基于最短增广链的最大流改进算法

赵礼峰,纪亚劲
(南京邮电大学 理学院,江苏 南京 210023)

摘 要:网络最大流是经典的组合优化问题,它的经典算法主要有三种,分别是 Ford-Fulkerson 算法、最短增广链算法(Dinic 算法)和预流推进算法。Ford-Fulkerson 算法中由于增广链的选取任意性而有时无法得到理想的最大流。最短增广链算法在分层剩余网络中寻找最短增广链,从而避免了增广链选取的任意性。但最短增广链算法在求解最大流过程中每次增广都需要重新寻找最短增广链,利用率不高。针对这一问题,提出了一种修复最短增广链的新算法。该算法在沿最短增广链调整流量之后,删除最短增广链流量为零的弧,且寻找合适的路径修复最短增广链,从而提高了最短增广链的使用效率,减少了最短增广链的搜索次数。应用新算法进行了 BA 无标度网络建模仿真。实验结果表明,该算法运行效率要高于最短增广链算法。

关键词:最大流;分层剩余网络;最短增广链;BA 无标度网络
中图分类号:TP301.6 **文献标识码**:A **文章编号**:1673-629X(2017)08-0088-04
doi:10.3969/j.issn.1673-629X.2017.08.018

Improved Algorithm of Maximum Flow with Shortest Augmenting Chain

ZHAO Li-feng,JI Ya-jin
(College of Science,Nanjing University of Posts and Telecommunications,Nanjing 210023,China)

Abstract:The maximum network flow is a classic combinatorial optimization problem,which mainly consists of Ford-Fulkerson algorithm,the shortest augmenting chain algorithm (Dinic algorithm) and preflow push algorithm. The desired maximum flow from Ford-Fulkerson algorithm could not be acquired because of the arbitrariness when choosing the augmented chain. The shortest augmenting chain algorithm can find the shortest augmenting chain in the remaining layered network to avoid the augmented chain selected arbitrary,however,it needs to search again shortest augmenting chain in maximum flow when augmenting with low using rate. Aimed at this problem,a new shortest augmenting chain repair algorithm is presented. After it has adjusted flow along the shortest augmenting chain the arc of zero flow on the augmented chain has been removed to retain the arc that the flow zero,which select the appropriate nodes to repair shortest augmenting chain in the remaining nodes for improving the efficiency and reducing the times of search shortest augmenting chain. The improved algorithm is verified through the modeling and simulation experimental in BA scale-free network,which shows that its efficiency is higher than the shortest augmenting chain algorithm.

Key words:maximum flow;remaining layered network;shortest augmenting chain;BA scale-free network

0 引 言

网络最大流问题是图论中极其重要的分支,是经典的组合优化问题,也可以看成特殊的线性规划问题[1]。它在运筹学、计算机、工程等众多科学领域中有着广泛的应用[2-3],例如,运输问题、分派问题、通信问题等都可以转化为网络最大流模型来解决。因此,研究网络最大流算法具有很重要的意义。

至今为止,网络最大流问题的研究已经有 50 多年的历史,现已建立了较为完善的理论并且提出了一系列经典算法。如 1956 年提出的 Ford-Fulkerson 算法[4],随后 Dinic,Edmonds 和 Karp 对 Ford-Fulkerson 算法进行了改进,提出了最短增广链算法[5-6]。该算法提出了分层剩余网络的概念,其主要思想是每次都是沿着最短增广链进行增广。1986 年,Karzanov 提出

收稿日期:2016-08-18 **修回日期**:2016-11-23 **网络出版时间**:2017-07-05
基金项目:国家自然科学基金青年基金项目(61304169)
作者简介:赵礼峰(1959-),男,教授,硕士研究生导师,研究方向为图论及应用、矩阵论;纪亚劲(1991-),男,硕士研究生,研究方向为图论及其在通信中的应用。
网络出版地址:http://kns.cnki.net/kcms/detail/61.1450.TP.20170705.1650.040.html

了预留推进算法[7],Goldberg 和 Tarjan 对此进行了深入研究并提出了一系列的改进算法[8-9];另外,还有大量的学者针对一些特殊网络提出了自己的算法[10-16],这些算法是如今研究大规模网络的基础。

Ford-Fulkerson 算法的优点是适用度广,但是每次都是任意寻找增广链,使得算法复杂度偏高;最短增广链算法在 Ford-Fulkerson 算法的基础上改进很多,即沿着最短增广链进行增广,在很大程度上降低了复杂度。但是算法仍存在缺陷,由于每次沿最短增广链增广之后需重新寻找最短增广链,所以利用率不高。

