004.942

МОДУЛЬ ПОИСКА УНИВЕРСАЛЬНОГО МАРШРУТА В УСЛОВИЯХ НЕОПРЕДЕЛЕННОСТИ

Д.М. Белобородов
Московский Государственный Технологический Университет «СТАНКИН»
Россия, 127055, Москва, Вадковский пер., 1
E-mail: schumaroz@gmail.com

А.И. Разумовский
Институт проблем управления им. В.А. Трапезникова РАН
Россия, 117997, Москва, Профсоюзная ул., 65
E-mail: razumowsky@yandex.ru

Ключевые слова: алгоритм, поисковая оптимизация, маршрут.

Аннотация: Планирование, поиск и адаптация универсального маршрута является актуальной задачей для автономных и полуавтономных транспортных средств. В построении маршрута на данный момент существует две основные проблемы: необходимость учета карт и особенностей местности для определенных логистических единиц, а также сложность перестроения маршрута при возникновении нештатных ситуаций или в условиях неопределенности. Указанные проблемы нельзя решить численно, при помощи таких алгоритмов как A* или алгоритм Дейкстры. Настоящее исследование представляет возможный подход к решению обозначенных проблем, используя алгоритм муравьиной колонии. Алгоритм муравьиной колонии реализован симуляции, максимально приближенной к реальным условиям среде, построенной с целью симуляции перемещения по маршруту в реальном времени.

1. Введение

Построение маршрутов всегда являлось важной частью мира. Поиск оптимального маршрута необходим для эффективного расходования ресурсов [1, 2], соблюдения планов и сроков [3- 5], составления расписаний [6] и прогнозирования [7]. В природной среде маршруты строятся постоянно. Например, насекомые и другие животные искусно находят пути к водоему, источнику пищи, а также возвращаются в удобное и безопасное место для отдыха. Пчелы и шмели строят свои маршруты по предпочтениям, редко их меняя [8]. Птицы ищут пути, реагируя на климатические факторы [9]. Экологические исследования предоставляют уникальные знания о коммуникации индивидов и сообществ [10]. Для людей оптимальные маршруты (с минимальными временными и экономическими затратами) так же актуальны, как и для животных. Они рассчитываются на основе специальных алгоритмов, в которых учитываются особенности местности [11-13]. Однако из-за непредвиденных обстоятельств следование расчетным маршрутом оказывается невозможным, и требуется его перестроение. Подобная коррекция либо занимает много времени, либо вовсе не осуществима. Для предупреждения этого используют алгоритмы, имитирующие поведение животных в поиске пути, в частности.

На тему точного построения маршрута существует множество исследований. Это связано с большим количеством «классических» математических алгоритмов, которые

могут быть применены для поиска пути, например, алгоритм Дейкстры [14], A*, а также алгоритм Флойда-Уоршелла [15]. Алгоритм оптимизации муравьиной колонии относится к другому виду алгоритмов – эволюционным или генетическим алгоритмам. Такие алгоритмы используют множество экземпляров объекта для исследования, в данном случае, окружения и нахождения маршрута до точки. Как правило, они основываются на различных природных явлениях. Например, коллективном поведении ястребов Харриса [16], колонии пчел [17] или распределении плесени [18].

Изначально алгоритм муравьиной колонии применялся для поиска оптимального пути на графе [19] и до сих пор находятся способы его улучшения. Эта статья предлагает реализацию алгоритма оптимизации муравьиной колонии для построения универсального маршрута на карте в 2D-пространстве.

Такая реализация может быть очень полезна для моделирования маршрутов каких-либо перевозок в различные места, поскольку она позволяет определить маршрут для любой логистической единицы и быстро изменить этот маршрут при необходимости.

2. Алгоритм оптимизации муравьиной колонии

Основой алгоритма муравьиной колонии является поведение муравьев при поиске еды для колонии. Для графа пути алгоритм представляет собой итерационное маркирование вершин с помощью «феромона» большим количеством экземпляров так называемых «муравьев» – объектов, каждый из которых имеет свои параметры и направления движения. Изначально это направление определяется вероятностным способом, на основании формулы (1):

$$P_i = \frac{l_i^q \cdot f_i^p}{\sum_{k=0}^{N} l_k^q \cdot f_k^p}, \quad (1)$$

где P_i – вероятность перехода по пути i, l_i – величина, обратная весу (длине) i-го перехода, f_i – количество феромона на i-м переходе, q – величина, определяющая влияние веса i-го перехода, p – величина, определяющая влияние феромонов i-го перехода.

