Модификация и обучение «муравьиного алгоритма» для планирования операций роя подвижных объектов

Я.А. Мостовой[1], В.А. Бердников[1]

[1]Самарский национальный исследовательский университет им. академика С.П. Королева, Московское шоссе 34А, Самара, Россия, 443086

Аннотация. Рой подвижных роботов, как система относительно простых взаимосвязанных управляемых объектов, выполняет общую задачу одновременно и распределённым образом. При планировании операций роя, связанных с созданием в зоне обслуживания сквозной фронтальной полосы из зон работы целевой аппаратуры – полезной нагрузки объектов роя, возникает необходимость учитывать возможность его оперативной перегруппировки, так как на момент планирования точная цель операции роя или ещё не определена, или представляет секрет, или определяется рядом случайных обстоятельств. Поэтому исполнение операции роя целесообразно проводить в две фазы, и первую предварительную фазу начинать ещё до разрешения упомянутых неопределённостей путем создания базовой случайной сети с относительно малой концентрации объектов роя в ней. На второй фазе операции после разрешения упомянутой неопределённости путем локальной перегруппировки объектов роя формируется конкретный программируемый перколяционный путь, обеспечивающий заданное покрытие зонами работы целевой аппаратуры объектов роя определённой зоны обслуживания. Решение этой задачи проводится методами теории программируемой перколяции. Рассмотрен и исследован модифицированный муравьиный алгоритм с эволюционным обучением с целью быстрого решения роем задачи поиска и формирования кратчайшего сквозного пути через зону обслуживания. В этом случае можно существенно сократить время на проведение операции.

1. Введение

Одним из самых быстро развивающихся направлений робототехники на сегодняшний день является групповая робототехника подвижных объектов (или робототехника роя) [1, 2, 13, 15, 16, 21]. Рой роботов-система взаимосвязанных роботов, по сравнению с обычным одиночным роботом, имеет ряд преимуществ, наиболее значимые из которых:

- большие быстродействие и радиус действия подобной системы за счет распределения роя по всей территории зоны обслуживания;
- высокая вероятность выполнения поставленного задания за счет возможности перераспределения целей и замены вышедшего из строя робота другим из роя;
- разнообразие вариантов достижения целей системы за счет перераспределения ролей между объектами роя;
- простота решаемых каждым роботом задач, что, тем не менее, при взаимодействии с множеством таких же объектов роя позволяет решать достаточно сложные задачи.

Рассматриваются операции роя, связанные с созданием в зоне обслуживания сквозной фронтальной полосы, состоящей из состыкованных зон работы целевой аппаратуры – полезной нагрузки объектов роя (аппаратуры поиска, наблюдения, связи, пожаротушения, воздействия на объекты зоны обслуживания и т.п.). При этом геометрическая интерпретация такой операции – создание сквозного пути или маршрута через зону обслуживания [4, 5, 6, 7, 8].

Данная задача роевой робототехники хорошо ложится на задачи теории перколяции. Теория перколяции обнаруживает сквозной «проводящий» путь – перколяционный путь, образующийся в случайной среде при росте концентрации «проводящих объектов» и достижения ею критического значения.

Классическая теория перколяции рассматривает матрицу со случайным заполнением, как модель случайной операционной среды в прямой геометрической интерпретации [3, 4, 9, 12, 17, 22, 23, 24]. В этой квадратной матрице с числом строк L случайная часть ячеек «черная», проводящая поток жидкости или газа, транспортный поток или информационный поток, а остальные ячейки – «белые», не проводящие поток. При росте концентрации (вероятности появления) черных ячеек некоторые из них случайным образом начинают соприкасаться ребрами, что можно интерпретировать как возникновение тесного взаимодействия, и сливаться. Соприкасающиеся ребрами «черные» ячейки образуют случайные проводящие кластеры, которые образуются и растут вместе с ростом концентрации «черных» ячеек [3, 9].

В рассматриваемой задаче размер квадратной ячейки матрицы, определяется вписанной в ячейку окружностью, которая является радиусом действия целевой аппаратуры объекта роя и вытекает из геометрического смысла понятия «кластера», как образования тесного (безпропускового) взаимодействия между объектами, размещёнными в ячейках и соприкасающимися ребрами в рамках авторской модели зоны обслуживания – матрицы.

В классической теории перколяции [3, 9, 23, 24] ищется концентрация объектов роя в зоне обслуживания – «черных» ячеек $p_{\text{п}}$ – порог стохастической перколяции, при которой образуется сквозной случайный маршрут по «черным» ячейкам через всю матрицу в заданном направлении – стохастический перколяционный кластер. Однако стохастический перколяционный кластер имеет рыхлую структуру, множество «мертвых ветвей» и явно избыточен с точки зрения количества необходимых объектов роя.

