第 25 讲
计算机网络的最小接通时间

□程继新

AMCM-94B 题是一个来源于实际的计算机网络信息传输问题. 原题简述如下：

某公司各部门间每天都要通过网络进行信息传输(各部门都配备了计算机，信息用文件表示). 信息网络可以用图来表示，顶点 $V_1, V_2, \cdots, V_m$ 表示计算机. 边 $e_1, e_2, \cdots, e_n$ 表示两台计算机之间需要传输的文件. $T(e_x)$ 为文件 e_x 的传输时间，$C(V_y)$ 是计算机 V_y 可以同时传输的最大文件数目. 例如 $C(V_y)=1$ 表示 V_y 每次只能与一台别的计算机进行文件传输. 针对下列三种情形，该公司希望有一个最佳的传输方案，使网络的接通时间(即所有文件传输全部完成的时间)最短.

情形 A：公司有 28 个部门，每个部门配有一台计算机(分别用图 25-1 中的顶点表示). 每天需传输 27 个文件(分别以图 25-1 中的边来表示). 在该网络中，对所有的 x、y，满足 $T(e_x)=1$ 和 $C(V_y)=1$.

情形 B：假设网络结构未变，但各部门间所传递的文件类型和大小各不相同，传输文件所需时间如表 25-1 所示，每台计算机的传输容量 $C(V_y)$ 仍为 1.

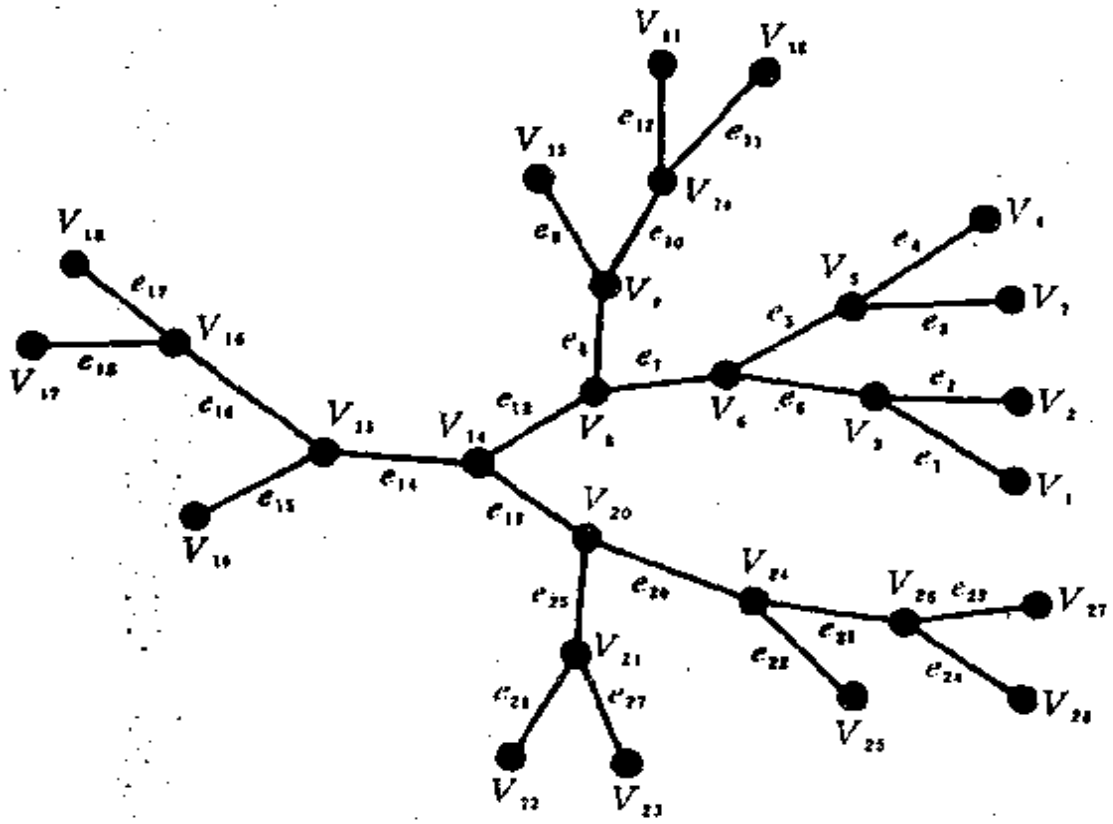

图 25-1　情形 A 的网络

表 25-1　情形 B 的文件传输时间

x	1	2	3	4	5	6	7	8	9	10	11	12	13	14
$T(e_x)$	3.0	4.1	4.0	7.0	1.0	8.0	3.2	2.4	5.0	8.0	1.0	4.4	9.0	3.2
x	15	16	17	18	19	20	21	22	23	24	25	26	27	
$T(e_x)$	2.1	8.0	3.6	4.5	7.0	7.0	9.0	4.2	4.4	5.0	7.0	9.0	1.2	

