实验目的与要求

1. 掌握最大流算法思想。
2. 学会用最大流算法求解应用问题。
3. 解释流网络的构造原理。
4. 解释为什么最大流能解决这个问题。
5. 给出上面四个球队的求解结果。
6. 尽可能实验优化的最大流算法。

一、实验原理

1. 棒球赛问题

(1) 介绍

1996 年 9 月 10 日，《旧金山纪事报》的体育版上登载了《巨人队正式告别 NL 西区比赛》一文，宣布了旧金山巨人队输掉比赛的消息。当时，圣地亚哥教士队凭借 80 场胜利暂列西区比赛第一，旧金山巨人队只赢得了 59 场比赛，要想追上圣地亚哥教士队，至少还得再赢 21 场比赛才行。然而，根据赛程安排，巨人队只剩下 20 场比赛没打了，因而彻底与冠军无缘。

有趣的是，报社可能没有发现，其实在两天以前，也就是 1996 年 9 月 8 日，巨人队就已经没有夺冠的可能了。那一天，圣地亚哥教士队还只有 78 场胜利，与洛杉矶道奇队暂时并列第一。此时的巨人队仍然是 59 场胜利，但还有 22 场比赛没打。因而，表面上看起来，巨人队似乎仍有夺冠的可能。然而，根据赛程安排，圣地亚哥教士队和洛杉矶道奇队互相之间还有 7 场比赛要打，其中必有一方会获得至少 4 场胜利，从而拿到 82 胜的总分；即使巨人队剩下的 22 场比赛全胜，也只能得到 81 胜。由此可见，巨人队再怎么努力，也不能获得冠军了。

在美国职业棒球的例行赛中，每个球队都要打 162 场比赛（对手包括但不限于同一分区里的其他队伍，和同一队伍也往往会有多次交手），所胜场数最多者为该分区的冠军；如果有并列第一的情况，则用加赛决出冠军。在比赛过程中，如果我们发现，某支球队无论如何都已经不可能以第一名或者并列第一名的成绩结束比赛，那么这支球队就提前被淘汰了（虽然它还要继续打下去）。从上面的例子中可以看出，发现并且证明一个球队已经告败，有时并不是一件容易的事。

关于这个事情有一个有趣的故事，下面是一段对话：

“看到上周报纸上关于爱因斯坦的那篇文章了吗？……有记者请他算出三角比赛的数学公式。你知道，一支球队赢得了很多剩余的比赛，其他球队则赢这个赢了那个。这个比赛到底有多少种可能性？哪个球队更有优势？”

“他到底知道吗？”

“显然他知道的也不多。上周五他选择道奇队没有选巨人队。”

TUESDAY, SEPTEMBER 10, 1996

San Francisco Chronicle

The Gate
Sports Online
► http://www.sfgate.com

SPORTING G

49ers, Young Get Big Bre

Quarterback m

By Gary Swan
Chronicle Staff Writer

The bye week has come at a perfect time for the 49ers and quarterback Steve Young. If they had a game next Sunday, there's a good chance Young would not play.

Giants Officially Leave the NL West Race

By Nancy Gay
Chronicle Staff Writer

With the smack of another National League West bat 500 miles away, the Giants' run at the division title ended last night, just as they were handing the visiting St. Louis Cardinals an even bigger lead in the NL Central.

CARDINALS 6
GIANTS 2

In San Diego, Greg Vaughn's three-run homer in the eighth pushed the Padres over the Pirates and officially shoved the rest of the Giants' season into the background. On the heels of their tedious 6-2 loss before an announced crowd of 10,307 at Candlestick Park, the Giants fell 19½ games off the lead.

As it is, the worst the Padres (80-65) can finish is 80-82. The Giants have fallen to 59-83 with 20 games left; they cannot win 80 games. Coming off a miserable 2-8 mark on a three-city road trip that saw their road record drop to 27-47, the Giants were hoping to get off on the right foot in their longest homestand of the year (15 games, 14 days).

Financing in Place For Giants' New Stadium
SEE PAGE B1, MAIN NEWS

"Where we are, you're going to be eliminated sooner or later," Baker said quietly. "But it doesn't alter the fact that we've still got to play ball. You've still got to play hard, the fans come out to watch you play. You've got to play for the fact of loving to play, no matter where you are in the standings.