Поэтому хорошо изученная классическая теория перколяции для плоских решеток для рассматриваемой задачи не пригодна и авторы развивают теорию программируемой (или искусственной) перколяции. В данной теории [4, 5, 6, 7, 8, 19, 20] сквозной перколяционный путь образуется не за счет повышения концентрации «проводящих черных ячеек» вплоть до наступления известной для плоской решетки концентрации стохастического пробоя [9], а активно организуется за счет заполнения интервалов между появившимися на матрице небольшими кластерами, дополняющими планируемый маршрут активными объектами роя.

При реализации программируемой перколяции [4, 5, 6, 7, 8, 19, 20] на первой фазе создаётся предварительная стохастическая основа из распределенных случайным образом в зоне обслуживания объектов при значениях их концентрации гораздо ниже порога стохастической перколяции, а на второй фазе строится сквозной перколяционный маршрут за счет внедрения (установки) дополнительных объектов в имеющиеся межкластерные интервалы (см. рис. 1). При этом концентрация стохастической основы выбирается таким образом, чтобы суммарные затраты на двухфазную операцию были минимальны. В [20] показано, что минимальные затраты на двухфазную операцию наступают при концентрации в районе 0.2 – 0.25.

2. Постановка задачи

Рассматривается рой роботов. Каждый робот может взаимодействовать с соседними в процессе решения некоторой задачи. Максимальное количество кластеров в рое объектов позволяет их легко коммутировать и объединять для совместного решения задач в любом требуемом месте зоны обслуживания. В ходе исследований подобной модели в [4, 5, 6, 7, 8, 19, 20] были обнаружены следующие статистические феномены.

Первый феномен, обнаруженный в результате численного статистического моделирования [4, 5, 6, 7, 8, 19], – это характерная зависимость среднего числа образующихся кластеров от

концентрации с максимальным значением при концентрации в районе 0.25. Такое значение концентрации физически объяснимо: при дальнейшем росте концентрации кластеры растут и начинают активно объединяться, при этом их количество падает (см. рис. 2).

а) б)

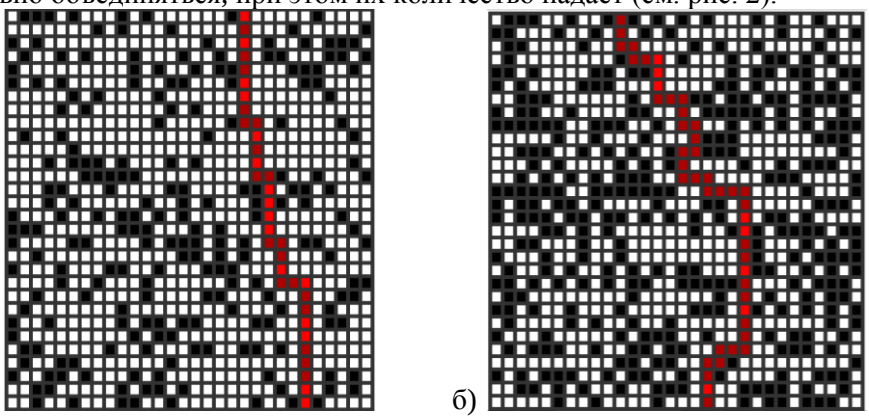

Рисунок 1. Пример решения задачи двухфазной операции на матрице при различной концентрации: а) концентрация $p = 0.25$, б) концентрация $p = 0.4$.

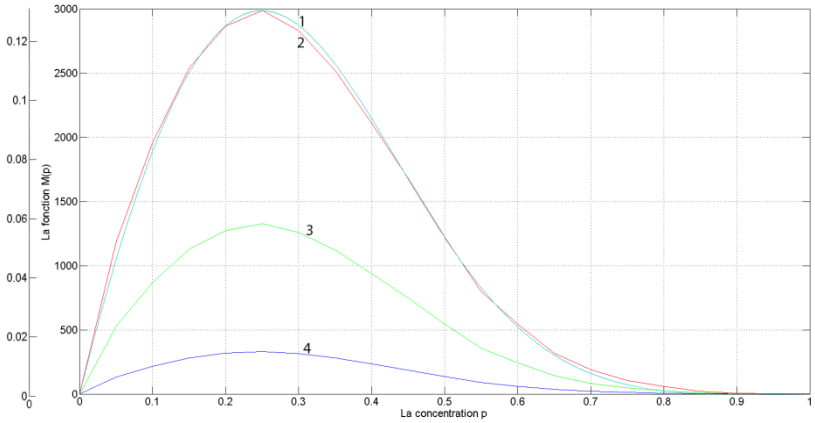

Рисунок 2. Среднее количество кластеров как функция вероятности нахождения объекта в ячейке – концентрации, полученное в результате статистического моделирования на матрицах различного размера; график 1 – нормированная функция среднего количества кластеров, график 2 – функция среднего количества кластеров для матриц 50×50, график 3 – функция среднего количества кластеров для матриц 100×100, график 4 – функция среднего количества кластеров для матриц 150×150.

