计算机研究与发展
Journal of Computer Research and Development
DOI:10.7544/issn1000-1239.2014.20130184
51(8):1845-1853,2014

分层法求解网络最大流的研究

赵 姝 苏建忠 刘倩倩 张燕平
(安徽大学计算机科学与技术学院 合肥 230601)
(计算智能与信号处理教育部重点实验室(安徽大学) 合肥 230601)
(zhaoshuzs2002@hotmail.com)

To Solve Maximum Flow Problem of Network with a Layering Method

Zhao Shu, Su Jianzhong, Liu Qianqian, and Zhang Yanping
(*School of Computer Science and Technology, Anhui University, Hefei* 230601)
(*Key Laboratory of Intelligent Computing and Signal Processing (Anhui University), Ministry of Education, Hefei* 230601)

Abstract Maximum flow problem is a classical combinational optimization problem. With the growing of the network scale, it is the key to improve the efficiency of the algorithm for the maximum flow problem. For a given single-source single-sink network, a novel layer method for getting its maximum flow is proposed in this paper. Firstly, we transform the original directed network into a layer network. Then we compute the maximum flow for each sub-network which contains two adjacent levels of nodes in the layer network. Finally, the minimum value of each sub-network's maximum flow is regarded as the estimated maximum flow of the original network. The experimental results on different networks show the efficiency of the proposed method. The maximum flow error is only about 1% averagely. Meanwhile, the running time of our method is only 11% of the classical algorithm (the Ford-Fulkerson algorithm) averagely. And in the best condition, the running time of our method is 2% of the classical algorithm and 25% of the improved two-phase capacity scaling algorithm.

Key words layering method; maximum flow; flow network; min-cut; the layer network

摘 要 网络最大流问题是经典的组合优化问题,随着网络规模的增加,提高算法效率成为解决问题的关键.为了降低求解大规模网络最大流的计算量,针对单源单汇网络提出基于网络分层的最大流问题求解新方法.分层法首先构造原有向网络对应的层次网络,接着在构造出的层次网络中计算各相邻结点层之间的最大流,以此为基础最终获得整个网络最大流的快速估算.分层法有效降低了计算的复杂性,为在大规模网络中快速获取最大流的求解提供了方便,并给出了一个解决最大流问题的新思路.不同网络上测试的实验结果显示,最大流的近似解误差可控制在1%左右,而平均运行时间仅为经典算法(Ford-Fulkerson算法)运行时间的11%,最好情况下的运行时间仅为经典算法运行时间的2%,是two-phase capacity scaling改进算法运行时间的25%,表明分层方法的有效性.

关键词 分层法;最大流;流网络;最小割;网络分层

中图法分类号 TP301.6; TP393

收稿日期:2013-02-28;**修回日期**:2013-06-19

基金项目:国家自然科学基金项目(61073117,61175046,61203290);安徽省自然科学基金项目(11040606M);安徽大学2011级研究生学术创新研究项目(01001770-10117700163)

通信作者:张燕平(zhangyp2@gmail.com)

最大流问题[1]是网络流理论的重要组成部分.它是组合优化中的经典问题,同时也可以看作特殊的线性规划问题[2-3].除了解决实际网络中的最大流问题外,最大流算法在许多领域都有广泛的应用[4-6].

解决最大流问题的经典算法是 Ford 和 Fulkerson 提出的增广路算法[7],随后许多学者提出改进算法,如 Edmonds 等人的最短增广路算法[8]、Dinic 提出的 Dinic 算法[9]、Karzanov 的预流推进算法[10]、Park-Pao 算法[11]等. Goldberg 等人对最大流问题进行了深入的研究,基于局部增广路重标记算法提出一系列改进算法[12-14].针对约束最大流问题,Cenk 提出多种基于 Capacity scaling 的算法[15-17],其中 two-phase capacity scaling 改进算法是近年来解决最大流问题较好的方法.研究者们也提出各种求解有向平面网络最大流问题的算法[18-21].尽管学者们的努力极大地推进最大流问题的研究进展,但是在改进算法的同时对网络规模的飞速增长考虑偏少,算法运行时间的减少比例远小于网络规模的增大比例.

近年来,由于各种网络的飞速发展,网络的规模呈指数级增长,对如何高效求解大规模网络的最大流提出更高的要求.目前主要有两类方法用以解决网络规模增大而带来的时间复杂度增加问题.一是使用并行计算的方法[22]求解最大流问题,另外一种是通过减小网络的规模降低求解最大流问题的复杂度的方法[23-24].虽然这两类方法简化了最大流问题的求解[25-26],但是并行计算方法要求初始网络可以转化为并行的模式,而第 2 种方法要对原网络做很大的改变,花费较长的时间,不能在短时间内求出最大流.如果网络的规模比较小,如几百或者几千个结点,它们可以有效地解决问题,但若处理大规模复杂网络[27-28],如结点数量达到 1 万以上,则需要很长的时间,有时第 2 种方法甚至是不可能的,不能满足短时间内求解大规模网络最大流的需求.

