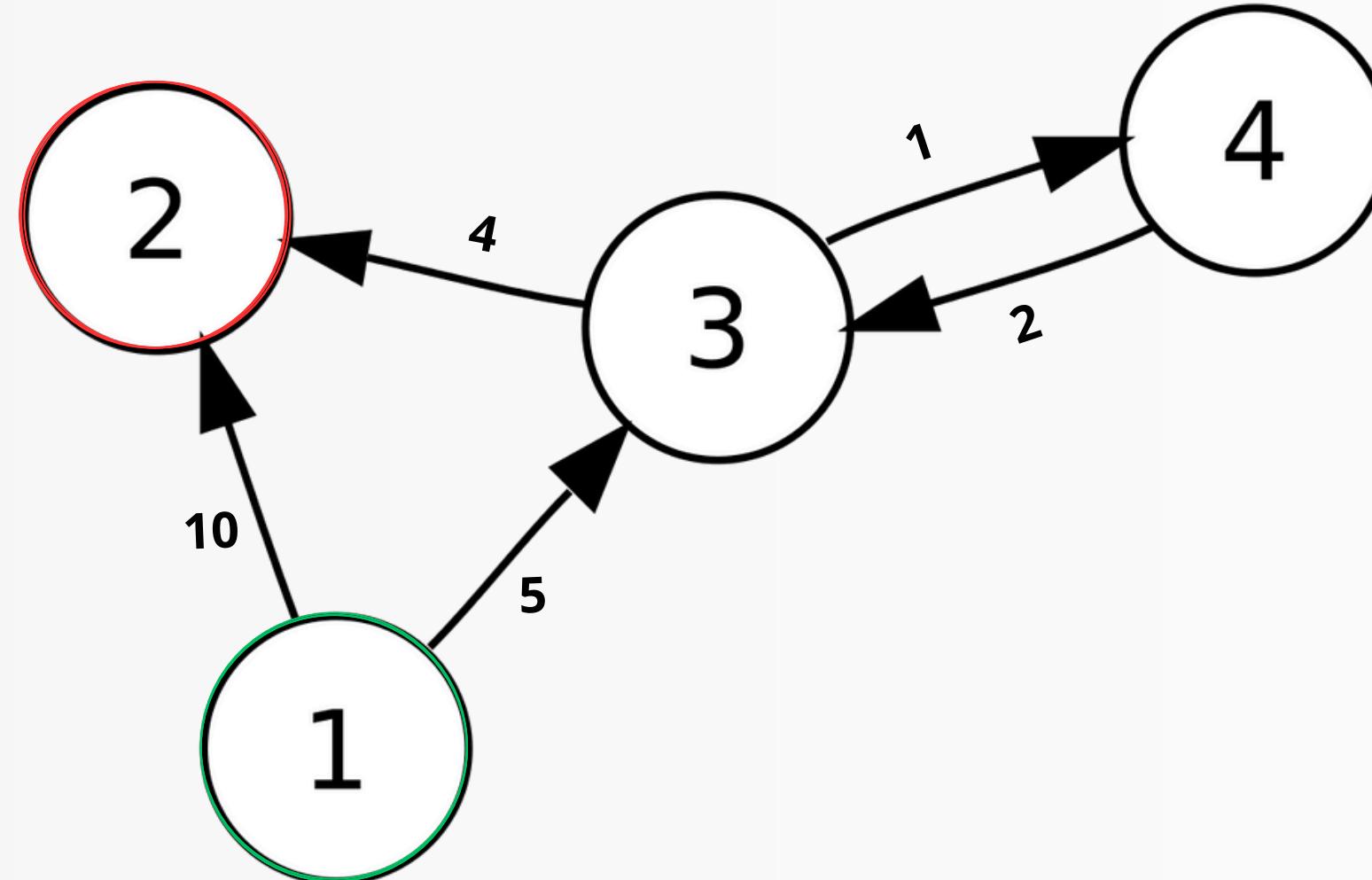
- Étape 4.1: Calculer $f(n) = g(n) + h(n)$ pour chaque voisin.