Второй статистический феномен – это наличие максимальной длины программируемого перколяционного пути при концентрации стохастического перколяционного пробоя. Здесь наблюдается максимальная извилистость пути, а количество добавленных в программируемый перколяционный путь ячеек равно нулю.

В ходе обширного математического эксперимента по исследованию двухфазной операции [4, 5, 6, 7, 8, 20] на перколяционной матрице конечного размера были установлены функции средней длины пути программируемой перколяции от концентрации и среднего количества добавленных объектов в программируемый перколяционный путь от концентрации (см. рис. 3 и 4). Данный эксперимент проводился следующим образом: на множестве перколяционных матриц проводился сначала засев объектов с различными концентрациями. Далее алгоритмом Хошена-Копельмана [14, 18] выделялись кластеры, а после строился оптимальный (с минимальным количеством добавленных ячеек) программируемый перколяционный путь (эксперимент проводился для матриц размером 50×50, 100×100 и 200×200). После алгоритмом «Молния-Дейкстра» [4, 5, 6, 7, 8, 19, 20] прокладывалось множество подозрительных на программируемый перколяционный путь путей, из который по критерию минимальности количества добавленных объектов выбирался лучший.

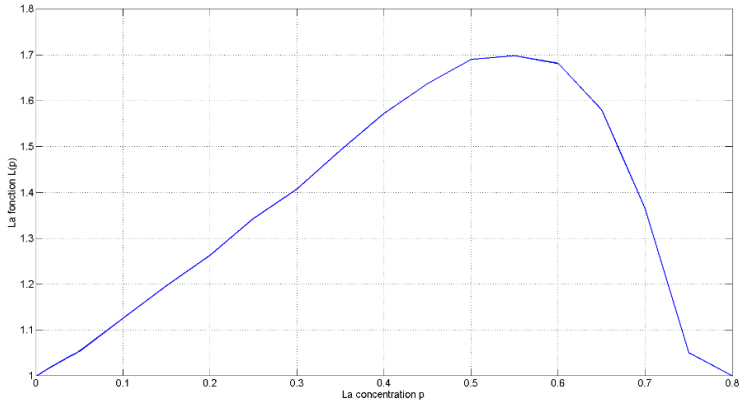

Рисунок 3. График функции $L(p)$ средней длины программируемого перколяционного пути.

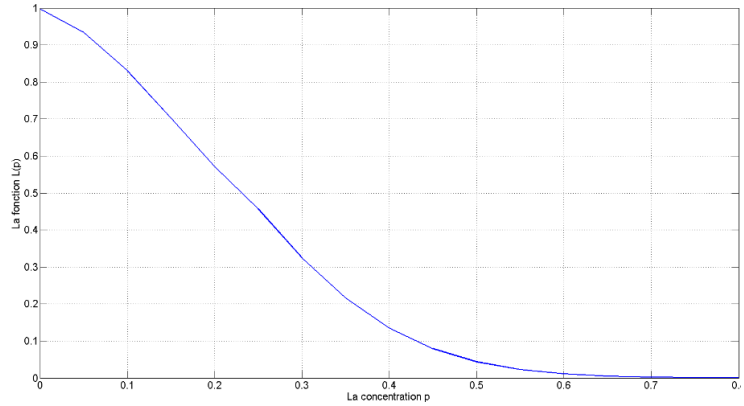

Рисунок 4. График функции $R(p)$ среднего количества добавленных объектов в межкластерные интервалы программированного перколяционного пути – добавленных ячеек в программируемый перколяционный путь.

При рассмотрении решения задачи программируемой перколяции большой сетью взаимосвязанных роботов, названной нами роем роботов, необходима согласованность функционирования отдельных объектов в системе объектов, приводящая к поведению системы объектов как целого [21]. Учитывая возможное большое количество объектов в рое (несколько сотен и более), процессы управления функционированием и возникающие при этом фронтальные структуры роя должны быть следствием самоорганизации роя без «управляющей руки», действующей на каждый объект роя извне. Внешнее целеопределение при этом возможно и должно быть доведено до каждого члена роя.