本文针对大规模网络最大流问题的求解,提出网络分层的概念,通过对原流网络中的结点进行分层,构造出原流网络对应的分层网络,同时结合网络流理论的最大流最小截定理[29],在层次网络中计算相邻层结点之间的流量,从而对原网络的最大流进行快速有效的估计,降低大规模网络最大流问题求解的复杂度,减少计算时间.

1 网络流和层次网络的基本概念及相关定理

定理 1[29]. 流网络 G 的最大流不大于其任一割的容量.

定理 2. 最大流-最小割定理[29]. 流网络 G 的最大流等于其最小割的容量.

定理 2 表明,网络从源点到汇点的各通路中,最小割是这些通路的瓶颈,其容量最小,它决定了整个网络的最大通过能力.

定义 1. 层次网络. 对一个流网络 $G=(V,E,C)$,其中 V 是由 n 个结点构成的集合,E 是包含 m 条边的集合,C 是边的容量集合. s 为源点,t 为汇点,令 $V_i=\{v\in V|$ 源点 s 到结点 v 的最短有向路径的长为 i,记作 $l(v)=i.\}$,V_i 值唯一. V_i 中的结点构成流网络 G 的第 i 层结点,则流网络 $G'=(\{V_i,i=0,1,\cdots\},E,C)$为 G 的层次网络.

假设源点 s 到汇点 t 的最短有向路径的长度为 n,则有:

1) $s\in V_0$, $t\in V_n$;

2) $V_i\cap V_j=\varnothing$, $i\neq j$.

说明:定义 1 中的有向路径的长度指该有向路径有向边的条数,两点间的最短有向路径指两点间含有边数最少的有向路径.

推论 1. 给定图 $G=(\{V_i,i=0,1\cdots\},E,C)$,若$\langle u,v\rangle\in E$,其中 $u\in V_i$, $v\in V_j$, $i\neq j$,则 $l(v)=l(u)+1$.

例 1. 图 1、图 2 分别给出一个网络和其对应的层次网络.

$V_0=\{s\}$, $V_1=\{1,2\}$, $V_3=\{3,4,5,t\}$.

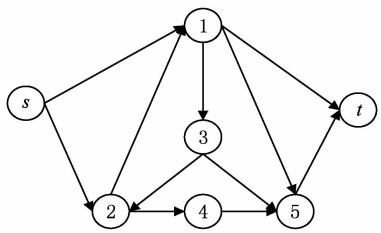


Fig. 1 The original network.

图 1 原网络图

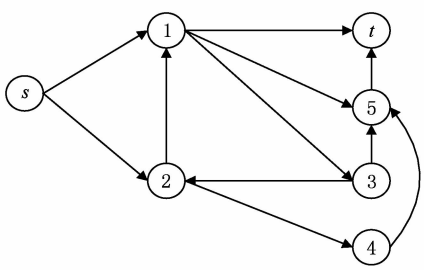


Fig. 2 The layer network of the original network.

图 2 原网络对应的层次网络

由图 1 和图 2 可知,原网络与其对应的层次网络是同构的,任意的单源单汇网络都可以转换为层次网络,本文的层次网络不改变原网络的结点和边,只需标记结点的层次.然而将原网络转换成层次网络后,各结点间的层次结构明显,各层之间的网络流流向清晰,使得用分层法求解最大流可行.

2　基于网络分层的最大流算法

结合层次网络的结构优势,本节以经典算法 FORD-FULKERSON 算法为基本求解算法,给出基于层次网络的最大流求解方法.

首先,给出构造层次网络的算法,使用广度优先搜索策略形成结点层次.

算法 1. 构造层次网络.

输入:流网络 $G=(V,E,C)$,s 为源点,t 为汇点,G 为单源单汇网络;

输出:网络分层后的流网络 $G'=(\{V_i,i=0,1,\cdots,n\},E,C)$.

步骤 1. 令 $V_0=\{s\}$,$S_0=V_0$,$i=0$;　/* V_i 为第 i 层结点集,$S_i=V_0\cup V_1\cup\cdots\cup V_i$) */

步骤 2. 若 $S_i=V$,分层结束,输出 $V_0,V_1,\cdots,V_n$.否则令 $V_{i+1}=\{u|\langle v,u\rangle,v\in V_i\}$,转步骤 3;

步骤 3. 令 $S_{i+1}=S_i\cup V_{i+1}$,并令 $i=i+1$,转步骤 2;

网络结点的分层算法在步骤 2 求 V_{i+1} 时需要遍历 V_i 的所有出边,最坏情况下每循环一次步骤 2 只能检查一条出边,且在算法执行过程中每条边都作为出边,算法需循环 $|E|$ 次,因此算法的时间复杂度为 $O(|E|)$.本文算法在分层时需确定每一结点所属层次,因此其空间复杂度为结点的个数,即 $O(n)$.