- $f(2) = h(2) + g(2) = 0 + 5 + 4 = 9$
- $f(4) = h(4) + g(4) = 5 + 6 = 11$

03 Comment Algorithme A* Fonctionne ?

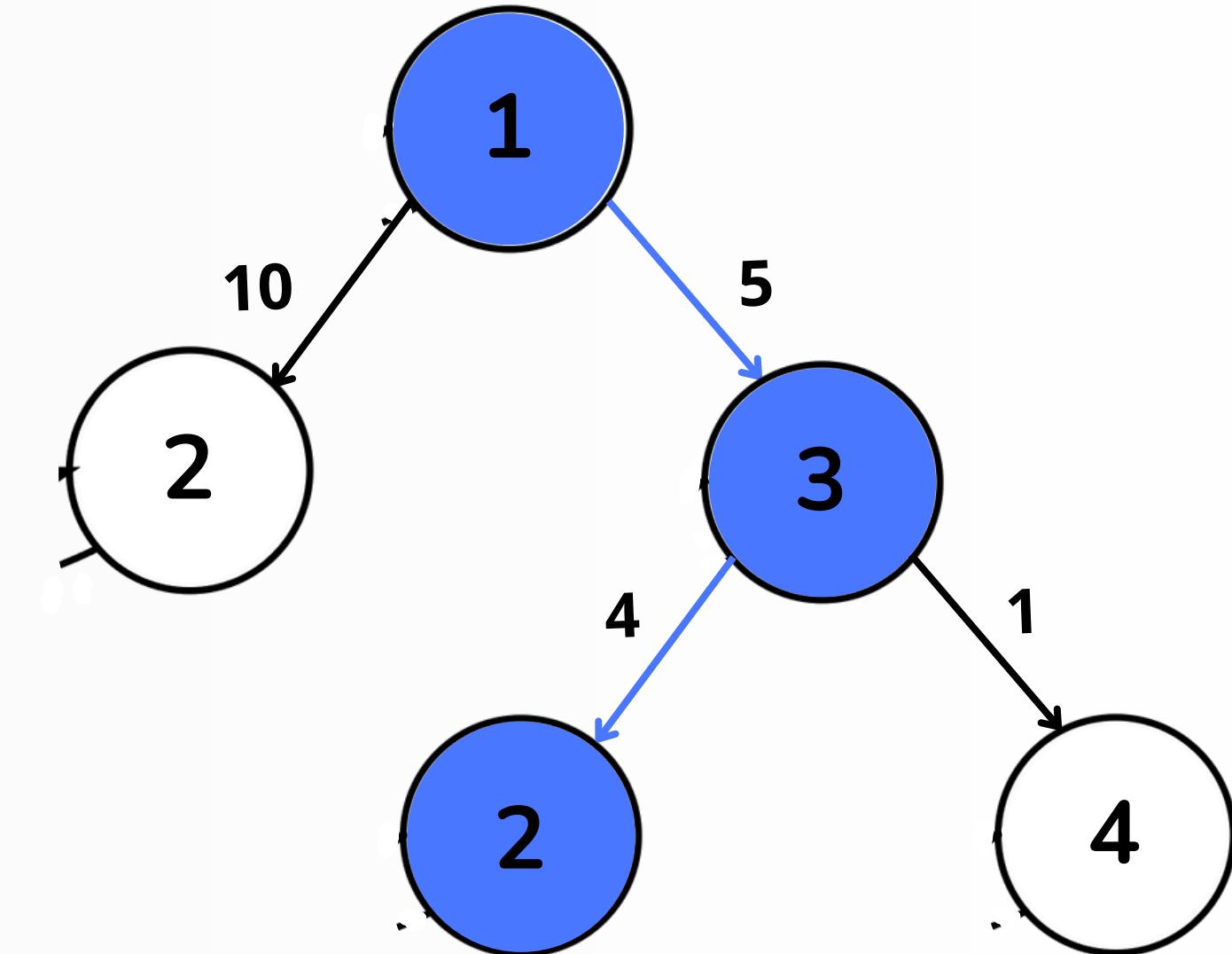
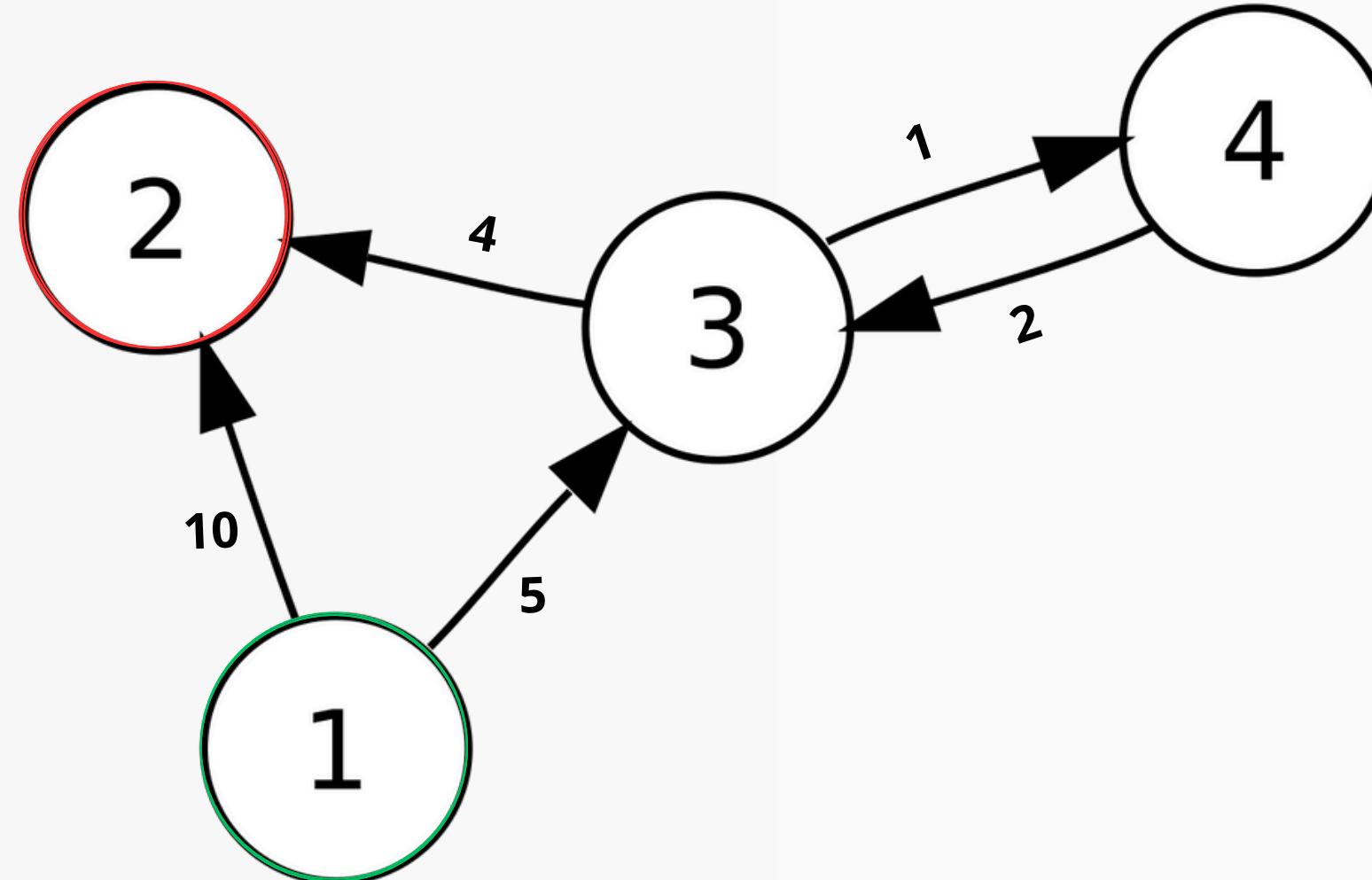
- Étape 5.1: Choisir le nœud avec le plus petit $f(n)$.



- $f(2)$ c'est la plus petite valeur donc le sommet sélectionné est 2.

03 Comment Algorithme A* Fonctionne ?

- L'algorithme s'arrête lorsque le sommet sélectionné est le sommet d'arrivée.



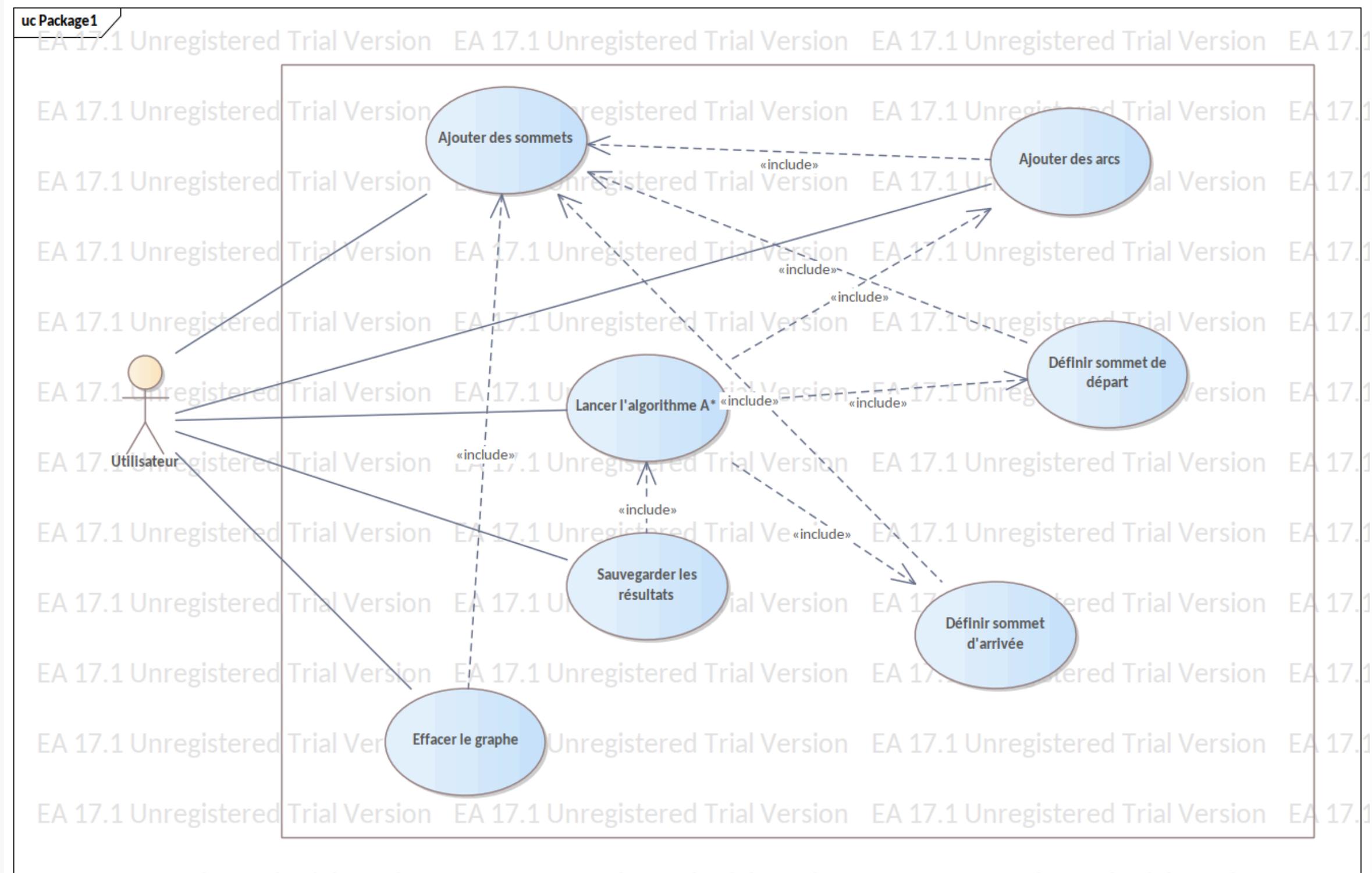
- Donc, le chemin le plus court pour aller de 1 vers 2 est [1, 3, 2] avec un coût de 9.

