Университет ИТМО
Факультет программной инженерии и компьютерной техники

Курс «Искусственный интеллект»

Лабораторная работа № 3
Вариант: 1

Работу выполнил
Студент группы P33101

Сыюань Хуан

Преподаватель
Игорь Александрович
Бессмертный

Санкт-Петербург
2021

Оглавление

Цель лабораторной работы

Исследовать алгоритмы решения задач методом поиска.

Описание предметной области

Имеется транспортная сеть, связывающая города СНГ. Сеть представлена в виде таблицы связей между городами. Связи являются двусторонними, т.е. допускают движение в обоих направлениях.
Необходимо проложить маршрут из одной заданной точки в другую.

Задание

Этап 1. Неинформированный поиск. На этом этапе известна только топология связей между городами. Выполнить:

1. поиск в ширину;
2. поиск глубину;
3. поиск с ограничением глубины;
4. поиск с итеративным углублением;
5. двунаправленный поиск.

Отобразить движение по дереву на его графе с указанием сложности каждого вида поиска. Сделать выводы.

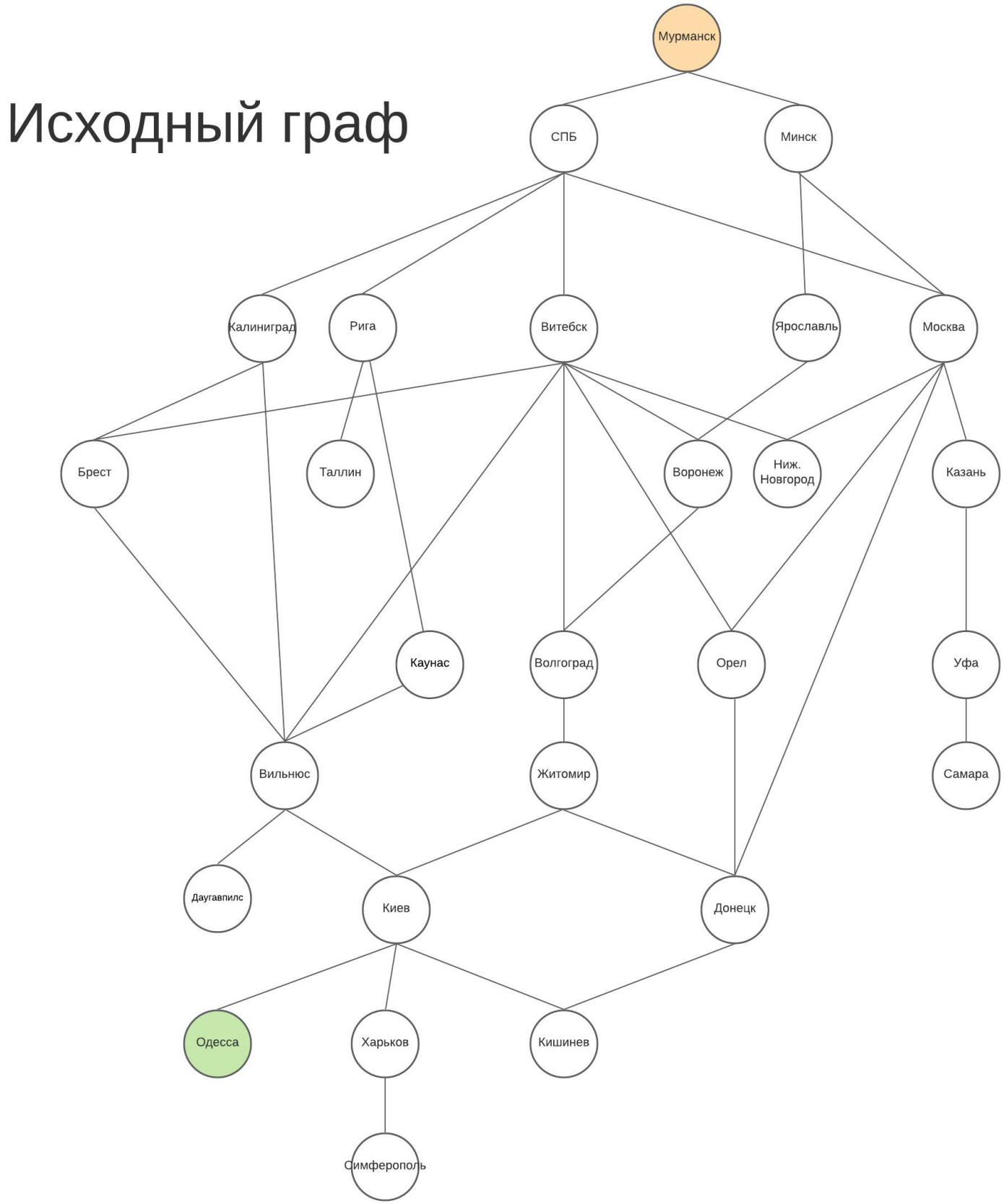
Этап 2. Информированный поиск. Воспользовавшись информацией о протяженности связей от текущего узла, выполнить:

1. жадный поиск по первому наилучшему соответствию;
2. затем, использую информацию о расстоянии до цели по прямой от каждого узла, выполнить поиск методом минимизации суммарной оценки A*.

