浅谈网络流算法的应用

关键字： 网络流、构造、优化

【 引　　言 】

【 正　　文 】

【 小　　结 】

湖南省长沙市长郡中学　金 恺

浅谈网络流算法的应用

引言

图论算法在信息学竞赛当中扮演着相当重要的角色，它的分支之多、应用范围之广令所有其它算法都望尘莫及。而网络流算法正是图论算法中的一个重要分支，它特点突出、作用显著，因此应用范围十分广范，在近年来的各级别信息学竞赛中更是层出不穷，并且它还将占据着越来越重要的地位。

本文旨在通过剖析若干应用实例，逐步阐述网络流算法的构造、优化原则和方法，对网络流算法作更深入、更彻底的认识。

浅谈网络流算法的应用

正文

相信大家早已对网络流算法的轮廓以及解法都进行了研究，这里只是我在平时做题过程中的一些体会，主要针对它的***应用范围***、具体***应用方法***以及诸多的***优化技巧***。

例一	赛车问题	例二	列车调度
例三	餐厅问题	例四	终极情报网

阿 P 与阿 Q 都是四驱车好手，他们各有 N 辆四驱车。为了一比高下，他们约好进行一场比赛。

这次比赛共有 M 个分站赛，赢得分站赛场次多的获得总冠军。

每个分站赛中，两人各选一辆自己的赛车比赛。分站赛的胜负大部分取决于两车的性能，但由于种种原因（如相互之间的干扰），性能并不是决定胜负的唯一指标，有时会出现 A 赢 B 、 B 赢 C 、 C 赢 D 、 D 又赢 A 的局面。幸好有一种高智能机器，只要给定两辆四驱车，就能立刻判断谁会赢，在总比赛前它就已经把阿 p 的每辆车与阿 q 的每辆车都两两测试过了，并且还把输赢表输入了电脑。

另外，由于比赛的磨损，每辆四驱车都有自己的寿命（即它们能参加分站赛的场次），不同的四驱车寿命可能不同，但最多不会超过 50 场。两辆四驱车最多只能交手一次。

现给定 N 、 M （ 1<=N<=100 ， 1<=M<=1000 ）、 N*N 的一个输赢表、每辆四驱车的寿命，并假设每次分站赛两人都有可选的赛车，请你计算一下阿 P 最多能够赢得多少个分站赛。