03 Analyse des besoins et conception

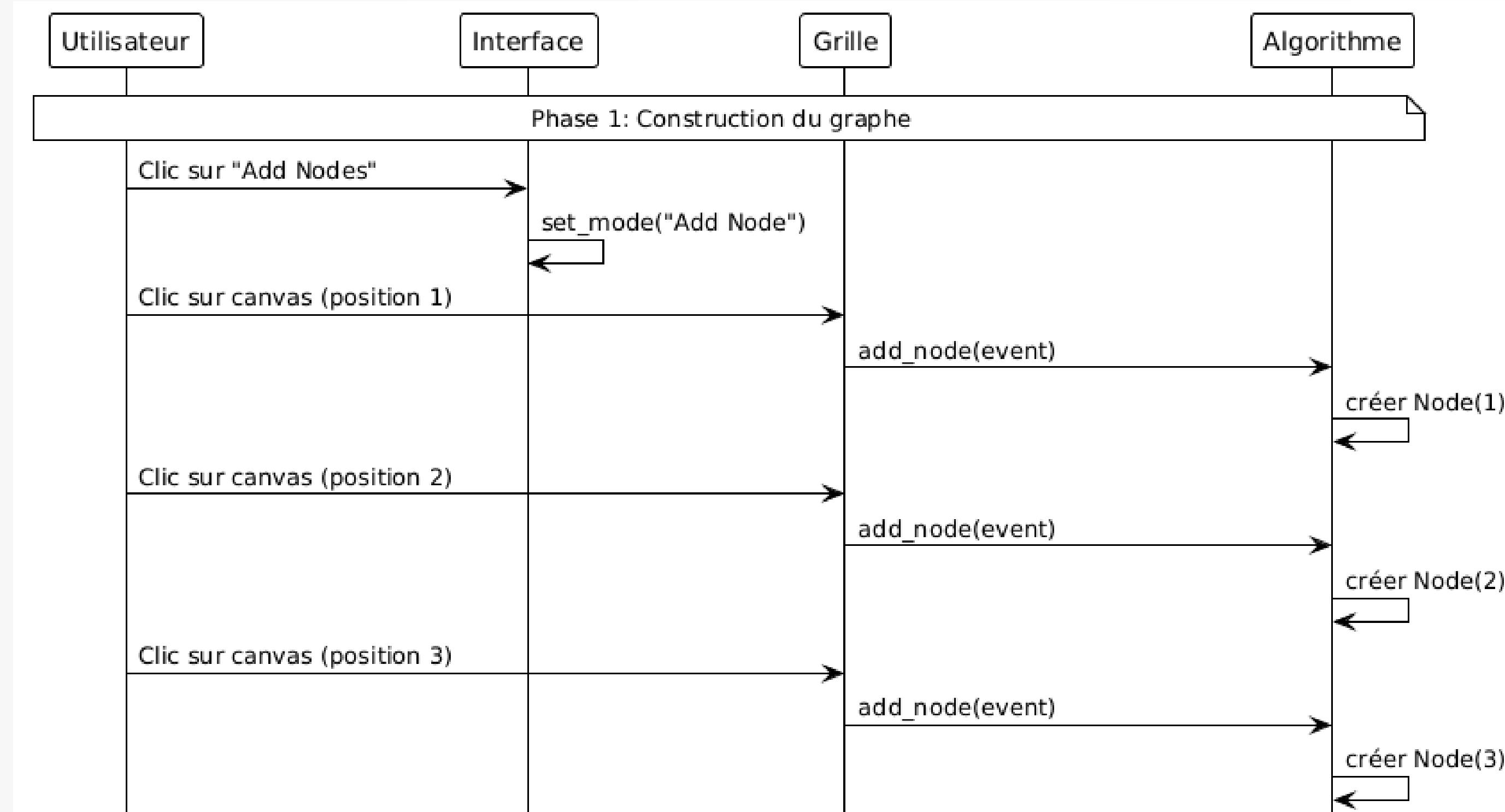
- Langage de modélisation unifié UML :



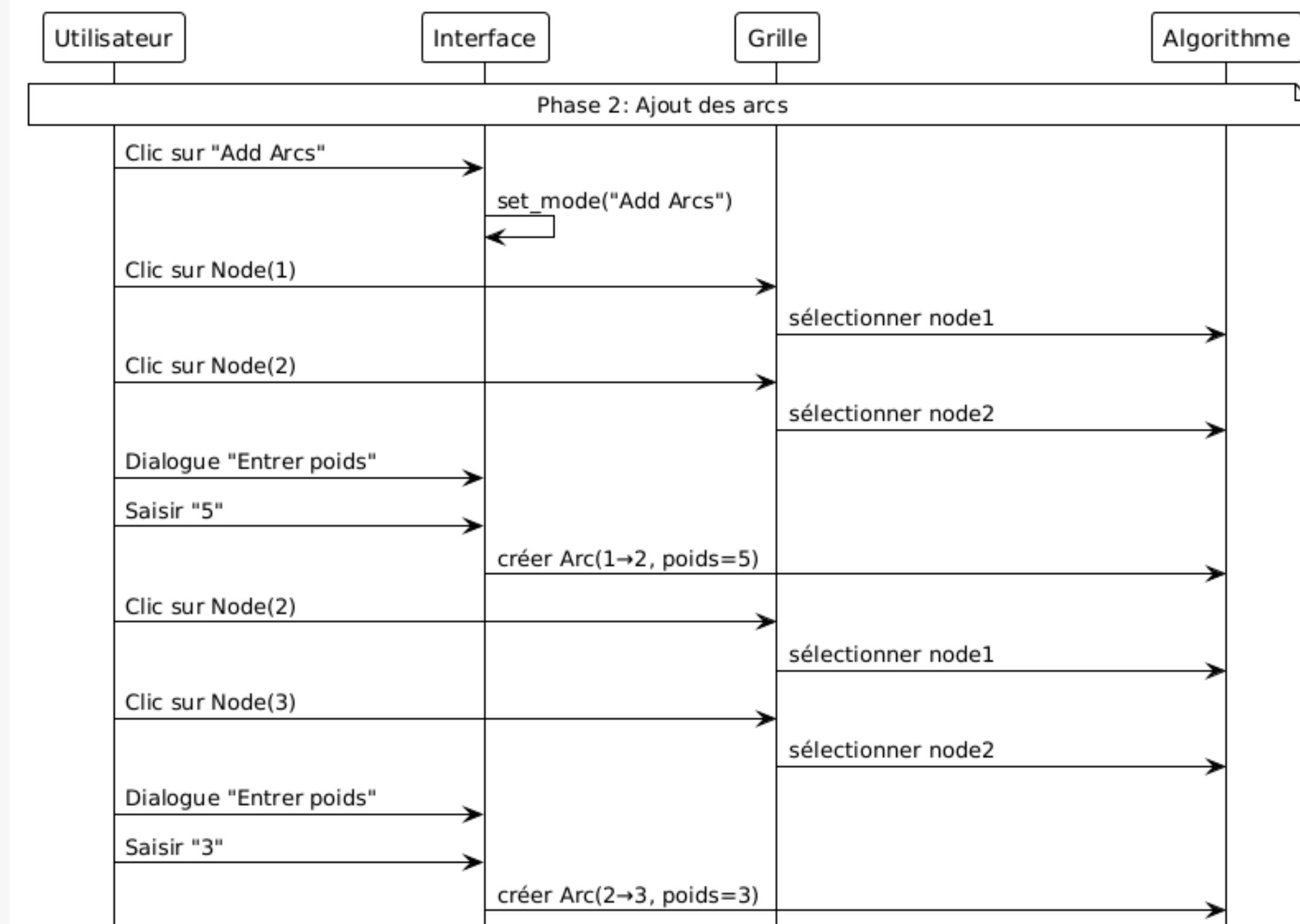
- Dans ce diagramme, l'acteur principal est l'utilisateur. Il peut ajouter des sommets, exécuter l'algorithme, etc.