Отобразить на графе выбранный маршрут и сравнить его сложность с неинформированным поиском. Сделать выводы.

Выполнение лабораторной работы

Исходный граф

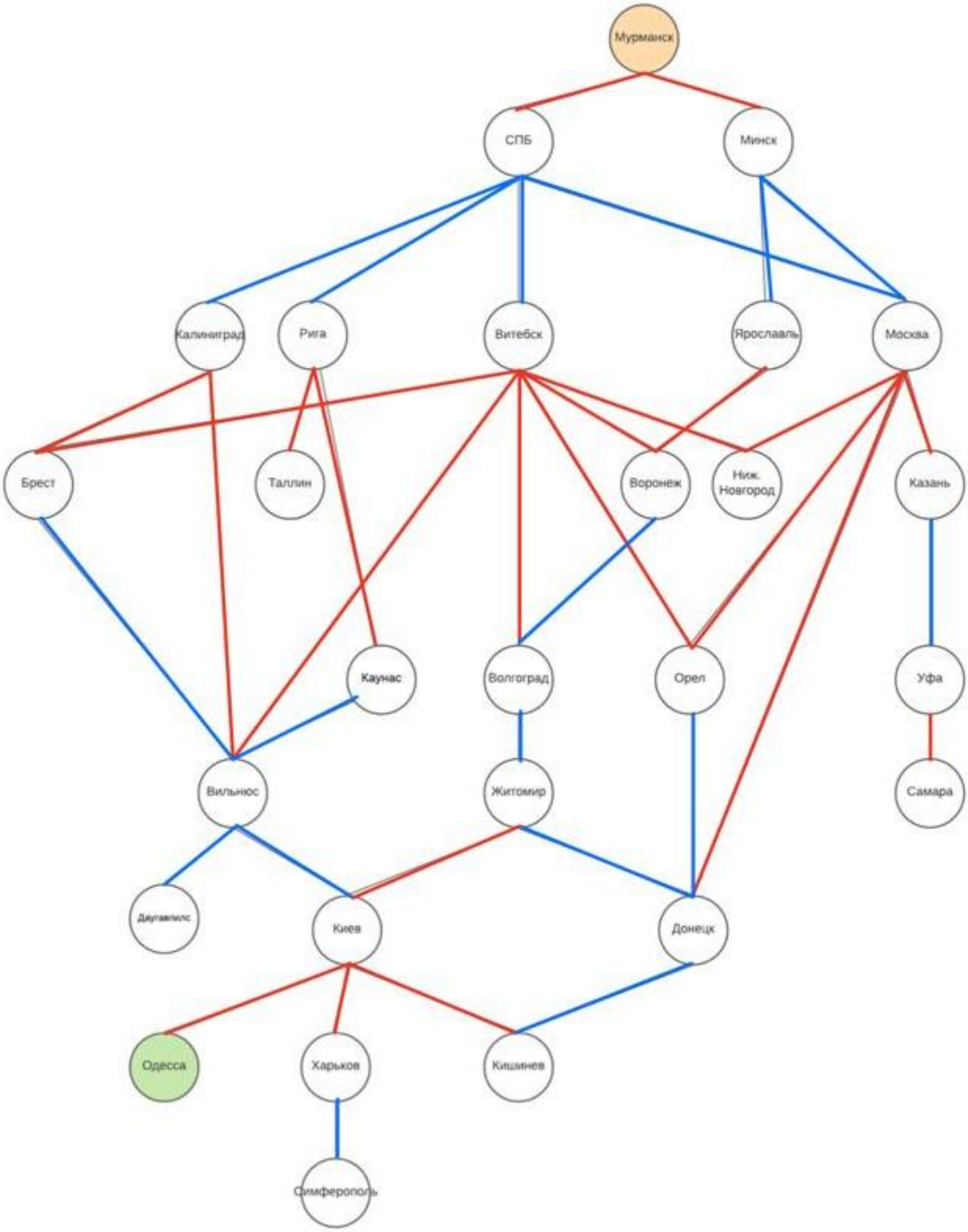


In accordance with variant 1, let it be searched between the cities of Murmansk and Odessa.

