АЛГОРИТМЫ ПОИСКА ПУТИ В ДИСКРЕТНОМ ПРОСТРАНСТВЕ

Домрачева Татьяна Сергеевна

студент, НГПУ им. Козьмы Минина, РФ, г. Нижний Новгород

Орловская Людмила Александровна

студент, НГПУ им. Козьмы Минина, РФ, г. Нижний Новгород

Романова Наталья Анатольевна

студент, НГПУ им. Козьмы Минина, РФ, г. Нижний Новгород

Шиганова Марина Викторовна

студент, НГПУ им. Козьмы Минина, РФ, г. Нижний Новгород

Гусев Игорь Владимирович

студент, НГПУ им. Козьмы Минина, РФ, г. Нижний Новгород

Гусев Вадим Владимирович

студент, НГПУ им. Козьмы Минина, РФ, г. Нижний Новгород

Поиск пути - это поиск компьютерным приложением кратчайшего маршрута между двумя точками. Эта область исследований в значительной степени основана на алгоритме Дейкстры для нахождения кратчайшего пути на взвешенном графе.

Большое применение технологии поиска пути получили в разного рода цифровых играх (компьютерные игры, игры для приставок, сматрфонов и прочие).

Оптимальный (быстрый и логичный в зависимости от необходимой длины) поиск пути возможен только на дискретном пространстве, то есть на том пространстве, которое было поделено специально для поиска пути. Итог дискретизации пространства – граф.

В математике (точнее в теории графов), граф представляет собой структуру, составляющую набор объектов, в которых некоторые пары объектов в некотором смысле «связаны». Объекты соответствуют математическим абстракциям, называемым вершинами (а также узлами или точками), и каждая из связанных пар вершин называется ребром (линиями).

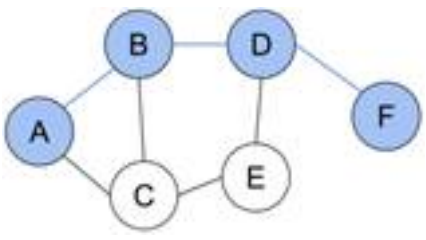

Рисунок 1. граф состоящий из вершин (круги с буквами внутри) и ребер (пары вершин связанные линией)

Возможные типы дискретизации пространства:

· Navigation regular grid (навигационная регулярная сетка)

· Waypoints (путевые точки)

· NavMesh (навигационная сетка)

· другие.

По сути, метод поиска пути ищет путь, начиная с одной вершины (стартовой) и исследуя соседние узлы до достижения целевого узла (конечно вершин), как правило, с целью поиска самого дешевого маршрута.

Две основные задачи поиска пути:

1. поиск пути между двумя узлами в графе;

2. кратчайший путь - найти оптимальный кратчайший путь.

Основные алгоритмы, такие как поиск по ширине и глубине, направлены на первую проблему, исчерпывая все возможности. Начиная с данного узла, они перебирают все потенциальные пути до тех пор, пока не достигнут узла назначения. Очевидно, эти методы не подходят для real-time приложений в виду малой скорости, и не подходят играм (в большинстве случаев) так как основная задача большинства – поиск кратчайшего пути.

Большая часть разновидностей модицификации алгоритмов A* (A star) и Дейкстры, в основном ищут пути посредством эвристики. Эти алгоритмы являются одними из лучших общих алгоритмов, которые ищут путь на графе без предварительной обработки.

Поиск пути алгоритмом Дейкстры. Общим примером алгоритма поиска путей на основе графа является алгоритм Дейкстры. Этот алгоритм начинается с начального узла и «открытого набора» узлов-кандидатов (соседей начального узла). На каждом шаге рассматривается узел в открытом наборе с самым низким расстоянием от начала. Рассматриваемый узел помечается как «закрытый», и все узлы, прилегающие к нему, добавляются в открытый набор, если они еще не были проверены. Этот процесс повторяется до тех пор, пока не будет найден путь к месту назначения. Поскольку узлы с наименьшим расстоянием проверяются первым, при первом обнаружении конечного узла, путь к нему будет самым коротким.

A* - модификация алгоритма Дейкстры, обычно используемого в играх. A* присваивает вес каждому открытому узлу, равному весу края, этого узла плюс приблизительное расстояние между этим узлом и концом. Это приблизительное расстояние определяется эвристикой и представляет собой минимально возможное расстояние между этим узлом и концом. Это позволяет устранить более длинные пути после обнаружения исходного пути. Если существует путь длины x между началом и концом, а минимальное расстояние между узлом и финишем больше, чем x, этот узел не нужно проверять.

Способом определения того, какую же вершину (из возможных соседей анализируемой вершины) использовать, является следующие выражение: F=G + H

Где: G - стоимость передвижения из стартовой вершины к данной, следуя найденному пути к этой клетке.

H - примерная стоимость передвижения от данной вершины до конечной. Это есть

эвристическая оценка пути. Стоимость H может быть вычислена множеством способов. В играх часто используется метод Манхеттена (Manhattan method), где считается общее количество вершин, необходимых для достижения целевой вершины от текущей, по горизонтали и вертикали, игнорируя любые препятствия, которые могут оказаться на пути. И умножается на некоторый коэффициент..

A* использует эту эвристику для улучшения поведения по сравнению с алгоритмом Дейкстры. Когда эвристика оценивается до нуля, A* эквивалентно алгоритму Дейкстры. По мере того как эвристическая оценка увеличивается и приближается к истинному расстоянию, A* продолжает находить оптимальные пути, но работает быстрее (в силу меньшего количественного анализа числа узлов). Когда значение эвристики точно соответствует истинному расстоянию, A* исследует наименьшее количество узлов. По мере увеличения значения эвристики A* исследует меньшее количество узлов, но больше не гарантирует оптимальный путь.

Алгоритм трассировки волн (волновой алгоритм) - алгоритм поиска кратчайшего пути на графе. Принадлежит к алгоритмам, основанным на методах поиска в ширине.

Состоит из двух этапов:

· распространение волны

· восстановление пути.

Распространение волны начинается от стартовой вершины в соседнюю, при этом проверяется, не принадлежит ли она ранее меченной в пути вершине.

Соседние вершины принято классифицировать двумя способами:

1. в окрестности фон Неймана соседними ячейками считаются только 4 ячейки по вертикали и горизонтали,

2. в окрестности Мура — все 8 ячеек, включая диагональные.

Если вершина не принадлжит к ранее помеченным в пути вершинам, то в ее атрибут записывается число, равное количеству шагов от стартовой вершины. Каждая такая вершина при следующей итерации распространяет свою волну.

Сборка кратчайшего пути происходит в обратном направлении: при выборе вершины, от финишной к стартовой, на каждом шаге выбирается вершина, имеющая атрибут расстояния от стартовой на единицу меньше текущей.

В целом, каждый из перечисленных алгоритмов находит свое применение, то есть в различных ситуациях определенный алгоритм. К примеру волновой хорошо подходит для критических игровых ситуаций когда требуется найти кратчайший путь до чего-то ближайшего для укрытия, а алгоритм Дейкстры используется наравне с A* (примерно), на некоторых форумах есть даже споры что лучше, ибо эвристика не всегда дает верные результаты.

Список литературы:

1. http://wikipedia.org/

2. http://www.gamedev.ru/

3. http://www.gamedev.net/

4. http://www.gamasutra.com

5. http://www.moddb.com

6. http://www.policyalmanac.org/

7. http://pmg.org.ru/