"You've got to play the role of spoiler, to not make it easier on

GIANTS, Page D5 Col. 2

图 1 实验介绍配图图

(2) 实验任务

Team i	Wins w_i	Losses l_i	To play r_i	Against = r_{ij}			
				Atl	Phi	NY	Mon
Atlanta	83	71	8	-	1	6	1
Philly	80	79	3	1	-	0	2
New York	78	78	6	6	0	-	0
Montreal	77	82	3	1	2	0	-

图 2 四个球队的比赛情况

图二是四个球队的比赛情况，现在的问题是哪些球队有机会以最多的胜利结束这个赛季？可以看到蒙特利尔队因最多只能取得 80 场胜利而被淘汰，但亚特兰大队已经取得 83 场胜利，蒙特利尔队因为$w_i + r_i < w_j$而被淘汰。费城队可以赢 83 场，但仍然会被淘汰。如果亚特兰大输掉一场比赛，那么其他球队就会赢一场。所以答案不仅取决于已经赢了多少场比赛，还取决于他们的对手是谁。

请利用最大流算法给出上面这个棒球问题的求解方法。

2. 最大流算法

(1) 算法介绍

首先明确问题是要求解**哪些球队有机会以最多的胜利结束这个赛季**，那么我们需要寻找的就是哪支队伍是 n 支队伍中排名第一的队伍，等同的，我们可以看作让不是第一的队伍都被淘汰，下面介绍两种淘汰方式。

第一种是明显的淘汰方式，例如图二，如果$w_i + r_i < w_j$，那么 i 队就一定会被淘汰。

第二种则是$w_i + r_i > w_j$，这种情况不能直接看出淘汰方式，我们就需要借助最大流来进行判断。例如图 2 若 Atl 队与 NY 队的对战会至少有一方获得 84 场胜利，总有一队超过 Phi 的最大可赢场数 83。

下面先介绍网络流的知识，首先，网络流的概念是建立在有向图上的，这张图包含一个起点 s（source 源点）、一个终点 t（sink 汇点）和一些管道（有向路径），管道中包含实际流量 f（flow）和容量 c（capacity）。而在实际操作中，我们还需要为管道标记上一个剩余量 r（residual = c - f）。而最大流算法的目标就是找到一个从起点流出量最多（=汇点接收量最多）的算法，而如果只是简单的不按任何给定算法来选择道路，可能会出现如图 3 一样的错误情况。

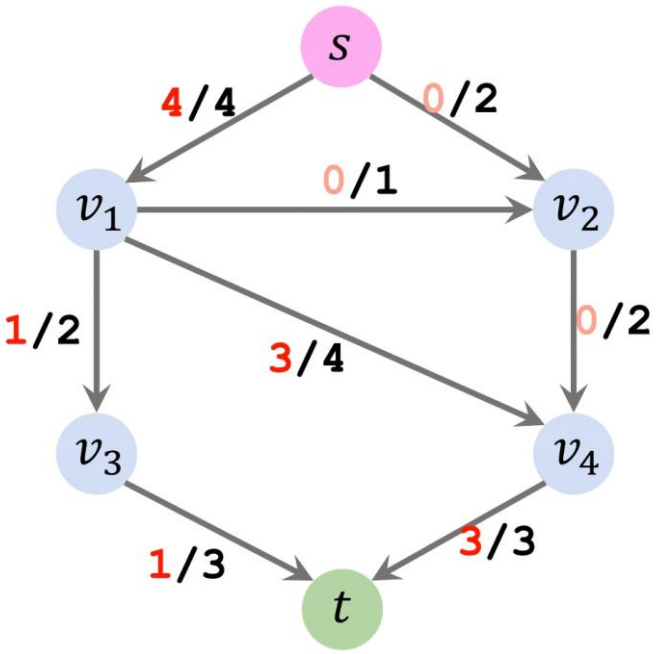


图 3 最大流为 5 的一种仅能获得 4(3+1)的解法

在操作中，我们一般会将剩余可流量 r=0 的边进行删除，但这就会造成图 3 这种错误：没有从起点到终点的路径了，但所得并不是最大流。

(2) Ford-Fulkerson 算法

而 Ford-Fulkerson 算法（简称 FF 算法）在 1956 年给出了一种解决方案，即添加一条容量=实际流量的“反流”，简单来说就是让流量可以倒流回去且每次递增 1，如图 4，但也可以看到，这种创建反流的方式是一次次递增的，可以达到最大流的大小，但这种创建的方式

