◎产品、研发、测试◎

基于 Bayes 推断的基因芯片探针特异性估计模型

彭 柳[1,2],冯圣中[1]
PENG Liu[1,2],FENG Sheng-zhong[1]

1.中国科学院 计算技术研究所 国家智能计算机研究开发中心 中国科学院计算机系统结构重点实验室,北京 100080
2.中国科学院 研究生院,北京 100039
1.Institute of Computing Technology,Chinese Academy of Sciences,Beijing 100080,China
2.Graduate School of Chinese Academy of Sciences,Beijing 100039,China
E-mail:pengliupl@ncic.ac.cn

PENG Liu,FENG Sheng-zhong.Probe specificity estimation model based on Bayes theory.Computer Engineering and Applications,2007,43(24):100-103.

Abstract: Sequence similarity is one of the pivotal problems in field of bioinformatics.This paper proposes a fast DNA sequence similarity estimation method based on Bayes theory. The algorithm is alignent-free,outperformances current similarity alignment algorithms,and can greatly improve the performance of Gen Chip design.
Key words: probe design;sequence similarity estimation;Bayes theory;specificity

摘 要:序列相似性计算是生物信息处理中的基本问题。针对基因芯片设计中的特异性评价问题,基于 Bayes 推断,建立了 DNA 序列快速估计算法,该算法不需要序列联配(alignment-free),性能好于广泛应用的相似性计算工具,可以大幅提高基因芯片设计性能。
关键词:探针设计;序列相似性估计;贝叶斯推断;特异性
文章编号:1002-8331(2007)24-0100-04 **文献标识码**:A **中图分类号**:TP311

1 引言

基因芯片是一种微阵列生物芯片,由数以万计的长度为 20~70 个碱基的探针组成。这些探针必须满足特异性(specificity)、敏感性(sensitivity)和一致性(consistency)要求。所谓特异性,即探针必须只与其设定目标基因杂交(hybridization),而不与非目标基因杂交;所谓敏感性,即探针本身不能形成二级结构;所谓一致性,就是所有探针-目标的杂交必须在相同的实验室环境(温度、溶液浓度等)进行。基因芯片设计的基本问题就是选择探针,使之满足这三个要求。其中,探针的特异性很大程度上影响基因芯片的性能。

杂交是基于碱基互补原理进行的,因此,探针的特异性检查就是计算其补序列和目标、非目标序列的相似性。理想的探针,其补序列必须和其目标序列高度相似,而同时和其非目标序列存在较大的距离。

以前的探针选择工具中特异性检查一般采用序列相似性计算工具,是基因芯片设计计算中最为耗时的部分。

针对以上问题,本文针对基因芯片设计中的特异性评价问题,基于 Bayes 推断,建立了 DNA 序列快速估计算法,实现了基于 spaced seed hashing 算法的探针选择工具,该算法不需要序列联配(alignment-free),性能好于广泛应用的相似性计算工具 BLAST,可以大幅提高基因芯片设计性能。

2 相关工作

2.1 现有探针选择工具研究

Li and Stormo[1]提出了一种启发式算法解决了探针的选择问题。为了提高时间效率,他们使用了下标矩阵的技术。然而,这个算法对于像基因群体等大规模的计算仍然不够快而且空间效率也不够高。例如,它用了几乎 4 天的时间才为大概有 9.5 Mbps 包含 6 343 个基因组的 S.cere 基因完成一个长度为 24 的探针集的设计。Kaderali and Schliep[2]专注于探针集精确性的设计,在他们的算法中,通过使用启发式动态规划,计算出了每个探针与其目标序列最稳定的联配,尽管他们的方法拥有更高的精确度,然而该算法却是非常慢而且也不适合大规模数据集。例如,它花了 9 个小时为了一总长度仅仅为 0.6 Mbps 58 HIV-1 设计探针集。Rahmann[3]提出了一种快速设计长度在 30 个核苷酸以内的短探针。其算法是通过计算探针的最长公共相邻子链来近似计算探针的不确定性,从而大大地节约了时间。该算法对于 Neuropora crassa 这样的大基因组的探针选择只需 4 个小时。然而,这种方法也有其自身的缺陷,它只能用于短探针的设计。并且,前面提到的它所用的近似计算也不够精确,可能丢失某些很好的探针。Chung et al[4]提出了一种为微生物群选择探针的方法,一种叫做 HPD 的分级探针设计方案,为非常保守的序列设计寡核苷酸探针。对于一个保守的功能基因的序列,HPD