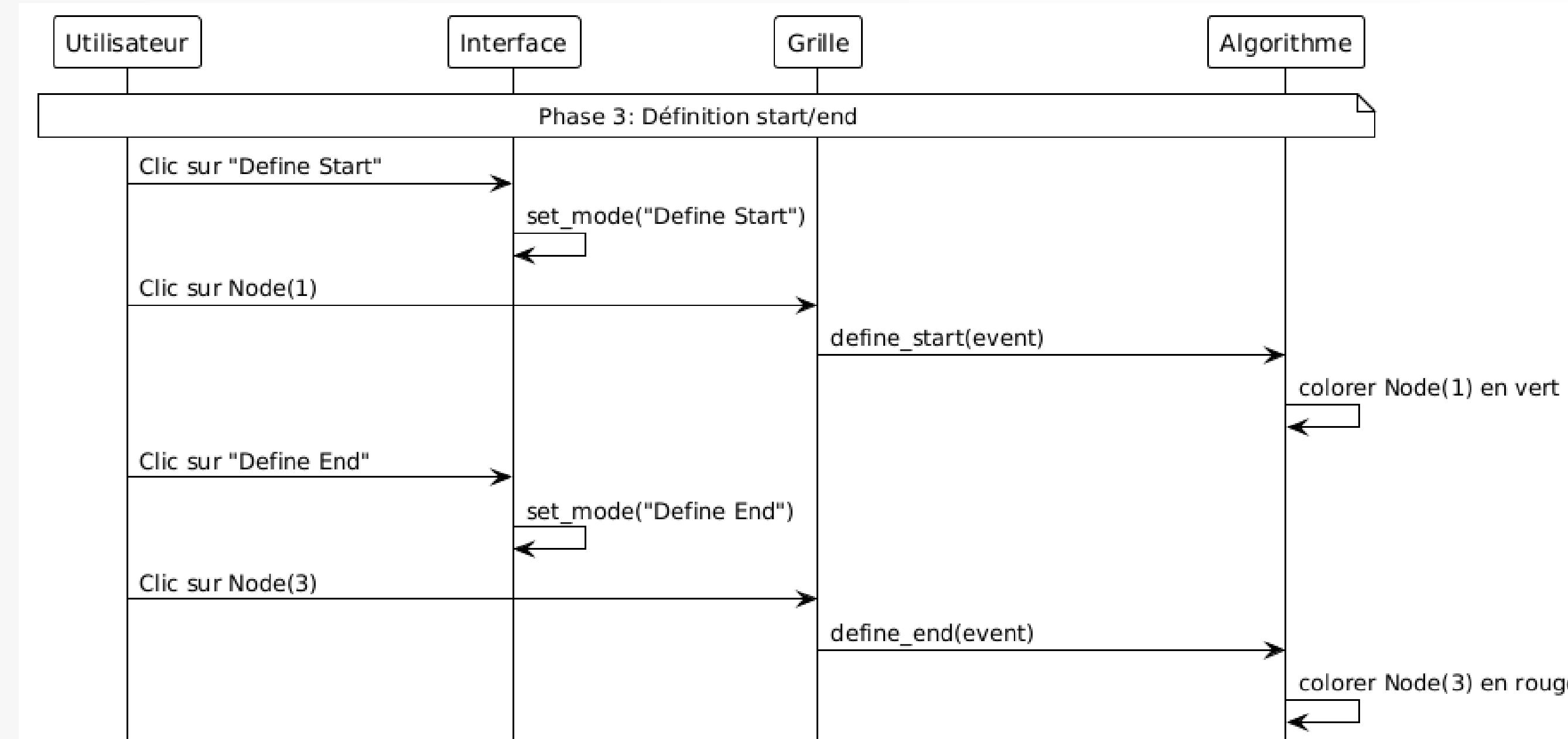
- Créer et placer des sommets sur la grille pour construire structure de base du graphe



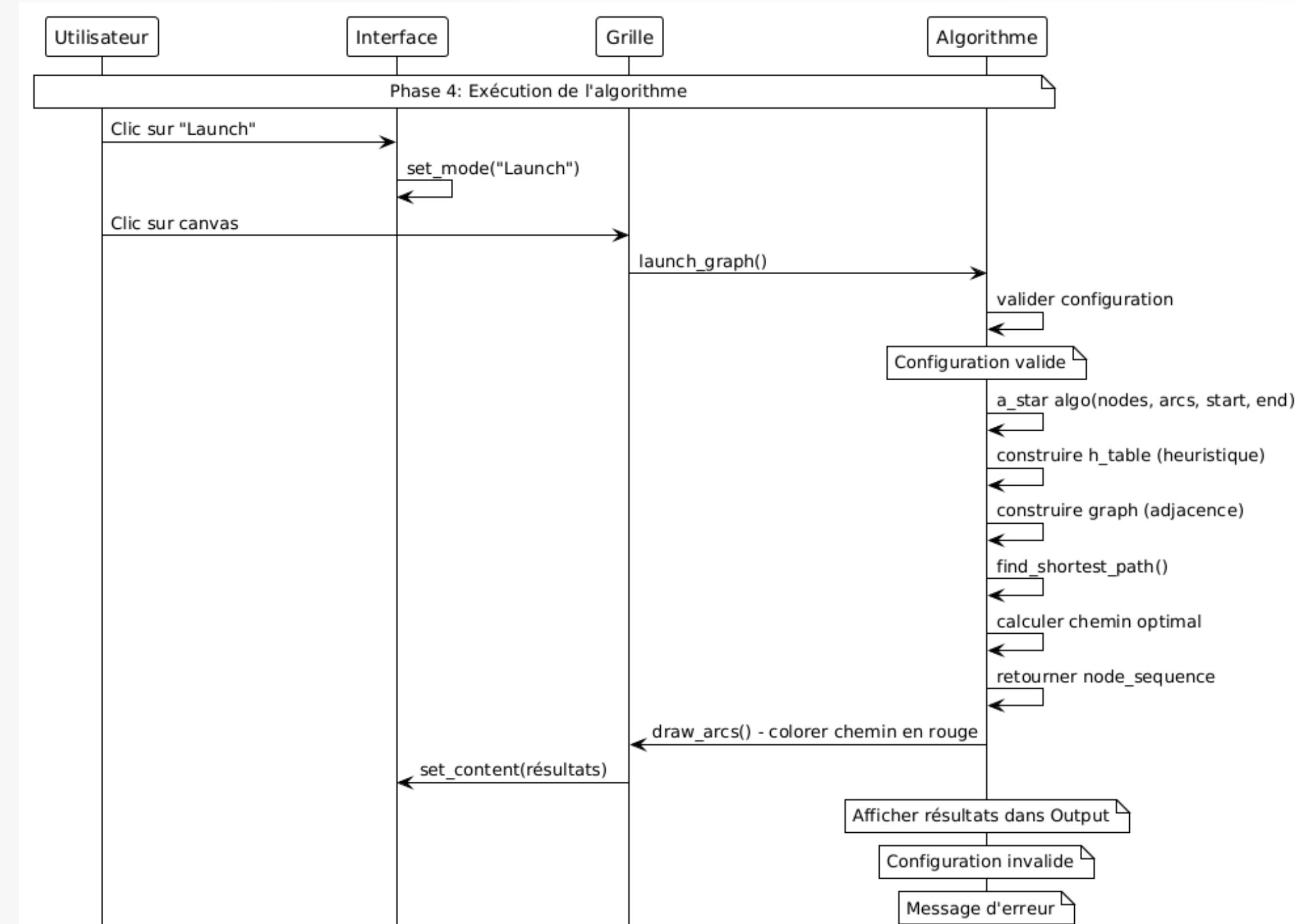
- Connecter les sommets existants avec des arcs pondérés pour former le graphe complet