Этап 1. Слепой поиск

b - коэффициент ветвления
d - глубина самого поверхностного решения
m - максимальная глубина дерева;
e - предел глубины

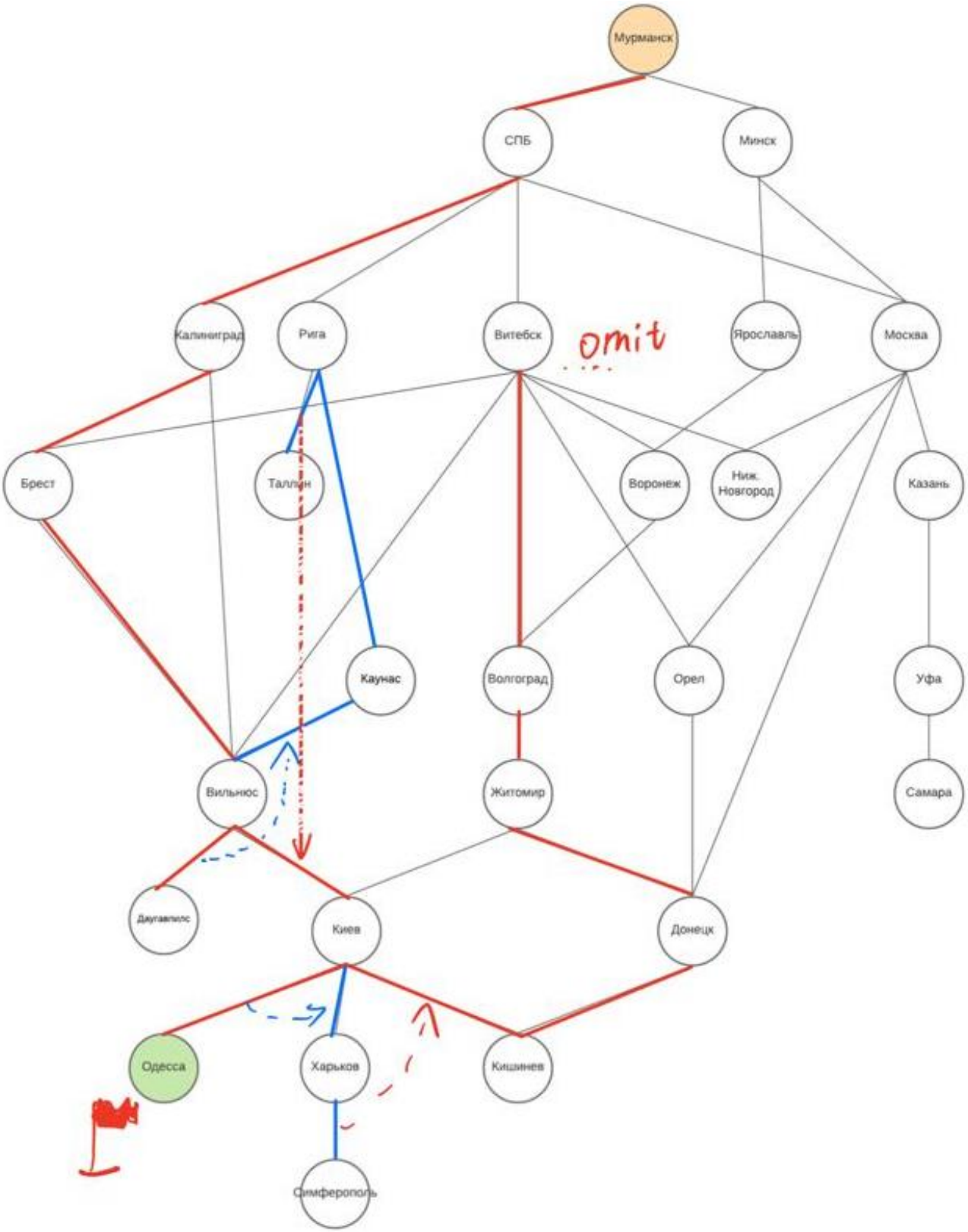
Поиск в ширину



Коэффициент ветвления возьмем максимальным для оценки верхней границы сложности. То есть b = 6.

The depth of the most superficial solution is d = 5.
Time complexity: $b^{d+1} = 6^{5+1} = 6^6$
Memory cost: $b^{d+1} = 6^6$