由算法 1 可得网络中各层的结点集合 $V_0,V_1,\cdots,V_n$,则原流网络可转换为如图 3 所示的层次网络结构.

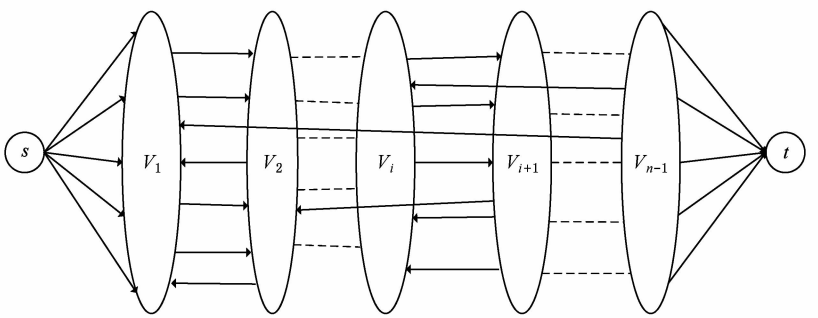


Fig. 3　The layer network.

图 3　层次网络

从层次网络的构造过程可以看出,层次网络和原流网络是同构的,构造的层次网络并没有改变原网络的拓扑结构.

网络中的结点分层后,网络中的边$\langle u,v\rangle$只有 3 种可能:

1) $u\in V_i$,$v\in V_{i+1}$;

2) $u,v\in V_{i+1}$;

3) $u\in V_i$,$v\in V_j$ 其中 $i>j+1$;

分层后的网络中不存在边$\langle u,v\rangle$($u\in V_i$,$v\in V_j$,$j>i+1$),即从某个结点流出的流量要想流到汇点必须由它的下一层结点进行中转.由割的定义[29]可知,在层次网络中,由结点集 V_i 到结点集 V_{i+1} 的边自然地构成了原流网络的一个割,因为如果把这些边从网络中移除,从源点到汇点将不可达,因此 $V_1,\cdots,V_n$ 共 $n-1$ 层中间结点构成了 $n-2$ 个割集.由定理 1 可知,原网络的最大流不会大于其任一割的容量,我们可看出原网络的最大流小于等于这 $n-2$ 个割中容量最小的割的容量.由定理 2 可知流网络 G 的最大流等于其最小割的容量,但是由于这 $n-2$ 个割集中不一定包含最小割,若只是简单地取这 $n-2$ 个割的容量的最小值作为对原网络最大流的估计,实际最大流值和估计值之间的误差将会比较大.

考虑上述因素,本文将利用 FORD-FULKERSON 算法对由 V_i 和 V_{i+1}(如图 4 所示)这两层结点构成的子网络作最大流的计算,如算法 2 所示,从而得到这些边上实际流量的一个较准确的估计,作为对整个网络最大值的估计值 f_i.

算法 2. 子网络最大流求解.

对网络中的每个结点 x,定义两个属性参数 $flowin$,$flowout$:$x_{flowin}=\sum_{l(w)<l(x)}c_{wx}$,表示 x 的邻居结点流进结点 x 的所有可能的流量和;$x_{flowout}=\sum_{l(x)<l(w)}c_{xw}$,表示从结点 x 流出到其邻居结点的所有

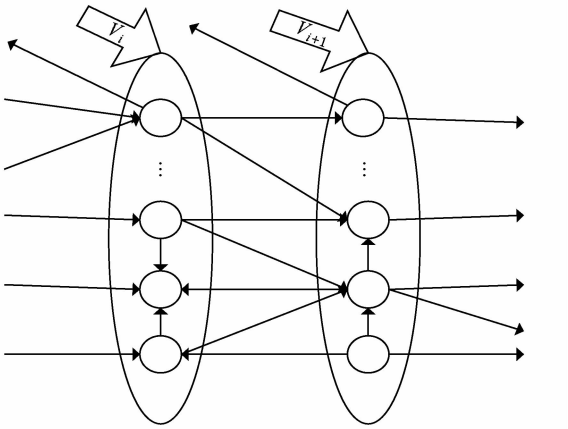


Fig. 4 The adjacent levels of the layer network.

图 4 层次网络中相邻的两层

可能的流量和.

输入：由相邻两层结点集 V_i 和 V_{i+1} 构成的局部网络流图 $G_i(\{V_i,V_{i+1}\},E,C)$；

输出：最大流值 f_i.