情形 C：公司正考虑扩展，一方面，每天有更多的文件需要传输，另一方面，公司的计算机系统也正在升级，有些部门将得到更高级的计算机(可以同时与几个部门进行文件传输)．这些变化分别显示于图 25-2、表 25-2、表 25-3 中．

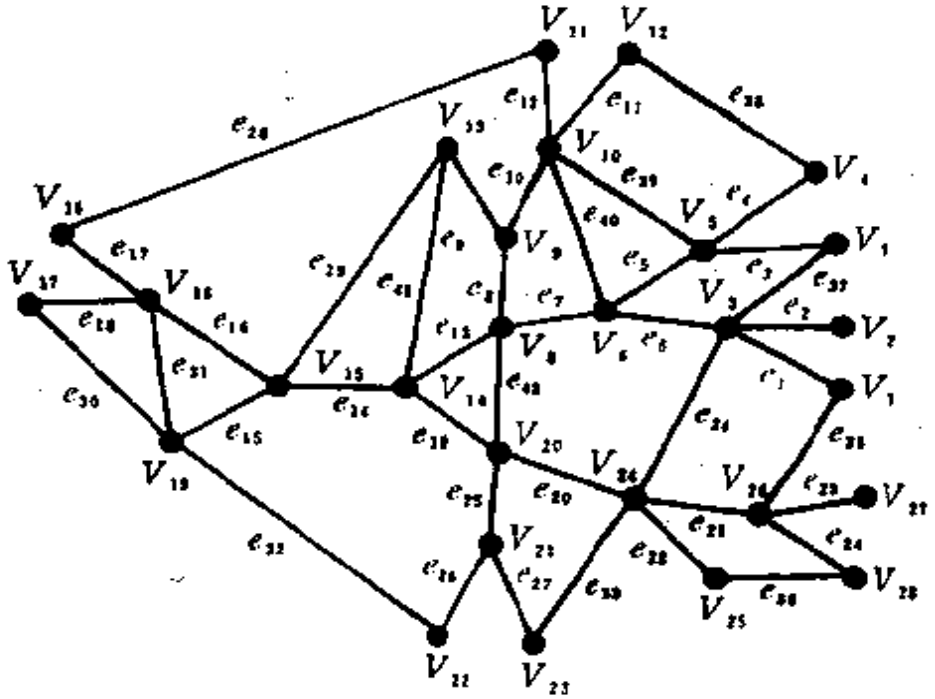

图 25-2　情形 C 的网络

表 25-2　情形 C 的文件传输时间

x	1	2	3	4	5	6	7	8	9	10	11	12	13	14
$T(e_x)$	3.0	4.1	4.0	7.0	1.0	8.0	3.2	2.4	5.0	8.0	1.0	4.4	9.0	3.2
x	15	16	17	18	19	20	21	22	23	24	25	26	27	28
$T(e_x)$	2.1	8.0	3.6	4.5	7.0	9.0	9.0	4.2	4.4	5.0	7.0	9.0	1.2	6.0
x	29	30	31	32	33	34	35	36	37	38	39	40	41	42
$T(e_x)$	1.1	5.2	4.1	4.0	7.0	2.4	9.0	3.7	6.3	6.6	5.1	7.1	3.0	6.1

表 25-3　情形 C 的计算机传输容量

y	1	2	3	4	5	6	7	8	9	10	11	12	13	14
$C(V_y)$	2	2	1	1	1	1	1	1	2	3	1	1	1	2
y	15	16	17	18	19	20	21	22	23	24	25	26	27	28

y	1	2	3	4	5	6	7	8	9	10	11	12	13	14
$C(V_y)$	1	2	1	1	1	1	1	2	1	1	1	2	1	1

（以下主要参考了中国科技大学获一等奖队的论文.）

§1 假　　设

1. 与文件传输时间相比，用于传输切换的时间(即一台计算机从一次传输转向另一次传输的时间)很小，可以忽略.

2. 所有文件传输都是独立的，即不存在某个文件必须在另一个文件之前或之后传输的情形.

3. 每一个文件都是以一个连续的整体被传输的.

4. 某次传输可以发生的充要条件是：参与该次传输的两台计算机都有足够的能力进行这次文件传输(即在该时刻计算机正在传输的文件数小于其可以传输的最大文件数).

5. 通讯网络是可靠的，无需为了验证或纠正错误而重复传输某个文件，因此，实际传输时间就是题中所给的时间 $T(e_x)$.

6. 在一个不长的时期内，各部门间每天要交换的信息量不会有大的波动，这意味着每天的文件传输时间基本上是一样的. 在稳定性分析中我们将考虑有较大波动的情形.

§2 符 号 约 定

$T(e_x)$：文件 e_x 的传输时间.

$C(V_y)$：计算机 V_y 的传输容量，即可以同时传输的最大文件数目.

$T_s(e), T_f(e)$：文件 e 开始和结束传输的时刻.

$\triangle$：图的度数.

χ'：边染色所需的最小色数.

$L(V_y)$：计算机 V_y 的传输负荷，定义为计算机完成所有传输需要的最短时间. 当 $C(V_y)=1$ 时，$L(V_y)=\sum_{e_x} T(e_x)$，当 $C(V_y)=n(n>1)$时，将计算机需要传输的文件分成 n 组，使传输时间总和最长的一组的时间尽可能短，$L(V_y)$即为该组所有文件的总传输时间.