Поиск в глубину

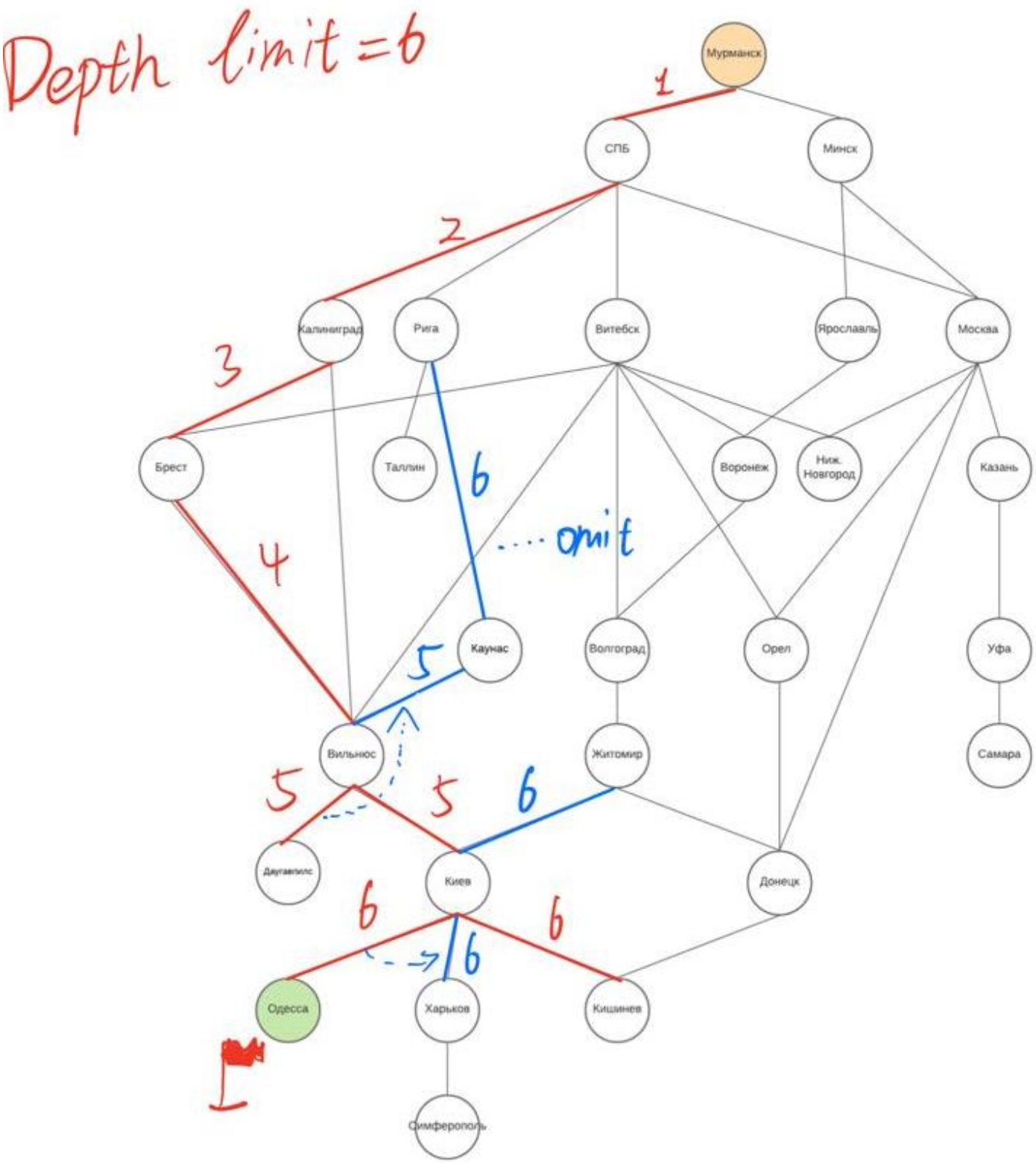


We take the branching factor as maximum to estimate the upper bound of complexity. That is, b = 6.
Maximum tree depth m = 15

Time complexity: $b^m = 6^{15}$

Memory cost: b * m = 6 * 15

Поиск с ограничением глубинны



Коэффициент ветвления возьмем максимальным для оценки верхней границы сложности. То есть b = 6.