Для реализации самоорганизующегося поведения каждый объект роя должен иметь систему управления, позволяющую ему занимать в пространстве заданное положение. Для этого каждый член роя должен знать свое собственное местоположение, имея автономную подсистему навигации. Это местоположение должно передаваться другим объектам роя по подсистеме связи. Таким образом, у каждого члена роя может быть сформирована в бортовой ЦВМ бинарная матрица роя, как модель зоны обслуживания, где значение 1 означает, что в данной точке пространства – ячейке матрицы – установлен член роя, 0 – в данной точке пространства объект роя отсутствует. После формирования этой матрицы и прокладки программируемого перколяционного пути из заданной точки операционной среды в бортовых ЦВМ каждый объект роя «узнает свой маневр» и автономно занимает требуемое положение.

Однако ограничения на вычислительные ресурсы каждого робота затрудняют использование описанного подхода к решению задач двухфазной операции, т.к. использование алгоритма «Молния-Дейкстра» требует значительных вычислительных затрат. Как следствие, целью настоящей статьи является создание алгоритма, позволяющего формировать кратчайший путь в заданном направлении через матрицу со случайным заполнением с любой граничной ячейки матрицы, проходящий через имеющиеся попутные кластеры с минимальным количеством

добавляемых в межкластерные интервалы объектов – «красных ячеек» – и обладающего меньшим временем работы при прочих равных условиях чем алгоритм Дейкстры.

Для решения этой задачи предлагается, во-первых, использовать «алгоритм муравьиной колонии» (объяснение основных причин выбора этого алгоритма в [8]) и, во-вторых, проводить предварительное обучение системы управления роботов на множестве матриц случайного размера и случайной концентрации. Это обучение должно улучшить параметры муравьиного алгоритма и привести к сокращению «колонии муравьев», что даст значительный выигрыш в скорости с допустимой потерей точности.

3. Исследование скорости работы алгоритма вычисления пути программируемой перколяции

Использование алгоритма Дейкстры как основного алгоритма поиска минимальных путей на перколяционной матрице является достаточно ресурсозатратным [8, 11]. Средняя скорость работы алгоритма Молния-Дейкстра оценивается как $O(L^5)$, где L – это высота перколяционной матрицы. Это легко показать:

Т.к. средняя скорость самого алгоритма Дейкстры оценивается как $O(n^2)$, где n – это количество вершин графа, то для матрицы (которую можно представить как граф с количеством вершин L^2) средняя скорость – $O(L^4)$. Для выбора наиболее минимального маршрута необходимо данный алгоритм запустить из каждой точки верхней строки матрицы, т.е. средняя скорость будет равна $O(L^5)$. В случае использования внутри алгоритма Дейкстры быстрой сортировки или бинарной кучи, то скорость рассматриваемого здесь алгоритма можно улучшить до $O(L^4 \log L)$.

Очевидно, что массовое использование такого алгоритма будет занимать достаточно длительное время и требовать серьезных процессорных мощностей, поэтому в [8] было предложено использовать муравьиный алгоритм. Очевидно, что использование стандартного муравьиного алгоритма [11] на перколяционной матрице затруднительно ввиду того, что алгоритм на первом этапе работы равновероятно выбирает путь. Чтобы увеличить скорость сходимости алгоритма, необходимо было скорректировать вероятность выбора, т.е. добавить некоторый модификатор m_{ij} к вероятности выбора кластера и движению вдоль направления программируемого перколяционного пути. Остальное тело классического муравьиного алгоритма не менялось [8, 11]. Таким образом, формула выбора направления движения муравья стала выглядеть следующим образом:

$$p_k = \frac{p + m_{ij} + ph_{ij}(n) + t\delta(i)}{\sum_{k=1}^{3} \; \substack{i=\begin{cases} i \\ i+1 \end{cases} \\ j=\begin{cases} j-1 \\ j \\ j+1 \end{cases}} p + m_{ij} + ph_{ij}\gamma(i) + t\delta(i)} \tag{1}$$

где p – вероятность выбора пути, заданная случайным числом ($p \in [0,1]$); m_{ij} – модификатор кластера, равный некоторому случайному числу, если ячейка матрицы принадлежит кластеру, и 0 в противном случае; $ph_{ij}(n)$ – значение феромона ($ph_{ij}(n) = ph_{ij}(n-1) - \Delta * L * L(P)$, Δ – скорость испарения феромона, n – номер итерации); t – модификатор движения по направлению перколяции; $\delta(i)$ – дельта-функция от i, равная $\delta(i) = \begin{cases} 1, & i = i+1 \\ 0, & \text{иначе} \end{cases}$; k – одно из возможных направлений перколяции (вниз, вправо, влево); $\gamma(n) = \begin{cases} 1, & ph_{ij} \geq 0 \\ 0, & \text{иначе} \end{cases}$, i – номер строки матрицы, j – номер столбца матрицы.