步骤 1. 为局部网络流图 $G_i(\{V_i,V_{i+1}\},E,C)$ 添加两个虚拟结点：源结点 s'、汇结点 t'，形成子网络流图，如图 5 所示；

步骤 2. 对任意结点 $x\in V_i$，建立一条边$\langle s',x\rangle$并设置其容量为 x_{flowin}；对任意结点结点 $x\in V_{i+1}$，则建立一条边$\langle x,t'\rangle$并设置其容量为 $x_{flowout}$；

步骤 3. 利用 FORD-FULKERSON 算法求解子网络的最大流值 f_i.

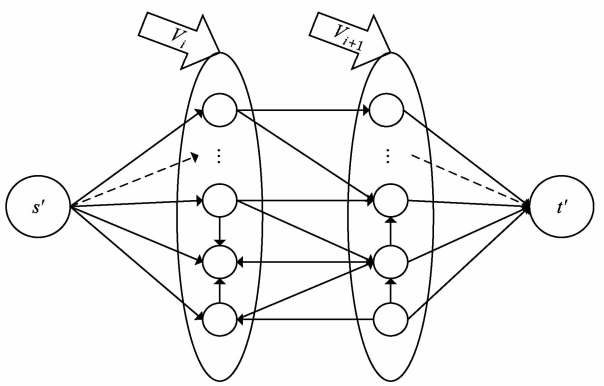


Fig. 5 The sub-network flow.

图 5 对应的子网络流图

由算法 2 得到的子网络流图 G_i 的最大流值 f_i 就是对这两层之间实际通过的流量的较准确的估计. 事实上，最小割作为网络的瓶颈，若原网络的最小割包含在 V_{i-1} 和 V_i 相邻层之间的边中，转换为如图 5 的子图后，最小割将作为虚拟源点的出边，同理，若原网络的最小割包含在 V_{i+1} 和 V_{i+2} 相邻层之间的边中，转换为如图 5 的子图后，最小割将作为虚拟汇点的入边. 提高子图的最大流估计值的准确率，由算法 2 求出的最大流值和原网络的实际最大流值是相等的.

算法 2 在步骤 1,2 需遍历子图中每一个结点，在增加了虚拟结点之后还需应用 FORD-FULKERSON 算法，因此在子图中，其时间复杂度略高于 FORD-FULKERSON 算法，而空间复杂度等同于 FORD-FULKERSON 算法.

在算法 2 的基础上，给出本文提出的基于层次网络的最大流求解算法.

算法 3. 基于层次网络的最大流求解算法.

输入：网络流图 $G=(V.E.C)$，s 为其源点，t 为其汇点；

输出：网络的最大流值 f.

步骤 1. 由算法 1，对原流网络的结点进行分层，构造其对应的层次网络；

步骤 2.

for ($i=1,i<n,i++$) /* n 为网络层次数 */

{用算法 2 求得 V_i 和 V_{i+1} 相邻层的最大流 f_i}；

步骤 3. 最大流值 $f=\min(f_i)$.

算法 3 由两个部分组成，因此其时间复杂度也包括两个部分：1)步骤 1 中算法 1 所需的时间复杂度；2)步骤 2 中 $n-1$ 次算法 2 所需的时间复杂度. 两部分之和为 $O(|E|)+(n-1)O(|E'|m'^2)$，其中 E' 表示子网络中的结点数，m' 表示子网络中的边数. 因此在大规模网络中，采用分层法将原网络分为一定的层次后，算法 3 的时间复杂度远小于 FORD-FULKERSON 算法的时间复杂度($O(|E|m^2)$). 算法 3 的空间复杂度相似于 FORD-FULKERSON 算法.

例 2. 计算图 6 所示网络的最大流.

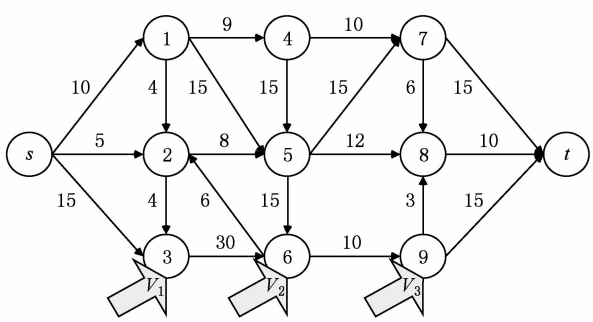


Fig. 6 The layer network of a simple flow network.