1、建立 N 个点代表阿 P 的 N 辆车，分别以 1 到 N 标号；

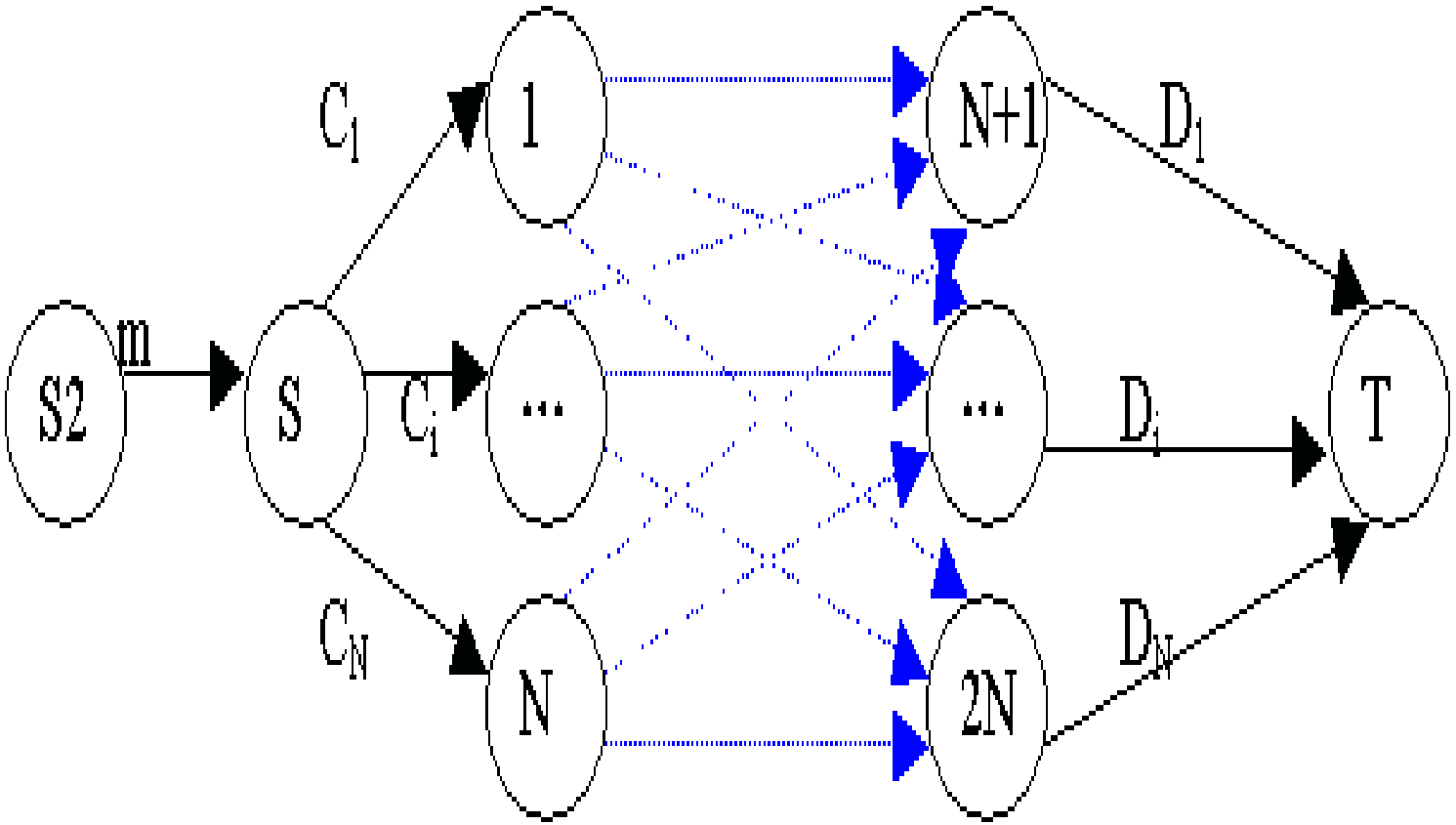
2、建立 N 个点代表阿 Q 的 N 辆车，分别以 N+1 到 2N 标号；

3、如果阿 P 的第 I 辆车能够跑赢阿 Q 的第 J 辆车，则加一条弧 I→N+J，容量为 1，表示两辆四驱车最多只能交手一次；

4、增加一个源点 S， S 与 1 到 N 中的每一个结点 I 都连一条弧 S→I，容量为阿 P 的第 I 辆车的寿命；

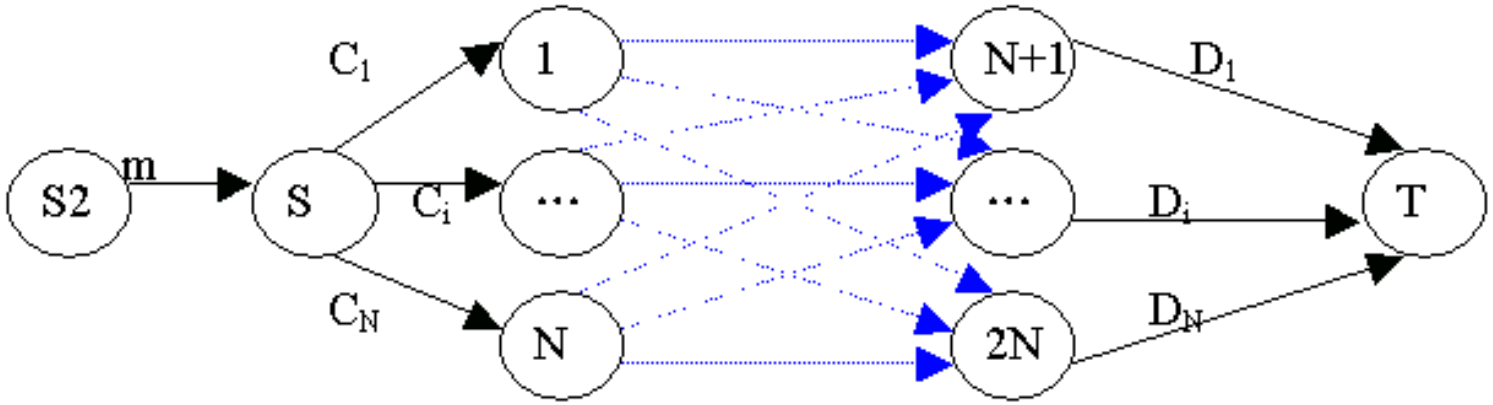
5、增加一个汇点 T， N+1 到 2N 中的每一个结点 N+J 到 T 都连一条弧 N+J→T，容量为阿 Q 的第 J 辆车的寿命；

6、再增加一个源点 S2，加一条弧 S2→S，容量为 M，表示最多 M 场分站赛 。



C_i表示阿 P 的第 I 辆车的寿命，D_i表示阿 Q 的第 I 辆车的寿命，蓝色弧的容量为 1

例一 | 赛车问题 | 问题描述 | 构　图 | 优　化



C_i表示阿 P 的第 I 辆车的寿命，D_i表示阿 Q 的第 I 辆车的寿命，蓝色弧的容量为 1

普通算法：保存流量与容量就需要（2N+3）*（2N+3）*2Bits

优化：总共只有四类弧：

1、S2→S　　2、S→I(i∈1..N)

3、I→J(i∈1..N，j∈N+1..2N)　　4、J→T(j∈N+1..2N)

最多也不过 1+N+N+N*N=（N+1）*（N+1）条弧

S2 这个源点与 S2→S 这条弧都可以不要，只需规定最多扩展 M 次流量即可

某货运车站有 **n**（ **n≤20** ）个车道，由于车道的长度有限，每个车道在某一时刻最多只能停靠一列货运列车。车站正常运行后，每天将有 **m**（ **m≤100** ）列货运列车从车站**经过**，其中第 **i** 列列车到达车站的时间为 **Reach[i]**，列车上装有价值 **Cost[i]** 的货物。