L_{max}：所有计算机负荷中的最大值.

§3 问题分析

我们的任务是给出文件传输方案，使得网络接通时间最短，限制条件可以用下式表示：

对任意 V_x，任意时刻

$$\sum_y g_{xy} \leqslant C(V_x),$$

其中 $g_{xy}=\begin{cases}1, V_x \text{ 和 } V_y \text{ 正处于信息交换中；}\\ 0, V_x \text{ 和 } V_y \text{ 之间无信息传输.}\end{cases}$

前人已对寻求最短完成时间(makespan)的问题有过许多研究. 例如，在不考虑优先权的情况下(即任何工作的进行都不必依赖于其他工作的完成)，在 m 台机器上安排 n 项不同的工作，特定的工作必须在特定的机器上完成，使最后完成时间最小化的问题已被证明是 NP-hard(Brian Thomaseek，1993). 关于该问题的一个近似算法是 LPT(最长过程优化处理)，这种方法所得结果的上限为实际最短完成时间的 $\left(\frac{4}{3}-\frac{1}{3m}\right)$ 倍. (Graham，1969.)

按照前面的定义，计算机的负荷为其完成全部文件传输所需的最短时间. 因为接通时间是所有计算机完成全部信息传输的时间，它应当主要决定于负荷较大的计算机的进程，而且 Lmax 是网

络接通时间的一个下限，因此，在寻求传输方案时，负荷重的计算机的文件传输应予以优先考虑.

在情形 A 的网络中，对所有的文件 e_x 和计算机 V_y 满足 $T(e_x)=1$ 和 $C(V_y)=1$. 我们注意到，一个时间单位内的所有文件传输事实上对应于图的一个匹配，反过来，每一个完美匹配对应于在一个特定时间单位内的所有传输. 这样，情形 A 可以转化为一个边染色问题(即以尽可能少的色数给每条边染色，使得相邻边具有不同的颜色)，网络的接通时间等于边染色所用色数.

对于更一般的情形 B 和 C，虽然可以用手算的办法凑出一个可能最优的解，但我们力求找到一个一般的算法处理 $T(e_x)$，$C(V_y)$ 和网络结构均为任意的情形. 由于该问题在一般情形下是 NP-hard(Brian Thomaseek 1993)，要得到一个有效的多项式算法是不可能的，这使我们不得不求助于近似算法的实施(Michel Gondran).

我们首先考虑了“叠代算法”，但很快发现这种方法不适用于解决我们的问题. “叠代算法”总是从一个已知的解出发，通过逐步的局部改进以达到尽可能优的解，然而在网络信息传输中，每一次传输的变动都将可能导致整个系统的混乱. 于是我们采用了“贪婪算法”的思想，即在任何时刻，所有可以被传输的文件都应处于传输状态，而不是等待状态，这样做的理由是试图通过局部的优化来达到整体的尽可能优化. “贪婪算法”无疑是找到最优传输方案的有效途径.

§4 模型构造

1. 边染色方法

如前所述，情形 A 相当于一个边染色问题. 对于二分图(无奇

圈的图)有下列定理(J. A. Bondy 1976)：

定理一 二分图 G 的边染色数 $\chi'=\triangle$，$\triangle$ 为 G 的度数.

二分图边染色的多项式算法如下：

① 运用匈牙利算法找出图的一个完美匹配，用一种未用过的颜色给所有参加匹配的边染色.

② 消去所有染色的边，剩下的图依然是二分图.

③ 回到第一步，直到所有边全部被染色.

由于匈牙利算法是多项式的，因而该算法也是多项式的.

情形 A 的网络是树状结构，容易证明它是二分的，其度数为 3，可以用三种不同颜色对所有边染色，也就是说，网络的接通时间为 3 个单位，同时，计算机的最大负荷即为图的度数 3，因此 3 是最短接通时间. 我们给出相应的传输方案如下.

表 25-4 情形 A 的最佳传输时间

$T_i=1$	e_1	e_5	e_8	e_{11}	e_{14}	e_{17}	e_{20}	e_{23}	e_{27}		
$T_i=2$	e_2	e_3	e_7	e_9	e_{12}	e_{15}	e_{18}	e_{19}	e_{22}	e_{24}	e_{26}
$T_i=3$	e_4	e_6	e_{10}	e_{13}	e_{16}	e_{21}	e_{25}				

对于一般的情形，网络未必是二分的. 如由三条边和三个顶点组成的三角形图，其染色数并不等于图的度数. Vizing 曾经证明了下列定理(Vizing 1964)：

定理二 如果 G 是一个简单图，那么 $\chi'=\triangle$ 或 $\triangle+1$.

有人(Keming Zhang 1983)已提出了用 $\triangle+1$ 种颜色对简单图进行边染色的多项式算法，遗憾的是，该算法无法保证在任意情况下得到最少可能的染色数，因而有可能得不到最短接通时间，这迫使我们转而寻求另外的办法解决更一般的情形.

