МАТЕМАТИКА

УДК 519.6

ПРИМЕНЕНИЕ АЛГОРИТМА А* ДЛЯ МИНИМИЗАЦИИ ПУТЕЙ В ЗАДАННОМ ГОМОЛОГИЧЕСКОМ КЛАССЕ

А.В. Галанин

Нижегородский госуниверситет им. Н.И. Лобачевского

al@galanin.nnov.ru

Поступила в редакцию 25.09.2013

Рассматриваются триангулированные замкнутые многообразия, реберные пути на них и группы гомологий по модулю 2. Разработана и практически опробована модификация для алгоритма поиска минимального пути, гомологичного заданному, основанная на использовании эвристического алгоритма А*.

Ключевые слова: симплекс, полиэдр, группа гомологий, алгоритм, минимизация, эвристика, алгоритм А*.

Постановка задачи и подходы к ее решению

В работе рассматриваются прямолинейные полиэдры в $\mathbf{R}^n$ и рёберные пути на них. Полиэдр предполагается двумерным, однако алгоритм может быть модифицирован для поиска минимального пути и в полиэдре размерности *m.* Используются группы гомологий с коэффициентами из поля $\mathbf{Z}_2$.

Дан полиэдр P, являющийся замкнутым многообразием, неотрицательная весовая функция $L: C_1(P) \to \mathbf{R}$, где вес ребра равен его длине в $\mathbf{R}^n$, и путь $x_0 \in C_1(P)$ с начальной вершиной s и конечной вершиной t. Требуется найти путь с минимальным весом среди всех путей с концами s и t, гомологичных x_0.

Для решения этой задачи в [1] разработан следующий алгоритм.

АЛГОРИТМ 1

1. Нахождение базиса $[y_1],\ldots,[y_r]$ группы гомологий $H_1(P)$.

2. Построение индексной вектор-функции $J: C_1(P) \to \mathbf{Z}_2^r$ относительно найденного базиса $[y_1],\ldots,[y_r]$.

3. Построение накрывающего полиэдра $\hat{P}$ с множеством вершин $\hat{V} = V \times \mathbf{Z}_2^r$ и множеством ребер $\hat{E} = \{[\hat{u}\hat{v}] = [(u,\xi)(v,\eta)] \,|\, [uv] \in E, \eta = \xi + J([uv])\}$, где V и E — наборы вершин и ребер исходного полиэдра P, а также проекции $\pi: \hat{P} \to P$ и весовой функции

$$\hat{L} = L \circ \pi : C_1(\hat{P}) \to \mathbf{R}.$$

4. Применение алгоритма Дейкстры (см. [2]) для поиска пути с минимальным весом на накрывающем полиэдре. При этом в качестве начальной точки на $\hat{P}$ выбирается пара $(s,0)$, а в качестве конечной – $(t, J(x_0))$.

5. Вычисление проекции найденного пути на P.

Наиболее трудоемким в алгоритме 1 является шаг 4, поскольку $\hat{P}$ содержит во много раз больше вершин и ребер, чем исходный полиэдр P. Поэтому представляется желательным использование любой возможности для ускорения выполнения этой процедуры.

Модификация алгоритма поиска минимального пути

Один из путей уменьшения перебора состоит в том, чтобы рассматривать в первую очередь варианты, являющиеся наиболее оптимальными в соответствии с эвристической информацией.

Поиск минимальных путей, соединяющих заданные вершины полиэдра, можно в некоторых случаях ускорить, использовав дополнительную информацию о задаче, например – информацию о том, как расположены вершины исходного полиэдра в пространстве. Для этого применим алгоритм А* (см. [2]).

В алгоритме А* каждой вершине графа, по которому ведётся минимизация, сопоставляется функция $f(v) = g(v) + h(v)$, где $g(v)$ – значение минимизируемой функции в вершине v, а $h(v)$ – оценка снизу минимизируемой функции между вершинами v и t.