如果准许列车 **i 进站**，则 **BackStreet** 车站将获得 **1%×Cost[i]** 的收益，但由于货物的搬运时间，该列车将在车站停留一段时间 **Stay[i]**，这段时间内，列车将占用车站中的某一个车道；否则列车直接出站，但这样车站将得不到一分钱。

你的任务就是：合理的安排列车的进站与出站，使得车站的总获利最大。

{ 如果列车 **a** 从第 **i** 车道离开时，列车 **b** 刚好到站（即 **Reach[a]+Stay[a] =Reach[b]** ），则它不能进入第 **i** 车道。 }

例：

有 2 个车道，5 列列车，它们的进站时间，车上价值，停留时间如下：

例 车编号	进站时间	车上价值	停留时间
1	2	5	2
2	3	4	2
3	5	6	3
4	2	2	2
5	6	3	2

那么最大获益为 0.18

{即让列车 1、3 进站走第一车道，列车 2、5 走第二车道}

例：

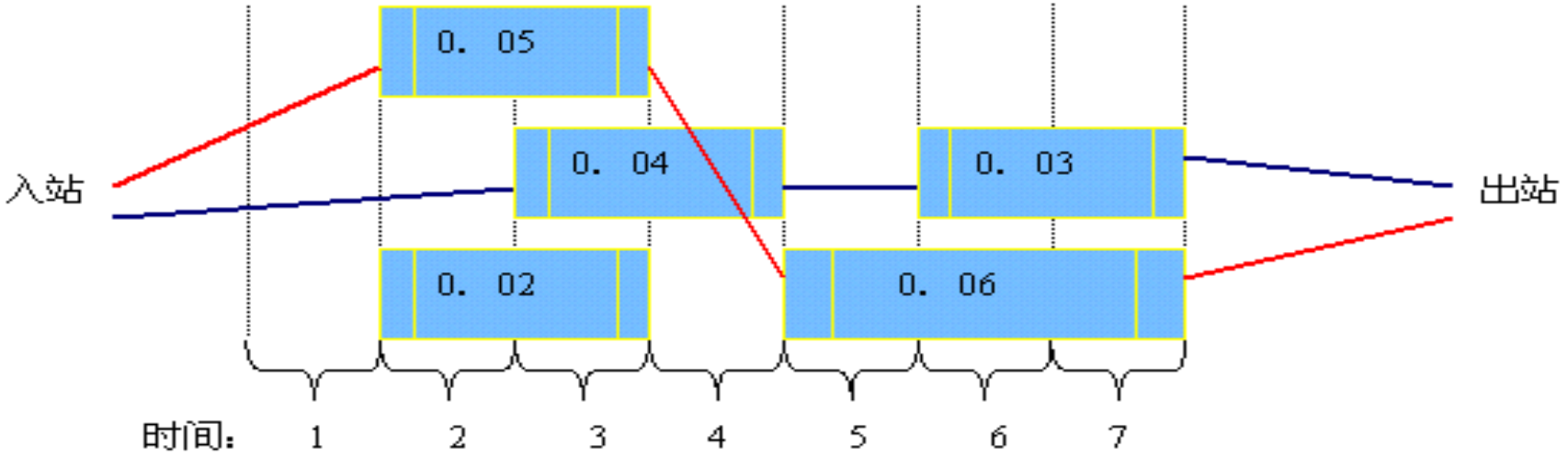
有 2 个车道，5 列列车，它们的进站时间，车上价值，停留时间如下：

例 车编号	进站时间	车上价值	停留时间
1	2	5	2
2	3	4	2
3	5	6	3
4	2	2	2
5	6	3	2

那么最大获益为 0.18