2. 考虑优先权的贪婪算法

一般情形下问题是 NP-hard，因而明智的选择是近似算法，我

们给出了寻求尽可能优的传输方案的有效方法，如下.

① 首先计算情形 B 和情形 C 中每台计算机的负荷(如表 25-5、表 25-6 所示)，将它们按递降顺序排列，负荷最大的计算机优先考虑其文件传输.

② 对于每一台计算机而言，将所有待传输的文件按传输时间递增顺序排列，时间较短的文件优先传输，这可以减少其它传输的等待时间.

③ 在整个文件传输过程中，按照优先顺序安排文件传输，直到不存在可以传输的文件处于等待状态为止，这样可保证在每一个局部阶段都能达到最优化.

④ 然而，局部优化未必保证整体优化，对于网络规模较小的情形，我们采用回溯的办法逐步改变优先权次序，在合理的时间寻求不同可能的传输方案，然后选择接通时间最短的方案付诸实施.

现在，建立于负荷优先和无等待思想基础上的深度优先搜索提供了解决情形 B 和情形 C 的有效算法. 这个算法适用于一般的情形，即 $T(e_x)$、$C(V_y)$和网络结构都是任意的. 下面的结论表明"考虑优先权的贪婪算法"的结果相当接近最短接通时间.

命题　$L\max$ 和 $2L\max-\min T(e_x)$分别是贪婪算法所得结果的上限和下限.

证明　下限为 $L\max$ 是显而易见的.

对于上限，考虑一个任意的文件 e，它在计算机 u 和 v 之间传输，$T_s(e)$和 $T_f(e)$分别为文件 e 开始和结束传输的时刻(起始时刻为 0)，下列关系成立：

$$T_f(e)=T_s(e)+T(e).$$

根据算法思想，文件 e 始终不处于等待状态，这意味着在 0～$T_s(e)$时间内，u 和 v 至少有一个处于忙碌状态. 因而有

$$\begin{aligned}T_s(e)&\leqslant[L(u)-T(e)]+[L(v)-T(e)]\\&\leqslant[L_{\max}-T(e)]+[L_{\max}-T(e)]\end{aligned}$$

$=2[L_{max}-T(e)]$,

$\therefore T_f(e)=T_s(e)+T(e)\leqslant 2L_{max}-T(e)\leqslant 2L_{max}-\min T(e_x)$.

由于文件 e 是任意的，假设它是最后完成传输的那个文件，这要，$T_f(e)$代表整个网络传输全部结束的时刻，因此网络接通时间的上限为 $2L_{max}-\min T(e_x)$.

注　下面的例子表明我们所给出的上限是贪婪算法的最小上限. 图 25-3 表示一个信息网络，采用考虑优先权的贪婪算法得到接通时间为 7，恰好为 $2L_{max}-\min T(e_x)=2\times 4-1$. 结合深度优先搜索可以给出最短接通时间为 6，这也说明深度优先搜索的必要性.

表 25-5　情形 B 计算机的负荷

y	1	2	3	4	5	6	7	8	9	10	11	12	13	14
$L(V_y)$	3.0	4.1	15.1	7.0	12.0	12.2	4.0	14.6	15.4	13.4	4.4	1.0	5.0	19.2
y	15	16	17	18	19	20	21	22	23	24	25	26	27	28
$L(V_y)$	13.3	16.1	4.5	3.6	2.1	21.0	17.2	9.0	1.2	20.2	4.2	18.4	4.4	5.0

表 25-6　情形 C 计算机的负荷

y	1	2	3	4	5	6	7	8	9	10	11	12	13	14
$L(V_y)$	9.0	4.1	23.8	13.6	17.1	19.3	10.3	20.7	8	9.5	10.4	7.6	9.1	12
y	15	16	17	18	19	20	21	22	23	24	25	26	27	28
$L(V_y)$	14.4	11.6	9.7	9.6	15.4	27.1	17.2	9.0	8.2	29.6	7.9	14.0	4.4	8.7

2.1 情形 A

运用上面的方法，我们很快给出了 30 种具有最短接通时间的传输方案(结果略去).