图 6 一个简单流网络对应的层次网络

图 6 为一个简单的流网络对应的层次网络，中间结点分布在网络流图的 3 个层上. 该网络的实际

最大流为 28,最小割集={⟨s,1⟩,⟨2,5⟩,⟨6,9⟩}. 由算法 2 得到这 3 层结点各相邻的两层结点所构成的两个子网络的最大流分别为 28 和 35,故最小值 28 为原网络最大流的估计值. 该估计值和原网络实际最大流值相等并非巧合,因为原网络的最小割包含在由结点集 V_1 和结点集 V_2 的这两层结点构建的子网络流图中,算法 2 能够发现该最小割.

例 3. 将图 6 中边⟨9,t⟩的容量由 15 改为 6,如图 7 所示,计算其最大流.

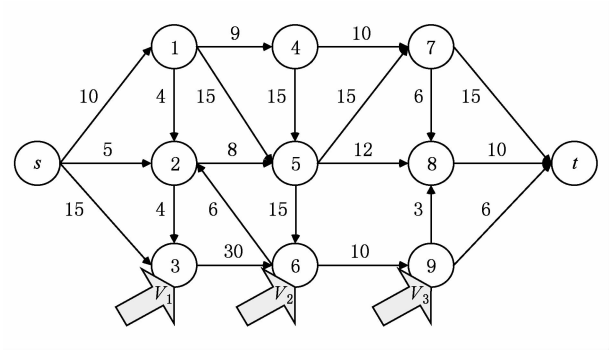


Fig. 7　A changed layer network.

图 7　调整后的一个层次网络

此时网络的实际最大流为 27,而最小割={⟨s,1⟩,⟨2,5⟩,⟨9,8⟩,⟨9,t⟩}. 由算法 2 得这 3 层结点各相邻的两层结点构成的子网络的最大流分别为 28 和 31,估计值仍为 28,比实际最大流值稍大,这是因为原网络的最小割⟨9,8⟩并不在某一个我们计算的子网络中,造成一定的误差.

3　实验结果及分析

本文使用国内外大家普遍采用的网络生成工具 NETGEN 和 RMFGEN 生成实验数据,网络生成工具的代码可以从文献[30]处下载.

NETGEN[31] 是 1974 年由 Klingman,Napier 和 Stutz 共同给出. 给定网络的结点数、边数以及边上的最大容量(整数),生成一个随机网络.

RMFGEN[32] 是由 Goldfarb 和 Grigoridis 在 1988 年给出,生成 3 维格点网络. 该算法需要 2 个参数 a 和 b,使用该算法生成的网格由 b 层组成,每层由 a^2 个结点组成 2 维网格.

网络的存储及实现均采用十字链表结构. 本文中所有的实验的硬件环境为 CPU:Intel Pentium® Dual-Core CPU E5500 @2.80 GHz,2.79 GHz,内存 1.96 GB. 编程环境为 MyEclipse7.0,使用 JDK 6.0.

表 1 给出在 RMFGEN 实例族上的实验结果.

Table 1　Experimental Results on the RMFGEN

表 1　在 RMFGEN 实例族上的实验结果

Network	n	m	i	c	f	f'	rt/s	rt'/s	er/%
Network 1	2 502	14 800	1	2 500	620 645	620 645	1.35	1.36	0.00
Network 2	5 002	136 348	2	2 500	620 167	620 167	21.05	21.07	0.00
Network 3	10 002	54 021	100	100	20 032	20570	17.11	0.63	2.69
Network 4	10 002	95 774	100	100	76 913	77 005	87.61	1.66	0.12
Network 5	10 002	158 601	100	100	168 245	169 949	257.69	3.24	1.01
Network 6	10 002	67 454	25	400	141 585	141 585	52.44	3.85	0.00
Network 7	10 002	96 218	25	400	362 787	366 981	136.78	7.41	1.16
Network 8	10 002	162 095	25	400	909 897	918 044	564.39	16.42	0.89
Network 9	20 002	131 939	200	100	33 283	33 381	117.18	1.94	0.29
Network 10	20 002	186 982	200	100	77 813	78 176	340.82	3.22	0.47
Network 11	20 002	291 761	200	100	166 030	167 368	890.92	5.85	0.81
Network 12	20 002	135 409	50	400	132 046	132 502	238.19	7.86	0.35
Network 13	20 002	195 048	50	400	365 641	369 223	678.03	14.91	0.98
Network 14	20 002	305 920	50	400	765 066	765 530	1545.58	35.34	0.06

Note: n and m represent the nodes and edges of the network; i represents the layers; c represents the nodes in each node-set; f and rt represent the maximum flow by ford-fulkerson algorithm and it's running time; f' and rt' represent ours; and er represents the $((f'-f)/f)\times 100$.

从表 1 的实验结果中我们可以看出,本文算法有较好的实用效果.

1) 非常适用于较大规模的网络.

