Алгоритмы поиска на не взвешенных графах на примере поиска путей в лабиринтах

Виногродский Серафим

26 мая 2022 г.

Содержание

1 Введение

...

2 Описание реализованных алгоритмов поиска

2.1 Поиск в глубину (Depth-first search, DFS)

2.1.1 Описание алгоритма

Поиск в глубину, как и поиск в ширину, является одним из базовых алгоритмов поиска на графах, в основе которого лежит использование стека для определения очерёдности посещения элементов графа. Для его реализации на C++ требуется 3 вспомогательные структуры:

(1) множество посещённых элементов графа;

(2) стек, определяющий очерёдность посещения;

(3) словарь родителей.

```
auto visited      = set<cell_t>({_start_pos});    // (1)
auto search_stack = stack<cell_t>({_start_pos}); // (2)
auto parents      = map<cell_t, cell_t>();        // (3)
```

В общем случае структура parents является необязательной, поскольку она используется только для восстановления пути до найденного элемента, что не всегда необходимо. Поскольку, в данной работе алгоритм DFS используется именно для поиска пути, а не для нахождения элемента, избавиться от структуры parents невозможно.

Далее, пока стек search_stack не пуст, из него изымается верхний элемент next. Если next подходит под условия поиска, то из словаря родителей строится путь до найденного элемента, после чего этот путь возвращается как результат поиска:

```
while (!search_stack.empty()) {
    cell_t next = search_stack.top();
    if (next == _dest_pos) {
        return build_path(parents, _start_pos, _dest_pos);
    }
    searh_stack.pop();

    // <...>
}
```

В ином случае, то есть если искомая вершина ещё не достигнута, все до этого не посещённые дети вершины next поочерёдно добавляются в стек search_stack, после чего поиск продолжается уже для следующей вершины (если таковая имеется):

```
while (!search_stack.empty()) {
    // <...>

    for (cell_t child : neighbors(next)) {
        if (!visited.contains(child)) {
            parents.insert({child, next});
            visited.insert(child);
            search_stack.push(child);
        }
    }
}
```

Следствием использования стека для определения очерёдности посещения является то, что дети только что посещённых узлов обрабатываются раньше детей узлов, посещённых до этого. Таким образом алгоритм DFS как бы старается забраться как можно глубже в структуру дерева (что вполне соответствует названию алгоритма), вместо того, что бы последовательно обрабатывать все узлы все большей и большей глубины, как это делает алгоритм BSF, который будет описан далее.

2.1.2 Оценка сложности

...

2.2 Поиск в ширину (Breadth-first search)

...

2.3 Поиск в ширину с двух сторон

...

3 Сравнение эффективности

...

4 Выводы

...