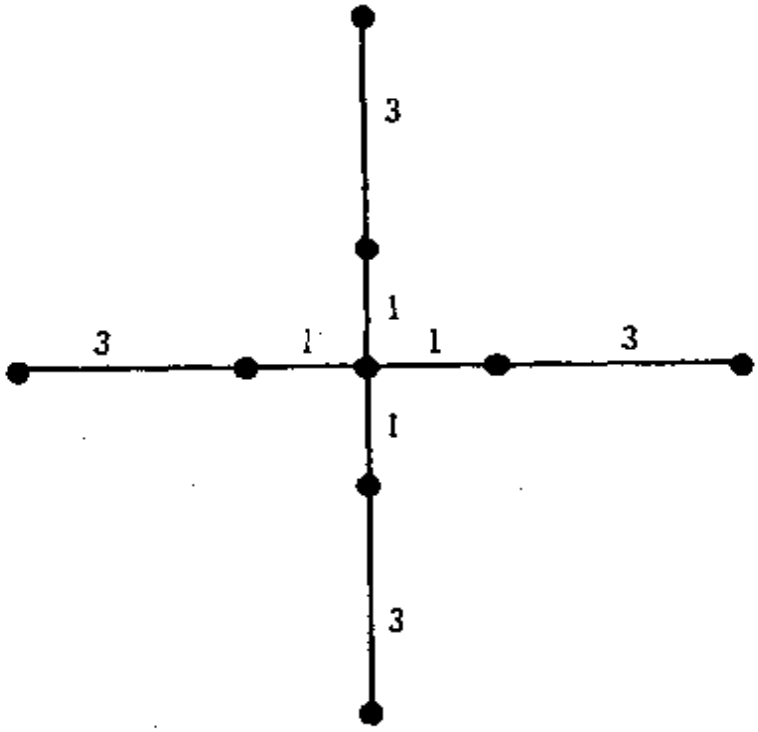

图 25-3

2.2 情形 B

深度优先搜索是一个穷举方法，将其完全实施是不切实际的，然而我们期望在合理的时间内找到足够好的结果. 对于情形 B，我们在一台 IBMPC386 机上运行用 $C++$ 语言编写的程序，在几秒钟内找到了 30 组最佳传输方案(接通时间为 23)，表 25-7 显示了其中的一组结果.

表 25-7 情形 B 的最佳传输方案

x	1	5	8	11	15	17	19	22	23	27	3	12	26	7
$T_s(e_x)$	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	1.0	1.2	2.1
$T_f(e_x)$	3.0	1.0	2.4	1.0	2.1	3.6	7.0	4.2	4.4	1.2	5.0	5.4	10.2	5.6
x	9	2	18	24	4	14	20	6	10	13	16	21	25	

$T_s(e_x)$	2.4	3.0	3.6	4.4	5.0	7.0	7.0	7.1	7.4	10.2	10.2	14.0	14.0	
$T_f(e_x)$	7.4	7.1	8.1	9.4	12.0	10.2	14.0	15.1	15.4	19.2	18.2	23.0	21.0	

利用下面的结论：**一个通讯网络的接通时间总不会小于其子网络的接通时间**. 我们将证明 23 是情形 B 中网络的最短接通时间.

证明　考虑原网络(图 25-1)的一个子图，如图 25-4 所示.

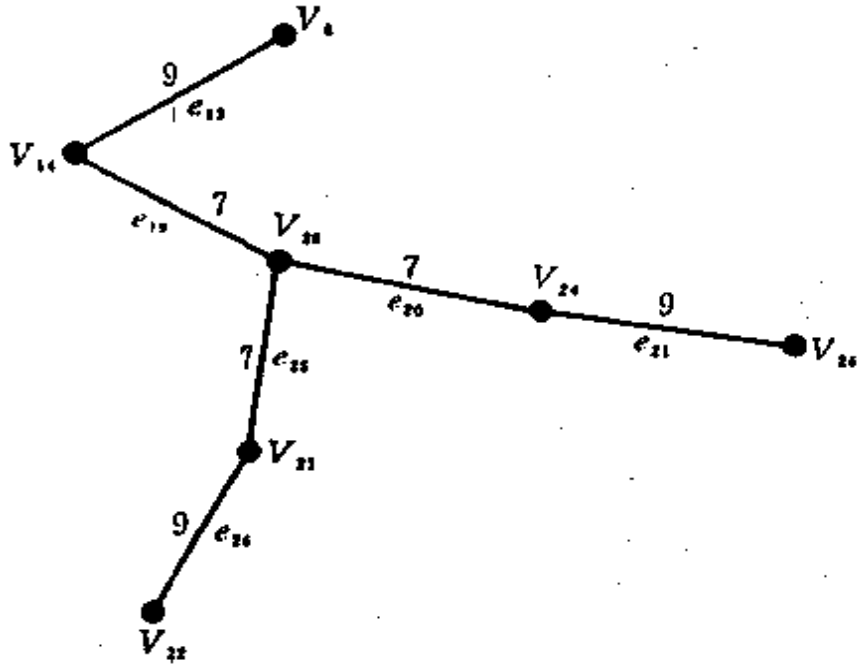

图 25-4

用一根时间轴来表示文件传输过程，边 e_{19}、e_{20}和 e_{25}具有公共顶点 V_{20}，它们必须在互不相交的时间区间内传输. 考虑到该子图的对称性，不妨假定 e_{20}在 e_{19}和 e_{25}之间传输. 因为 e_{21}与 e_{20}两边相邻，它必须在 e_{20}之前或之后传输(如图 25-5、图 25-6 所示)，从图 25-5 和图 25-6 明显看出，完成这 4 个文件的总传输时间不会小于 23，也即 23 是情形 B 接通时间的一个下限.

因此，我们给出的结果(表 25-7)即为情形 B 中的最佳传输方

案之一.