针对上述不足,提出了一种新的改进的最短增广链算法。该算法通过一种方法来修复最短增广链[17],避免了反复重新寻找新的最短增广链,以提高算法效率。

1　基本概念

1.1　最大流的数学模型

给定一个容量网络 $G=(V,A,c)$,其中 V 是顶点集,A 是弧集,c 是弧的容量,$f(a)(a\in A)$ 称为通过弧 a 的流量。在网络 G 中定义两个顶点 v_s 和 v_t,v_s 为 G 的发点,v_t 为 G 的收点。

网络最大流模型:

$$\max \sum_{v_j\in N^+(v_s)} f_{sj}-\sum_{v_j\in N^-(v_s)} f_{js}$$

$$\text{s.t.}\begin{cases}\sum\limits_{v_j\in N^+(v_i)} f_{ij}-\sum\limits_{v_j\in N^-(v_i)} f_{ji}=\begin{cases}f_{st}, i=s\\0, i\in V\backslash\{v_s,v_t\}\\-f_{st}, i=t\end{cases}\\0\leqslant f_{ij}\leqslant c_{ij},\forall (v_i,v_j)\in A\end{cases}$$

1.2　分层剩余网络

对于一个容量网络 $G=(V,A,c)$ 及 G 上的可行流 f,令

$$A_+(f)=\{(v_i,v_j)\mid (v_i,v_j)\in A, f_{ij}<c_{ij}\}$$

$$A_-(f)=\{(v_i,v_j)\mid (v_j,v_i)\in A, f_{ji}>0\}$$

称 $A_+(f)$ 为前向弧集,$A_-(f)$ 为后向弧集,且记 $A_+(f)\cup A_-(f)=A(f)$,令

$$c_{ij}(f)=\begin{cases}c_{ij}-f_{ij}, (v_i,v_j)\in A_+(f)\\f_{ji}, (v_i,v_j)\in A_-(f)\end{cases}$$

称 $c_{ij}(f)$ 为弧 (v_i,v_j) 关于 f 的剩余容量。

定义 1:由 V,$A(f)$ 和 $c(f)$ 组成的网络 $G(f)=(V,A(f),c(f))$ 称为网络 G 关于 f 的剩余网络。

定义 2:对于剩余网络 $G(f)=(V,A(f),c(f))$,规定关于 $G(f)$ 的子网络 $\mathrm{AG}(f)=(V'(f),A'(f),c(f))$,如下:

$$V'(f)=\{v_t\}\cup\{v_i\in V\mid h(v_i)<h(v_t)\}$$

$$A'(f)=\{(v_i,v_j)\in A(f)\mid h(v_j)=h(v_i)+1<h(v_t)\}\cup\{(v_i,v_j)\in A(f)\mid h(v_i)=h(v_t)-1\}$$

则 $\mathrm{AG}(f)$ 称为 G 的关于 f 的分层剩余网络[18]。

2　一种改进的最大流算法

2.1　算法思想

首先从容量网络 D 的任一个可行流 f_1(例如零流)开始,构造 G 的关于 f_1 的分层剩余网络 $\mathrm{AG}(f_1)$,在 $\mathrm{AG}(f_1)$ 中使用深度优先搜索算法选取一条 (v_s,v_t) 路径 P_1,沿 P_1 对 f_1 进行增广,并相应修改 P_1 上的容量,在 $\mathrm{AG}(f_1)$ 中删去 P_1 上容量为零的弧,且同时在原网络中删去相应的弧,然后对 P_1 进行修复,修复的方法是:从发点 v_s 和收点 v_t 出发,沿 P_1 向中间逐点遍历,若遇断点便停止遍历并分别记为断点 v_i 和 v_j,考察在 $\mathrm{AG}(f_1)$ 中是否存在从 v_i 到 v_j 的路径,若存在,则修复最短增广链 P_1,继续沿 P_1 对 f_1 进行增广,直至不能修复,如图 1 所示。之后在 $\mathrm{AG}(f_1)$ 中重新选取增广链 P_2,重复上述操作。经过有限次增广,使得余下网络不再有 (v_s,v_t) 路径,从而得到新的可行流 f_2。v_t 在 $G(f_2)$ 中的层数大于 v_t 在 $G(f_1)$ 中的层数,重新构造分层剩余网络 $\mathrm{AG}(f_2)$,对于 $\mathrm{AG}(f_2)$ 重复以上的做法,得到可行流 f_3,一直做下去,直到得到可行流 f_k,使得 $G(f_k)$ 中不存在 (v_s,v_t) 路径,此时 f_k 即为 G 的最大流。