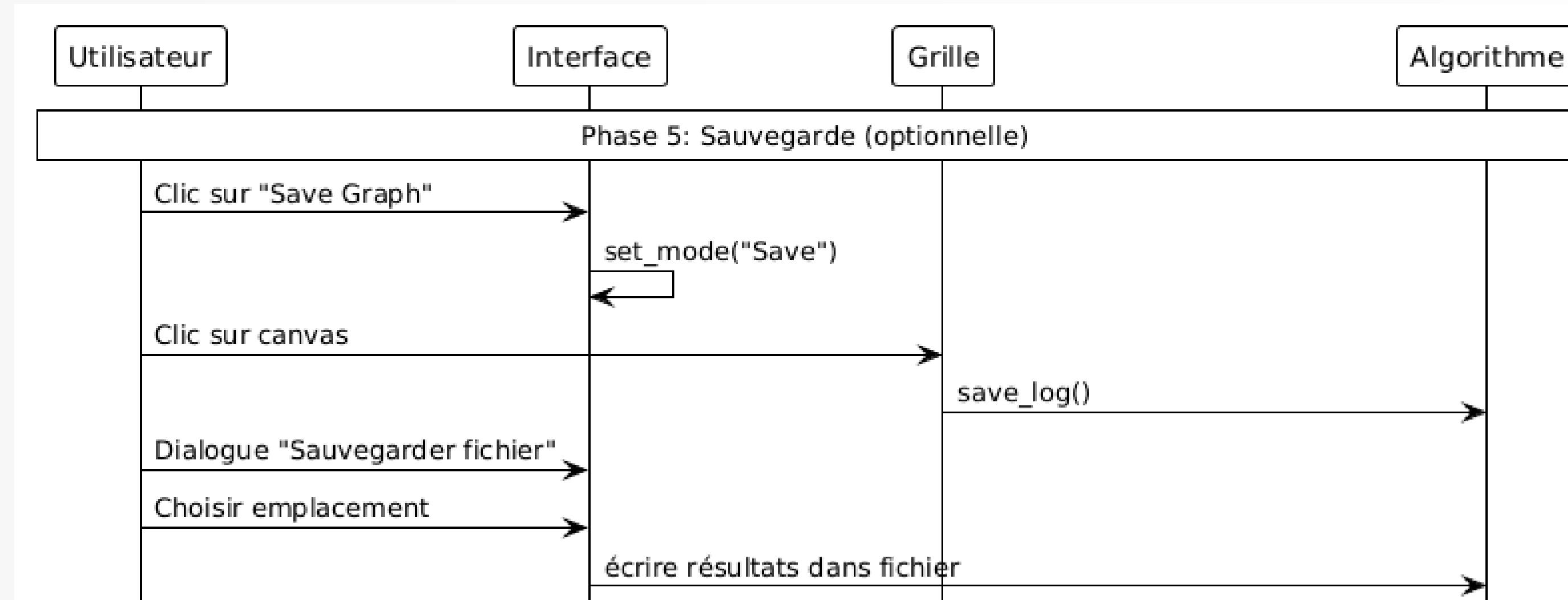
- Définir les sommets de départ et d'arrivée dans le graphe.



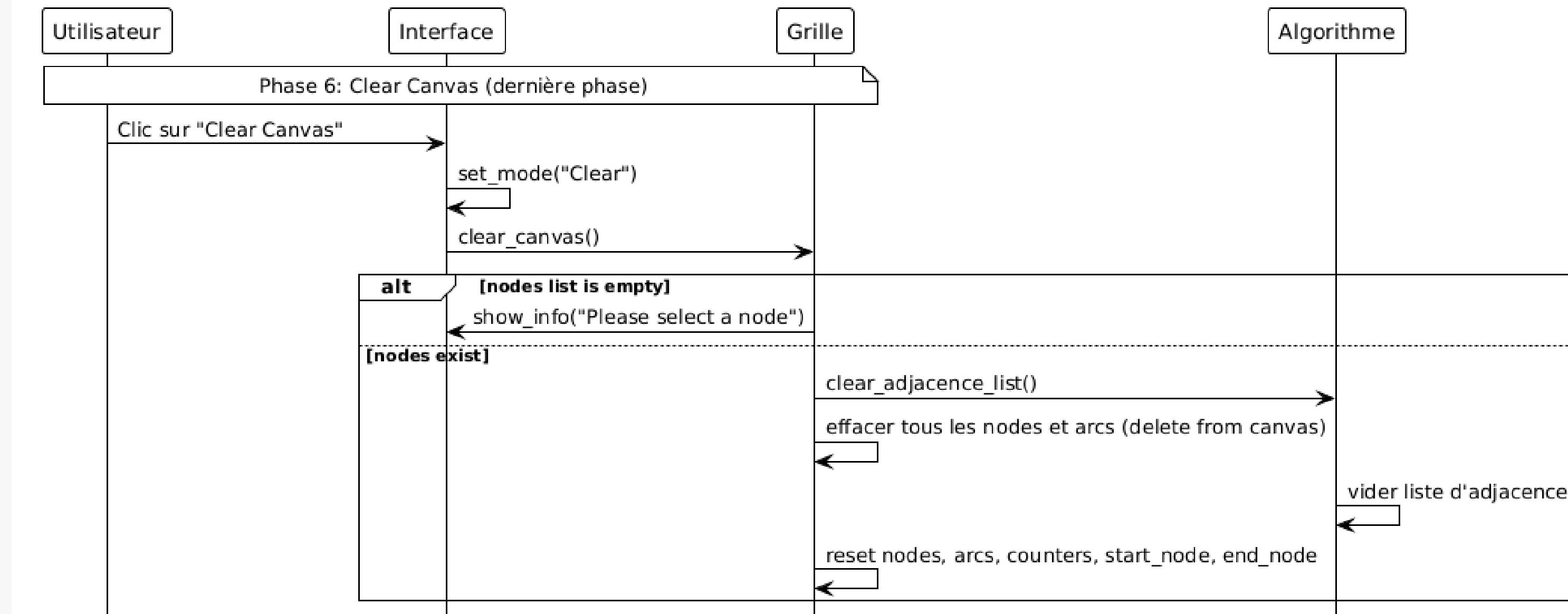
- Lancer l'algorithme de recherche du plus court chemin et afficher le résultat optimal