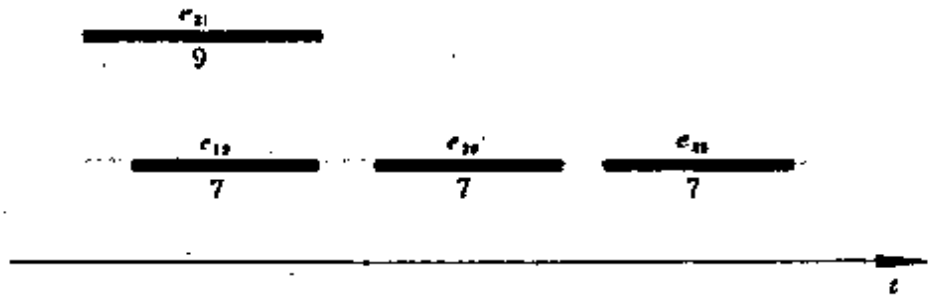

图 25-5

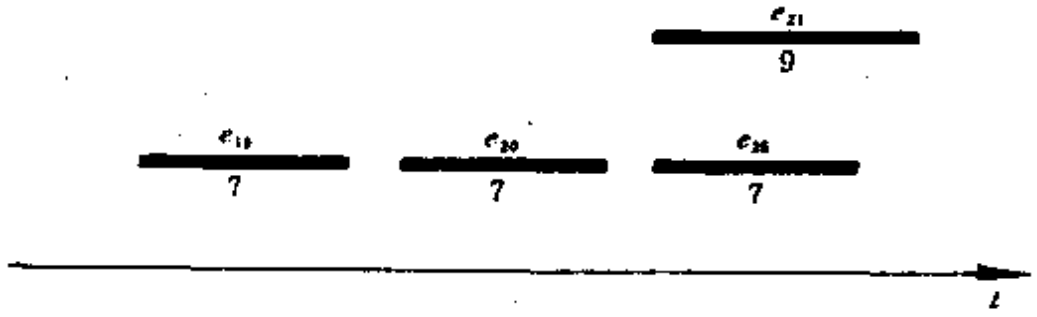

图 25-6

2.3 情形 C

在扩展的网络中，V_{24}具有最大负荷 $L(V_{24})=29.6$. 这意味着 29.6 为接通时间的下限. 注意到 V_{24}的传输容量并没有提高(仍为 1)，一种直觉使我们在假定所有计算机均未升级的情况下去寻求最佳传输方案. 令人惊讶的是，第 5 次搜索的结果便给出了接通时间为 29.6 的结果，传输方案如表 25-8 所示.

这无疑是一个意想不到的结果！它表明新计算机的配置没有有效地减少接通时间. 要尽可能缩短接通时间，必须合理配置新计算机，正如我们下面指出的那样.

表 25-8　情形 C 的最佳传输方案

x	5	11	15	17	23	27	34	36	41	42	3	12	38	26
$T_t(e_x)$	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	1.0	1.0	1.2
$T_f(e_x)$	1.0	1.0	2.1	3.6	4.4	1.2	2.4	3.7	3.0	6.1	5.0	5.4	7.6	10.2
x	30	1	33	9	14	24	2	28	39	7	16	19	8	21
$T_t(e_x)$	2.1	2.4	2.4	3.0	3.0	4.4	5.4	5.4	5.4	6.1	6.2	6.2	9.3	9.4
$T_f(e_x)$	7.3	5.4	9.4	8.0	6.2	9.4	9.5	11.4	10.5	9.3	14.2	13.2	11.7	18.4
x	37	32	4	40	13	25	29	31	6	10	18	22	35	20
$T_t(e_x)$	9.5	10.2	10.5	10.5	13.2	13.2	14.2	14.2	17.6	17.6	18.3	18.4	18.4	22.6
$T_f(e_x)$	15.8	14.2	17.5	17.6	22.2	20.2	15.3	18.3	25.6	25.6	22.8	22.6	27.4	29.6

§5　进一步的考虑

1. 新计算机的重配置

如果新计算机是用来减少接通时间，它们就应该配置到那些传输信息量较大的部门去. 我们先假设所有计算机的传输容量均为 1，计算出每台机器的负荷为 $\sum_{e_x} T(e_x)$，然后选出负荷最大的 8 台计算机所在的部门，传输容量为 3 的新计算机被派到负荷最大的地方(V_{24})，其余 7 台容量为 2 的新计算机配置到另外 7 个负荷较大的部门去. 重新分配后各个部门计算机的传输容量如表 25-9 所示.

表 25-9　新计算机重新配置后各部门计算机的传输容量

y	1	2	3	4	5	6	7	8	9	10	11	12	13	14
$C(V_y)$	1	1	2	1	1	1	1	2	1	2	1	1	1	2
y	15	16	17	18	19	20	21	22	23	24	25	26	27	28
$C(V_y)$	1	2	1	1	1	2	1	1	1	3	1	2	1	1

在改进后的计算机系统中，计算机的最大负荷为 19.3，用同样的算法得到最小接通时间为 19.3(相应传输方案如表 25-10 所示)，这与原来的 29.6 相比，减少了 35%.

