УДК 519.7 DOI 10.17223/2226308X/12/52

ПРИБЛИЖЁННЫЙ АЛГОРИТМ ПОИСКА ОПТИМАЛЬНОГО МАРШРУТА В СЕТИ С ОГРАНИЧЕНИЕМ

А. А. Солдатенко

Предлагается приближённый алгоритм RevTree решения NP-трудной задачи RCSP (Resource Constrained Shortest Path). Задача RCSP является расширением задачи о кратчайшем пути в ориентированном графе $G = (V, E)$, когда для каждой дуги $e \in E$ кроме основной весовой функции $w(e)$ дополнительно задаются функции $r_i(e)$, $i = 1, \ldots, k$, численно отражающие потребности в ресурсах, которые необходимы для передвижения по этой дуге. Задача RCSP возникает при проектировании и эксплуатации компьютерных и мультисервисных сетей. Показано, что алгоритм RevTree всегда находит допустимое решение задачи RCSP, если оно существует, за полиномиальное время, отклоняясь от оптимального решения на величину, зависящую от значений $w(e)$ и $r_i(e)$, $e \in E$.

Ключевые слова: *ресурсоограниченный кратчайший путь, алгоритмы на графах, оптимальная маршрутизация, компьютерные и мультисервисные сети.*

Введение

Задача RCSP традиционно формулируется в терминах теории графов и целочисленного линейного программирования. В теоретико-графовой формулировке рассматриваемая сеть представляется ориентированным графом, вершины которого соответствуют узлам сети, а дуги — каналам связи. Предполагается, что всякая дуга обладает основным весом, например стоимостью, а также некоторыми дополнительными весами, отражающими потребности в ресурсах, которые нужны для передвижения по этой дуге. Требуется найти кратчайший путь между двумя заданными узлами сети, который удовлетворял бы заданным ограничениям на итоговые ресурсные затраты, необходимые для прохождения этого пути. Известно, что даже при одном ограничении задача RCSP является NP-трудной [1].

В настоящее время выделяют три класса методов и соответствующих им алгоритмов, способных находить точное или приближённое решение задачи RCSP: 1) методы ранжирования путей [2], 2) методы маркировки вершин [1, 3, 4] и 3) методы лагранжевой релаксации [5, 6]. Первые два класса основаны на теоретико-графовой постановке задачи, в то время как методы третьего класса исходят из постановки задачи RCSP на языке целочисленного линейного программирования. К сожалению, большинство существующих алгоритмов решения задачи RCSP медленно работают на сетях большой размерности, для многих современных приложений совершенно неприемлемы временные характеристики данных алгоритмов и их многочисленных усовершенствованных версий [3]. Поэтому остаётся актуальной разработка приближённых алгоритмов решения задачи RCSP, способных быстро находить решение на сетях большой размерности [3, 6].

1. Теоретико-графовая постановка задачи

Пусть сеть описана взвешенным ориентированным графом (далее — просто графом) $G = (V, E)$ без кратных дуг и петель, в котором каждая вершина $v \in V$ представляет узел сети, а каждая дуга $e \in E$ — канал связи между соответствующими узлами сети, при этом $n = |V|$ и $m = |E|$. Считаем, что на множестве дуг графа $G = (V, E)$ задана функция $w(e)\colon E \to \mathbb{R}^+$, ставящая в соответствие каждой дуге $e \in E$ её вес

$w(e) > 0$. Пусть для вершин $s, d \in V$ в графе G существует путь P, идущий от вершины s к вершине d. Полагаем, что вес этого (s,d)-пути P вычисляется как сумма весов всех входящих в него дуг:

$$w(P) = \sum_{e \in P} w(e), \tag{1}$$

т. е. функция $w(e)$ является аддитивной. Если $w(e)$ — стоимость передвижения по дуге e, то $w(P)$ можно интерпретировать как стоимость прохождения (s,d)-пути P в графе G.