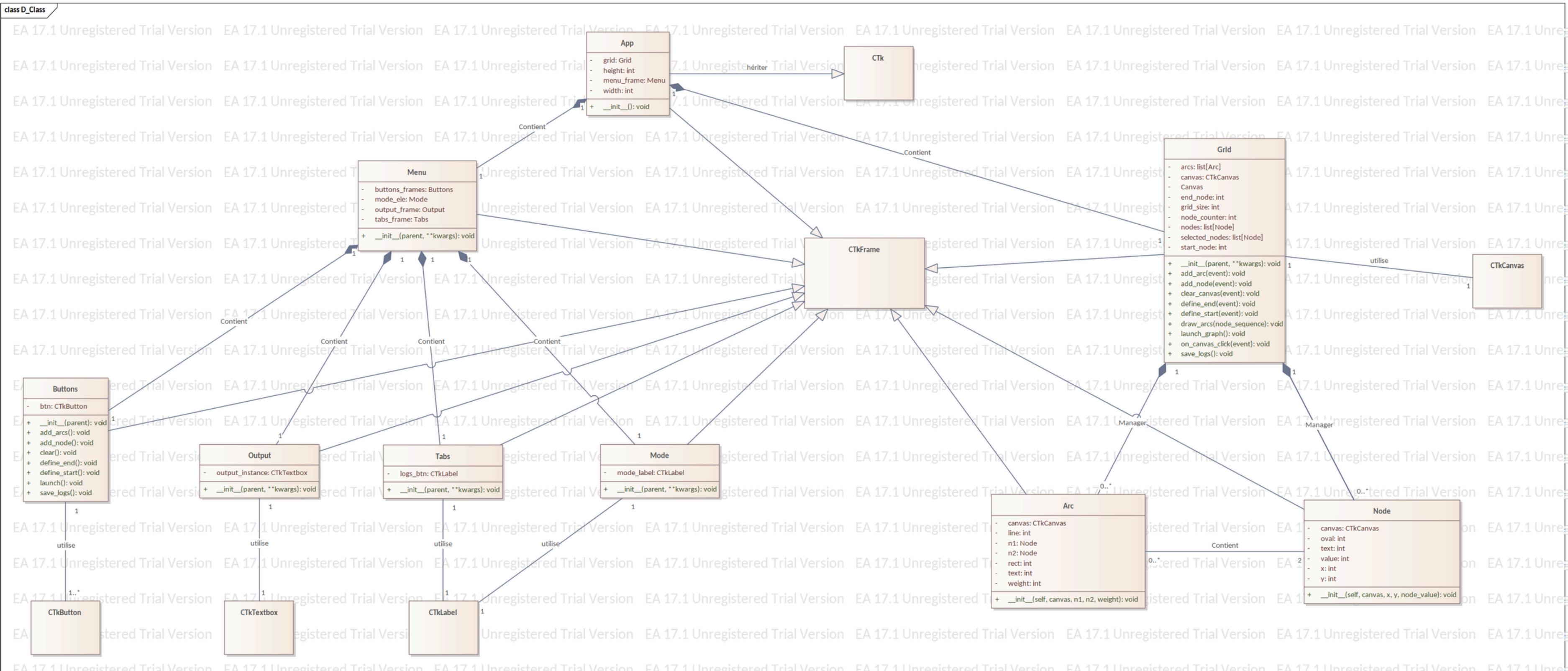
- Sauvegarder la configuration du graphe et les résultats obtenus pour une utilisation ultérieure