自动地产生聚类树所有节点的探针。但是,HPD 是在 windows 平台上面通过使用 ClustalW[5,6]和 NCBI-BLAST 实现的,故其计算速度很难提高,并且当数据规模增大到成千上万的序列时,计算就变得很慢了。

2.2 探针杂交条件研究

探针-目标的相似性,主要采用碱基相同比率和连续相同碱基长度这两个参数来评价。对于 50-mer 的探针,Kane[7]指出,为了获得特异性高的探针,必须满足以下两个条件:

(1)探针和非目标序列的相似度不可以超过 75%~85%。

(2)探针和非目标序列之间,碱基连续相同的长度不可以超过 15。

Jizhong Zhou 在 Kane 的基础上作了更深入细致的研究,提出了更完善的探针设计标准[8]。对于序列特异性探针,探针和非目标的相似性必须小于 85%,对于长度为 50 碱基的探针,连续相同碱基长度不可以超过 16,对于长度为 70 碱基的探针,连续相同碱基长度不可以超过 20。他们的研究还表明,探针和目标/非目标序列的杂交,存在或然性。探针-目标/非目标之间的相似性越高,杂交可能性越大;反之,越不相似,越不易杂交。

仔细分析这些杂交条件,可以发现,在探针特异性检查中,并不需要计算出探针-目标/非目标相似性的确切数值,而是需要判断探针-目标相似性是否大于某个阈值(如 88%),或者探针-非目标相似性是否小于某个阈值(如 75%)。这就是本文相似性快速估计得以应用的基础。

3 基于 Bayes 的序列相似性估计

由于 DNA 序列均由 A、T、G、C 四种碱基组成,并且每种碱基的分布是存在先验概率的(简单而言 25%)。那么对于长度为 n 个碱基的等长序列 y_1、y_2,如果其中 m 个位置两序列对应碱基相同,$m<n$,依据 Bayes 理论,在存在先验概率条件下,序列 y_1、y_2 的相似性可以估计。详细叙述如下:

对于长度为 n 个碱基的等长序列 y_1、y_2,检查其中 m 个位置,$m<n$;用 x 表示两个序列间的 Hamming 距离。定义如下事件:

事件 A:随机选择 y_2 中 m 个位置,发现并不是所有碱基与 y_1 对应位置碱基都一致。

事件 B:随机选择 y_2 中 m 个位置,发现所有碱基与 y_1 对应位置碱基都一致。

$x<d$:描述该序列被探针命中。

由于序列 y_1 和 y_2 相似与否可以通过随机检查序列 y_1 和 y_2 的 n 个位置中的 m 个位置是否都一致来表示,那么,探针和非目标序列的杂交概率可以表示为 $P(x\leqslant d|A)$,探针和目标序列的杂交概率可以表示为 $P(x\leqslant d|B)$。依据 Bayes 公式[9],可得:

$$P(x\leqslant d|A)=\frac{P(x\leqslant d|A)}{P(A)}=\frac{\sum_{j=0}^{d}P(x=j,A)}{P(A)}=$$

$$\frac{\sum_{j=0}^{d}P(x=j)P(A|x=j)}{P(A)}=\frac{\sum_{j=0}^{d}C_n^j p^j(1-p)^{n-j}\left(1-\frac{C_{n-j}^m}{C_n^m}\right)}{1-(1-p)^m} \quad (1)$$

同样,可以得到:

$$P(x\leqslant d|B)=\frac{\sum_{j=0}^{d}C_n^j p^j(1-p)^{n-j}\cdot\frac{C_{n-j}^m}{C_n^m}}{(1-p)^m} \quad (2)$$

为了得到特异性高的探针,希望 $P(x\leqslant d|B)$ 尽可能的大,同时,$P(x\leqslant d|A)$ 尽可能的小。

4 SSH 算法与高特异性探针构造

4.1 基本定义

定义输入为 $Y=\{y_1,y_2,\cdots,y_k\}$,其中 y_i 是一个 DNA 序列,用字母表 $\sum=\{A,C,G,T/U\}$ 上的字符串表示,k 是集 Y 的势。

离散种子的模式:长度为 n,权值为 m。

DNA 序列编码:设 y_i 是一个 DNA 序列,用字母表 $\sum=\{A,C,G,T\}$ 上的字符串表示。对序列 y_i 编码,令 A,T,G,C 分别用 00,01,10,11 编码,那么序列"ACGGTCG" 变为"00 11 10 10 01 11 10"。令 $y_i[j]$ 表示序列 y_i 中第 j 个元素,每个元素为 2 bit。