Пусть для каждой дуги графа G также заданы функции $r_i(e)\colon E \to \mathbb{R}^+$, $i = 1, \ldots, k$, отражающие ресурсные потребности, которые необходимы для передвижения по этой дуге, и всегда $r_i(e) > 0$. Предполагается, что все эти функции аддитивные, т. е. для любого (s,d)-пути P верны равенства

$$r_i(P) = \sum_{e \in P} r_i(e), \ \ i = 1, \ldots, k. \tag{2}$$

Кроме того, определены величины $R_i \in \mathbb{R}^+$, $i = 1, \ldots, k$, задающие ресурсные ограничения рассматриваемой мультисервисной сети. Всякий (s,d)-путь P называется допустимым, если он удовлетворяет ресурсным ограничениям:

$$r_i(P) \leqslant R_i, \ \ i = 1, \ldots, k. \tag{3}$$

Требуется найти (s,d)-путь P, который минимизирует целевую функцию (1) и удовлетворяет ограничениям (3). Такой путь определяет оптимальное решение задачи RCSP и называется ресурсноограниченным кратчайшим (s,d)-путём в графе $G = (V,E)$. Вес этого пути обозначим через OPT. Всякий допустимый (s,d)-путь можно рассматривать в качестве приближённого решения данной задачи. В [7] предложен эвристический алгоритм поиска приближённого решения. В настоящей работе предлагается алгоритм RevTree, который находит допустимое решение задачи RCSP за полиномиальное время, отклоняясь от оптимального решения на величину, зависящую от значений $w(e)$ и $r_i(e)$, $e \in E$, т. е. получены оценки его сложности и точности.

2. Описание алгоритма RevTree

Под размерностью задачи RCSP понимаются значения n и m — число вершин и дуг графа $G = (V,E)$ соответственно, а под её параметрами — значения функций $w(e), r_i(e)$. Для краткости вместо $r_i(e)$ и R_i будем писать $r(e)$ и R. Опишем алгоритм RevTree для случая, когда имеется только один ресурс R_i и $w(e), r_i(e)$ — положительные вещественнозначные функции.

Алгоритм RevTree состоит из двух фаз, на каждой из которых однократно выполняется известный алгоритм Дейкстры [8]. На первой фазе, исходя из функций $r(e)$, $e \in E$, вычисляется дерево путей минимального веса с корнем в вершине d. Это дерево определяет для каждой вершины $u \in V$ такой (u,d)-путь, вес которого указывает минимальный ресурс для прохождения (u,d)-пути. Обозначим (u,d)-путь через P_2.

Пусть P_1 — путь из стартовой вершины s в текущую вершину v, найденный как некоторое допустимое решение задачи RCSP для вершин s и v. Согласно формуле (2), для прохождения этого (s,v)-пути затрачен ресурс величины $r(P_1)$. Множество $\Gamma(v)$ определяет возможные направления дальнейшего движения по дугам графа G из вершины v. Тогда для перемещения из вершины v в вершину $u \in \Gamma(v)$ необходимо выполнение условия

$$r(P_1) + r(v,u) + \pi(u) \leqslant R, \tag{4}$$

где $\pi(u) = \sum\limits_{e \in P_2} r(e)$. Условие (4) гарантирует, что путь (P_1, e, P_2), где $e = (v, u) \in E$, является допустимым решением задачи RCSP.

На второй фазе вновь выполняется алгоритм Дейкстры, но только с усечёнными окрестностями вершин. Усечение окрестности $\Gamma(v)$ для текущей вершины $v \in V$ осуществляется следующим образом: если для $u \in \Gamma(v)$ нарушается условие (4), то она удаляется из $\Gamma(v)$. Таким образом, алгоритм RevTree уменьшает мощность множеств $\Gamma(v)$ с учётом ресурсного ограничения R, что позволяет находить допустимое решение задачи RCSP, если оно существует (алгоритм 1).

Алгоритм 1. RevTree

Вход: граф $G = (V, E)$, $w(e)$, $r(e)$ для всех $e \in E$, величина R, вершины $s, d \in V$.
Выход: значение веса и последовательность вершин допустимого (s, d)-пути.