{即让列车 1、3 进站走第一车道，列车 2、5 走第二车道}

先看样例，我们很容易做出如下的图例。



红色直线表示先停入列车 1 再停入列车 3

蓝色直线表示先停入列车 2 再停入列车 5

建图方法：

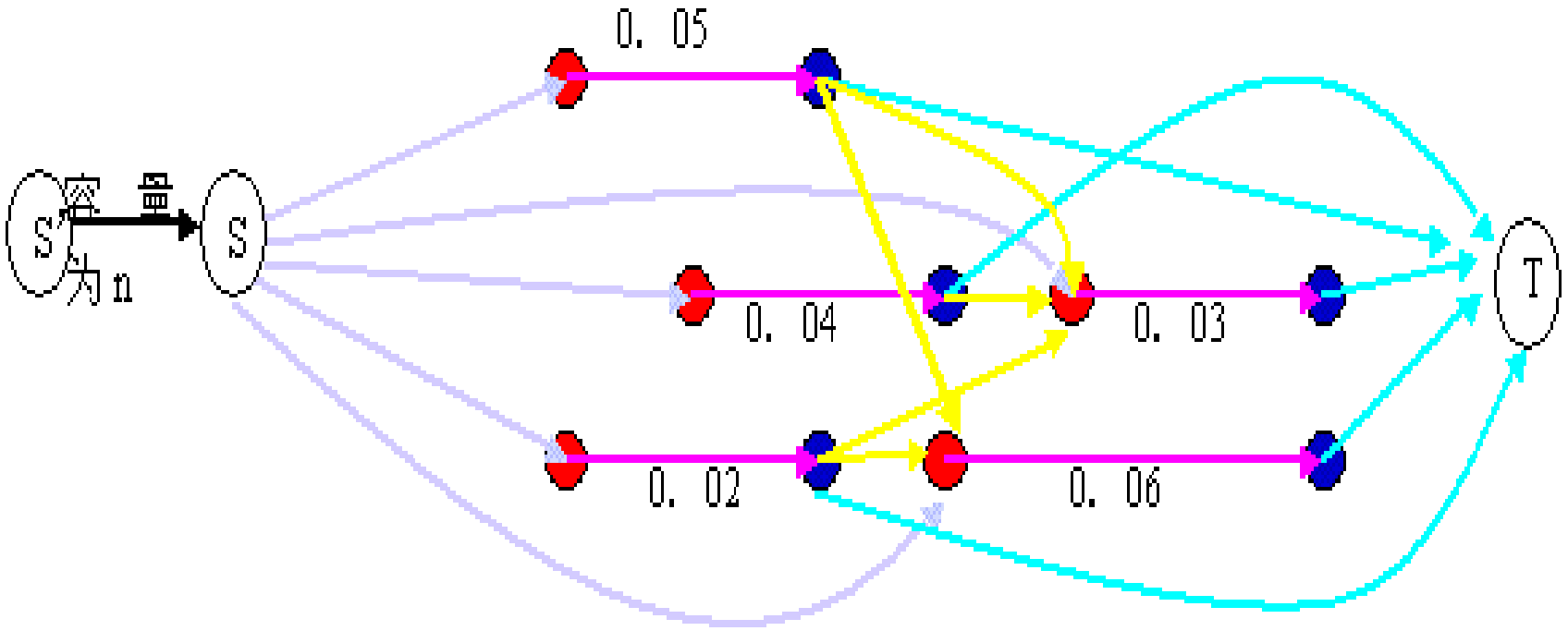
1、将每一列列车拆成两个点，如第 i 列列车拆成点 i 和 i'； i 到 i' 之间加一条容量为 1，费用为 **1%**×Cost[i] 的弧 i→i'；｛若 i→i' 的弧的流量为 1，则表示该列火车进站，并获得 **1%**×Cost[i]｝

2、增加一个源点 S， S 与每个点 i 连一条容量为 1 费用为 0 的弧 S→i；｛如果 S→i 的流量为 1，则表示列车 i 作为某个车道的第一列入站的列车｝

3、再增加一个源点 S'， S'→S 的容量为 n，表示有 n 个车道；

4、增加一个汇点 T，每个点 i' 与 T 连一条容量为 1 费用为 0 的弧 i'→T；｛如果 i'→T 的流量为 1，则表示列车 i 作为某个车道的最后一列入站的列车｝

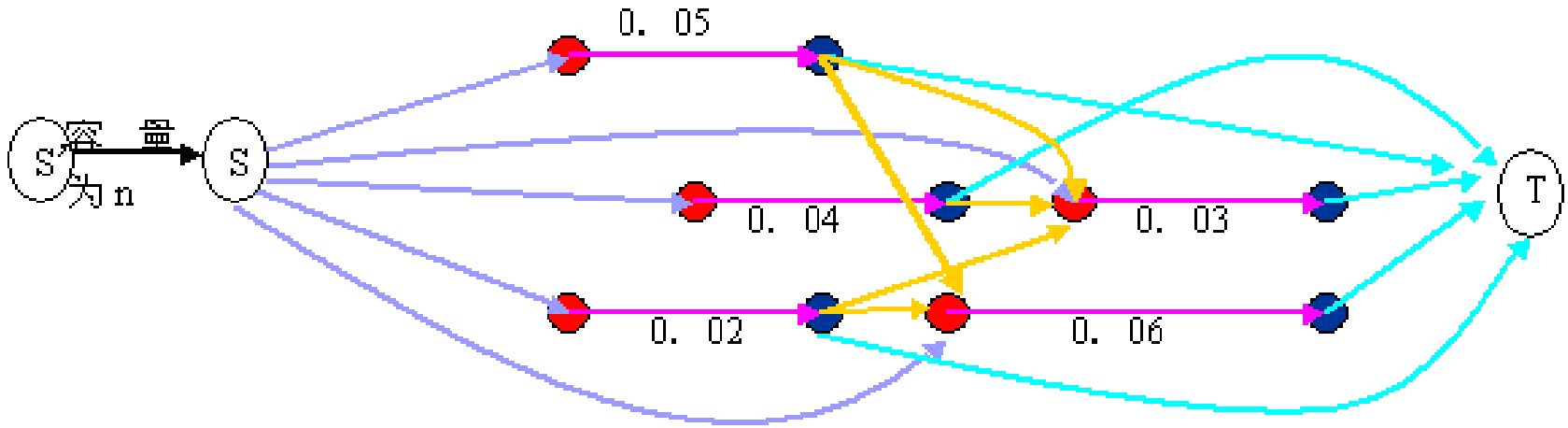
5、对于所有的 i 和 j（i≠j，且 i， j∈1..m），如果 Reach[i]+Stay[i]



