

Алгоритм A^*

Что это такое

- Алгоритм **A*** (англ. *A star*) — алгоритм поиска, который находит во взвешенном графе маршрут наименьшей стоимости от начальной вершины до выбранной конечной.
- Исследует граф путём просмотра путей через наиболее перспективные узлы, выбираемые в соответствии с указанным правилом.

Преимущества относительно других алгоритмов поиска минимального пути

- обходит меньшее количество вершин, благодаря тому, что он работает с «*оптимистичной*» оценкой пути через вершину
- A* проходит наименьшее количество вершин графа среди допустимых алгоритмов, использующих такую же точную (или менее точную) эвристику

Использование

- при разработке игр, когда нужно находить кратчайшие пути между заданными точками
- при построении пути навигатором
- и многое другое

Описание алгоритма

- Для выбора очередной раскрываемой вершины используется эвристическая функция:

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

- $g(x)$ – наименьшая стоимость до x из стартовой вершины
- $h(x)$ – эвристическая оценка стоимости пути из x до конечной вершины

Выбор эвристической функции

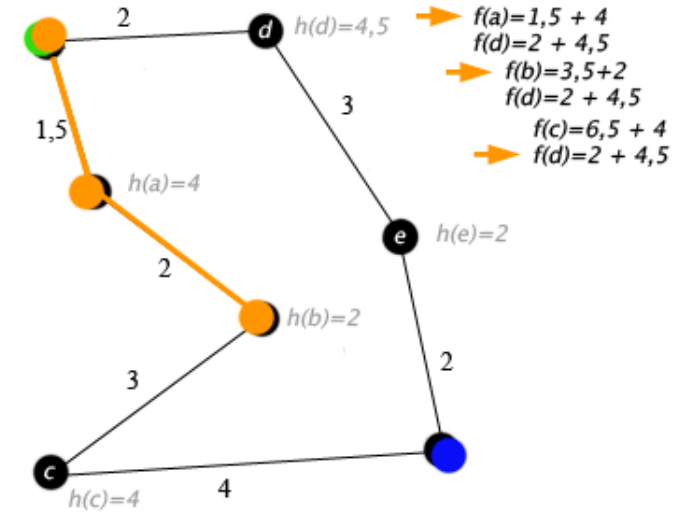
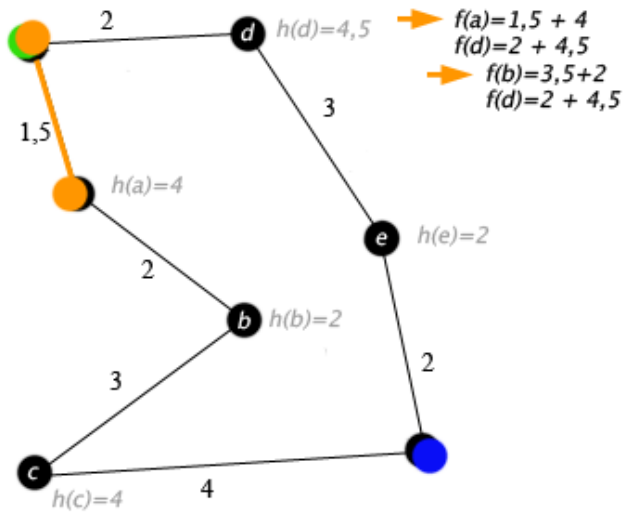
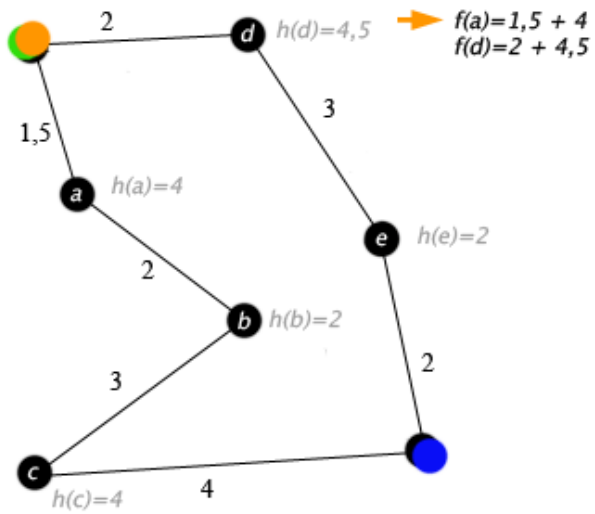
- $h(x)$ должна быть **допустима**: для любой вершины x $h(x)$ не больше веса кратчайшего пути от x до конечной вершины.
- $h(x)$ должна быть **монотонна**: если y – потомок x , то верно
$$f(x) \leq f(y)$$
- Примеры эвристических функций:
 - 1) манхэттенское расстояние
 - 2) расстояние Чебышева
 - 3) евклидово расстояние по прямой

Основные шаги

- 1) Добавляем стартовую вершину s в множество рассматриваемых вершин U .
Полагаем $g(s) = 0, f(s) = h(s)$.
- 2) Пока U не пусто, выбираем из U вершину x с $\min f(x)$, добавляем ее в множество рассмотренных вершин Q , просматриваем ее соседей, при этом:
 - Если сосед x_0 ранее просматривался и $g(x_0) \leq g(x) + d(x, x_0)$, то переходим к следующему
 - Если сосед x_0 не просматривался или значение $g(x_0) \geq g(x) + d(x, x_0)$, то меняем предка x_0 на x и пересчитываем значения $g(x_0), f(x_0)$. Если x_0 не принадлежит U , то добавляем ее в U .

Детали реализации

- Используется очередь с приоритетом (приоритет на основе эвристической функции)
- Для каждой вершины хранится ссылка на родителя
- Эвристическая функция должна быть порядка $O(1)$
- В зависимости от реализации очереди, может производиться обход в глубину (если реализуется LIFO) или в ширину (FIFO)



Пример работы

https://upload.wikimedia.org/wikipedia/commons/5/5d/Astar_progress_animation.gif

Псевдокод

- Q — множество вершин, которые требуется рассмотреть,
- U — множество рассмотренных вершин,
- $f[x]$ — значение эвристической функции для вершины x ,
- $g[x]$ — стоимость пути от начальной вершины до x ,
- $h(x)$ — эвристическая оценка расстояния от вершины x до конечной вершины

```
bool A*(start, goal):
    U = ∅
    Q = ∅
    Q.push(start)
    g[start] = 0
    f[start] = g[start] + h(start)
    while Q.size() != 0
        current = вершина из Q с минимальным значением f
        if current == goal
            return true
        Q.remove(current)
        U.push(current)
        for v : смежные с current вершины
            tentativeScore = g[current] + d(current, v)
            if v ∈ U and tentativeScore >= g[v]
                continue
            if v ∉ U or tentativeScore < g[v]
                parent[v] = current
                g[v] = tentativeScore
                f[v] = g[v] + h(v)
                if v ∉ Q
                    Q.push(v)
    return false
```