И н и ц и а л и з а ц и я
$w[s] := 0$; $r[s] := 0$; $p[s] := \texttt{null}$; $Passed := \varnothing$; $\pi_r[d] := 0$.
П е р в а я ф а з а
Для всех $v \in V \setminus \{d\}$
  $\pi_r[v] := \infty$.
Пока $\exists u \notin Passed$
  выбираем $u \notin Passed$ с минимальным $\pi_r[u]$; $Passed := Passed \cup \{u\}$.
  Для всех $v \notin Passed$ & $e = (u, v) \in E$:
    Если $\pi_r[v] > \pi_r[u] + r(e)$, **то**
      $\pi_r[v] := \pi_r[u] + r(e)$.
$Passed := \varnothing$.
В т о р а я ф а з а
Для всех $v \in V \setminus \{s\}$
  $w[v] := \infty$; $p[v] := \texttt{null}$; $r[v] := 0$.
Пока $\exists u \notin Passed$
  выбираем $u \notin Passed$ с минимальным $w[u]$ и $r[u] + \pi_r[v] < R$;
  $Passed := Passed \cup \{u\}$.
  Для всех $v \notin Passed$ & $e = (u, v) \in E$:
    Если $w[v] > w[u] + w(e)$, **то**
      $w[v] := w[u] + w(e)$; $r[v] := r[u] + r(e)$; $p[v] := u$.

3. Оценки сложности и точности алгоритма RevTree

Известно, что алгоритм Дейкстры требует $\mathcal{O}(n^2)$ времени и $\mathcal{O}(n)$ памяти [8]. На первой фазе с помощью алгоритма Дейкстры находится дерево путей минимального веса с корнем в вершине d, для сохранения которого необходимо $\mathcal{O}(n)$ памяти. На второй фазе в алгоритм Дейкстры добавляется проверка условия (4), которая не влияет на оценку вычислительной сложности. Поскольку обе фазы алгоритма RevTree выполняются последовательно, в целом для нахождения решения задачи RCSP затрачивается $\mathcal{O}(n^2)$ времени и $\mathcal{O}(n)$ памяти.

Оценим точность приближённого решения, формируемого алгоритмом RevTree. Для этого вычислим $\lambda_{\min} = \min\limits_{e \in E} (r(e)/w(e)) > 0$, $\lambda_{\max} = \max\limits_{e \in E} (r(e)/w(e)) > 0$. Заметим, что функции $\lambda_{\max}, \lambda_{\min}$ определены для любого $e \in E$, поскольку в формулировке задачи RCSP предполагается, что $w(e) > 0$ и $r(e) > 0$ для всех $e \in E$.

Пусть алгоритм RevTree на некотором шаге нашёл путь P_1, идущий из стартовой вершины s в текущую вершину v, как оптимальное решение задачи RCSP для вершин s и v. Если вершина v совпадает с целевой вершиной d, то найдено оптимальное решение исходной задачи. Если вершина v совпадает с s, то путь P_1 содержит пустое множество дуг и для него $w(P_1) = r(P_1) = 0$. В общем случае вершина v определяет начало ещё не пройденной части искомого (s, d)-пути. Рассмотрим неусечённую окрестность $\Gamma(v)$ текущей вершины v как множество концов дуг, исходящих из вершины v и обладающих временными метками. Пусть $x \in \Gamma(v)$ — вершина, для которой $w(v, x) = \min\limits_{u \in \Gamma(v)} w(v, u)$, но не выполняется условие (4), т. е. верно соотношение

$$r(P_1) + r(v, x) + \pi(x) > R. \tag{5}$$

Именно эту вершину выбирает алгоритм Дейкстры, если окрестность $\Gamma(v)$ не подверглась усечению. При выборе вершины $u \in \Gamma(v)$, удовлетворяющей условию (4), алгоритм RevTree отклоняется от кратчайшего пути в смысле (1) без учёта ресурсного ограничения (3). Обозначим (v, d)-путь, найденный с помощью неусечённых окрестностей, как P^*_{rest}, а с помощью усечённых окрестностей — P_{rest}. Для путей (P_1, P_{rest}) и (P_1, P^*_{rest}) справедливы следующие неравенства:

$$w(P_1) + w(P^*_{\text{rest}}) \leqslant \text{OPT} \leqslant w(P_1) + w(P_{\text{rest}}). \tag{6}$$