Hamming 距离:设 y_i 和 y_j 由 A、T、G、C 四种碱基组成的 DNA 序列,$y_i=y_{i1},y_{i2},\cdots,y_{im}$,$y_j=y_{j1},y_{j2},\cdots,y_{jn}$:当 $m=n$ 时,y_i 和 y_j 之间的 Hamming 距离 $x=1/2\Sigma yik\oplus yjk$,$k=\{1,2,\cdots,n\}$;当 $m<n$ 时,y_i 和 y_j 之间的 Hamming 距离 $x=\min\{1/2\ \Sigma\ yik\oplus y(j+p)k$,$k=\{1,2,\cdots,m\}\}$,$j=\{0,1,2,\cdots,n-m+1\}$。

序列 y_i 字 w_r:位于序列 y_i 的位置 r 上的长度为 m 的子序列。

序列 y 的一个候选探针:序列 y 的一个字或者多个字的串连。如果一个候选探针是两个或者多个字的串连,那么这些字之间应该允许重叠或者他们之间允许有空格,通常一个空格应该小于 3 bp。显然,当字具有特异性是,候选探针也具有特异性。

4.2 Spaced Seed Hashing 算法

目前已有的探针设计工具有很多,例如:HPD[10]、OligoWiz[11]、PROBEmer[12]、OligoPicker[13]、OligoArray 2.1[14]等。它们相似性计算中主要存在两大问题:一方面,其相似性计算大多采用动态规划 BLAST 算法,非常耗时:当探针长度为 p,目标序列总长度为 m,非目标序列总长度为 n 时,其时间复杂度为 $O(p*p*m*n)$;另一方面,由于 BLAST 是以连续的最小长度为 7 bp 的字命中为基础计算相似性的,采用 BLAST 还可能丢掉一些好的 probe。

针对目前以上两大问题,提出用 Spaced Seed hashing 算法计算相似性。一方面,由于 Spaced Seed hashing 算法采用 hashing 算法,故能够使时间复杂度降为 $O(m+n)$;另一方面,Spaced Seed hashing 利用要求相似又不要求完全匹配的事实,通过采用 2 中的 Bayes 理论,选择恰当的 spaced seed 模式,就可以得到特异性很高的探针,尽量不遗漏好的探针。

4.3 高特异性探针构造算法

高特异性探针构造算法分三步:第一步,通过 Bayes 理论获得能够产生高特异性模式探针的 spaced seed;第二步,基于选择的 spaced seed 通过 Spaced Seed Hashing 算法获得特异字;第三步,通过特异字生成探针。具体步骤如下:

第一步:确定 spaced seed 的模式。

根据前面的分析,可以知道,为了获得高特异性的探针,必须让探针与目标序列杂交的概率大,同时让序列与非目标序列杂交的概率小,即让 $P(x<=d|B)$ 尽可能的大同时让 $P(x<=d|A)$ 尽可能的小,构造函数:

$$F=P(x<=d|B)-P(x<=d|A) \quad (3)$$

根据经验,选择 seed 长度在 15 左右,权值在 10 左右进行测试(按照公式(1)、(2))。

从图 1 中可以看出,length 为 15,weight 为 13 的 spaced seed 其对应的 F 值最大,故生成的探针应该特异性很好,而 length 为 13,weight 为 10 的 spaced seed 其对应的 F 值最小,

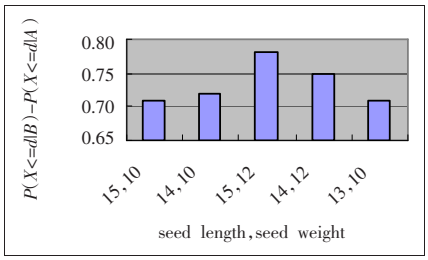

图 1 不同 spaced seed pattern 对应的 F 值

故生成的探针应该特异性较差。这里选择 length 为 15,weight 为 13 作为 spaced seed 的模式。

第二步:获得每个序列的特异字。

这一步的目的是产生每个序列 yi 的有可能的特异字。

序列 / ID / 字	1	2	3	…	M
1	−1	1	−1		−1
2	1	1	−1		−1
…				h_{ij}	
N	1	−1	−1		−1

图 2 命中矩阵,其所有元素的初始值均是−1

如果对于给定的离散种子，第 i 个字在第 j 个序列中出现,则将 hij 置为 1。对于第 i 个字 wi,如果对于序列 yj,,hij=1,而对于其他任何 yk,都有,hik=−1,那么字 wi 就是序列 yj 的特异字。依据这个规则,可以得到任何序列的所有特殊字。