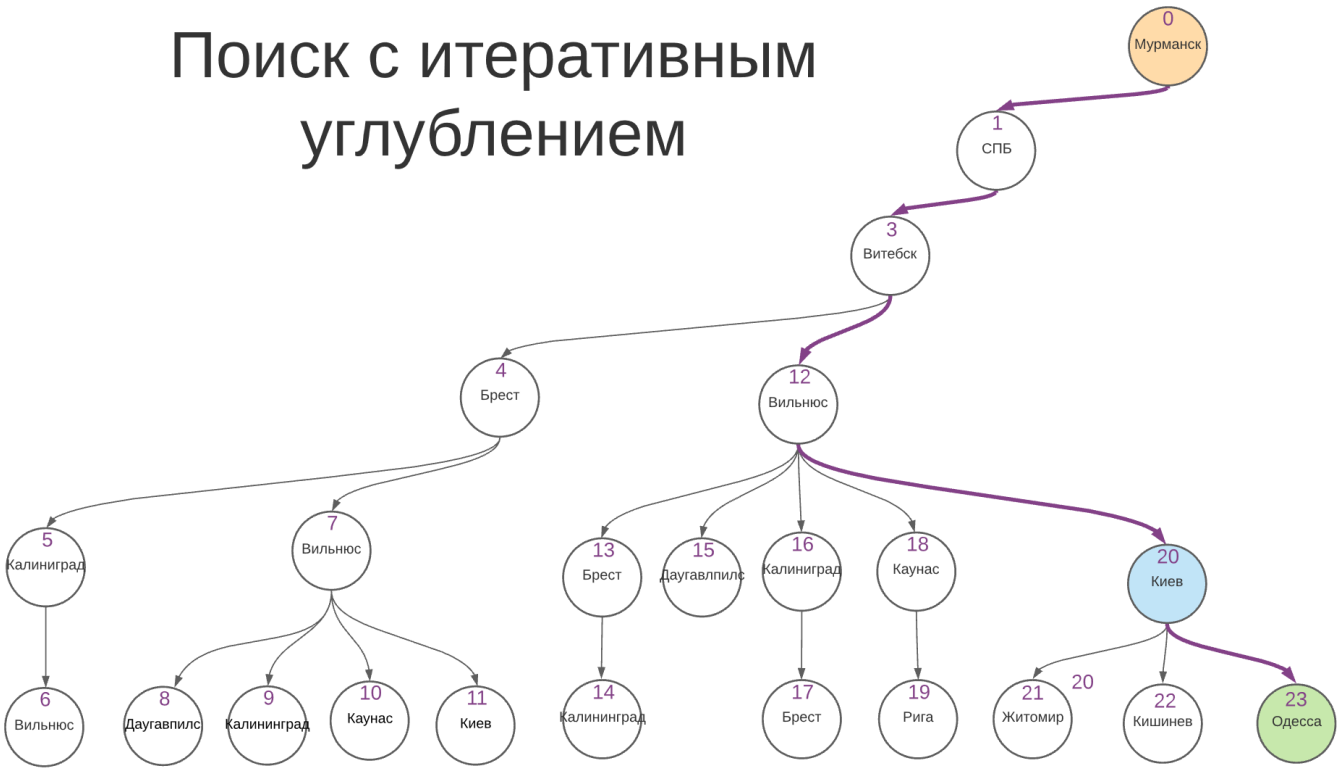
Depth limit e = 6

Time complexity: $b^e = 6^6$

Memory cost: b * e = 6 * 6

If the depth limit is 5, we still can get the “Одесса”.

Поиск с итеративным углублением



Коэффициент ветвления возьмем максимальным для оценки верхней границы сложности. То есть b = 6.

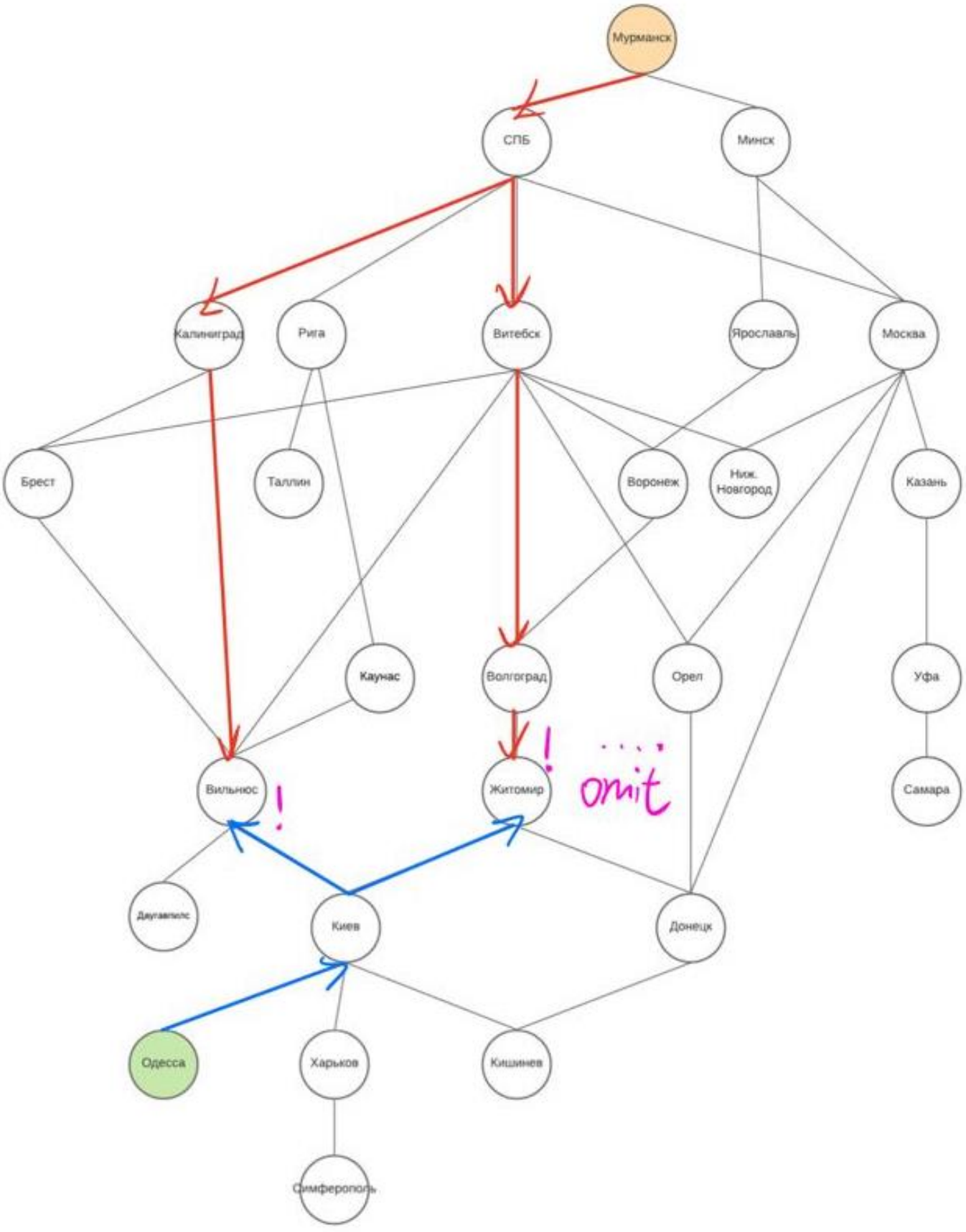
The depth of the most superficial solution: d = 5