表 25-10　情形 C 中新计算机重新配置后的最佳传输方案

x	1	2	3	7	8	11	12	14	17	18	19	20	21	22
$T_s(e_x)$	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
$T_f(e_x)$	3.0	4.1	4.0	3.2	2.4	1.0	4.4	3.2	3.6	4.5	7.0	7.0	9.0	4.2
x	23	26	38	9	6	13	15	39	37	33	36	28	35	16
$T_s(e_x)$	0.0	0.0	1.0	2.4	3.2	3.2	3.2	4.0	4.1	4.2	4.2	4.4	4.4	5.3
$T_f(e_x)$	4.4	9.0	7.6	7.4	11.2	12.2	5.3	9.1	10.4	11.2	7.9	10.4	13.4	13.3
x	30	42	10	41	24	25	4	34	31	40	29	32	27	5
$T_s(e_x)$	5.3	7.0	7.4	7.4	9.0	9.0	9.1	10.4	10.5	11.2	13.3	14.6	16.0	18.3
$T_f(e_x)$	10.5	13.1	15.4	10.4	14.0	16.0	16.1	12.8	14.6	18.3	14.4	18.6	17.2	19.3

2. 随机传输模型

我们的算法基于一个基本假定：文件传输时间是已知的，并且不随每天而变化，然而在实际工作中，日传递信息量有时会有较大的变化甚至根本不知道，这时制定一个整体传输方案是没有意

义的.假定计算机可以通过网络监测其他计算机的工作状态，我们认为每台计算机只需遵守下列规则：只要相邻的顶点(计算机)允许传输，就立刻与之进行文件传输，如果有多种选择，则任意选择其中一个.

我们对情形 B 用上述方法模拟了 500 次，结果表明最短接通时间为 23，最长接通时间为 31.2.图 25-7 显示了各种接通时间的出现几率，从中看出，大部分传输过程所需要的接通时间都接近或等于 23.

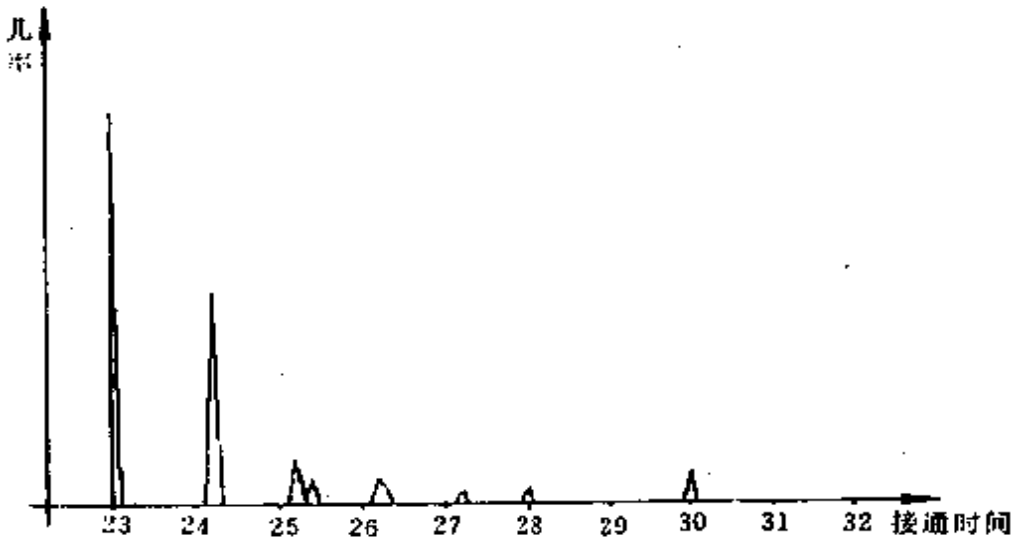

图 25-7

§6 稳定性的提高

由于各种实际因素的作用，文件传输时间的波动是不可避免的.如果某次传输发生了延迟，那么接在其后的所有文件的传输都得受到影响.(甚至会导致整个系统的混乱!)为了提高系统的稳定性，我们假定文件传输时间 $T(e_x)$的波动范围为 $T(e_x)(1\pm5\%)$，用 $T(e_x)$的波动上限代替 $T(e_x)$.对于情形 A、B 和 C，我们用同样

的算法得到接通时间为 3.2,23.8 和 30.8. 以情形 B 为例,其传输方案如表 25-11 所示.

表 25-11　提高稳定性后的情形 B 传输方案

x	1	5	8	11	15	17	19	22	23	27	3	12	26	7
$T_s(e_x)$	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.1	1.1	1.3	2.5
$T_f(e_x)$	3.0	1.0	2.4	1.0	2.1	3.6	7.0	4.2	4.4	1.2	5.1	5.5	10.3	5.7
x	9	2	18	24	4	14	20	6	10	13	16	21	25	
$T_s(e_x)$	2.5	3.2	3.8	4.6	5.3	7.4	7.4	7.5	7.8	10.8	10.8	14.8	14.8	
$T_f(e_x)$	7.5	7.3	8.3	9.6	12.3	10.6	14.4	15.5	15.8	19.8	18.8	23.8	21.8	

用传输时间的波动上限来代替文件传输时间后,接通时间稍有加长,但某次传输的延迟所造成的影响将随着传输过程的进行很快地被驰豫掉. 我们以情形 B 中 e_{19} 不同程度的波动(以 $\triangle T$ 表示)来检验整个系统的稳定性. 表 25-12 说明修正后的模型具有相当好的稳定性.