Решение, полученное при использовании такого алгоритма при первой итерации, может быть одним из худших, однако, итерационное применение алгоритма обычно даёт результат, близкий к оптимальному.

Оригинальный алгоритм муравьиной колонии, предназначенный только для графов, сильно ограничен в своем использовании. Он показывает общий принцип работы, но не может быть применен на практике в чистом виде. Для решения этой проблемы, в настоящем докладе представлен пример адаптации идеи алгоритма муравьиной колонии для решения практической задачи нахождения маршрутов при изменяющихся условиях.

3. Реализация алгоритма поиска маршрута в условиях неопределенности

Для решения проблемы, обозначенной выше, алгоритма муравьиной колонии, предназначенного для графа, недостаточно. Из оригинального алгоритма был взят основной принцип построения маршрута на основе феромонов, однако функционал алгоритма существенно перестроен.

3.1. Движение

В рамках более расширенного пространства, где каждая точка представляет собой возможное положение перемещающейся транспортной единицы (далее – агента), движение агента определяется множеством параметров, которые симулируют приближенное к реальности его поведение.

Выбор направления движения происходит следующим образом. Каждую итерацию, агент проверяет клетки вокруг себя на наличие в них феромонов нужного ему типа (далее – маркеров). Если такие маркеры есть, то их расположение может повлиять на коррекцию вектора движения агента с вероятностью 95%. В таком случае координаты вектора направления определяются формулами (2), (3):

(2) $$D_x = \cos\left(\frac{M+\pi}{180}\right),$$

(3) $$D_y = \sin\left(\frac{M+\pi}{180}\right),$$

где D – вектор желаемого направления движения; M – угол, указывающий направление к наиболее интенсивному маркеру.

Агент проверяет 8 соседних с ним ячеек, получая информацию об их интенсивности (интенсивность маркеров на рисунке обозначена в углах ячеек). Ячейки могут содержать два типа маркеров: ведущих к пункту назначения (далее – узел, обозначены зеленым цветом), или к точке выхода (далее – хаб, обозначены красным цветом). В зависимости от типа агента, он проверяет интенсивность маркера того или иного типа.

На рис. 1 маркер с максимальной интенсивностью типа «к хабу» находится в ячейке 1, имеющий значение 40. Самый интенсивный маркер типа «к узлу» находится в ячейке 8 со значением 45.

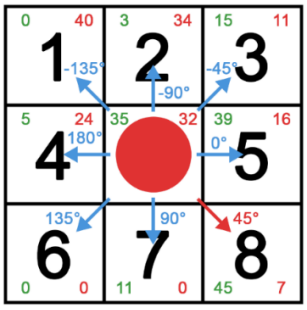

Рис. 1. Углы для перемещения на ячейки с различными интенсивностями.

В данном случае агент движется из хаба и ищет узел. Он оставляет после себя маркеры типа «к хабу» и осуществляет выбор той ячейки с маркером «к узлу», которая имеет максимальную интенсивность. Ячейка 8 здесь имеет наивысшую интенсивность. Следовательно, этот агент выберет своей следующей точкой для перемещения ячейку 8. В соответствии с выбранной ячейкой формируется и угол для перемещения (рис. 1).

Если маркеров нет или же агент «свободен», на этой итерации (то есть не зависит от маркеров), то он либо продолжит движение в произвольно-выбранном направлении, либо в направлении, выбранном на предыдущей итерации (4), (5):

(4) $$D_x = \cos\left(\frac{R+\pi}{180}\right) \cdot W,$$

(5) $$D_y = \sin\left(\frac{R+\pi}{180}\right) \cdot W,$$

где D – вектор желаемого направления движения; R – угол, сгенерированный случайным образом ($0 \le R \le 360$); W – коэффициент, определяющий силу поворота при смене направления.

После получения вектора направления движения, определяется очередная точка перемещения. Её координаты вычисляются по формуле (6):

(6) $$P = (m \cdot D - v) \cdot F_S,$$

где P – вектор, указывающий на координаты следующей точки перемещения; m – константа, устанавливающая стандартную скорость движения агента; D – вектор желаемого направления движения, который находится по формулам (2) и (3) или (4) и (5); v – скорость, с которой, на данный момент, двигается агент; F_s – коэффициент силы поворота.