粉红色箭头上的数字表示费用

未标数字的弧的费用为 0

未标明容量的弧的容量为 1



优化：

1. 其实没有必要增加一个源点 S’ 及一条弧 S’→S，因为每次修改可增广轨上弧的流量时，都是以 1 作为可修改量，故只要规定最多找 n 次增广轨，就可以确保占用的车道数小于等于 n 了。

2．第 1 步优化以后，所有弧的容量都变为 1，非常便于处理：可以把每条弧的容量和流量用 1 个 Byte 类型存储：

0—— 容量为 0，流量为 0；

1—— 容量为 1，流量为 0；

2—— 容量为 1，流量为 1；

公司在连续的 n 天内，每天对毛巾有一定的需求量，第 i 天需要 Ai 个。毛巾每次使用前都要消毒，新毛巾已消毒。消毒有两种方式， A 种方式的需要 a 天时间， B 种方式 b 天时间（ b>a ） ,2 种方式的价格分别为 fa 、 fb ，购买一条新毛巾价格为 f （ f>fa>fb ） , 求用最少的钱满足每天的需要。

构点：

按照常规的方法，我们把每一天拆成两个点。即第 i 天拆成顶点 i 和顶点 i’（i∈1..N），分别看成一天的开始与结束，另加一个源点 S 和一个汇点 T 。

连边：

1 、源点 s 至顶点 1 之间连一条弧，该弧的容量为 0 到∞，费用为 f ，它的流量表示总共购买的新毛巾数，因为在任意一天购买新毛巾与在第一天购买等效，所以可以在第一天买下需要的所有新毛巾；

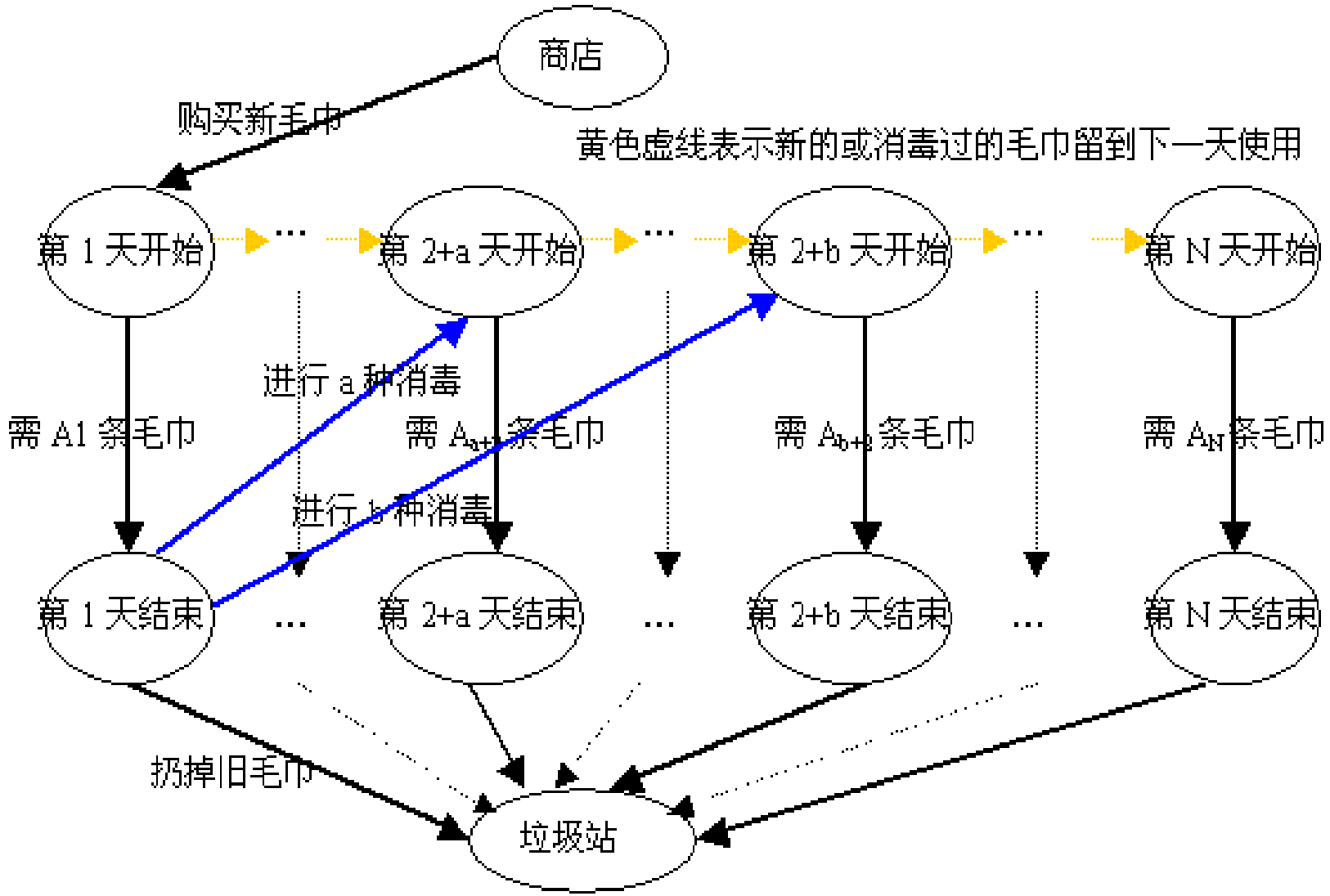
2 、顶点 i （ i≤n-1 ）与 i+1 间连一条弧，容量为 0 到∞，费用为 0 ，表示第 i 天开始时的新毛巾或消过毒的可以留到下一天使用；