В (6) величина $w(P_1)+w(P^*_{\text{rest}})$ определяет значение целевой функции для кратчайшего пути без учёта ресурсных ограничений, а $w(P_1)+w(P_{\text{rest}})$ — значение целевой функции пути, найденного алгоритмом RevTree. Предполагается, что решение задачи RCSP существует. Следовательно, найдётся хотя бы один допустимый (s, d)-путь, а значит, всегда существует определённый выше путь P_{rest}. Для прохождения всякого (v, d)-пути доступен ресурс $R_{\text{rest}} = R - r(P_1)$. Согласно условию (4), справедливо неравенство $R_{\text{rest}} \geqslant \pi(v)$. Исходя из определения функции $\lambda(e)$, для всякой дуги $e \in E$ имеют место соотношения

$$0 < \lambda_{\min} w(e) \leqslant r(e) \leqslant \lambda_{\max} w(e).$$

Поскольку функции $w(e)$ и $r(e)$ аддитивные, получим подобные соотношения для любого (v, d)-пути, в том числе P^*_{rest} и P_{rest}:

$$0 < \lambda_{\min} w(P^*_{\text{rest}}) \leqslant r(P^*_{\text{rest}}) \leqslant \lambda_{\max} w(P^*_{rest}); \tag{7}$$

$$0 < \lambda_{\min} w(P_{\text{rest}}) \leqslant r(P_{\text{rest}}) \leqslant \lambda_{\max} w(P_{\text{rest}}). \tag{8}$$

Путь P^*_{rest} по построению проходит через вершину x, поэтому с учётом (5) верны соотношения

$$r(P^*_{\text{rest}}) \geqslant r(v, x) + \pi(x) > R - r(P_1) = R_{\text{rest}}.$$

Здесь $\pi(x)$ — минимально необходимый ресурс для передвижения из вершины x в вершину d. Следовательно, $r(P^*_{\text{rest}}) > R_{\text{rest}}$. Отсюда с учётом (7) получим

$$R_{\text{rest}}/\lambda_{\max} < w(P^*_{\text{rest}}). \tag{9}$$

Из соотношений (8) имеем неравенство $w(P_{\text{rest}}) \leqslant r(P_{\text{rest}})/\lambda_{\min}$. Поскольку всегда $w(P^*_{\text{rest}}) \leqslant w(P_{\text{rest}})$ и $r(P_{\text{rest}}) \leqslant R_{\text{rest}}$, то

$$w(P^*_{\text{rest}}) \leqslant r(P_{\text{rest}})/\lambda_{\min} \leqslant R_{\text{rest}}/\lambda_{\min}. \tag{10}$$

Из (9) и (10) следует

$$R_{\text{rest}}/\lambda_{\max} < w(P^*_{\text{rest}}) \leqslant R_{\text{rest}}/\lambda_{\min}. \tag{11}$$

Оценим решение, найденное алгоритмом RevTree, исходя из неравенств (6). В результате получим

$$\frac{w(P_1) + w(P_{\text{rest}}) - \text{OPT}}{\text{OPT}} \leqslant \frac{w(P_{\text{rest}}) - w(P^*_{\text{rest}})}{w(P_1) + w(P^*_{\text{rest}})}. \tag{12}$$

Наибольшее отклонение найденного решения от оптимального достигается при $v = s$ и $w(P_{\text{rest}}) = R_{\text{rest}}/\lambda_{\min}$. Тогда неравенство (12) принимает вид

$$\frac{R_{\text{rest}}/\lambda_{\min} - \text{OPT}}{\text{OPT}} \leqslant \frac{R_{\text{rest}}/\lambda_{\min} - w(P^*_{\text{rest}})}{w(P^*_{\text{rest}})}.$$

Согласно (11), справедливо $R_{\text{rest}}/\lambda_{\max} < w(P^*_{\text{rest}})$. Отсюда окончательно имеем

$$\frac{R_{\text{rest}}/\lambda_{\min} - \text{OPT}}{\text{OPT}} \leqslant \frac{R_{\text{rest}}/\lambda_{\min} - R_{\text{rest}}/\lambda_{\max}}{R_{\text{rest}}/\lambda_{\max}} = \frac{\lambda_{\max}}{\lambda_{\min}} - 1 = \varepsilon.$$

Таким образом, допустимое решение, найденное алгоритмом RevTree, отклоняется от оптимального решения задачи RCSP не более чем на величину $\varepsilon = \lambda_{\max}/\lambda_{\min} - 1$.