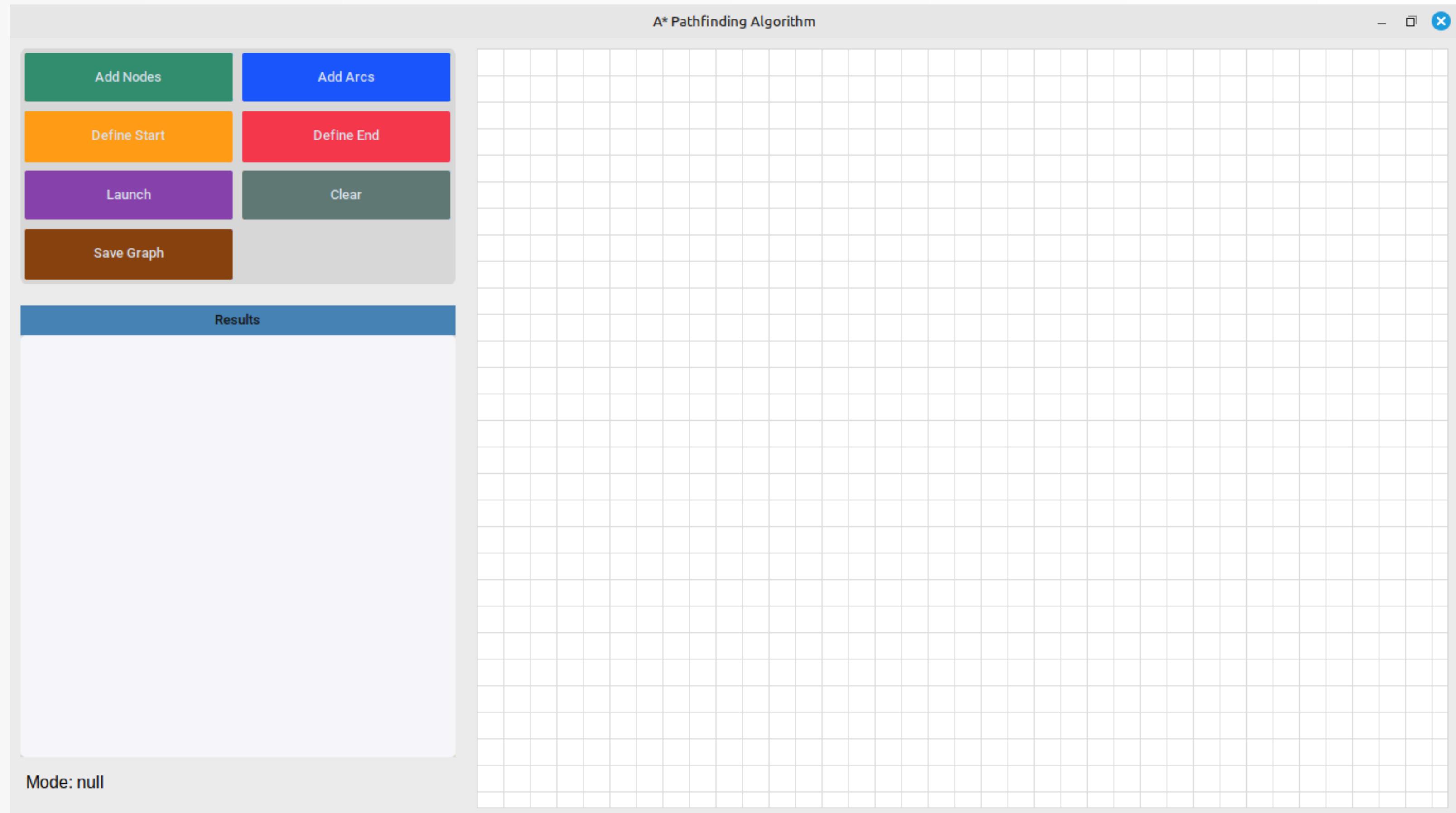


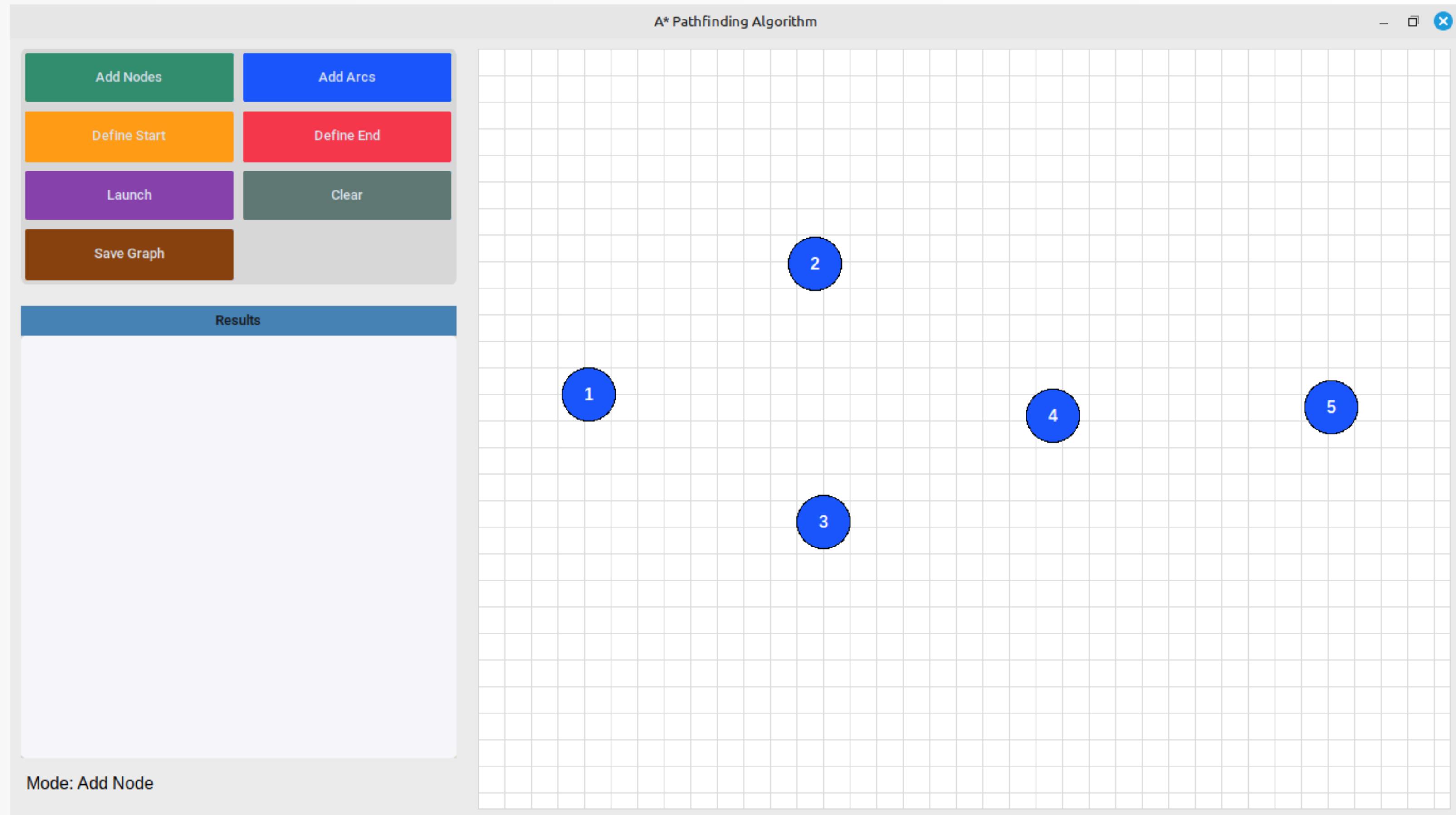
- Effacer complètement le graphe actuel pour préparer un nouveau départ



- Ce diagramme montre les classes, leurs attributs, leurs méthodes et leurs relations.







05 Résultats et démonstration de l'application

Sélection du mode d'ajout des arcs

A* Pathfinding Algorithm

Add Nodes Add Arcs

Define Start Define End

Launch Clear

Save Graph

Results

Mode: Add Arcs

Arc Weight

Enter weight != 0 between Node 4 and Node 5:
3

OK Cancel

The application interface for the A* Pathfinding Algorithm. On the left, there's a toolbar with buttons for 'Add Nodes' (green), 'Add Arcs' (blue), 'Define Start' (orange), 'Define End' (red), 'Launch' (purple), 'Clear' (dark green), and 'Save Graph' (brown). Below the toolbar is a blue bar labeled 'Results'. At the bottom left, it says 'Mode: Add Arcs'. The main area is a grid where a graph is being drawn. The graph consists of five nodes (1, 2, 3, 4, 5) and several directed edges. Edges and their weights are: (1, 2) weight 1, (1, 3) weight 4, (2, 1) weight 2, (2, 3) weight 5, (2, 4) weight 12, (3, 2) weight 2, (3, 4) weight 2, (4, 5) weight 3. A modal dialog titled 'Arc Weight' is open, asking 'Enter weight != 0 between Node 4 and Node 5:' with the value '3' entered. There are 'OK' and 'Cancel' buttons at the bottom of the dialog.

```
graph LR; 1((1)) -- 1 --> 2((2)); 1 -- 4 --> 3((3)); 2 -- 2 --> 1; 2 -- 5 --> 3; 2 -- 12 --> 4((4)); 3 -- 2 --> 4; 3 -- 2 --> 5((5)); 4 -- 3 --> 5;
```

A* Pathfinding Algorithm

Add Nodes Add Arcs

Define Start Define End

Launch Clear

Save Graph

Results

Mode: Define Start

The interface displays a graph with five nodes labeled 1 through 5. Node 1 is highlighted in green and serves as the start node. Node 5 is the goal node. The graph consists of the following arcs:

- From node 1 to node 2 (arc value 1)
- From node 1 to node 3 (arc value 4)
- From node 2 to node 1 (arc value 2)
- From node 2 to node 3 (arc value 2)
- From node 2 to node 4 (arc value 5)
- From node 2 to node 5 (arc value 12)
- From node 3 to node 1 (arc value 2)
- From node 3 to node 2 (arc value 2)
- From node 3 to node 4 (arc value 3)
- From node 4 to node 3 (arc value 2)
- From node 4 to node 5 (arc value 3)

A* Pathfinding Algorithm

The interface shows a graph with nodes labeled 1 through 5. Node 1 is green and serves as the start node. Node 5 is red and serves as the end node. There are several arcs connecting the nodes, each labeled with a value: (1, 2) = 1, (1, 3) = 4, (2, 1) = 1, (2, 3) = 2, (2, 4) = 5, (2, 5) = 12, (3, 2) = 2, (3, 4) = 3, and (4, 5) = 3. The A* algorithm has calculated f-values for each node: f(1) = 0, f(2) = 1, f(3) = 2, f(4) = 5, and f(5) = 3. The mode is currently set to "Define End".

Add Nodes Add Arcs

Define Start Define End

Launch Clear

Save Graph

Results

Mode: Define End

A* Pathfinding Algorithm

Add Nodes Add Arcs

Define Start Define End

Launch Clear

Save Graph

Results

Algorithm Results:

** Heuristic Table:
{'1': 0, '2': 0, '3': 0, '4': 0, '5': 0};

** Graph [Adjacency List]:
{'1': [('2', 1), ('3', 4)], '2': [('3', 2), ('4', 5), ('5', 12)], '3': [('4', 2)], '4': [('5', 3)]}

** Node Sequence:
1 → 2 → 3 → 4 → 5

** Total Path Cost From 1 to 5: 8

Mode: Launch

The diagram illustrates the execution of the A* pathfinding algorithm on a graph. The graph consists of five nodes (1, 2, 3, 4, 5) connected by arcs. Node 1 is green and serves as the start node. Node 5 is red and serves as the end node. The algorithm uses a heuristic table where all nodes have a value of 0. The current state of the search is shown with blue circles for open nodes and a red circle for the closed set. The open set contains nodes 1, 2, 3, and 4. The closed set contains node 5. Red arrows indicate the path from the start node 1 through nodes 2, 3, and 4 to the goal node 5. Small boxes labeled with numbers (1, 2, 3, 4, 5, 12) are placed along the red arrows, likely representing the f-values or specific cost values assigned by the heuristic function.