3、顶点 i 到顶点 i’间连一条弧，弧的上界为 Ai， 下界也为 Ai，费用为 0，下界为 Ai 表示这一天至少需要 Ai 条毛巾，又因为超过 Ai 条是没有必要的，所以上界也为 Ai；

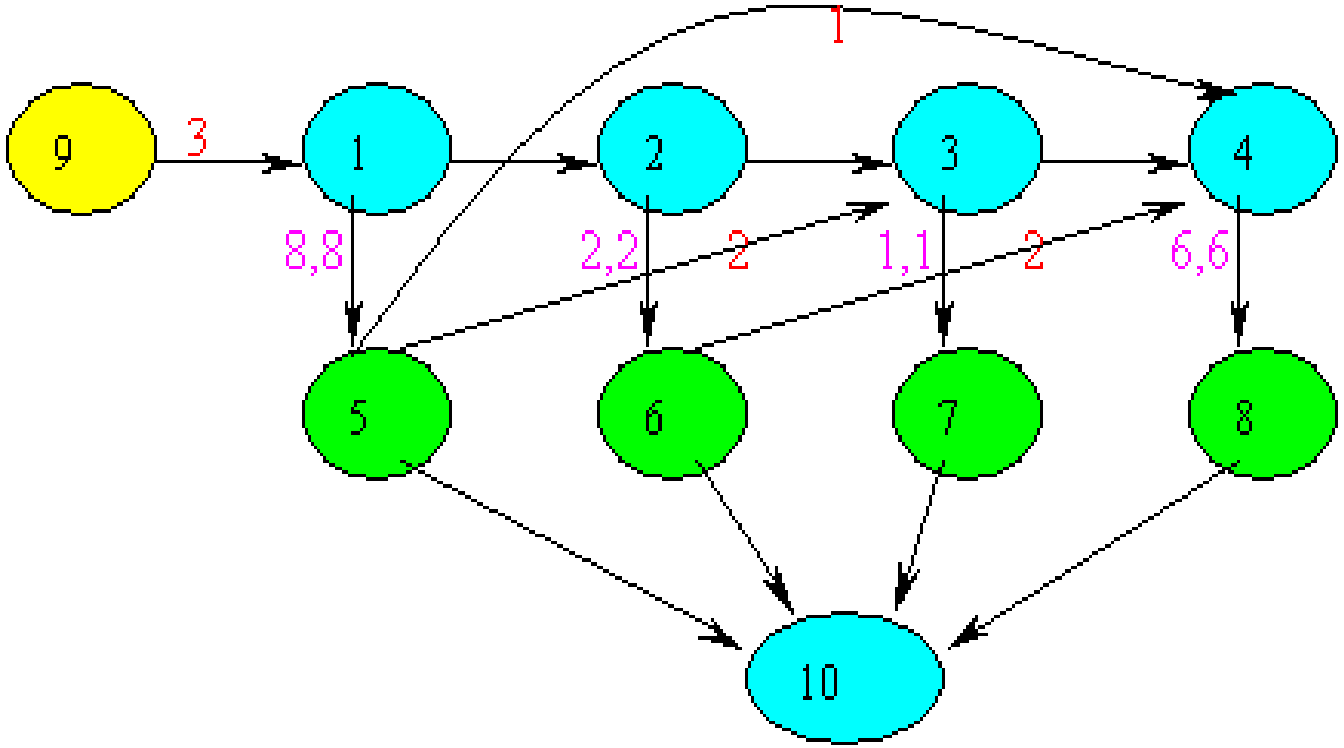
4、i’与汇点 t 间连一条弧，该弧的容量为 0 到∞，费用为 0，表示第 i 天用过了的毛巾可以不再进行消毒而直接扔掉；

5、i’(i<n-a) 到 i+a+1 连一条弧，弧的容量为 0 到∞，费用为 fa。表示第 i 天使用的毛巾可以进行 A 种消毒，并在 a 天之后可以重新使用；

6、i’(i<n-b) 到 i+b+1 连一条弧，弧的容量为 0 到∞，费用为 fb。表示第 i 天使用的毛巾可以进行 B 种消毒，并在 b 天之后可以重新使用 。


商店
购买新毛巾
黄色虚线表示新的或消毒过的毛巾留到下一天使用
第 1 天开始
第 2+a 天开始
第 2+b 天开始
第 N 天开始
进行 a 种消毒
需 A1 条毛巾
需 A_{a+1} 条毛巾
需 A_{b+2} 条毛巾
需 A_N 条毛巾
进行 b 种消毒
第 1 天结束
第 2+a 天结束
第 2+b 天结束
第 N 天结束
扔掉旧毛巾
垃圾站