明显地降低了计算网络最大流的运行时间,对

结点数相同、层数相同的网络,随着网络中边的数目的增加,FORD-FULKERSON 算法的运行时间急剧增加,而本文算法的运行时间则维持相对稳定.对规模相近的网络(结点数和边数都相似),网络的层数越多,本文算法的运行时间越少,最大流的误差仍控制在很小的范围内.例如 Network 5 和 Network 8,它们的结点数相同、边数相似,Network 5 层数多出 Network 8 层数的 3 倍,Network 8 的运行时间比 Network 5 的运行时间多出 3 倍,它们的误差都在 1%左右.从表 1 中的数据分析得出,对于结点数和边数相近的网络,网络层数多出 n 倍,运行时间也节省 n 倍,而最大流的误差跟网络的层数没有必然的联系.

2) 误差小.

由本文的算法计算得到的网络最大流的估计值和实际最大流值之间的误差控制得较好,其中最大的误差仅为 2.69%,其余都在 1%左右,大部分误差值小于 1%.

事实上,本文的方法对简单网络的效果不明显,如表 1 中的两个网络 Network 1 和 Network 2,使用本文的算法运行时间比使用经典算法的运行时间略高,主要是因为这两个网络结构的特殊性,其中 Network 1 的网络结构只有一层,Network 2 的网络结构只有两层,加上构造层次网络时间,本文方法略大于经典算法的运行时间.

为了进一步说明本文算法在时间上的优越性,本文参考了近年来求解最大流问题较好的方法的运行时间,表 2 列出了基于 Capacity Scaling 算法[33-34]求解 RMFGEN 实例族最大流的运行时间,其中数据引用于文献[33].

Table 2 Experimental Results on RMFGEN of other Algorithm

表 2 其他算法在 RMFGEN 实例族上的实验结果

Network	n	m	Running Time/s				rt/s
			2-Phase I	2-Phase II	1-Phase	2-Phase′	
Network 15	1024	4080	0.065	0.066	0.023	0.021	0.28
Network 16	2048	8176	0.212	0.203	0.067	0.053	0.68
Network 17	4096	16368	0.521	0.550	0.173	0.141	1.13
Network 18	8192	32752	3.107	3.327	1.468	1.433	0.52
Network 19	16384	65520	14.655	15.522	8.811	7.945	1.96
Network 20	32768	131056	63.978	67.855	43.132	36.513	8.21

Note: n and m represent the nodes and edges of the network; and rt represents our algorithm.

从表 1 和表 2 中选取 4 组相同数量级的网络最大流求解时间进行比较得图 8,其中横坐标为所选取的 4 组网络,纵坐标表示运行时间(单位:s).可以看出,本文算法的运行时间大部分优于流行的最大流算法,特别是对大数据网络,本文的运行时间只要 2-phase Capacity Scaling 改进算法运行时间的 25%.对小数据网络,由于本的算法首先处理网络的分层,花费一定的时间,因此本文的算法略差.

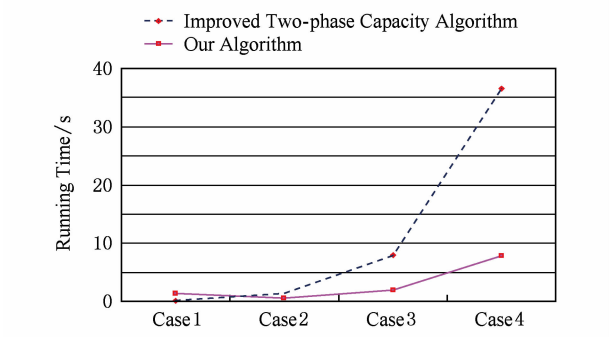


Fig. 8 A comparison of two algorithms on the same network.

图 8 2 种算法在规模相近的网络上的比较

表 3 列出在 NETGEN 实例族上的实验结果.

由表 3 可知,大部分的误差都低于 1%,最大误差也仅为 2.09%.

从算法的运行时间看,若网络的饱和度较大,且相应的网络的层数较少,则局部网络的结构必然变得相对复杂,增大了本文方法计算局部网络最大流的时间复杂度,但准确率没有必然联系.例如表 3 中的 Network 22 和 Network 23,相同的结点数和网络层数,Network 23 比 Network 22 的结构复杂,Network 23 比 Network 22 的运行时间多,而它们的准确率跟此没有关系.因此本文算法在层数较少的网络上的时间效果差于层数较多并且层与层之间结构简单的网络.

从准确率看,若网络的层数较少,更容易在相邻

层中发现最小割,通过局部的计算得到原网络的最大流更为精确.例如 Network 22,Network 23,Network 25 和 Network 26,这 4 个网络的层数相对较少,它们的误差值也相对较小.因此本文算法在相同条件下对相对层数较少的网络,具有较高的准确率.