Time complexity: $b^d = 6^5$

Memory cost: b * d = 6 * 5

It is very similar to the previous algorithm and combines the advantages of the BFS

Двунаправленный поиск



Коэффициент ветвления возьмем максимальным для оценки верхней границы сложности. То есть b = 6.

The depth of the most superficial solution: d = 5

Time: $b^{d/2} = 6^{5/2}$

Cost memory: $b^{d/2} = 6^{5/2}$

Вывод

BFS:

advantages:

If there is any solution, BFS will provide the solution. If there are multiple solutions to a given problem, BFS will provide the smallest solution that requires the fewest steps.

disadvantages:

It requires a lot of memory because each level of the tree must be saved into memory to expand the next level. If the solution is far from the root node, BFS takes a lot of time.

DFS:

advantage:

DFS requires a little memory because it only needs to store a bunch of nodes on the path from the root node to the current node. It takes less time to reach the target node than the BFS algorithm (if it moves in the correct path).

disadvantages:

Many conditions may continue to occur, and there is no guarantee that a solution will be found. The DFS algorithm is used for in-depth search, and sometimes it may enter an infinite loop.

Depth-limited search:

advantage:

- The depth limit search is memory efficient.

disadvantages:

- Depth-limited search also has the disadvantage of incompleteness.
- If there are multiple solutions to the problem, it may not be the best choice.

Iterative deepening depth first search:

advantage:

It combines the advantages of BFS and DFS search algorithms in terms of fast search and memory efficiency.

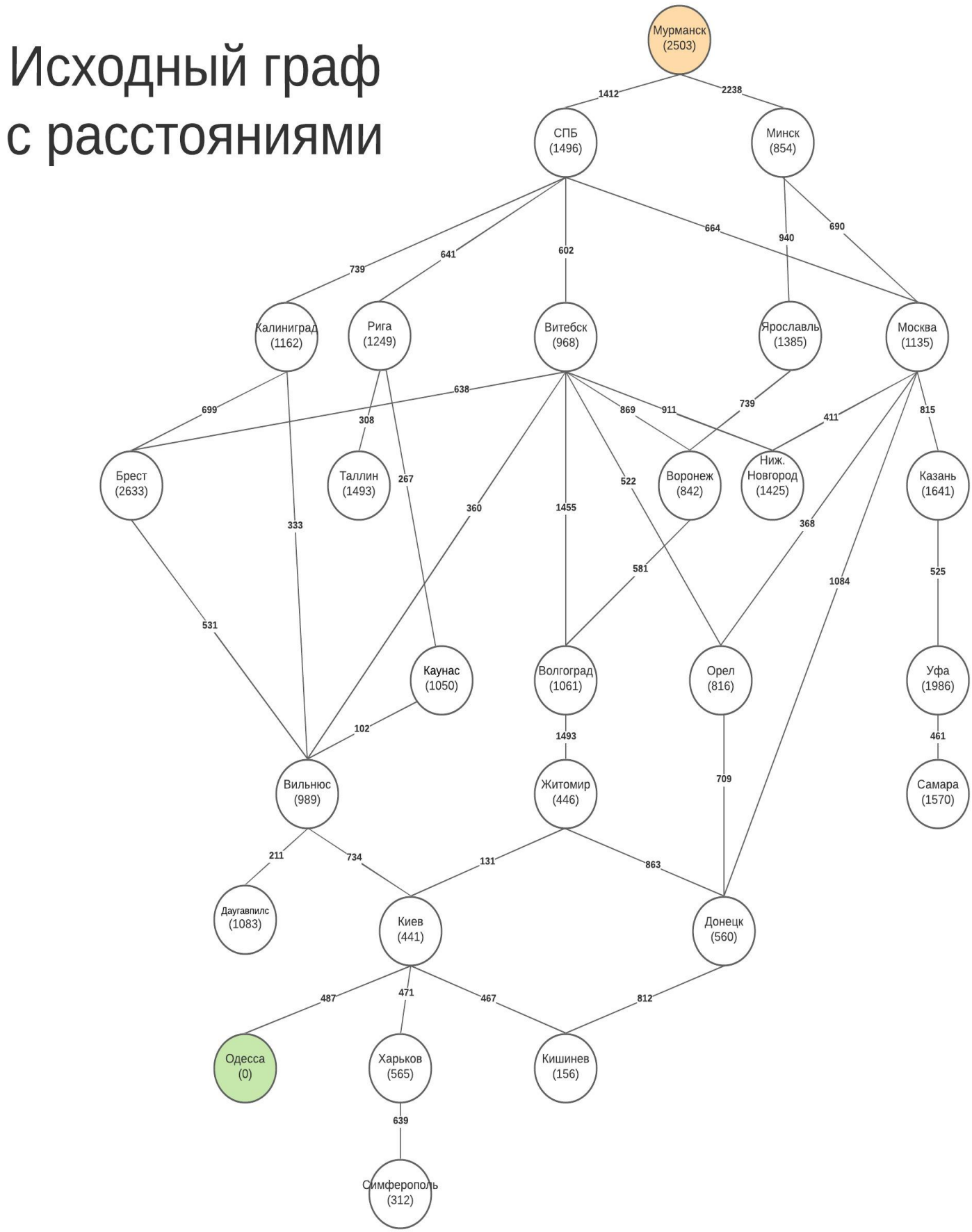
disadvantages:

The main disadvantage is that it repeats all the work of the previous phase. So only if we have no any idea about the depth, we use IDDFS.

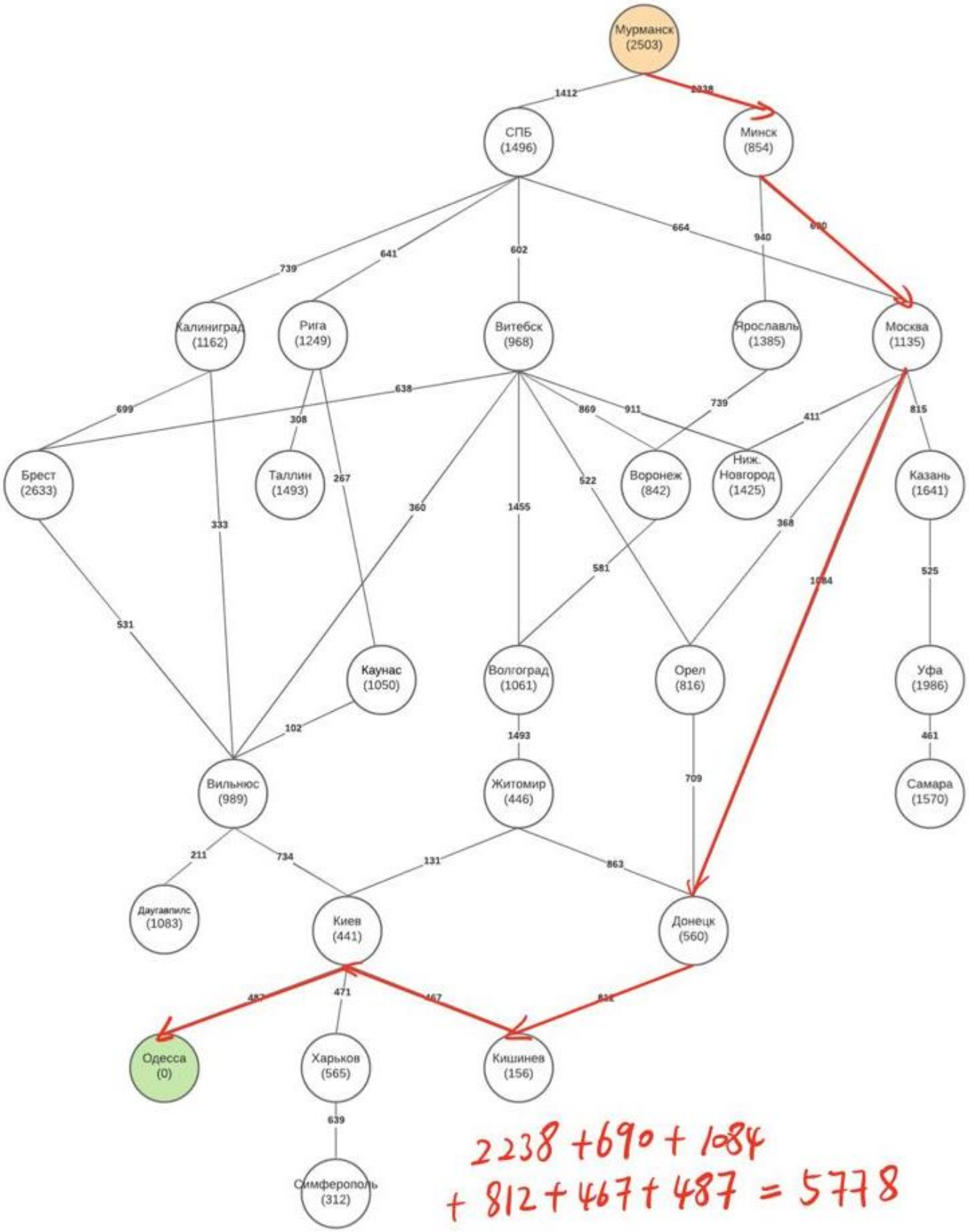
Bidirectional Search(Двунаправленный поиск):
advantage:
Two-way search is fast. Two-way search requires less memory
disadvantages:
The realization of the bidirectional search tree is difficult. In a two-way search, the target state should be known at the first. That’s the hardest point.