例如：当n=4 a=1 b=2 f=3 fa=2 fb=1 A1=8 A2=2 A3=1 A4=6时，可构造如下的网络：



桔红色数字为费用，没有标记的弧的费用为0；
粉红色数字为上下界，没有标记的弧的下界为0上界为∞

1．构造初始流

由于上述方法构造的是一个容量有上下界的网络，初始流不能设为零流，那怎么办呢？倘若采用《图论》书上的构造附加网构造初始流的话，不仅增添了许多代码，浪费了编程时间，而且空间上难以承受，程序运行时间上也作了无谓的浪费。应该“因题而异”，针对本算法构造出初始流。

构造的核心思想为：不消毒，所有需求的毛巾都经过购买得到。

所有的 i→i+n(1≤i≤n) 的弧的流量设为 Ai

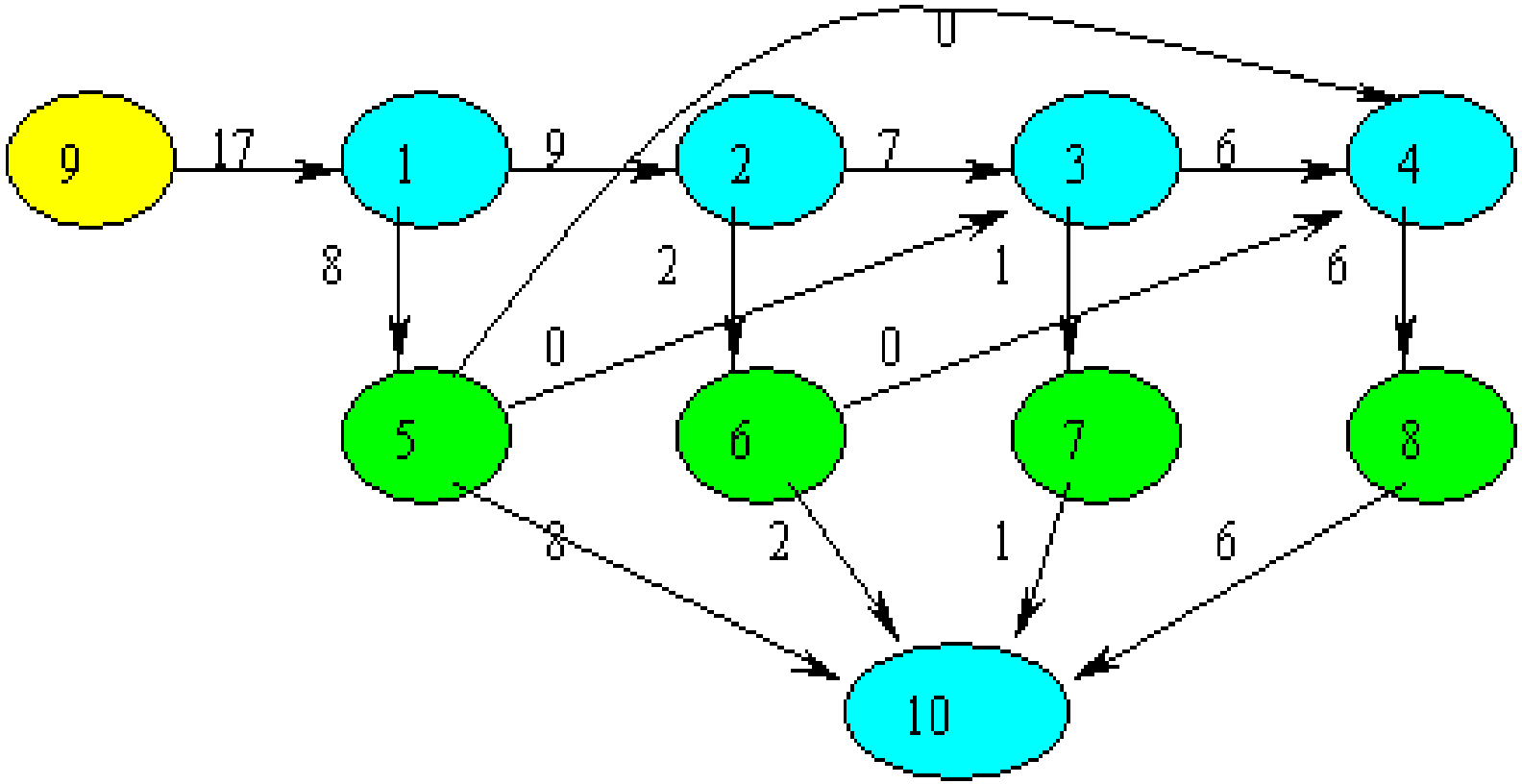
n-1→n 的弧的流量设为 An

i→i+1 （ i∈1..n-2 ）的弧的流量设为 i+1→i+2 的流量加上 i+1→i+n+1 的流量

源点 s 到 1 的弧的流量设为 1→2 的弧流量加上 1→n+1 的弧的流量

顶点 i+n+1(1≤i≤n)→ 汇点 t 的流量设为 Ai

例如：上图中的网络，它的初始流可设为：



弧上的数字为它的初始流量

2．空间存储

以往的网络流算法，都是采用邻接矩阵或邻接表来存储，这题 n 的范围 1 到 1000，肯定不能采用邻接矩阵来存，倘若用邻接表存的话，编程的复杂度又会增大许多，有没有两全其美的办法呢？