Максимальная скорость (т.е. тот случай, где выбранный путь идет по всем ячейкам матрицы) такого алгоритма может быть оценена как $O(L^3K)$, где L – высота матрицы, K – количество муравьев. В среднем (т.е. при хорошо выбранных параметрах) оценить скорость работы муравьиного алгоритма можно $O\big(KL^2L(p)\big)$. Сравнение среднего времени работы

алгоритмов, реализованных на языке C++, представлено на рисунках 5 и 6. Все результаты представлены для процессора Intel Core i7 CPU 870.

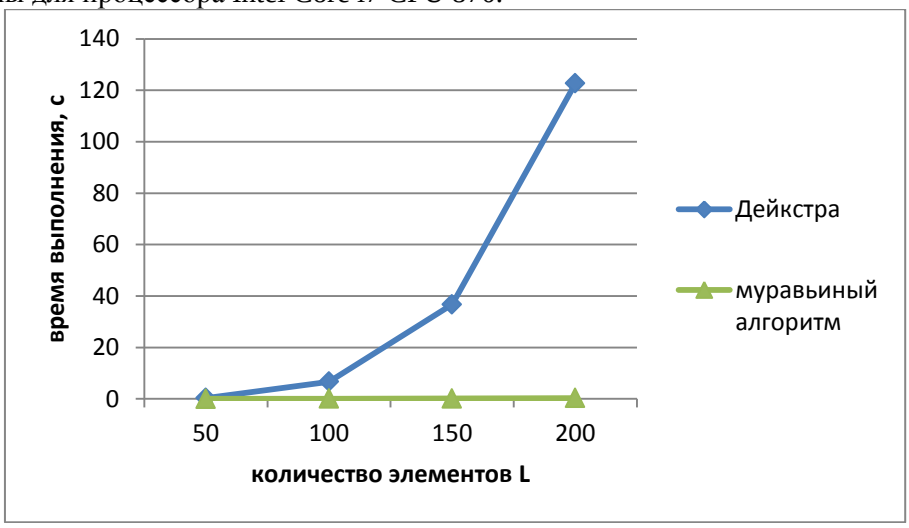

Рисунок 5. Среднее время работы алгоритмов.

Рисунок 6. Среднее время работы муравьиного алгоритма для 100 «муравьев», выпущенного из одной ячейки на матрице.

Однако муравьиный алгоритм очень сильно зависит от качества выбора параметров (скорость испарения феромонов, начальное значение феромона и количество добавляемого феромона на каждом шаге, размер колонии), которые сильно влияют на «случайность» поведения алгоритма. Так при низком значении феромона прокладываемые пути могут не сойтись (колония кончилась), при высоком – увеличивается вероятность выбора неверного пути и т.п. Сам же выбор параметров является трудной и неочевидной задачей ввиду относительно большого количества этих параметров и неочевидности их взаимодействия друг с другом [11]. В таком случае удобно будет воспользоваться эволюционным подходом к выбору параметров.

4. Эволюционное обучение муравьиного алгоритма

В качестве фенотипа оставим само тело муравьиного алгоритма, т.е. не будем вносить никаких изменений в работу данного алгоритма и не будем менять в ходе обучения правило выбора оптимального пути [10].

В качестве генотипа воспользуемся следующими параметрами: m_{ij} (модификатор кластера), ph_{ij} (значение феромона), t (модификатор движения вдоль перколяционного пробоя), Δ (скорость испарения феромона) и K (количество муравьев) (см. выражение 1).

Правило обучения – минимизация следующей ошибки:

$$\xi = \frac{0.1 * (L_{real} - L(p)L) + 0.9 * (R_{real} - R(p)L)}{L} \quad (2)$$

где L_{real} – полученная длина искусственного перколяционного пробоя; $L(p)$ – эталонная нормированная по L длина искусственного перколяционного пробоя; R_{real} – полученное количество добавленных ячеек; $R(p)$ – эталонное нормированное по L количество добавленных ячеек, L – длина матрицы.

Обучение генетического алгоритма происходило по следующему правилу:

1. Проведение двухфазной операции на основе материнской колонии, подсчет ее ошибки ξ_m;
2. Создание потомков на основе материнской колонии с внесенными генетическими мутациями (в значения параметров генотипа вносились случайные изменения), подсчет ошибок потомков ξ_{e_i}, где i – номер потомка;
3. Выбор той колонии (материнская или кто-то из потомков), чья ошибка обучения ξ минимальна.

