УДК 519.17:519.816:004.715

АЛГОРИТМ МУРАВЬЯ ДЛЯ РЕШЕНИЯ ЗАДАЧИ КОММИВОЯЖЕРА

Б.О. Кузиков, студ.; С.П. Шаповалов, доц.
Сумський государственный университет

Данная статья рассматривает применимость алгоритма муравьиной колонии (ACS) к задачи коммивояжера (TSP). В данном алгоритме используется система кооперирующихся интеллектуальных агентов, названных муравьями, для нахождения решений указанной задачи. Для кооперации агенты используют «фермент», оставляемый на гранях транспортной сети, в процессе поиска оптимального решения. Алгоритм показывает хорошую производительность, как для симметричной, так и для асимметричной задачи коммивояжера.

Эффективность созданных человечеством алгоритмов в главном своем предначертании проверяется практической их реализацией, а решение задач прикладного характера сопровождается детализацией используемых при этом алгоритмов [1].

Алгоритмы муравья [2,3], основанные на применении нескольких агентов, позволяют решать самые разнообразные проблемы статики и динамики. Предложим детализацию этого алгоритма к проблеме маршрутизации в сетях, рассмотрев задачу коммивояжера.

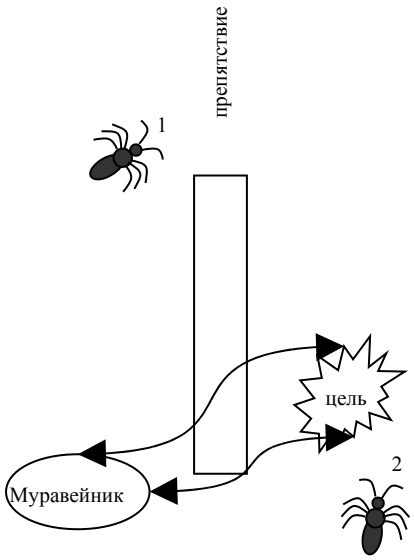
Общая задача коммивояжера (traveling salesmen problem, TSP) [4,5] состоит в следующем: используя заданную систему транспортных соединений (дорог и т.п.) между пунктами (городами, фирмами и т.п.) в конкретной зоне обслуживания, посетить все пункты в такой последовательности, чтобы пройденный путь был минимальным [1,5]. В терминах теории графов задачу можно сформулировать следующим образом: в конечном взвешенном графе G=(V,E) найти гамильтонов цикл наименьшего веса. То есть для каждого ребра $(u,v) \in E$ известна его стоимость $d(u,v)$. Необходимо построить цикл A, проходящий по всем вершинам ровно один раз, причем $d(A) = \sum_{(u,v)\in A} d(u,v) \rightarrow \min$. В случае $d(u,v) \neq d(v,u)$ задача называется асимметричной (ATSP). Симметрия и выполнение неравенства треугольника $d(u,v) \leq d(u,w) + d(w,v)$ в значительной мере облегчают решение задачи, однако они не всегда применимы.

Общая задача коммивояжера является NP-полной, то есть не разрешимой за полиномиальное время [1]. В случае небольшой дорожной сети она разрешима методом перебора с использованием различных вариантов его сокращения (например, метод ветвей и границ). Однако в большинстве практических приложений задача имеет большую размерность, и потому переборные алгоритмы слабо применимы.

Принцип алгоритмов муравья (Ant algorithms), или оптимизации по принципу муравьиной колонии, был предложен Марко Доринго (Marco Doringo) [2,3,4]. Идея алгоритмов состоит в том, что хотя муравьи слепы, они умеют ориентироваться на сложной местности, находя оптимальный путь между муравейником и внешними точками. При этом в качестве маркера используется фермент, который тем концентрированней, чем больше муравьев прошли по данному пути.

Проиллюстрируем вышесказанное.

Обозначим муравья, идущего по верхней грани, – первым, а по нижней – вторым. Пусть верхняя грань вдвое длиннее нижней. Первоначально путь от муравейника к цели неизвестен. К тому времени как второй муравей достигнет цели, первый пройдет только пол пути. На пути в муравейник второй муравей пойдет по нижней грани, так как только она помечена ферментом. К тому моменту как первый муравей достигнет цели, верхняя грань будет помечена ферментом 1 раз, а нижняя уже 2. А значит, муравей выберет нижнюю, более короткую грань.



Рассмотрим термины данного алгоритма применительно к задаче коммивояжера.

Путь муравья – данная нам по условию задачи транспортная сеть. Исходный граф двунаправлен, значит, муравей может путешествовать вдоль грани в любом направлении.

Муравей – программный агент, который является членом большой колонии, используемой при решении некоторой задачи. Для того чтобы удовлетворить свойства гамильтонова цикла, каждый муравей поддерживает список табу – список узлов, которые он уже посетил. Узлы в этом списке располагаются в порядке их посещения – позже этот список будет использован для определения длины пути. Фермент муравьи «оставляют» на пройденных ребрах графа.