观察上图，很容易发现，网络中的所有弧只分为 5 种，即：

1、顶点 i-1（1≤i≤n）到 i；（0→1 表示源点 s 到 1）

2、顶点 i（1≤i≤n）到顶点 i+n；

3、 i+n（1≤i≤n）到 t；

4、 i+n(（1≤i<n-a）到 i+a+1；

5、 i+n(（1≤i<n-b）到 i+b+1

且容量、费用、第 2 类弧的流量都是固定，所以只要用四个 1..n 的数组分别存储剩下的四类弧的流量即可，这样只需要 4*1000*2/1024≈8KB。

见 CTSC2001 第一试第一题。（Agent）

下面是建立这个网络模型的主要步骤：

① 添加一个源点 p；

② 添加一个辅助节点 p'，令 $C_{p,p'} = K$，$W_{p,p'} = 1$，如果盟军总部能够与编号为 i 的间谍进行联系，即 $AM_i > 0$，那么令 $C_{p',i} = AM_i, W_{p',i} = AS_i$，$(i = 1,2,3\ldots\ldots n)$；

③ 如果编号为 i 的间谍与编号为 j 的间谍之间能够进行信息传递，即 $M_{i,j} > 0$，那么令 $C_{i,j} = M_{i,j}$，$W_{i,j} = S_{i,j}$，$(i = 1,2,3\ldots\ldots n, j = 1,2,3\ldots\ldots n)$；

④ 添加一个汇点 q，如果编号为 i 的间谍能够与德军情报部门联系，则令 $C_{i,q} = K$，$W_{i,q} = 1$，$(i = 1,2,3\ldots\ldots n)$。

最小费用最大流问题是求所有弧的（费用 * 流量）和最小，本题则是求所有弧的费用流量的积最大。

那么是否能够利用求最小费用最大流的算法解决这个问题呢？

用 $F_{i,j}$ 表示结点 i 与结点 j 之间传递信息的实际数目。那么这道题目求解的目标为：

$$Max\left\{\prod_{i=1}^{n}\prod_{j=1}^{n}S_{i,j}^{F_{i,j}}\cdot\prod_{i=1}^{n}AS_i^{F_{p',i}}\right\} \qquad (1)$$

通过取对数，这个式子可以转化为：

$$Min\left\{\sum_{i=1}^{n}\sum_{j=1}^{n}(-F_{i,j}\cdot\ln(S_{i,j}))+\sum_{i=1}^{n}(-F_{p',i}\cdot\ln(AS_i))\right\}, \qquad (2)$$

从这个式子可以看出，问题从求积转化成为了求和，显然满足最小费用最大流问题模型的特点，故可以采用求费用流的算法进行解决。

若$(a,b)\in A$，令$W_{a,b}=-\ln(W_{a,b})$。

另一方面，因为只要$M_{i,j}>0$，就有$0<S_{i,j}\le 1$，那么$-Ln(S_{i,j})>0$，即对于网络中的有向弧(i,j)，有$W_{i,j}>0$。因此在求解代价最小的增广轨时，仍可以采用大家平时解决费用流时所采用的迭代法。

最终的求解目标为：$\prod_{i=1}^{n}\prod_{j=1}^{n}e^{W_{i,j}\cdot(F_{i,j}-C_{i,j})}\cdot\prod_{i=1}^{n}e^{W_{p',i}\cdot(F_{p',i}-C_{p',i})}$

采用邻接表来记录整个网络，占用的空间一般比采用邻接矩阵要少，而且效率会有所提高。

一个要注意的地方就是对误差的处理，因为问题在求解的过程当中都是对实数进行操作，而每条弧上的代价又是一个非常小的正实数，如果采用普通的判断实数大小的方法，很有可能在利用迭代法求最短路的时候出现死循环的情况。

为了避免这种情况的出现，设 $Limit = 10^{-12}$，在每次比较实数 a、b 大小的时候，将 $a > b$ 改成 $a > b + Limit$，将 $a < b$ 改成 $a < b - Limit$，将 $a \geq b$ 改成 $a > b - Limit$，将 $a \leq b$ 改成 $a < b + Limit$ 这样就能够较好的避免误差。

浅谈网络流算法的应用

小结

缺点：相对于其它图论算法来说，它的模型更加复杂，编程复杂度也更高，而且算法的系数较大。

优点：网络流模型可以容纳的要素很多，特别是权的类型众多：不仅有表示容量和流量的权，还有表示费用的权；容量不仅有上界，还可以有下界，这些多变的因素，使网络流模型在现实问题的解决和图论问题的研究中有了十分广泛的应用。它经常能够很好地解决一些搜索与动态规划无法解决的，看似 NP 的难题。

浅谈网络流算法的应用

小结

难点：网络流在具体问题中的应用，最具挑战性的部分是模型的构造，其次是算法的优化。

构造没有现成的模式可依，只能根据题目的具体条件来分析。这需要对各种网络流的性质了如指掌，并且归纳总结一些经验，发挥我们的创造性。

一般来说，用得最多的方法是拆点法。

优化是算法的重要环节，它并非朝夕之功就能提高的，必须靠经验的积累。