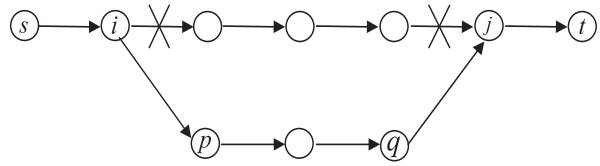

图 1　最短增广链修复过程

定理 1:分层剩余网络 $\mathrm{AG}(f)$ 中 (v_s,v_t) 路就是容量网络 G 中关于 f 的最短增广链。若沿着最短增广链增广后,通过该方法修复的路径仍为最短增广链。

证明:增广链的定义是,对于 G 中一条 (v_s,v_t) 链 P,若 P 的前向弧为 f 非饱和弧,后向弧为 f 正弧,则称 P 为关于 f 的 (v_s,v_t) 的增广链。

由剩余网络的定义知,剩余网络中弧的容量为:

$$c_{ij}=\begin{cases}c_{ij}-f_{ij}, (v_i,v_j)\in A_+(f)\\f_{ji}, (v_i,v_j)\in A_-(f)\end{cases}$$

设 P 是剩余网络 $G(f)$ 中的一条 (v_s,v_t) 路,则 P 中任一条弧都满足增广链的定义。

又根据分层的规则易知,$G(f)$ 中任何最短 (v_s,v_t) 路都在 $\mathrm{AG}(f)$ 中,且 $\mathrm{AG}(f)$ 中任何 (v_s,v_t) 路都是 $G(f)$ 的最短 (v_s,v_t) 路,即 $\mathrm{AG}(f)$ 中任何 (v_s,v_t) 路都是容量网络 G 中的最短增广链。而通过该算法修复的路径仍是 $\mathrm{AG}(f)$ 中的 (v_s,v_t) 路径,所以修复的路径仍是最短增广链。

2.2 算法步骤

最大流算法:

输入:原容量网络 $G=(V,A,c)$ 与指定的发点 v_s 、收点 v_t ;

输出:最大流 f 。

Step1:在 G 中取初始可行流 f_1(可以取零流),令 $k=1$ 。

Step2:先构造剩余网络 $G(f_k)$,再利用广探法构造分层剩余网络 $AG(f_k)$,若 $AG(f_k)$ 中 v_t 得不到标号,结束,f_k 就是 G 的最大流,否则转 Step3。

Step3:在 $AG(f_k)$ 中寻找 (v_s,v_t) 路 P(深度优先原则),转 Step4,若不存在,则令 $f_{k+1}=f_k,k=k+1$,转 Step2。

Step4:沿 P 对 f_k 进行增广,相应修改 $AG(f_k)$ 中 P 上弧的容量,删去 P 上容量为零的弧和原网络中相应的弧。

Step5:对增广链 P 进行修复,转 Step4,若不能修复,转 Step3。

2.3 算法可行性分析

该算法运行时,每次在分层剩余网络中找到或修复一条最短增广链后,都从发点 v_s 出发沿不饱和弧进行增广,一直推进到收点 v_t ,增广后最短增广链上至少有一条饱和弧。设网络 $G=(V,A,c)$ 中共有 m 条弧,那么该算法最多经过 m 次增广后,网络 G 饱和,此时网络 G 无法找到或修复一条最短增广链,算法终止,得到网络最大流。所以该算法会在有限的步骤之后终止。

2.4 算法复杂度分析

设 G 的顶点数为 n ,弧数为 m 。因为算法中构造的分层剩余网络 $AG(f_k)$ 的层数随着 k 单调增加,所以 Step2 中构造分层剩余网络 $AG(f_k)$ 最多执行 $n-1$ 次,又由广探法知,每次构造分层剩余网络 $AG(f_k)$ 的复杂度为 $O(m)$ 。在 $AG(f_k)$ 中寻找增广链后都要删去至少一条弧,所以至多找 m 次 (v_s,v_t) 路,每次寻找 (v_s,v_t) 路的计算量为 $O(n)$,于是得到算法的时间复杂度为: $O(n+n\cdot m+n\cdot n\cdot m)=O(n^2m)$ 。