4. Вычислительные эксперименты

Для оценки результативности алгоритма RevTree проведены вычислительные эксперименты на компьютере с процессором Intel Core i7-7700K Processor (8 MB Cache, 3,60 ГГц) и ОЗУ объёмом 16 Гбайт. Осуществлялось сравнение программной реализации алгоритма RevTree и пакета IBM ILOG CPLEX [9] по времени работы, числу выполненных запросов, точности найденного решения. Заметим, что обе программы находят решение задачи RCSP, если множество допустимых решений не пусто, при этом CPLEX находит оптимальное (точное) решение. Эксперименты проводились на случайно сгенерированных графах G_1–G_3 для последовательности из 1000 случайно сгенерированных (s, d)-запросов (табл. 1). Для случайной генерации графов применялся метод Ваксмена с параметрами $\alpha = 0{,}15$, $\beta = 0{,}25$, который традиционно используется для генерации графов, топологически схожих с реальными компьютерными сетями [10]. Запрос считался выполненным, если для него было найдено допустимое решение задачи RCSP. Результаты экспериментов представлены в табл. 2. Согласно этим результатам, алгоритм RevTree для рассматриваемых графов находит столько же допустимых решений, что и CPLEX, при этом они совпадают и являются оптимальными. Однако во всех случаях алгоритм RevTree работает значительно быстрее пакета программ CPLEX.

Таблица 1

Размерность и параметры задачи RCSP

Название графа	n — число вершин	m — число дуг	$\lambda_{\max}$	$\lambda_{\min}$	ε
G_1	500	3923	0,75	0,6	0,25
G_2	1000	16345	0,75	0,6	0,25
G_3	1500	36655	0,75	0,6	0,25

Таблица 2

Результаты сравнения алгоритма RevTree и пакета CPLEX

Название графа	CPLEX		RevTree		
	Время обработки серии запросов, с	Число выполненных запросов	Время обработки серии запросов, с	Число выполненных запросов, из них для q найдено приближённое решение	
				Всего	q
G_1	428,265	910	2,69	910	0
G_2	1607,58	967	47,524	967	0
G_3	7571,22	676	94,217	676	0

Заключение

Предложен полиномиальный приближённый алгоритм RevTree решения задачи RCSP при наличии только одного ресурса. Алгоритм имеет сложность по времени $\mathcal{O}(n^2)$ и всегда находит допустимое решение задачи, если оно существует, отклоняясь от оптимального решения на величину, не превышающую $\varepsilon = \lambda_{\max}/\lambda_{\min} - 1$, зависящую от параметров задачи RCSP. Эксперименты подтверждают результативность алгоритма RevTree и положений, высказанных в п. 3. Целесообразны дальнейшие исследования специальной маркировки вершин, заложенной в алгоритме RevTree.

ЛИТЕРАТУРА

1. *Joksch H. C.* The shortest route problem with constraints // J. Math. Analysis Appl. 1966. V. 14. P. 191–197.
2. *Di Puglia Pugliese L. and Guerriero F.* A survey of resource constrained shortest path-problems: exact solution approaches // J. Networks. 2013. V. 62. Iss. 3. P. 183–200.
3. *Zhu X. and Wilhelm W. E.* A three-stage approach for the resource-constrained shortest path as a sub-problem in column generation // J. Computers & Operations Research. 2012. V. 39. Iss. 2. P. 164–178.
4. *Dumitrescu I. and Boland N.* Improved preprocessing, labeling and scaling algorithms for the weight-constrained shortest path problem // J. Networks. 2003. V. 42. P. 135–153.
5. *Jepsen M., Petersen B., Spoorendonk S., and Pisinger D.* A branch-and-cut algorithm for the capacitated profitable tour problem // J. Discrete Optimization. 2014. V. 14. P. 78–96.
6. *Horvath M. and Kis T.* Solving resource constrained shortest path problems with LP-based methods // J. Computers & Operations Research. 2016. V. 73. P. 150–164.
7. *Солдатенко А. А.* Алгоритм оптимальной маршрутизации в мультисервисных телекоммуникационных сетях // Прикладная дискретная математика. Приложение. 2018. № 11. С. 122–127.
8. *Кормен Т. Х., Лейзерсон Ч. И., Ривест Р. Л., Штайн К.* Алгоритмы. Построение и анализ. М.: Вильямс, 2018. 1328 с.
9. `https://www.ibm.com/support/knowledgecenter/SSSA5P_12.6.2/ilog.odms.studio.help/pdf/usrcplex.pdf` — IBM Corp.: IBM ILOG CPLEX Optimizer Studio. CPLEX User's Manual. Version 12 Release 6, 2015.
10. *Pathan A. K., Monowar M. M., and Khan S.* Simulation Technologies in Networking and Communications: Selecting the Best Tool for the Test. CRC Press, 2017. 648 p.