Выберем в качестве $g(v)$ длину кратчайшего пути от вершины s до вершины v, а в качестве $h(v)$ – какую-либо функцию, не переоценивающую расстояние между текущей вершиной v и конечной вершиной t.

Также на вершины, помещаемые в очередь, можно наложить дополнительное ограничение $f(v) \le M$, где M – длина исходного пути перед минимизацией. Подобное отсечение не повлияет на допустимость алгоритма А*, так как путь, имеющий длину более начального, заведомо не оптимален.

Как и всякий эвристический алгоритм, алгоритм А* не гарантирует того, что вершины будут рассмотрены в оптимальном порядке, поэтому можно подобрать примеры, в которых А* будет работать много дольше алгоритма Дейкстры. Но для большинства рассмотренных моделей модифицированный алгоритм показал 30–40-кратный прирост в скорости работы и 6–9-кратное снижение количества итераций.

К сожалению, прямое применение такого подхода к шагу 4 алгоритма 1 невозможно. Дело в том, что на шаге 3 строится только симплициальная схема накрывающего полиэдра $\hat{P}$. Ни о какой реализации этой абстрактной схемы в пространстве нет и речи. Поэтому необходимо сначала построить оценочную функцию $h(\hat{v})$ на $\hat{V} = V \times \mathbf{Z}_2^r$, для чего требуется добавить в алгоритм 1 дополнительный шаг.

В предлагаемом нами модифицированном алгоритме шаг 4 алгоритма 1 заменяется на следующие действия.

4.1. Нахождение оценки минимальных расстояний $h(v)$ от произвольных точек $v \in V$ до конечной точки t пути x_0 на одномерном остове полиэра P. Это может быть расстояние в декартовом пространстве или кратчайшее расстояние, найденное с помощью алгоритма Дейкстры (заметим, что в этом месте алгоритм Дейкстры применяется до перехода к накрытию).

4.2. Поиск с применением алгоритма А* пути с минимальным весом на одномерном остове накрывающего полиэдра $\hat{P}$, соединяющего вершины $\hat{s} = (s, 0)$ и $\hat{t} = (t, J(x_0))$. При этом в качестве оценочной функции может быть использована композиция $\hat{h} = h \circ \pi$.

Применение функции $\hat{h}$ в алгоритме А* на $\hat{P}$ корректно, так как для любой вершины $\hat{v} = (v, \eta) \in \hat{V}$ и минимального пути $\hat{x}$ из $\hat{v}$ в $\hat{t}$ путь $x = \pi(\hat{x})$ имеет начало $v = \pi(\hat{v})$ и конец $t = \pi(\hat{t})$, и потому $\hat{L}(\hat{x}) = L(x) \ge h(v) = \hat{h}(\hat{v})$.

Поскольку шаг 4.2 является существенно новым, опишем его подробнее.

```
procedure search_A_star (ŝ, t̂)
  for each v̂ ∈ Ĝ
      visited[v̂] ← false
      d[v̂] ← ∞
  ENQUEUE(Q, ŝ, ĥ(ŝ, t̂))
  d[ŝ] ← 0
  from[ŝ] ← ŝ
  while Q ≠ ∅
      v ← POP(Q)
      if visited[ v̂ ] then
            continue
      visited[v̂] ← true
      for each ŵ ∈ Adj[v̂]
            path_weight ← d[v̂] + L(v̂, ŵ)
            if path_weight < d[ŵ] then
                  ENQUEUE(Q, ŵ, path_weight + ĥ(v̂))
                  d[ŵ] ← path_weight
                  from[ŵ] ← v̂
```

Функция search_A_star ($\hat{s}$, $\hat{t}$) для вершины $\hat{s}$ находит кратчайший путь до конечной вершины $\hat{t}$ с помощью алгоритма А*. Граф считается заданным списком смежности *Adj*.