3.2. Механизм маркеров

Маркеры представляют из себя информацию, содержащуюся в ячейках, которую агенты могут использовать для навигации по карте. Если в ячейку занесен маркер, то он обязательно будет содержать информацию о своем типе, и интенсивность.

Тип имеет два возможных значения: “к хабу” или “к узлу”. Значение этих типов для агентов описано выше.

Значение интенсивности при появлении маркеров равняется значению актуальности агента. Актуальность – это особый параметр агента, который определяет, насколько долго после выходя из хаба или узла данные агента будут значимы. Чем дольше путешествует агент – тем меньше становится его актуальность.

Если маркер уже существует на карте, то при прохождении через него агента, сравниваются значения текущей интенсивности маркера и актуальности агента. Если актуальность больше текущей интенсивности, то в качестве значения интенсивности будет взята актуальность агента. Таким образом, наибольшую интенсивность будут иметь маркеры, отображающие наикратчайший маршрут между точками.

Каждую итерацию маркеры снижают свою интенсивность, а агенты – свою актуальность на определенное значение. При снижении интенсивности до минимального значения, маркер полностью удаляется из ячейки, а при потере актуальности агентом он перестает оставлять какие-либо маркеры. Это симулирует испарение феромонов, что позволяет убирать неактуальные маршруты.

4. Результаты

Алгоритм был реализован в виде подключаемой динамической библиотеки DLL на языке C++ в среде Microsoft Visual Studio. Для визуализации использовалась библиотека SFML.

В качестве входных данных используется карта, имеющая сетку 200 на 200 клеток, на которой установлены точки начала и конца маршрута, а также непреодолимые препятствия. При этом, карта может динамически изменяться пользователем при работе программы. Препятствия и маркеры на карте отображаются закрашенными клетками; агенты – свободно-перемещающимися точками, пункты назначения – квадратами из клеток 3x3.

Параметры симуляции представлены в таблице 1. Результаты пяти экземпляров симуляции с данными параметрами и одинаковым расположением объектов представлены в таблице 2.

Таблица 1. Значение параметров.

Объект	Параметр	Значение
Map Grid	height	200
	width	200
Hub	pos_x	100
	pos_y	150
Node	pos_x	650
	pos_y	550
Agent	speed (m)	140
	steer_strength (F_s)	2
	wander_strength (W)	0,15
	relevance	50
	type	“ToNode”

Таблица 2. Результаты симуляции.

Параметр	Симуляция				
	1	**2**	**3**	**4**	**5**
Время от хаба до узла на первом проходе	7,7	7,4	10,1	8,2	9,6
Время от узла до хаба на первом проходе	9,3	9,5	9,2	8,6	9,4
Число установленных маршрутов после 30 секунд симуляции	3	5	6	4	3
Время от хаба до узла на установленном маршруте	6,9	6,6	6,2	7,4	8,3
Время от узла до хаба на установленном маршруте	7	8,1	6,9	10,8	7,5
Время для нахождения нового маршрута в условиях неопределенности	14,3	11,7	9,8	13,3	14,5
Время для ре-оптимизации текущего маршрута	15,4	13,1	10,8	14,1	17,2

«Условие неопределенности» в параметре 6 обозначает заблокированное состояние главного установленного маршрута, и он показывает время, затраченное на нахождение нового такого маршрута. «Ре-оптимизация» в параметре 7 показывает время, затраченное на нахождение нового маршрута при открытии прохода, который позволяет значительно улучшить текущий маршрут.

5. Выводы

Расчет универсального маршрута при неопределенных обстоятельствах является крайне актуальной задачей в сегодняшних условиях все более сильного продвижения эксплуатации беспилотного транспорта. В данном исследовании была смоделирована ситуация внезапной неопределенности, реагирование на которую ведет к перестроению маршрута движения.

В реализованном виде представленный модуль может быть интегрирован как DLL с различными информационными системами и приложениями. В частности, перспективными областями его использования являются логистические расчеты, стабилизация транспортных потоков и грузоперевозок. Кроме того, разработанный модуль способен решать проблемы планирования, навигации и составления карт. Универсальное решение поиска маршрута позволяет использовать его в разнообразных целях, а эволюционный алгоритм позволит найти маршрут в непредсказуемых и неопределенных обстоятельствах.