График сходимости обучения представлен на рисунках 7 и 8 для различных условий обучения: в первом случае обучение модифицированного муравьиного алгоритма проводилось для фиксированной концентрации ($p = 0{,}25$) на множестве матриц случайного размера, во втором случае – обучение проводилось на множестве случайных матриц случайного размера со случайной концентрацией объектов роя. В обоих случаях начиная с 96-го поколения (одинаковые номера поколений при обучении – не более чем совпадение) обучение прекратилось, шум ошибки стабилизировался в соответствии с дисперсиями для функций средней длины программируемой перколяции ($L(p)$) и среднего количества добавленных объектов ($R(p)$), которые являлись эталонными в выражении 2.

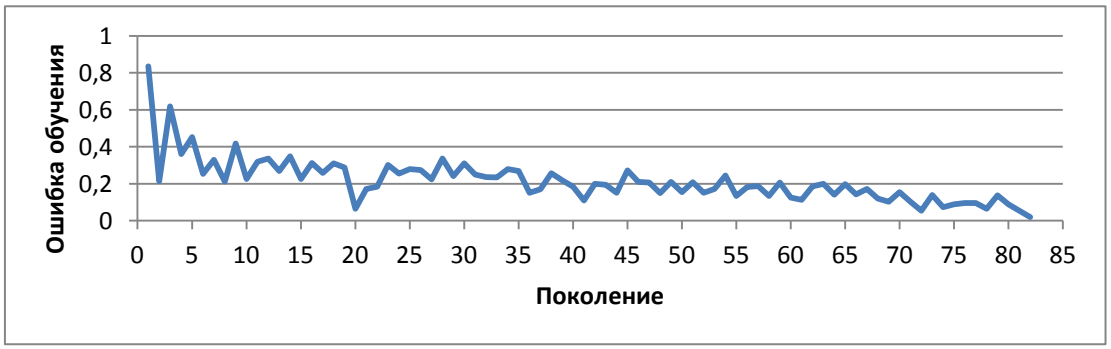

Рисунок 7. Динамика ошибки обучения эволюционным алгоритмом муравьиного алгоритма (фиксированная концентрация, случайный размер матрицы).

Рисунок 8. Динамика ошибки обучения эволюционным алгоритмом муравьиного алгоритма (случайная концентрация, случайный размер матрицы).

Также во время обучения было замечено, что начиная с некоторого шага (поколения), муравьиный алгоритм практически переставал зависеть от количества муравьев в колонии, т.е. он «эволюционировал» в стандартный квазиоптимальный алгоритм поиска оптимального пути.

Это позволило снизить количество муравьев до 3-х в колонии, что существенно увеличило скорость работы данного обученного алгоритма. Выбор такого количества агентов в колонии обусловлен следующими соображениями: вероятность выбора ячейки, посещенной предыдущим (как было замечено в ходе эксперимента), уже на третьем шаге для обученных эвристических параметров стремится к единице, т.е.:

$$\begin{cases} \dfrac{p + m_{ij} + ph_{ij} + t\delta(i)}{\sum^{3}_{\substack{k=1 \\ i=\begin{cases} i \\ i+1 \end{cases} \\ j=\begin{cases} j-1 \\ j \\ j+1 \end{cases}}} p + m_{ij} + \big(ph - 2\Delta * L * L(P)\big)\gamma(i) + t\delta(i)} \to 1, \qquad p \in [0,1], ph_{ij} > 0 \\ \dfrac{\big(p_1 + m_{ij} + ph_{ij} + t\delta(i)\big) + \big(p_2 + m_{ij} + t\delta(i)\big) + \big(p_3 + m_{ij} + t\delta(i)\big)}{\sum^{3}_{\substack{k=1 \\ i=\begin{cases} i \\ i+1 \end{cases} \\ j=\begin{cases} j-1 \\ j \\ j+1 \end{cases}}} p + m_{ij} + \big(ph - 2\Delta * L * L(P)\big)\gamma(i) + t\delta(i)} = 1 \end{cases} \tag{3}$$

Средняя скорость обученного муравьиного алгоритма оценивается как $O\big(3L^2L(p)\big) \approx O\big(L^2L(p)\big)$. Результат обучения представлен на рисунке 9.