在改进算法运行过程中,Step5 中每一次对最短增广链进行修复,就减少了在 $AG(f_k)$ 中最短增广链的搜索次数,最终降低了时间复杂度。

3 实例分析

例:求图 2 中从 v_s 到 v_t 的网络最大流。

解:(1)对于网络 G ,取零流 f_1 作为初始可行流,令 $k=1$。

(2)构造剩余网络 $G(f_1)$,然后利用广探法构造 $AG(f_1)$,见图 3。

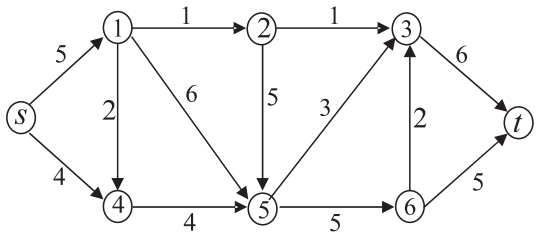

图 2 容量网络 G

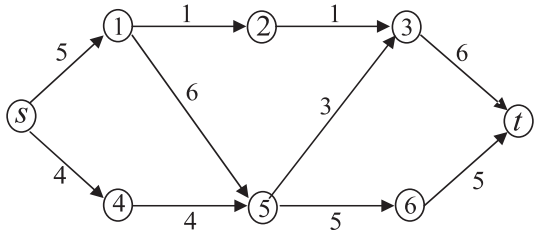

图 3 分层剩余网络 $AG(f_1)$

(3)在 $AG(f_1)$ 选取最短增广链 $P_1=v_sv_1v_2v_3v_t$, $\delta=\min\{5,1,1,6\}=1$,沿 P_1 对 f_1 进行增广流值 1,得到新的可行流仍记为 f_1,修改 $AG(f_1)$,删去 $AG(f_1)$ 和原网络中的弧 $(v_1,v_2),(v_2,v_3)$ 。

(4)对最短增广链 P_1 进行修复,断点为 v_1,v_3,修复后的最短增广链为 $P_1=v_sv_1v_5v_3v_t$, $\delta=\min\{4,6,3,5\}=3$,沿 P_1 对 f_1 进行增广,流值为 3,得到的可行流仍记为 f_1,修改 $AG(f_1)$,删去 $AG(f_1)$ 和原网络中的弧 (v_5,v_3) 。

(5)对于最短增广链 $P_1=v_sv_1v_5v_3v_t$ 不能修复,则转到步骤(3)重新寻找最短增广链,并且在之后的步骤中不需要进行修复,所以之后的算法执行情况和最短增广链算法相同。最后得到容量网络 G 的最大流 f_2,且最大流流值为 $v(f_2)=9$,见图 4。

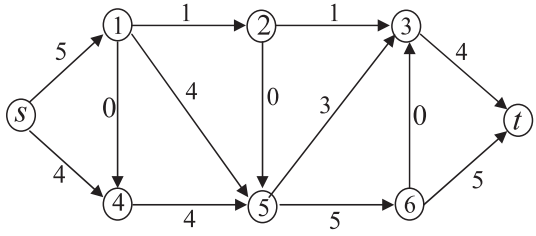

图 4 最大流 f_2

4 算法的仿真与分析

为比较改进算法与最短增广链算法的运行时间,分别在网络规模为 300,600,900,1 200,1 500,1 800 个节点的 BA 无标度网络上进行仿真比较,编程环境为 Matlab2012b。

BA 无标度网络邻接矩阵生成过程如下:

(1)先使用 $p=0.5$ 的概率生成一个规模为 50×50 的完全随机网络,并用 0 表示对应的弧不存在,1 表示对应的弧存在;

(2)求出邻接矩阵的行和作为每个节点的度数;

(3)新增节点,并且给每个新增的节点生成 50 条边;

(4)计算节点度数累计概率并用赌轮法将新增节点与原有网络节点连接并更新邻接矩阵,直到达到给定的规模停止更新;

(5)将 BA 无标度网络邻接矩阵中数值为 1 的元素替换为一定范围内的随机数作为弧容量。

对于每种规模,进行 5 次仿真实验,然后取平均运行时间进行比较,结果见表 1。

表 1　两种算法在不同规模网络上的实验结果

网络规模(节点数量)	最大流值		平均运行时间/s	
	最短增广链算法	改进算法	最短增广链算法	改进算法
300	81	81	0.680 2	0.667 4
600	183	183	2.025 8	1.920 0
900	126	126	2.996 3	2.488 4
1 200	103	103	4.227 9	3.425 4
1 500	78	78	6.609 4	4.908 8
1 800	106	106	14.244 4	11.435 5