Table 3　The Experimental Results on NETGEN

表 3　在 NETGEN 实例族上的实验结果

Network	n	m	i	f	f'	rt/s	rt'/s	er/%
Network 21	5 000	52 451	18	30 757	31 289	7.63	1.17	1.72
Network 22	5 000	101 883	9	163 974	164 586	51.75	10.34	0.37
Network 23	5 000	247 783	9	487 922	487 922	453.44	78.58	0.00
Network 24	10 000	106 685	22	54 054	54 524	55.08	4.05	0.87
Network 25	10 000	244 310	14	251 663	251 663	358.59	32.91	0.00
Network 26	10 000	360 564	13	382 207	382 330	890.13	120.27	0.03
Network 27	20 000	173 689	87	13 782	14 071	67.82	4.25	2.09
Network 28	20 000	410 682	39	127 722	127 914	682.14	34.75	0.15
Network 29	20 000	604 091	30	244 488	244 488	1420.66	122.78	0.00

Note: n and m represent the nodes and edges of the network; i represents the layers; c represents the nodes in each node-set; f and rt represent the maximum flow by ford-fulkerson algorithm and it's running time; f' and rt' represent ours; and er represents the $((f'-f)/f)\times 100$.

综上,本文方法的特点为:

1) 适用于具有一定层数(最少 4 层)的网络.

层数越多,虽然网络的层数变多,但是相邻层的网络相对简单,使用算法 2 计算子网络最大流的运行时间相对较少,时间总和也相对更少.若不能将原网络分至 3 层以上,本文的算法则不适用,每个子网络结构复杂,求解其最大流时不占优势,并有额外的分层耗费,导致本文算法的运行时间会略有增加.

2) 更适用于相邻层之间的稠密度高的网络.

相同的结点数和边数,若相邻层网络之间的边越多,它的运行时间越少.本文在使用算法 2 时,减少了层次间的弧(在处理子网络时,将同一结点的弧收缩),在广度搜索出边时节省大量的时间.本文利用 FORD-FULKERSON 算法求相邻层的最大流,需要大量的时间,因此本文算法更注重于相邻层的网络流,使得最小割分布在相邻层之间的概率较大,更容易发现最小割,增加准确率,也会使运行时间变小.

4　结　　论

求解网络最大流是组合优化的基本问题之一,随着网络规模的不断增大,为了满足求解时间的需求,在较短的时间内以最小的代价求出网络的最大流,本文提出基于网络分层的求解最大流的方法,通过构造原网络对应的层次网络,然后在层次网络中计算各相邻层结点之间的流量,从中选取最小值作为对整个网络最大流的估计.实验结果表明本文的方法可以快速有效地计算出原网络的近似最大流,其误差基本控制在 1%左右,尤其适用于层次结构分明的大数据网络,针对层数较少的网络其准确率更高,而针对相邻层之间稠密度较高的网络其运行时间更低.

我们的下一步工作将讨论层次的划分及子网络的构成,以便更快速、有效地求解大规模网络的最大流.

参　考　文　献

[1] Zhang Xianchao, Chen Guoliang, Wang Yinyu. Research on the maximum network flow problem [J]. Journal of Computer Research and Development, 2003, 40(9): 1281-1292 (in Chinese)
(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] Berge C. Graphs and Hypergraphs [M]. Amsterdam: North-Holland Publishing Company, 1973

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

[4] Royset J O, Wood R K. Solving the bi-objective maximum-flow network-interdiction problem [J]. Informs Journal on Computing, 2007, 19(2): 175-184

[5] Zhang Xianchao, Jiang He, Liu Xinyue, et al. Maximum flows in undirected planar networks with unit capacities [J]. Journal of Computer Research and Development, 2008, 45 (suppl I): 40-42 (in Chinese)
(张宪超, 江贺, 刘馨月, 等. 无向平面单位容量网络中的最大流[J]. 计算机研究与发展, 2008, 45(增刊 I): 40-42)

[6] 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): 1-5

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

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

[9] Dinic E A. Algorithm for solution of a problem of maximum flow in networks with power estimation [J]. Soviet Mathematics Doklady, 1970, 11(8): 1277-1280

[10] 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

[11] Park G H, Pao Y H. Neural-net approach to the maximum flow problem [J]. Electronics Letters, 1998, 34(8): 786-787

[12] Goldberg A V, Tarjan R E. A new approach to the maximum-flow problem [J]. Journal of the ACM (JACM), 1988, 35(4): 921-940

[13] Goldberg A V. The partial augment - relabel algorithm for the maximum flow problem [C] //Proc of the 16th Annual European Symp on Algorithms. Berlin: Springer, 2008: 466-477

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

[15] Caliskan C. A faster polynomial algorithm for the constrained maximum flow problem [J]. Computers & Operations Research, 2012, 39(11): 2634-2641