A* Pathfinding Algorithm

Add Nodes Add Arcs

Define Start Define End

Launch Clear

Save Graph

Results

Algorithm Results:

** Heuristic Table:
{'1': 0, '2': 0, '3': 0, '4': 0, '5': 0};

** Graph [Adjacency List]:
{'1': [('2', 1), ('3', 4)], '2': [('3', 2), ('4', 5), ('5', 12)],
: [('4', 2)], '4': [('5', 3)]}

** Node Sequence:
 $1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5$

** Total Path Cost From 1 to 5: 8

Mode: Save

Save As

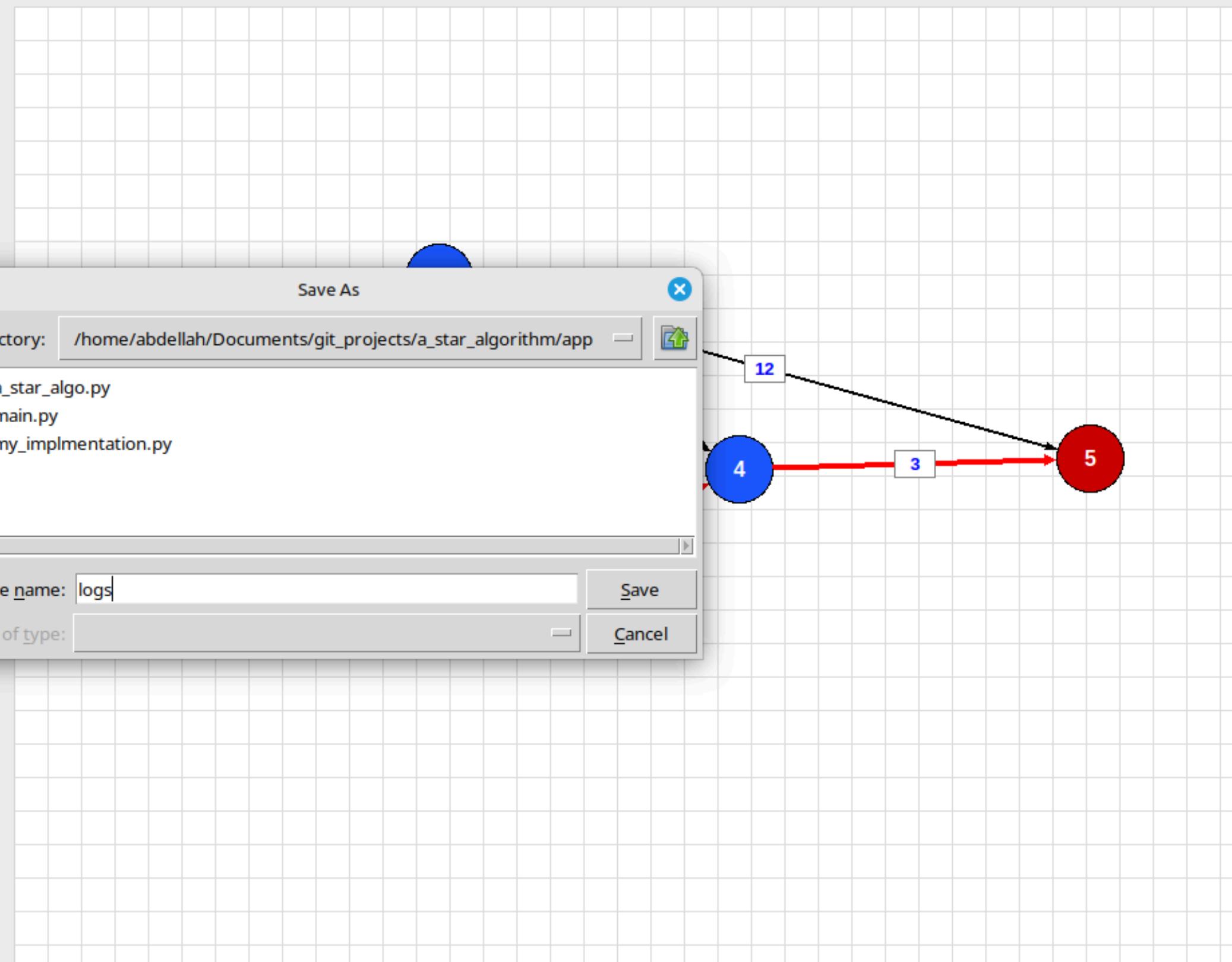
Directory: /home/abdelah/Documents/git_projects/a_star_algorithm/app

a_star_algo.py
main.py
my_implementation.py

File name: logs

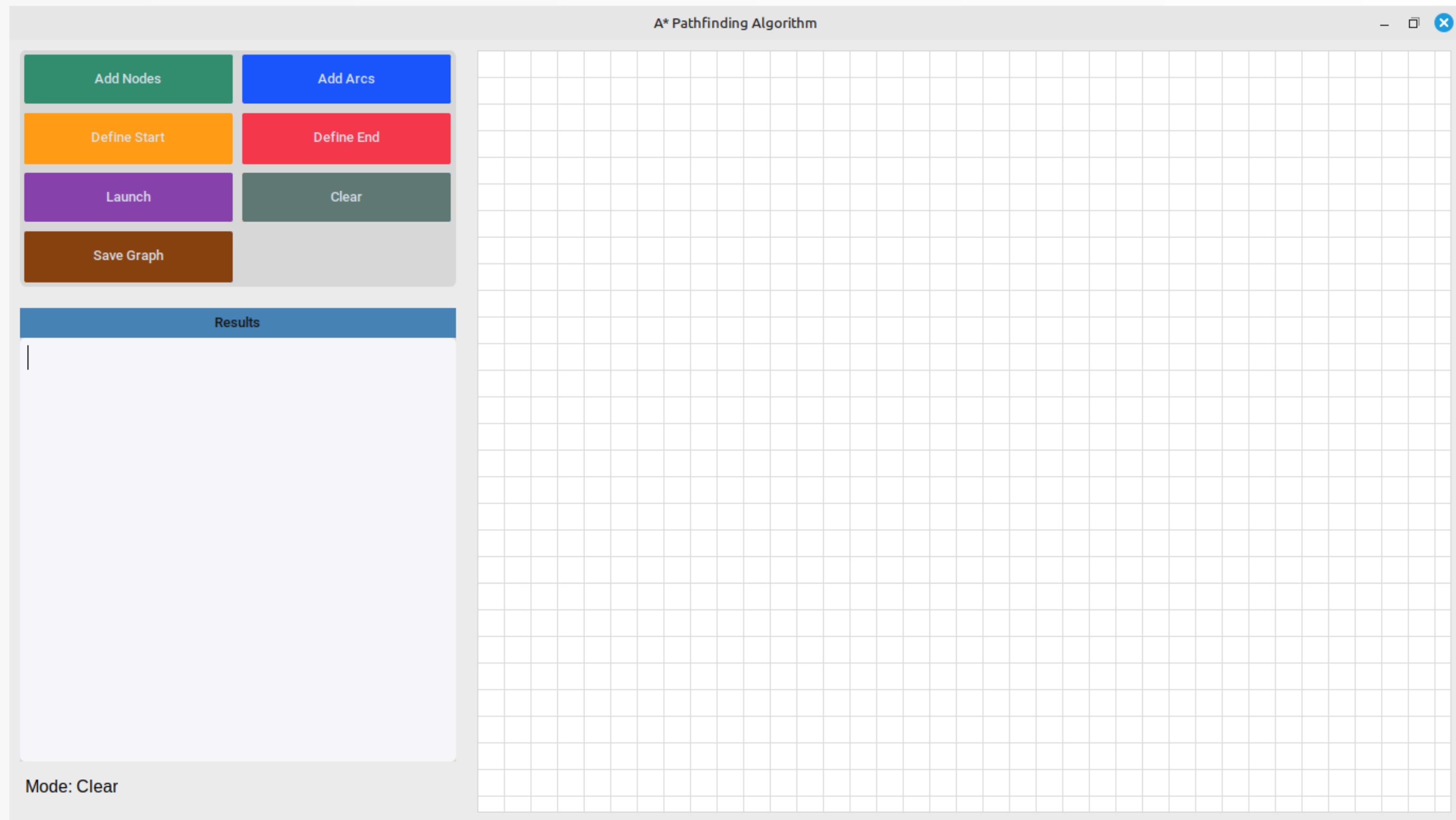
Save Cancel

Files of type:



05 Résultats et démonstration de l'application

Effacement du graphe



Merci Pour Votre Attention