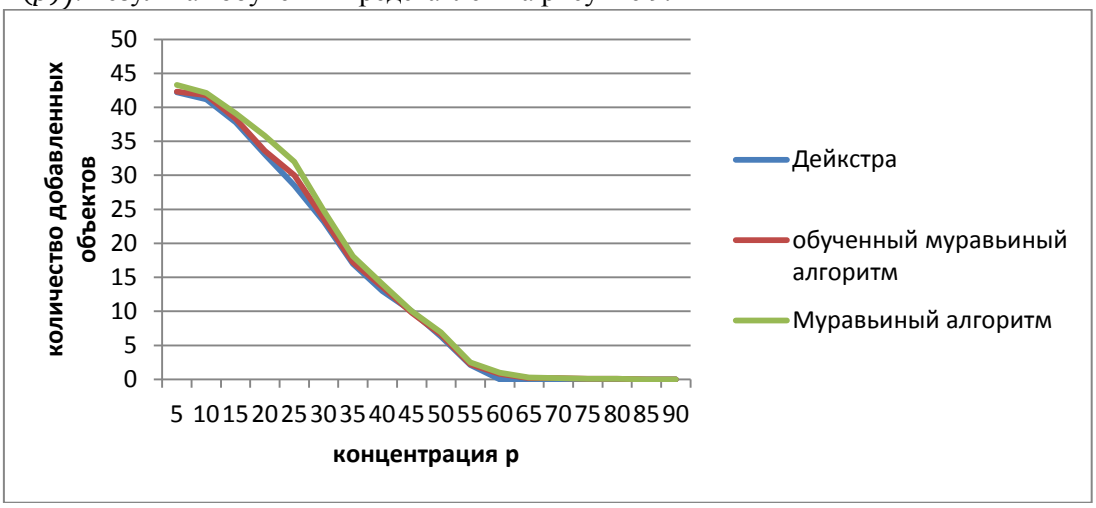

Рисунок 9. Среднее количество добавленных объектов рассмотренными алгоритмами.

Анализируя данный график, можно сказать, что средняя погрешность необученного муравьиного алгоритма много выше, чем у алгоритма Дейкстры и обученного модифицированного муравьиного алгоритма. В ходе эволюционного процесса муравьиный алгоритм стал практически идеально совпадать с результатами алгоритма Дейкстры, а также приобрел значительный выигрыш в скорости, т.к. удалось резко сократить количество муравьев в колонии. Однако ~~всегда~~ существует вероятность, что обученный муравьиный алгоритм изначально выберет неоптимальный путь. Длительное обучение может только снизить вероятность наступления подобного исхода, но не исключить его.

5. Выводы

В результате проделанной работы можно сделать следующие выводы:

1. Возможное большое количество объектов в рое (несколько сотен и более), сложные процессы управления целевым функционированием роя с образованием различных фронтальных структур должны быть следствием самоорганизации роя без «управляющей руки», действующей на каждый объект роя извне. Это требует повышения эффективности по

времени работы алгоритма прокладки программируемого перколяционного пути, которую не может предоставить алгоритм «Молния-Дейкстра».

2. Исследована скорость алгоритма «Молния-Дейкстра» для решения задачи формирования сквозного маршрута программируемой перколяции через матрицу со случайным заполнением и предложен адекватный вариант его замены модифицированным муравьиным алгоритмом с сохранением приемлемой точности выходных результатов.

3. Специально для рассматриваемой задачи разработан модифицированный генетический муравьиный алгоритм, предложен эволюционный метод его обучения и проведено это обучение, в результате которого удалось сильно увеличить скорость работы модифицированного муравьиного алгоритма за счет снижения количества муравьев в колонии (результат эволюции) и настроить его параметры (скорость испарения феромонов, начальное значение феромона, количество добавляемого феромона на каждом шаге и дополнительные модификаторы, описанные в выражении 1).