从表 1 中可以看出,改进算法和最短增广链算法同样都能精确地求出网络最大流,并且改进算法的运行速度比最短增广链算法快。

两种算法的平均运行时间对比曲线如图 5 所示。

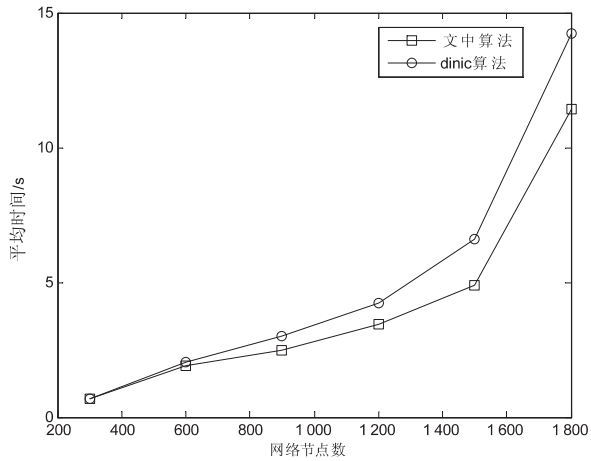

图 5　两种算法的平均运行时间

从图 5 中更能明显看出,改进算法的运行效率比最短增广链算法高,并且节点数也多,优化的效率更明显。所以对于大型网络新算法的适用性更强。

5　结束语

网络最大流在众多领域中应用广泛,而最短增广链算法是求解网络最大流问题的经典算法之一。与 Ford-Fulkerson 算法相比,其避免了增广链选取的任意性,从而减少了算法复杂度。但最短增广链算法在求解最大流的过程中,每条最短增广链只能增广一次,效率较低。文中提出的改进算法通过修复最短增广链提高了最短增广链的使用效率。仿真结果表明,该算法与其他最短增广链算法相比,在不同规模的随机网络中运行速度都较快,因此求解网络最大流的效率更高。

参考文献:

[1] 张宪超,陈国良,万颖瑜. 网络最大流问题研究进展[J]. 计算机研究与发展,2003,40(9):1281-1292.

[2] Pardalos P M,Resende M G C. Handbook of applied optimization[M]. New York:Oxford University Press,2002:363-374.

[3] Schrijver A. On the history of the transportation and maximum flow problems[J]. Mathematical Programming,2002,91(3):437-445.

[4] Ford J L R,Fulkerson D R. Maximum flow through a network [J]. Canadian Journal of Mathematics,1956,8(5):399-404.

[5] Edmonds J,Karp R M. Theoretical improvements in algorithmic efficiency for networks flow problems[J]. Journal of ACM,1972,19(2):248-264.

[6] Dinic E A. Algorithm for solution of a problem of maximum flow in a network with power estimation[J]. Soviet Mathematics Doklady,1970,2(5):1277-1280.

[7] Karzanov A V. Determining the maximum flow in a network by the method of pre-flows[J]. Soviet Mathematics Doklady,1974,15(3):434-437.

[8] Goldberg A V. The partial augment-relabel algorithm for the maximum flow problem[C]//European symposium on algorithms. Berlin:Springer,2008:466-477.

[9] Goldberg A V,Rao S. Beyond the flow decomposition barrier [J]. Journal of ACM,1998,45(5):783-797.

[10] 张宪超,陈国良. 小容量网络上的最大流算法[J]. 计算机研究与发展,2001,38(2):194-198.

[11] 邱伟星,王以凡,沈金龙. 一个求无向网络最大流的算法[J]. 南京邮电学院学报,1997,17(4):170-172.

[12] 郭　强. 无向网络最大流问题研究[J]. 计算机工程与应用,2005,41(9):76-78.

[13] Weihe K. Maximum (s,t)-flows in planar networks in O(|V| log|V|) - time[J]. Journal of Computer System Science,1997,55(3):454-476.

[14] Borradaile G,Klein P. An O(nlogn) algorithm for maximum st-flow in a directed planar graph[J]. Journal of ACM,2009,56(2):524-533.

[15] Negruseri C S,Pasoi M B,Stanley B,et al. Solving maximum flow problems on real world bipartite graphs[J]. Journal of Experimental Algorithmics,2011,16(3):14-28.

[16] Erickson J. Maximum flows and parametric shortest paths in planar graphs[C]//ACM-SIAM symposium on discrete algorithms. Austin,Texas,USA:ACM,2010:794-804.

[17] 赵礼峰,严子恒. 基于增广链修复的最大流求解算法[J]. 计算机应用,2015,35(5):1246-1249.

[18] 谢　政. 网络算法与复杂性理论[M]. 长沙:国防科技大学出版社,2003.