3．Dinkelbach 算法的分析

在这一节中，我们假定 Dinkelbach 算法终止于第 k 次。我们可以得到一个参量序列(L1,L2…Lk)和一个 0-1 值的向量(**x1**,**x2**…**xk**).z(L)的凸度暗示了下面的不等式：

dx1>**dx2**>…>**dxk-1**>=**dxk**>0

cx1+nC**dx1**>**cx2**+nC**dx2**>...>**cxk-1**+nC**dxk-1**>=**cxk**+nC**xk**>=0

由于函数 z(L)是严格递减的，也很容易发现

z(L1)<z(L2)<...<z(Lk) = 0 并且 L1>L2>...>Lk

引理 **1**

如果 0>=z(Lr)>-1/nD (2<=r<=k) 那么 z(Lr)=0

证明 由于 Lr=**cxr-1**/**dxr-1**

z(Lr)=**cxr-** Lr **dxr=cxr-**(**cxr-1dxr**)/(**dxr-1**)=(**cxrdxr-1 – cxr-1dxr**)/(**dxr-1**)

假定 z(Lr)<0，那么(**cxrdxr-1 – cxr-1dxr**)<=-1。

因此不等式 0<**dxr-1**<=nD 暗示了 z(Lr)<=-1/nD。它是矛盾的！

上面的引理来源于权向量 **c** 和 **d** 的完整性。这个引理暗示了如果 z(Lr)>=-1/nD 那么 z(Lr)=0，因此该算法会中止于第 r 次。

引理 **2** 如果 0<=**cxr**+nC**dxr**<1,那么 z(**cxr**/**dxr**)=0

证明 由于 c**xr**+nC**dxr** 是整数，如果 0<=**cxr**+nC**dxr**<1，那么 **cxr**+nC**dxr**=0 并且 **cxr**/**dxr**=-nC。由于最值 L*>=-nC, **xr** 是分数规划的最优解并且 L*=**cxr**/**dxr**=-nC。那么显然有 z(**cxr**/**dxr**)=z(L*)=0

上面的引理证明了如果 **cxr**+nC**dxr** < 1，那么算法就在 r 或者 r+1 次终止。

现在给出主要引理：

引理 **3**

如果 1<=r<=k-1 那么|z(Lr+1)|<=(1/2)|z(Lr)|或 **cxr+1**+nC**dxr+1** <=(1/2)(**cxr**+nC**dxr**)将满足。

证明 如果 Lr+nC<=0 那么 Lr=L*=-nC 并且它暗示着 z(Lr)b=(1/2)z(Lr+1)=0

现在假定 Lr+nC>0。就出现了两种情况：

情况(i)　首先我们来考虑 z(Lr+1)(Lr+nC)<=z(Lr)/2*(Lr+1+nC)。这个情形如图 2 所示。在这个图中，直线 z=**cxr**-L**dxr** 被记作 lr。这里我们将用到图 2 的符号。令 M 为线段 PR 的中点。那么点 M 的坐标为(Lr+1 , z(Lr)(Lr+1+nC)/2/ Lr+nC)。因此条件 z(Lr+1)(Lr+nC)<=z(Lr)*(Lr+1+nC)/2 暗示着点 Q=(Lr+1,z(Lr+1))在线段 MR 上。在这个条件下，我们证明不等式 **cxr+1**+nC **dxr+1**<=(1/2)(**cxr**+nC, **dxr**)成立。这意味着直线 lr+1 与线段 MR 相交，lr+1 也与线段 M'R'相交，这里 M'是线段 P'R'的中点。现在我们证明这个不等式：

(Lr-Lr+1)(**cxr+1**+nC **dxr+1**)
=(**cxr+1**-Lr+1 **dxr+1**) (Lr+nC)- (**cxr+1**-Lr **dxr+1**) (Lr+1+nC)
=z(Lr+1) (Lr+nC) - (**cxr+1**-Lr **dxr+1**) (Lr+1+nC)
<= z(Lr) (Lr+1+nC)/2- (**cxr**-Lr **dxr**) (Lr+1+nC)
=-(1/2) (**cxr**-Lr **dxr**) (Lr+1+nC)=-(1/2)(**cxr**/ **dxr** - Lr) **dxr**(**cxr**/ **dxr**+nC)
=-(1/2)(Lr+1-Lr) (**cxr**+nC **dxr**)=(1/2) (Lr-Lr+1) (**cxr**+nC **dxr**)