表 25-12　稳定性提高后情形 B 中 e_{19} 传输时间波动造成影响的分析

$\triangle T(e_{19})$	受影响的文件传输	接通时间波动
<0.0		0.0
0.0～0.4	e_{14}　e_{20}	0.0
0.4～0.8	e_{14}　e_{20}　e_{21}　e_{25}	0.0
$0.8+t$	e_{14}　e_{20}　e_{21}　e_{25}	t

§7　优缺点分析

模型提供了一个普适的算法,它适用于网络结构、文件传输时间和计算机传输容量都是任意的情形. 图 25-8 是一个任意构造的

网络，其中计算机的最大负荷为 22（接通时间下限）. 用同样的算法我们很快得到了接通时间为 22 的传输方案，证实了模型的普适性.

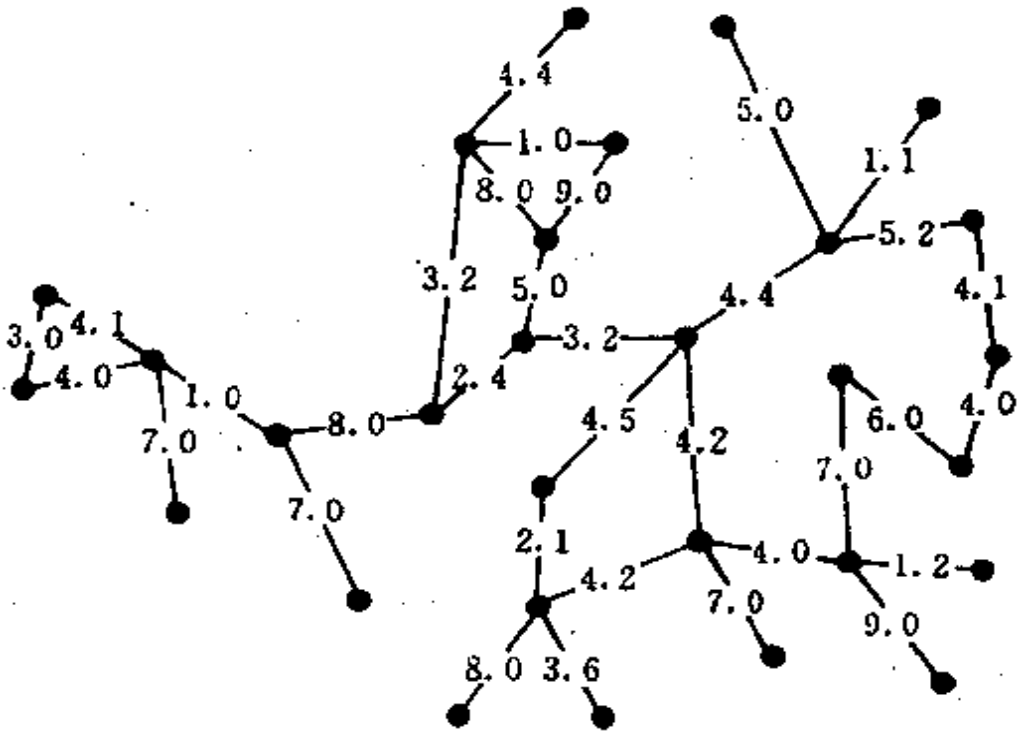

图 25-8　一个随意构造的计算机网络

当网络规模很大时，深度优先搜索是不切实际的，因而我们的算法并不能保证得到最优解，事实上，NP-问题是不存在有效的多项式算法的. 尽管如此，考虑优先权的贪婪算法还是可以被接受的，因为算法结果的上限为 $2L_{max}-\min T(e_i)$，而且正如随机传输模型的模拟结果所示，在大多数情况下，我们都能得到很好的结果.

必须指出，我们的模型不适用于文件传输存在相互依赖性的情形.

参 考 文 献

[1] Brian Thomas ECK and Machael Pinedo, 1993, *On The Minimization of The Makespan Subject to Flowtime Optimality*, Operations Research, 797—801.

[2] Graham R. L, 1976, *Bound on Multiprocessing Timing Anomalies*, SIAM J. App. Math. 17, 416—429.

[3] Michel Gondran and Michel Minoux, 1979, *Graphs and Algorithms*, John Wiley & Sons.

[4] J. A. Bondy and U. S. R. Murty, 1976, *Graph Theory with Applications*, the Macmillan Press Ltd.

[5] Vizing V. G., 1964, *On an Estimation of the Chromatic Class of a P-Graph* (*Russian*), Diskret. Analiz. 3,25—30.

[6] Keming Zhang et. al, *Solution to Exercises in Graph Theory with Applications*, Qinghua University Press.