Список литературы

1. Kobayashi Y., Kiyama N., Aoshima H., Kashiyama M. A route search method for electric vehicles in consideration of range and locations of charging stations // 2011 IEEE intelligent vehicles symposium (IV). 2011. P. 920-925.
2. Kono T., Fushiki T., Asada K., Nakano K. Fuel consumption analysis and prediction model for “eco” route search // 15th World Congress on Intelligent Transport Systems and ITS America's 2008 Annual Meeting. 2008.
3. Xu J., Gao Y., Liu C., Zhao L., Ding Z. Efficient route search on hierarchical dynamic road networks // Distributed and Parallel Databases. 2015. Vol. 33, No. 2. P. 227-252.
4. Chen L., Shang S., Yao B., Li J. Pay your trip for traffic congestion: Dynamic pricing in traffic-aware road networks // Proceedings of the AAAI Conference on Artificial Intelligence. 2020. Vol. 34. P. 582-589.
5. Sugimura T., Yamaguchi H., Yabuki H.. Development and Implementation of an Arctic Sea Route Search System // International Conference on Offshore Mechanics and Arctic Engineering. 2021. Vol. 85178. P. V007T07A003.
6. Feng G., Su G., Sun Z. Optimal route of emergency resource scheduling based on GIS // Proceedings of the 3rd ACM SIGSPATIAL Workshop on Emergency Management Using. 2017. P. 1-5.
7. Hein K., Xu Y., Wilson G., Gupta A.K. Coordinated optimal voyage planning and energy management of all-electric ship with hybrid energy storage system // IEEE Transactions on Power Systems. 2021. Vol. PS-36, No. 3. P. 2355-2365.
8. Makino T.T., Sakai S. Findings on spatial foraging patterns of bumblebees (Bombus ignitus) from a bee-tracking experiment in a net cage // Behavioral Ecology and Sociobiology. 2004. Vol. 56, No. 2. P. 155-163.
9. Weimerskirch H., Bishop C., Jeanniard-du-Dot T., Prudor A., Sachs G. Frigate birds track atmospheric conditions over months-long transoceanic flights // Science. 2016. Vol. 353 (6294). P. 74-78.
10. Costa-Pereira R., Moll R.J., Jesmer B.R., Jetz W. Animal tracking moves community ecology: Opportunities and challenges // Journal of Animal Ecology. 2022. Vol. 91, No. 7. P. 1334-1344.
11. Zhang W., Gong X., Han G., Zhao Y. An Improved Ant Colony Algorithm for Path Planning in One Scenic Area With Many Spots // IEEE Access. 2017. Vol. 5. P. 13260-13269.
12. Wang X., Zhang H., Liu S., Wang J., Wang Y., Shangguan D. Path planning of scenic spots based on improved A* algorithm // Scientific Reports. 2022. Vol. 12, No. 1. P. 1320.
13. Alivand M., Hochmair H., Srinivasan S. Analyzing how travelers choose scenic routes using route choice models // Computers, Environment and Urban Systems. 2014. Vol. 50. P. 41-52.
14. Akiba T., Iwata Y., Kawarabayashi K., Kawata Y. Fast Shortest-path Distance Queries on Road Networks by Pruned Highway Labeling // 2014 Proceedings of the Sixteenth Workshop on Algorithm Engineering and Experiments (ALENEX). 2014. P. 147-154.
15. Xu R., Miao D., Liu L., Panneerselva J. An Optimal Travel Route Plan for Yangzhou Based on the Improved Floyd Algorithm // 2017 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData). 2017. P. 168-177.
16. Heidari A.A., Mirjalili S., Faris H., Aljarah I., Mafarja M., Chen H.. Harris hawks optimization: Algorithm and applications // Future generation computer systems. 2019. Vol. 97. P. 849-872.
17. Karaboga D., Gorkemli B. A combinatorial Artificial Bee Colony algorithm for traveling salesman problem // 2011 International Symposium on Innovations in Intelligent Systems and Applications. 2011.
18. Jones J., Adamatzky A. Computation of the travelling salesman problem by a shrinking blob // Natural Computing. 2013. Vol. 13, No. 1. P. 1-16.
19. Dorigo M., Gambardella L.M. Ant colony system: a cooperative learning approach to the traveling salesman problem // IEEE Transactions on Evolutionary Computation. 1997. Vol. EC-1, No. 1. P. 53-66.