由于 Lr>Lr+1,那么不等式 **cxr+1**+nC **dxr+1**<= (1/2) (**cxr**+nC **dxr**)已经被证明。

情况(ii) 接着，考虑 z(Lr+1)(Lr+nC)>z(Lr)/2*(Lr+1+nC)

|z(Lr+1)|=- z(Lr+1)< - z(Lr)(nC+ Lr+1)/2/(nC+ Lr)

=|z(Lr)|(1-(Lr- Lr+1)/ (nC+Lr))/2<=1/2 * |z(Lr)|

注意无论|z(L)|还是 **cx**+nC**dx** 在过程中都是不增长的。

在第一次，|z(L)|的值小于等于 2n^2*CD。通过引理 1，显然如果存在 O(log(2n^2CD/(1/nD)))<=O(log(nCD))次，他们每个都能至少将 z(L)减少 50%那么，然后就能得到最优解。同样，引理 2 暗示了将 **cx**+nC**dx** 减少 50%的次数 O(log(2n^2CD))<=O(log(nCD))是最坏情况。引理 3 证明了每次将|z(L)|或 **cx**+nC**dx** 减少 50%。因此，重复总次数的界限是 O(log(nCD))。

定理 4　Dinkelbanch 算法最坏情况的运行次数是 O(log(nCD))<=O(log(nM))，这里 M=max{C,D}。

上面的定理证明了 Dinkelbanch 算法最坏运行次数是 O(log(nCD))。它暗示了，如果对于 Q(L)存在强多项式算法，Dinkelbach 算法就能在多项式时间内解决分数规划问题。然而，当我们用多项式算法解决了子问题后，我们需要估计目标函数 Q(L)的系数的输入规模。在下一节，我们将通过分析最优比率生成树和分数调配问题来讨论这一点。

4．讨论

Chandrasekaran[4]提出了最优比率生成树的算法，它是基于二分搜索的 。Dinkelbach 算法可以在 O(T(v,e)log(vCD))的时间解决该问题，这里 v 是点的个数，e 是边的个数，并且用 T(v,e)表示计算普通最小生成树的强多项式算法。很容易将 Chandrasekaran 的算法延伸到一般带有分数目标函数的矩阵胚规划问题。在这种情况下，在这种情形下，函数 z(L)的断点数最大为 n(n-1)/2（参见[4]）因此，当可行域 Ω 是矩阵胚基础特征向量的集合。Dinkelbach 算法就会在 O(min{n^2,log(nCD)})后终止。

对于调配问题，已经研制了许多算法。大概最有名的算法就是 Hungarian 方法，并且它在最坏情况下的复杂度是 O(v(v log v+e)) [9]。使用 Hungarian 方法，Dinkelbach 算法可以用 O(v(v log v+e)log(vCD))的时间解决分数调配问题。在[19]中，Orlin 和 Ahuja 提出了一个 O(sqrt(v)e*log(vW))的算法来解决调配问题而且据说他们的算法因为强多项式算法而具有竞争性（参见[2]也可）。在他们的算法中，它假定边权为整数，并且 W 代表边权的最大绝对值。为了将这个算法与 Dinkelbach 算法相结合，我们需要将在运行第 r 次解决的的子问题 Q(L)用下面式子代替

(**dxr-1**) **cx**-(**cxr-1**) **dx**

这里 **xr-1** 表示代表运行第 i 次得到的最优解。因此，在每次运行中，Dinkelbach 算法解决边权绝对值小于等于 2v^2CD 的调配问题。它暗示了 Dinkelbach 算法对于调配问题的最坏情况下的时间复杂度为

O(sqrt(v)e(log(2v^3CD))(log(eCD)))<=O(sqrt(v)e(log(vCD))^2)

我们说，如果一个具有线性目标函数的 0-1 整数规划问题存在多项式算法，我们可以利用上述目标函数在多项式时间内解决它的 0-1 分数规划问题。