Популяция. Перед началом движения муравьев случайным образом распределяют по узлам графа. Этим самым повышается вероятность быстрого нахождения оптимального маршрута. Оптимизация количества муравьев в популяции будет рассмотрена ниже, наряду со значениями остальных параметров.

Движение муравья (state transition rule) Направление движения муравья задается вероятностным уравнением. Формула 1 устанавливает вероятность, с которой *k*-й муравей системы, находящийся в городе r, отправится в город s:

$$p_k(r,s) = \begin{cases} \dfrac{\tau(r,s)\cdot\eta(r,s)^{\beta}}{\sum\limits_{u\in J_k(r)}\tau(r,u)\cdot\eta(r,u)^{\beta}} & s\in J_k(r), \\ 0 & s\notin J_k(r). \end{cases} \qquad (1)$$

В уравнении (1) s понимается как случайная величина. Для повышения эффективности при дальнейшем развитии алгоритма для выбора пункта назначения была введена формула (2):

$$s = \begin{cases} \arg\max\left[\tau(r,s)\cdot\eta(r,s)^{\beta}\right] & \text{åñëè } q \le q_0 \quad (\text{èñïîëüçîâàíèå ëó÷øåãî}), \\ S & \text{åñëè } q > q_0 \quad (\text{ïîèñê íîâûõ ðåø åíèé}), \end{cases} \qquad (2)$$

где S – результат, полученный при использовании формулы (1);

$\tau(i,j)$ – интенсивность фермента на грани между узлами *i* и *j*;

$\eta(i,j) = 1/d(i,j)$ – функция, обратная длине соответствующей грани;

$J_k(r)$ – множество узлов, еще не посещенных муравьем *k*, находящимся в узле *r*;

β – параметр, контролирующий относительный приоритет фермента на пути над видимостью следующего города;

q – случайное число в диапазоне [0...1], выбираемое каждый раз при пересчете формулы;

$0<q_0<1$– параметр, показывает относительную важность поиска новых маршрутов, над использованием старых.

Путешествие муравья. Заметим, что при движении циклы невозможны благодаря списку табу. При переходе муравья от одного узла к другому уровень фермента изменяется по формуле (3):

$$\tau(r,s) \leftarrow (1-\rho)\cdot\tau(r,s) + \rho\cdot\Delta\tau(r,s), \qquad (3)$$

где 0<*p*<1 – параметр. В зависимости от применяемых эвристик величина $\Delta\tau(r,s)$ формулы (3) может находиться по следующим формулам: (I) $\Delta\tau(r,s) = \gamma \max_{z\in J_k(s)} \tau(s,z)$, (II) $\Delta\tau(r,s) = \tau_0$ или (III) $\Delta\tau(r,s) = 0$. В первом случае алгоритм носит название Ant-Q, второй – используется для повышения производительности и называется ACS. Первый и второй алгоритмы дают аналогичные результаты. Третий случай может быть использован в случае симметричной задачи коммивояжера, хотя и резко снижает производительность алгоритма и потому редко используется на практике [2,4].

Испарение фермента (global updating rule). После завершения всеми муравьями их путешествий выделяется наиболее оптимальный путь. Для повышения эффективности поиска пути на следующей итерации ко всем граням применяется формула (4):

$$\tau(r,s) \leftarrow (1-\alpha)\cdot\tau(r,s) + \alpha\cdot\Delta\tau(r,s), \qquad (4)$$

где $\Delta\tau(r,s) = \begin{cases} (L_{gb})^{-1}, & \text{åñëè } (r,s) \quad \text{ëåæèò íà êðàò÷àéøåì} \\ 0 & \text{â îñòàëüíûõ ñëó÷àÿõ;} \end{cases}$

L_{gb} – длина лучшего маршрута, найденного на данный момент, 0<α<1 – параметр. Вместо L_{gb} может быть использована длина лучшего маршрута текущей итерации. Согласно практическим наблюдениям разница между этими двумя вариантами незначительна.

Повторный запуск. После того как путь всех муравьев завершен, а количество фермента на гранях обновлено согласно (4), происходит повторный запуск алгоритма. При повторном запуске необходимо очистить список табу и длину пути каждого из муравьев. Повторный запуск осуществляется оговоренное число раз или до момента, когда на протяжении нескольких запусков не было отмечено изменений.
В целом алгоритм применительно к задаче коммивояжера можно сформулировать следующим образом:

Инициализация переменных
Loop /* итерация поиска решения*/
    Разметить всех муравьев
    Loop /* шаги поиска*/
        Для каждого муравья выбрать следующий узел, используя (2)
        переместить муравья
        обновить уровень фермента, используя (3)
    Until все муравьи закончат построение пути
    Применить испарение фермента ко всей сети, используя (4)
Until выполнено нужное число итераций

Между алгоритмами муравья и генетическими алгоритмами можно провести некую параллель. Аналог наследственной информации о лучшем решении в данном случае сохраняется в виде фермента на гранях. Величина q вносит элемент случайности в действие популяции.