第三步:产生候选探针。

序列 yi 的候选探针是它的一个特殊字或者多个特异字的串连。如果候选探针是两个或者多个特异字的串连,那么这些字之间允许有重叠或者空格，通常空格的大小应该小于 3 bp。将第二步中的每个特异字和其前后的相邻特异字进行串联,如果相邻连个特异字之间的空格小于 3 bp,并且联接之后子串的长度在 20 mer~70 mer 之间，那么将子串加入到该序列的候选探针集中。最后用如下条件过滤所有的候选指针:(1)低复杂性条件;(2)CG content 要求。如果条件和要求不能满足,那么将其从候选探针列表中除去。

5 实验结果与分析

本章介绍三个实验:探针特异性比较、计算性能比较和不同 spaced seed 模式对探针特异性甄别效果的比较。探针特异性比较实验主要用于证明基于 Spaced Seed Hashing 算法的探针设计工具相对于基于 BLAST 的探针设计工具能够获得特异性更好的探针。计算性能比较实验主要用于证明基于 Spaced Seed Hashing 算法的探针设计工具相对于基于 BLAST 的探针设计工具有更好的计算性能。不同 spaced seed 模式对探针特异性甄别效果的比较实验主要用于证明基于 Bayes 推导选择出来的 spaced seed 相对于其他的 spaced seed 能够产生特异性更好的探针。

5.1 测试数据集与参数设置

实验平台:AMD Opteron 242 2P 1.6 GHz,1 GB,但实验中只使用了一个 CPU,操作系统为 SUSE LINUX 2.6。

实验数据:序列采用 E.coli,从 NCBI 网站下载。

参数设置:除非特殊声明,实验均使用缺省参数。

5.2 探针特异性对比

实验使用序列 E.coli 作为输入，分别通过基于 Spaced Seed Hashing 的 ProDesign[15]产生探针集 S_1 和通过基于 BLAST 的 OligoPicker 产生探针 S_2。首先,检查探针和目标序列的相似度:分别在 S_1 和 S_2 中随机选择了 100 个探针,使用 EMBOSS[16]将每个探针和其对应的目标序列做连配并计算相似度,发现探针和目标序列的相似度都在 90%左右,没有太大的差别。然后,检查探针和非目标序列的相似度：分别在 S_1 和 S_2 中各选择一个探针,将两个探针分别与随机选择的 100 条非目标序列使用 EMBOSS 做连配并计算相似度,实验结果如下图 3、4 所示:

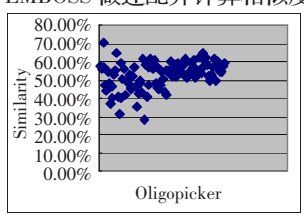

图 3 OligoPicker 生成的探针和非目标序列的相似度

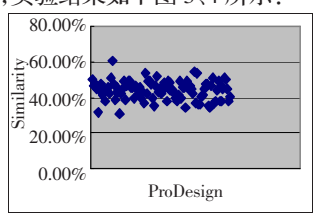

图 4 ProDesign 生成的探针和非目标序列的相似度

从图 3 和图 4 中可以看出，通过基于 Spaced Seed Hashing 的 ProDesign 生成的探针与非目标序列的相似度大多数都在 30%到 50%,而通过 OligoPicker 基于 BLAST 生成的探针与非目标序列的相似度大多数都在 40%到 60%，同时它们和目标序列的相似度基本相同，故基于 Spaced Seed Hashing 的 ProDesign 生成的探针交互杂交的概率较小,特异性较好,即基于 Spaced Seed Hashing 作相似性计算生成的探针相对于基于 BLAST 作相似性计算生成的探针具有更高的特异性。

5.3 计算性能对比

实验将基于 Spaced Seed Hashing 的探针设计工具 ProDesign 与基于 BLAST 的探针设计工具 OligoPocker 的计算性能进行比较:以序列 E.coli 作为输入,比较不同的探针设计工具对相同的输入序列生成探针集所需要的时间,实验结果如表 1 所示:

表 1 计算性能比较

工具名字	探针长度/mer	时间/min
OligoPicker	70	14.10
ProDesign	20~70	1.67

从表 1 种可以看出,基于 Spaced Seed Hashing 的 ProDesign 比基于 BLAST 的 OligoPicker 对相同的输入序列 E.coli 生成探针所需要时间少很多,故其计算性能更好。

5.4 Spaced Seed 对探针特异性甄别效果的影响

从图 1 中可以看出,length 为 15,weigh 为 13 的 Spaced Seed 其对应的 F 值最大,依据 3 中的推导,其生成的探针应该特异性应该较好，而 length 为 13,weight 为 10 的 Spaced Seed 其对应的 F 值最小,其生成的探针应该特异性较差。实验如下:输入序列均使用 E.coli,实验设置不同的 Spaced Seed 模式,模式 1 的 length 和 weight 分别为 15 和 13，模式 2 的 length 和 weight 分别为 13 和 10,分别通过 ProDesign 生成各自的探针集 S_1 和 S_2，在 S_1 和 S_2 中随机选择了 100 个探针，使用 EMBOSS 将每个探针和其对应的目标序列做连配并计算相似度,发现探针和目标序列的相似度都在 90%左右,没有太大的差别。然后,检查探针和非目标序列的相似度：分别在 S_1 和 S_2 中选择一个探针，将两个探针分别与随机选择的 100 条非目标序列使用

EMBOSS 做连配并计算相似度,实验结果如图 5、6 所示:

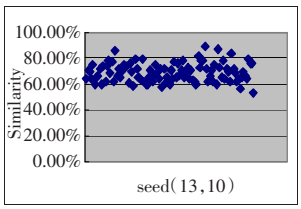

图 5　length 为 13,weigh 为 10 的 seed 生成的探针和非目标序列的相似度

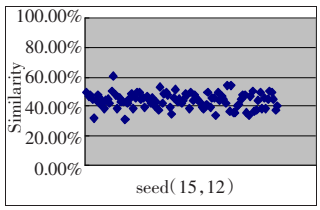

图 6　length 为 15,weigh 为 10 的 seed 生成的探针和非目标序列的相似度

从图 5 和图6 中可以看出,length 为 15,weight 为 12 的 Spaced Seed 生成的探针与非目标序列的相似度大多数都在 30%到 50%,length 为 13,weight 为 10 的 seed 生成的探针与非目标序列的相似度大多数都在 60%到 80%,而它们和目标序列的相似度都在 90%左右,故模式 1 相对于模式 2 生成的探针交互杂交的概率较小, 即模式 1 的 Spaced Seed 生成的探针相对于模式 2 的 Spaced Seed 生成的探针特异性较好, 即通过 3 中 Bayes 推导得到的 Spaced Seed 能够生成特异性更好的探针。

6　结论

本文基于 Bayes 推断,提出了面向基因芯片设计的序列相似性估计方法, 并基于 Bayes 推导和 Space Seed Hashing 算法,建立了高特异性探针构造算法。理论分析与实验结果表明,本文提出方法,能够大幅提高基因芯片设计运算速度,并设计出更高特异性的探针。(收稿日期:2007 年 4 月)

参考文献:

[1] Li F,Stormo G.Selection of optimal DNA oligos for gene expression arrays[J].Bioinformatics,2001,17:98-99.

[2] Kaderali L,Schliep A.Selecting signature oligonucleotides to identify organisms using DNA arrays[J].Bioinformatics,2002,18:1340-1349.

[3] Rahmann S.Rapid large-scale oligonucleotide selection for microarrays[C]//Proceedings of the First IEEE Computer Society Bioinformatics Conference (CSB),IEEE,2002:54-63.

[4] Chung W.Design of long oligonucleotide probes for functional gene detection in a microbial community[J].Bioinformatics,2005,21:4092-4100.

[5] Wilbur W J,Lipman D J.Rapid similarity searches of nucleic acid and protein data banks[C]//Proc Natl Acad Sci,USA,80:726-730.

[6] Myers E W,Miller W.Optimal alignments in linear space[J].Comput Applic Biosci,1988,4:11-17.

[7] Kane M D,Jatkoe T A,Strumpf C R,et al.Assesment of the sensitivity and specificity of oligonucleotide(50mer) microarrays[J].Nucleic Acids Res,2000,28:4552-4557.

[8] Wu He L,Li X,Fields M W,et al.Empirical establishment of oligonucleotide probe design criteria[J].Appl Environ Microbiol.