可能增加搜索路径的时间，时间复杂度来到$O(em)$，其中 e 是路径的数量，m 是最大流的大小。

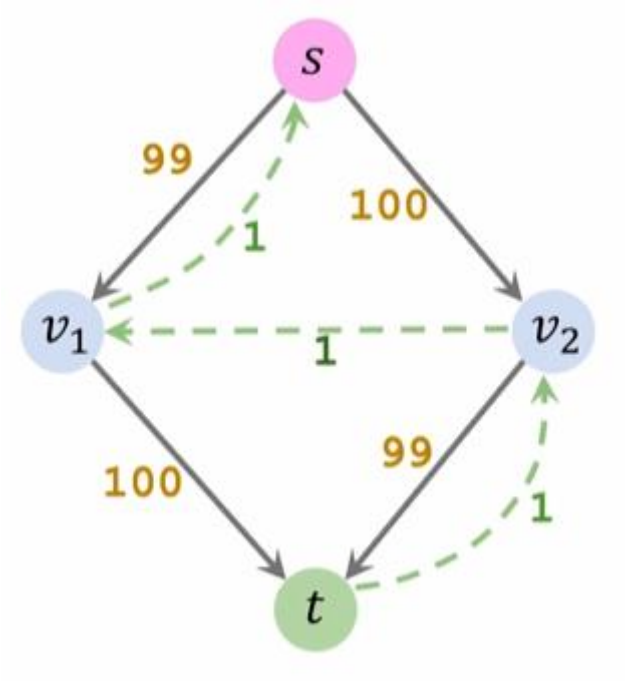


图 4 FF 算法例子

(3) Edmonds–Karp 算法

后人就考虑优化，Edmonds–Karp 算法（简称 EK 算法）在 1972 年被提出，如图 4，FF 算法可能存在搜索到较长路径（只考虑可达，不考虑长度），而 EK 算法则是寻找最短长度（将每条边看作相等长度路径），EK 算法常被看作 FF 算法的特例，它的时间复杂度是$O(e^2n)$，其中 e 是边数，n 是点数，他相比 FF 算法的优点在于，不会因为最大流的大小影响时间复杂度。

(4) Dinic 算法

而实际上，在 1970 年 Dinitz 就提出了 Dinic 算法，这种算法依靠 BFS 分层来标记每个点到原点的距离，如图 5，每次寻找当前层级可达的所有点，并将他们标记为下一层级，注意，分层的时候不标记相同层级间的道路，这种算法在每次建造反流后都对他进行分层，然后再寻找可达通路，时间复杂度来到$O(n^2e)$，其中 e 是边数，n 是点数。

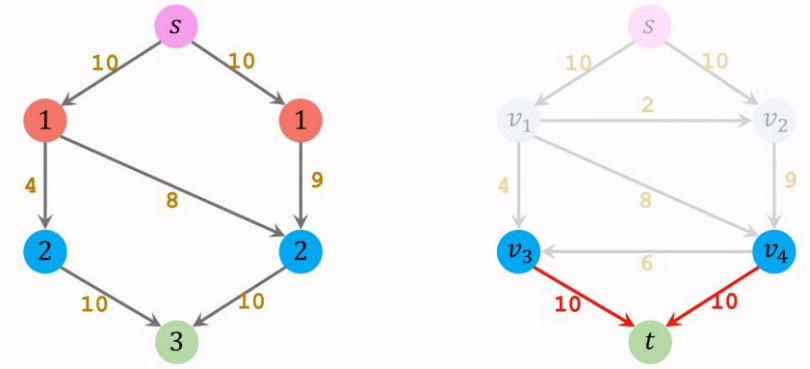


图 5 Dinic 算法例子

3. 为什么最大流能解决这个问题

(1) 简单介绍

每个队伍都是有获胜上限的，与最大流的思想类似。

(2) 对应关系

首先明确，我们需要对每一个队伍建图，每个图需要包含 4 个部分

1.源点 S，虚拟起点。

2.比赛点，除了自身（标号 k）以外的 n-1 个队伍两两比赛，共有$\frac{(n-1)(n-2)}{2}$场，而源点到比赛点的权值（容量）则是 a[i][j]（图 2 中的r_{ij}）。