Отдельно стоит оговорить подбор параметров, так как они существенно влияют на скорость нахождения качественного решения. Подчеркнем также, что значение параметров слабо зависит от размерности задачи.

Параметр β определяет важность выбора более короткой грани. Кроме того, этот параметр позволяет проследить ценность функции η для нахождения эффективных решений за приемлемое время. На практике часто используется $\beta = 2$.

Параметр q_0 отвечает за нахождение новых путей, α и ρ - за испарение фермента. Для задач небольшой размерности (n>100) используют значения q_0=0,9, α=ρ=0,1.

Для определения значения τ_0 используют эвристику $\tau_0 = (n\cdot L_{nn})^{-1}$, где L_{nn} – оценка длины маршрута; *n* – количество узлов.

Отдельным вопросом является также численность популяции. Зависимость величины найденного решения от количества муравьев при фиксированном числе итераций можно представить графиком, приведенном на рис. 2.

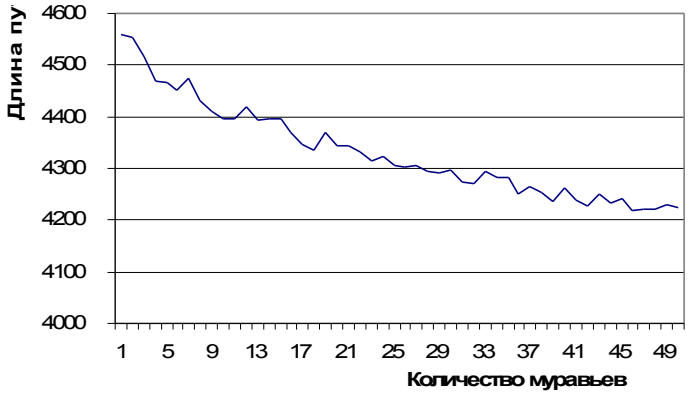


Рисунок 2 - Зависимость длины найденного пути от количества муравьев

График был получен при тестировании алгоритма по следующей методике. Исходные данные: набор из 50, случайно сгенерированных городов, расстояние между которыми не превышает 500. Полученная таблица инцидентности является симметричной. Алгоритм в каждом случае производил 20 итераций поиска. Результат был усреднен между 50 запусками с одинаковым количеством муравьев.

Из графика видно, что с увеличением количества муравьев растет и качество решения. Однако не следует забывать, что время работы алгоритма линейно зависит от числа агентов. Практические исследования показывают, что оптимальное соотношение качества решения и времени работы алгоритм дает при соотношении $m/n \approx 0.4$, где m – численность муравьев; n – количество узлов [2].

ВЫВОД

В работе рассмотрены алгоритм колонии муравьев и его применение к задаче коммивояжера. Алгоритм имеет хорошую производительность как в случае симметричной, так и асимметричной задачи коммивояжера. Важно также подчеркнуть возможность распараллеливания его работы между различными вычислительными ресурсами [3]. В работе рассмотрена оптимизация алгоритма с помощью эвристики изменения коэффициента β.

Для иллюстрации качества работы алгоритма проведено сравнение результатов работы с результатами, полученными при помощи алгоритма MST-Prim. В среднем за аналогичное время были получены результаты, на 24% качественнее.

SUMMARY

This paper content common arrangement of traveling salesman problem (TSP), introduces ant colony system (ACS) and distributed algorithm that is applied to the TSP. In ACS, a set of cooperating agents called ants cooperate to find good solutions to TSPs. Ants cooperate using an indirect form of communication mediated by pheromone they deposit on the edges of the TSP graph while building solutions. The results show that ACS outperforms other nature-inspired algorithms such as simulated annealing and evolutionary computation. It's appear for some of the best performing algorithms for symmetric and asymmetric TSPs.

СПИСОК ЛИТЕРАТУРЫ

1. Кормен Т., Лейзерсон Ч., Ривест Р. Алгоритмы: построение и анализ. – М.: МЦНМО, 2000. -960 с.: ил.
2. Dorigo M., Gambardella L.M. Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem. IEEE Transactions on Evolutionary Computation, 1997 1(1):53-66.
3. Dorigo M., G. Di Caro, Gambardella L. M. Ant Algorithms for Discrete Optimization. Artificial Life, 1999 5(2):137-172.
4. Джонс М.Т. Программирование искусственного интеллекта. – М.:ДМК Пресс, 2004. – 312 с.:ил.
5. Кутковецкий В.Я. Дослідження операцій. - К.: Професіонал, 2004.

Поступила в редакцию 6 декабря 2005 г.