[9] http://plato.stanford.edu/entries/bayes-theorem/.

[10] Chung W.Design of long oligonucleotide probes for functional gene detection in a microbial community[J].Bioinformatics,2005,21:4092-4100.

[11] Nielsen H B.Design of oligonucleotides for microarrays and perspectives for design of multi-transptome arrays[J].Nucleic Acids Res,2003,31:3491-3496.

[12] Emrich S J.PROBEmer:a web-based software tool for selecting optimal DNA Oligos[J].Nucleic Acids Res,2003,31:3746-3750.

[13] Wang X,Seed B.Selecting of Oligonucleotide probes for protein coding sequences[J].Bioinformatics,2003,19:796-802.

[14] Rouillard J M.OligoArray 2.0:design of oligonucleotide probes for DNA microarrays using a thermodynamic approach[J].Nucleic Acids Research,2003,31:3057-3062.

[15] Feng Shengzhong,Xu Yongbai,Wang Zhuozhi,et al.A fast and flexible heuristic approach to oligo probe design for gene family detection in microbial community,2006.

[16] Rice P,Longden I,Bleasby A.EMBOSS:the european molecular biology open software suite [J].Trends in Genetics,2002,16(6):276-277.

(上接 93 页)

$p_{32}=4, p_{42}=2, u(3,4)=5, u(4,2)=3$

指出了调整路径;经算法步骤(9)~(11)搜寻到调整量 $\delta=2$;经算法步骤(12)、(13)按指出的调整路径和搜寻到调整量执行

$x_{35}\Leftarrow x_{35}-\delta, x_{45}\Leftarrow x_{45}+\delta; x_{43}\Leftarrow x_{43}-\delta, x_{23}\Leftarrow x_{23}+\delta$

便获得了新的可行解,见表 8。

以后再次循环运算,在算法步骤(7)中,将始终是 $\lambda\geqslant 0$,因此算法步骤(14)输出的最优解,便是表 8 所示的结果:

$x_{12}=8, x_{22}=2, x_{23}=4, x_{24}=2, x_{31}=2, x_{36}=4, x_{43}=3, x_{45}=5$ 其它 $x_{ij}=0$;$y_1=8, y_2=8, y_3=6, y_4=8$ 依次为 $A_1、A_2、A_3、A_4$ 承担的任务数量。

6　结束语

由模型(1)知,当 $m=n, a_j=1, (j=1,2,\cdots,n), L=1$ 时,O-M 指派问题便退化成经典指派问题,因此,O-M 指派问题的算法也可以求解经典指派问题(参见文献[9])。其实,构建 O-M 指派问题的算法的方法还可以运用于经典运输问题(参见文献[10]),及文献[1-3]描述的指派问题,而由此建立的算法会更加便捷,更适合于计算机运行。至于具体如何在文献[1-3]描述的指派问题中应用,将另文论述。而能否在文献[4-7]描述的指派问题中应用,并获得更好的效果,值得进一步研究。(收稿日期:2006 年 11 月)

参考文献:

[1] Bai Guo-zhong.B-assignment problems [J].Journal of Systems Science and Systems Engineering,1997,6(1):1-4.

[2] 白国仲,毛经中.C 指派问题[J].系统工程理论与实践,2003,23(3):108-111.

[3] 石忠民.广义指派问题[J].运筹与管理,1999,8(1):21-26.

[4] Romeijn H E,Morales D R.Generating experimental data for the generalized assignment problem [J].Operations Research,2001,49(6):866-878.

[5] Nauss R M.Solving the generalized assignment problem:an optimizting and heuristic approach [J].INFORMS Journal on Computing,2003,15(3):249-266.

[6] Mutsunori Yagiura,Toshihide Ibaraki,Fred Glover.An ejection chain approach for the generalized assignment problem [J].INFORMS Journal on Computing,2004,16(2):133-151.

[7] 李引珍,郭耀煌.一类带时间约束指派问题的分枝定界算法[J].系统工程理论与实践,2005,25(6):39-42.

[8] 宋业新,陈绵云,张暑红.多目标指派问题及其在军械物资供应中的应用[J].系统工程理论与实践,2001,21(11):141-144.

[9] 郭强.分配问题的一种新的迭代算法[J].系统工程与电子技术,2004,26(12):1915-1916.

[10] 郭强.运输问题的一种新的迭代算法[J].计算机工程与应用,2004,40(11):57-58.