Этап 2. Информированный поиск

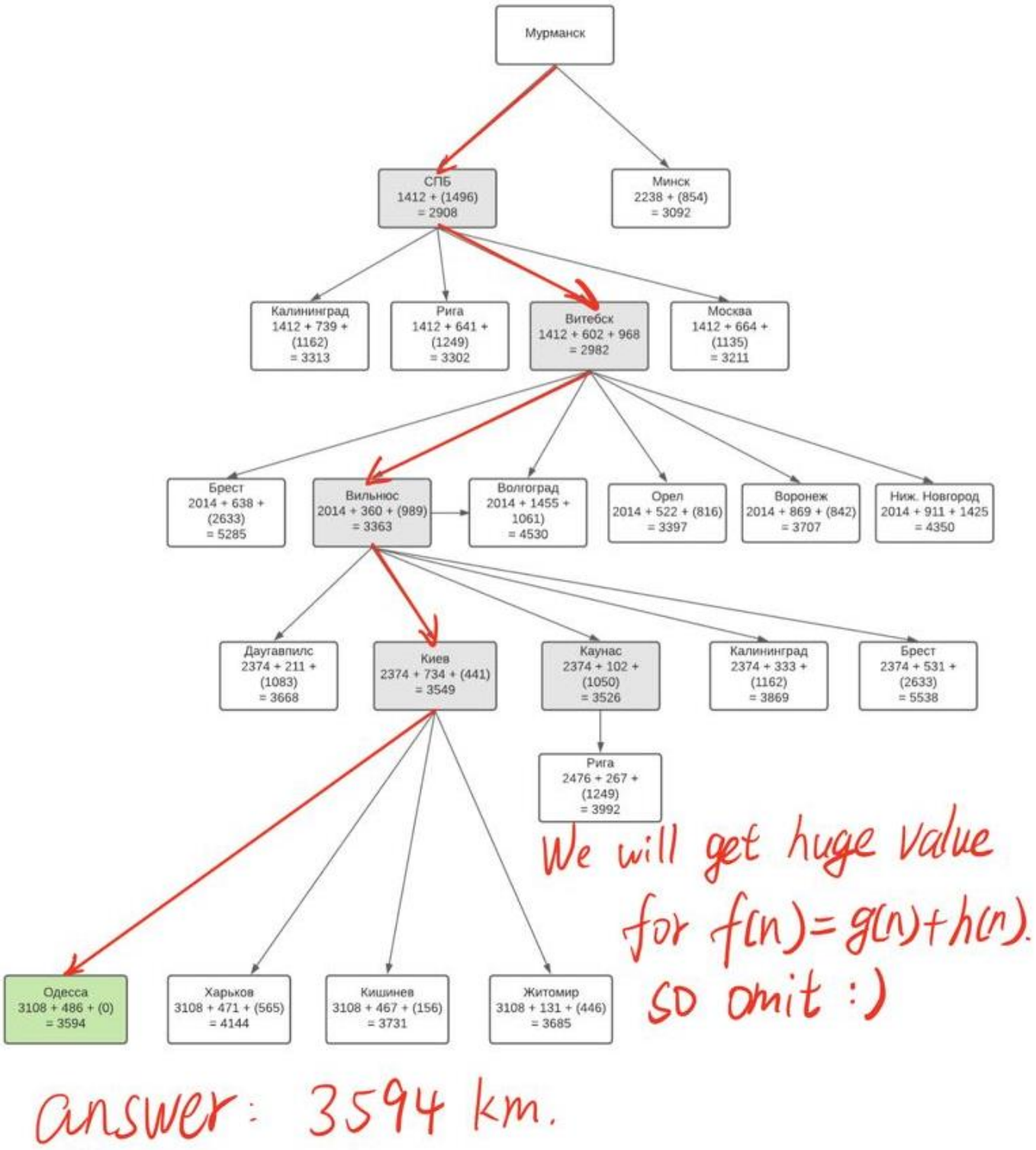
Исходный граф с расстояниями



Жадный поиск



Поиск методом минимизации суммарной оценки A*



Вывод

Greedy Search

advantages:

Greedy search can switch between BFS and DFS by obtaining the advantages of the two algorithms. This algorithm is more effective than the BFS and DFS algorithms.

disadvantages:

In the worst case, it can behave as an unguided depth-first search. It can fall into a loop like DFS. The algorithm is not optimal.

A* Search
advantage:
A * search algorithm is a better algorithm than other search algorithms. A * The search algorithm is optimal and complete. This algorithm can solve very complex problems.
disadvantages:
It does not always produce the shortest path because it is mainly based on heuristics and approximations. A * The search algorithm has some complexity issues. The main disadvantage of A* is the memory requirement, because it stores all generated nodes in memory, so it is impractical for various large-scale problems. l