В случае если во время работы алгоритма попалась ещё не рассмотренная вершина $\hat{v}$, она добавляется в очередь с приоритетом Q на проверку. Приоритет вершины в очереди вычисляется с помощью функции $\hat{f}(\hat{v}) = \hat{g}(\hat{v}) + \hat{h}(\hat{v})$, где $\hat{g}(\hat{v})$ – длина кратчайшего пути от вершины $\hat{s}$ до $\hat{v}$. Элементы очереди Q рассматриваются в порядке возрастания приоритета.

В массиве *visited* хранятся булевские значения, обозначающие посещённые вершины, массив *d* хранит текущее минимальное значение весовой функции на пути до вершины $\hat{v}$, массив *from* используется для сохранения предыдущей вершины в графе предшествования.

Сравнение производительности алгоритмов поиска

В рамках работы было проведено сравнение модифицированного алгоритма с оригинальным. На различных моделях полиэдров выбирался рёберный путь, и к нему применялся алгоритм минимизации, описанный в [1], и алгоритм, описанный выше. В качестве $h(v)$ выбрано расстояние в декартовом пространстве между текущей вершиной v и конечной вершиной t. Такая оценка может быть использована в алгоритме A*, так как длина любого пути между вершинами v и t не может быть меньше, чем расстояние между вершинами.

Таблица

Таблица времени исполнения алгоритмов и количества итераций при различных начальных условиях (в скобках приведено количество рёбер в исходном пути)

Данные	Оригинальный		Модификация	
	Время	Шаги	Время	Шаги
Тор (19)	0.03 с	735	0.05 с	226
Узел (58)	0.18 с	2599	0.34 с	2346
Лестница (44)	19.108 с	24937	2.42 с	7485
Лестница (50)	12.85 мин	151253	1.91 мин	42231
Лестница (54)	8.35 мин	118308	23.05 с	19048
Лестница 2 (89)	4.13 час	712062	5.76 мин	76257

Модифицированный алгоритм показал примерно 30–40-кратный прирост в скорости работы и 6–9-кратное снижение количества итераций. Точные результаты сравнения времени исполнения и количества итераций рассматриваемых алгоритмов приведены в таблице.

Выводы

Предложена модификация алгоритма поиска кратчайшего пути с заданным индексом, основанная на применении эвристической оценочной функции, зависящей от расстояния в пространстве между текущей и конечной вершинами. Проведено сравнение с оригинальным алгоритмом, показавшее возможность применения модифицированного метода, дающего на некоторых начальных условиях прирост по скорости работы в 30–40 раз, а по количеству итераций в 6–9 раз. В наихудшем рассмотренном случае затраченное время превышает время оригинального алгоритма в 2 раза, а количество итераций остаётся сравнимым.

Работа выполнена при финансовой поддержке Минобрнауки РФ в рамках государственного задания на оказание услуг в 2012–2014 гг. подведомственными высшими учебными заведениями (шифр заявки 1.1907.2011).

Список литературы

1. Lapteva A.V. and Yakovlev E.I. Index Vector-Function and Minimal Cycles // Lobachevskii Journal of Mathematics. 2006. Vol. 22. P. 35–46.
2. Лорьер Ж.-Л. Системы искусственного интеллекта / Пер. с фр. и ред. В.Л. Стефанюка. М.: Мир, 1991.

APPLICATION OF THE ALGORITHM A* TO PATH MINIMIZATION IN A GIVEN HOMOLOGY CLASS

A.V. Galanin

Triangulated closed manifolds, their edge paths and modulo 2 homology groups are considered. A modification to the search algorithm for the shortest path (homologous to a given one) based on the use of a heuristic algorithm A* has been developed and practically tested.

Keywords: simplex, polyhedron, homology group, algorithm, minimization, heuristics, algorithm A*.

References

1. Lapteva A.V. and Yakovlev E.I. Index Vector-Function and Minimal Cycles // Lobchevskii Journal of Mathematics. 2006. Vol. 22. P. 35–46.
2. Lor'er Zh.-L. Sistemy iskusstvennogo intellekta / Per. s fr. i red. V.L. Stefanjuka. M.: Mir, 1991.