3.队伍点，比赛点到队伍点的权值（容量）是无限大的，这是因为需要传达正确的比赛结果到队伍中，比赛点到汇点的权值（容量）应是 w[k]+r[k]-w[j]，这表示 k 队的最大可赢场-j 队的已赢场（j 队全输，k 队全赢的情况，即 k 队可能从 j 队取得的胜场）。

4.汇点 T，收集所有的流量，若收集到的流量>=可发出的流量，表明结果是可分配的，那么 k 队就是可以被接受的（未被淘汰）。

二、实验步骤

实验先把给定的图例转化为输入数据如图 6 所示，进行测试，使用 C++中的 chrono 头文件进行记时。

```
4
Atlanta 83 71 8 0 1 6 1
Philly 80 79 3 1 0 0 2
NewYork 78 78 6 6 0 0 0
Montreal 77 82 3 1 2 0 0
```

图 6 给定的输入数据

三、实验结果与分析

1. 最大流算法（Dinic 实现）

(1) 结果截图

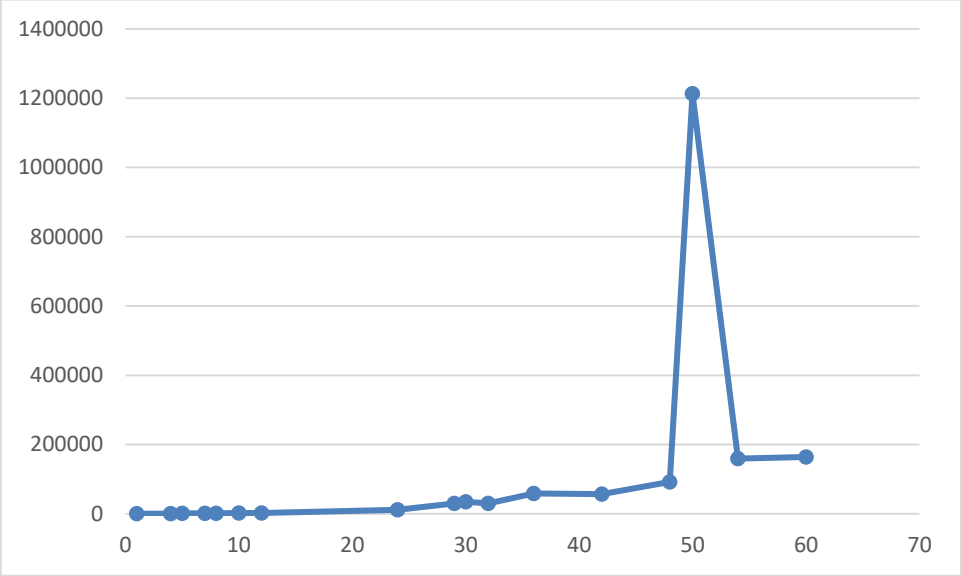
给定的输入数据（图 2、6）的运行结果如图 7 所示：

图 7：最大流算法 给定的图 2、6 实验结果

下面对普林斯顿大学给定的测试数据进行测试并分析运行时间，如表 1 所示

(2) 结果分析

表 1 最大流算法 运行时间与规模表



其中，可以看到，运行时间基本与点数成正相关，而算法的时间复杂度是$O(n^2e)$，在 n=50 处的不寻常数据点就是因为该数据建的边较多。

四、实验结论与体会

本次实验复习了图的连通性相关内容、掌握网络流和最大流的基本原理与三种不同的求最大流算法（FF、EK、DC），加强了实际问题到程序的抽象能力，使用最大流思想解决了棒球比赛问题。