6. Литература

[1] Каляев, И.А. Модели и алгоритмы коллективного управления в группах роботов / И.А. Каляев, А.Р. Гайдук, С.Г. Капустян – М.: Физматлит, 2009.
[2] Каляев, И.А. Стайные принципы управления в группе объектов/ И.А. Каляев, А.Р. Гайдук // Мехатроника, автоматизация, управление. – 2004. – Т. 12. – С. 27-38.
[3] Москалев, П.В. Математическое моделирование пористых структур / П.В. Москалев, В.В. Шитов – Москва: Физматлит, 2007. – 120 с.
[4] Мостовой, Я.А. Статистические феномены больших распределенных кластеров наноспутников // Вестник Самарского Государственного Университета имени академика С.П. Королева (национального исследовательского университета). – 2011. – Т. 26, № 2. – С. 80-89.
[5] Мостовой, Я.А. Двухфазные операции в больших сетях наноспутников // Компьютерная оптика. – 2013. – Т. 37, № 1. – С. 120-130.
[6] Мостовой, Я.А. Управляемая перколяция и оптимальные двухфазовые операции в больших сетях наноспутников // Инфокоммуникационные технологии. – 2013. – Т. 11, № 1. – С. 53-62.
[7] Мостовой, Я.А. Моделирование оптимальных двухфазных операций в случайных операционных средах // Автометрия. – 2015. – Т. 51, № 3. – С. 35-41.
[8] Мостовой, Я.А. Муравьиные алгоритмы в планировании двухфазных операций на больших сетях / Я.А. Мостовой, В.А. Бердников // Проблемы управления и моделирования в сложных системах, 2017.– С. 445-454.
[9] Тарасевич, Ю.Ю. Перколяция: теория, приложения, алгоритмы – Москва: УРСС, 2002. – 109 с.
[10] Рутковская, Д. Нейронные сети, генетические алгоритмы и нечеткие системы / Д. Рутковская, М. Пилиньский, Л. Рутковский – М.: Горячая линия-Телеком, 2008. – 452 с.
[11] Штовба, С.Д. Муравьиные алгоритмы // Exponenta Pro. Математика в приложениях. – 2003. – № 4. – С. 70-75
[12] Alexandrowicz, Z. Critically branched chains and percolation clusters // Physics Letters A. – 1980. – Vol. 80(4). – P. 284-286.
[13] Abraham, A. Swarm Intelligence in Data Mining / A. Abraham, C. Grosan, V. Ramos // Studies in Computational Intelligence – Springer Verlag, Germany, 2006.
[14] Babalievski, F. Cluster counting: the Hoshen-Kopelman algorthm vs. Spanning three approach // International Journal of Modern Physics C. – 1998. – Vol. 9(1). – P. 43-61.
[15] Beni, G. From Swarm Intelligence to Swarm Robotics // Swarm Robotics SAB International Workshop. Lecture Notes in Computer Science. – 2005. – Vol. 3342. – P. 1-10.
[16] Bonabeau, E. Swarm Intelligence: A Whole New Way to Think About Business / E. Bonabeau, Ch. Meyer // Harvard Business Review. – 2001. – Vol. 79(5). – P. 106-114.
[17] Galam, S. Universal formulas for percolation thresholds / S. Galam, A. Mauger // Phys. Rev. E. – 1996. – Vol. 53(3). – P. 2177-2181.

[18] Hoshen, J. Percolation and cluster distribution. I. Cluster multiple labeling technique and critical concentration algorithm / J. Hoshen, R. Kopelman // Phys. Rev. B. – 1976. – Vol. 14. – P. 3438-3445.
[19] Mostovoy, Y.A. Analytical and numerical modeling of the process for cluster emergence of objects in a random environment / Y.A. Mostovoy, V.A. Berdnikov // Journal of Physics: Conference Series. – 2018. – Vol. 1096(1).
[20] Mostovoy, Y.A. Statistical modeling of a scale network of nanosatellites / Y.A. Mostovoy, V.A. Berdnikov // Journal of Physics: Conference Series. – 2018. – Vol. 1096(1).
[21] Sahin, E. Swarm robotics: From sources of inspiration to domains of application // Swarm Robotics. Lecture Notes in Computer Science. – 2005. – Vol. 3342. – P. 10-20.
[22] Sarshar, N. Scalable Percolation Search in Power Law Networks / N. Sarshar, P.O. Boykin, V.P. Roychowdhury // Proceedings of the Fourth International Conference on Peer-to-Peer Computing – Zurich, 2004.
[23] Stauffer, D. Scaling theory of percolation clusters // Physics Reports. – 1979. – Vol. 54. – P. 1-74.
[24] Stauffer, D. Introduction to Percolation Theory / D. Stauffer, A. Aharony – London: Taylor & Francis, 1992.

Modification and training of ant algorithm for planning swarm operations of moving objects

Y.A. Mostovoi[1], V.A. Berdnikov[1]

[1]Samara National Research University, Moskovskoe Shosse 34A, Samara, Russia, 443086

Abstract. A swarm of mobile robots, as a system of relatively simple interconnected controlled objects, performs a common task simultaneously and in a distributed manner. When planning swarm operations associated with the creation in the service area through the frontal band of the zones of operation of the target equipment there is a need to consider the possibility of its operational regrouping, since at the time of planning the exact purpose of the swarm operation has not yet been determined, or is a secret, or is determined by a number of random circumstances. The execution of the swarm operation is advisable to carry out in two phases. The first phase starts even before the resolution of these uncertainties by creating a basic random network with a relatively small concentration of robots. In the second phase, after resolving the uncertainty, a programmable percolation route is formed by local rearrangement of the swarm objects, which provides a predetermined coverage of the target equipment of the robots with a certain service zone. A modified ant algorithm with evolutionary learning is considered in order to quickly solve the swarm problem of two-phase operation. In this case, you can significantly reduce the time for the operation.