[16] Caliskan C. A computational study of the capacity scaling algorithm for the maximum flow problem [J]. Computers & Operations Research, 2012, 39(11): 2742-2747

[17] Caliskan C. A double scaling algorithm for the constrained maximum flow problem [J]. Computers & Operations Research, 2008, 35(4): 1138-1150

[18] Weihe K. Maximum (s, t)-flows in planar networks in $O(|V|\log|V|)$ time [C] //Proc of the 35th Annual Symp on Foundations of Computer Science. Piscataway, NJ: IEEE, 1994: 178-189

[19] Borradaile G, Klein P. An $o(n \log n)$ algorithm for maximum st-flow in a directed planar graph [J]. Journal of the ACM, 2009, 56(2): 524-533

[20] Zhang Yanping, Hua Bo, Jiang Juan, et al. Research on the maximum flow in large-scale network [C] //Proc of the 7th Int Conf on Computational Intelligence and Security (CIS). Piscataway, NJ: IEEE, 2011: 482-486

[21] Pham T L, Lavallee I, Bui M, et al. A distributed algorithm for the maximum flow problem [C] //Proc of IEEE 2005 the 4th Int Symp on Parallel and Distributed Computing. Piscataway, NJ: IEEE, 2005: 131-138

[22] Halim F, Yap R H C, Wu Yongzheng A mapreduce-based maximum-flow algorithm for large small-world network graphs [C] //Proc of IEEE the 31st Int Conf on Distributed Computing Systems (ICDCS 2011). Piscataway, NJ: IEEE, 2011: 192-202

[23] Liers F, Pardella G. Simplifying maximum flow computations: The effect of shrinking and good initial flows [J]. Discrete Applied Mathematics, 2011, 159(17): 2187-2203

[24] Zhang Yanping, Xu Xiansheng, Hua Bo, et al. Contracting community for computing maximum flow [C] //Proc of 2012 IEEE Int Conf on Granular Computing (GrC). Piscataway, NJ: IEEE, 2012: 651-656

[25] Fujishige S. A maximum flow algorithm using ma ordering [J]. Operations Research Letters, 2003, 31(3): 176-178

[26] Lorente S, Bejan A. Heterogeneous porous media as multiscale structures for maximum flow access [J]. Journal of Applied Physics, 2006, 100(11): 1149090-1149098

[27] Zhang Xinmeng, Jiang Shengyi. Complex network community detection based on core graph incremental clustering [J]. Acta Automatica Sinica, 2012, 38(9): 1-8 (in Chinese)
(张新猛, 蒋盛益. 基于核心图增量聚类的复杂网络划分算法[J]. 自动化学报, 2012, 38(9): 1-8)

[28] Huang Wenliang, Liu Yong, Zhong Zhiqiang, et al. Complex network based sms filtering algorithm [J]. Acta Automatica Sinica, 2012, 35(7): 990-996 (in Chinese)
(黄文良, 刘勇, 钟志强, 等. 基于复杂网络的垃圾短信过滤算法[J]. 自动化学报, 2012, 35(7): 990-996)

[29] Xie Jinxing, Xing Wenxun. Network Optimization [M]. Beijing: Tsinghua University Press, 2000 (in Chinese)
(谢金星, 邢文训. 网络优化[M]. 北京: 清华大学出版社, 2000)

[30] Source codes for problem instance generators [OL]. [2010-05-10]. http://elib. zib. de/pub/mp-testdata/generators/index. html

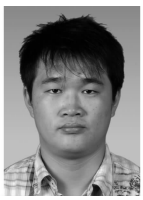
[31] Klingman D, Napier A, Stutz J. NETGEN: A program for generating large scale capacitated assignment, transportation, and minimum cost flow network problems [J]. Management Science, 1974, 20(5): 814-821

[32] Goldfarb D, Grigoriadis M D. A computational comparison of the dinic and network simplex methods for maximum flow [J]. Annals of Operations Research, 1988, 13(1): 81-123

[33] Caliskan C. A computational study of the capacity scaling algorithm for the maximum flow problem [J]. Computers & Operations Research, 2012, 39(11): 2742-2747

[34] Caliskan C. A faster polynomial algorithm for the constrained maximum flow problem [J]. Computers & Operations Research, 2012, 39(11): 2634-2641

Zhao Shu, born in 1979. Received her PhD from Anhui University, associate professor. Her main research interests include machine learning, quotient space theory and social network.



Su Jianzhong, born in 1989. Master in Anhui University. His main research interests include network optimization and social network.

Liu Qianqian, born in 1989. Master in Anhui University. Her main research interests include social network and network optimization.

Zhang Yanping, born in 1962. PhD and professor in Anhui University. Her main research interests include machine learning, quotient space theory and intelligent computing.