基于智能 Agent 系统的集成路由算法*)

张艳华[1] 贺 红[1] 马绍汉[2]

(山东理工大学计算机学院 淄博255012)[1] (山东大学计算机系 济南250100)[2]

Integrated Routing Algorithms Based on Intelligent Agent Systems

ZHANG Yan-Hua[1] HE Hong[1] MA Shao-Han[2]

(Computer Department of Shandong University of Technology, Zibo 255012)

(Computer Department of Shandong University, Jinan 250100)

Abstract In order to ensure serve quality of Internet, and enhance Internet resource utilizing, we take advantage of NN-Algorithm and common algorithm, absorb NN-Algorithm's merit, such as fast and veracity, and overcome shortcoming high cost. We absorb common algorithm's lower cost and overcome it's shortcoming. We put forward integrated routing algorithms based on intelligent Agent systems. In different condition we trade off NN-Algorithm and common algorithm, so we can enhance serve quality of Internet. Compared with the common network model, this paper also considers the weight of network vertexes. So the intelligent routing algorithm is more practical.

Keywords Weight of network vertexes, Intelligent Agent, Neural network, Automatic alternate, Integrated routing algorithms

1 引言

路由策略是计算机网络协议中最复杂最重要的部分，要使传统的计算机网络发展成为高性能的网络，就必须在保持与扩展原有网络功能的前提下，提高网络应用的性能。要做到这一点，对路由策略的研究，是一项极为重要的工作。现有的路由选择策略通常是采用 Dijkstra 算法和 Bellman-Ford 算法，这样的路由选择机制已经不能满足分布式多媒体应用对于网络服务质量的需求。

目前的路由算法，大都建立在链路加权的网络模型基础上，对于早期的带宽资源不足、链路速度较低的网络状况，忽略节点处理速度的这种模型是能够满足要求的。对于现在 Internet 上互联网络的能力差异很大、高速光纤链路普及，并且可以通过改善编码技术实现"窄带宽用"的情况，给我们的路由策略研究带来了新的思路：带宽不是路由的"瓶颈"，主要的传输成本从网络拓扑的边上转移到节点上，所以忽略节点的因素就不能很好地抽象出网络特性，找到最佳的路由。因此，应将节点的权和链路的权一起考虑，改进已有的网络模型，同时对建立在新模型上的路由算法作相应的改进。

针对上述问题，本文引入人工智能领域最新发展成果之一的智能 Agent 理论，来研究新的路由算法。

2 智能 Agent 概述

智能 Agent 技术是 IT 领域当前的一个研究热点，有关的研究工作方兴未艾。随着有关 Agent 的研究在各个领域的逐步深入，来自不同研究领域的研究人员根据自己的知识背景和领域中需要解决的问题，从不同角度赋予了 Agent 各种不同的内涵。Agent 意味着自主性(autonomous)和智能性(intelligent)。其中 Franklin 和 Graesser 是这样描述 Agent 的，"自动 Agent 是处于一个环境之中，并作为这个环境一部分的一个系统。它随时可以感测这个环境，并执行相应的动作，同时建立自己的活动规划以应付未来可能感测到的环境变化。"

因为存在多种关于智能的定义，使智能性无法作为设计目标，这样一来，具有智能性的 Agent 也无法与其它所谓的智能软件相区分，而自主性似乎比智能性更容易区分 Agent 软件与其它软件。Wooldring 和 Jennings 认为：自主性是 Agent 的基本特性，其含义为："具有周期性动作自发执行和主动性，Agent 能够采取抢先的和独立的最终有利于用户的行动。"通常情况下，Agent 都具有以下几个主要特点：

自主性：Agent 一般都具有自己的资源和局部于自身的控制机制，能够在没有外界直接操纵的情况下，根据自身的内部状态以及感知到的外部环境信息，决定和控制自身的行为；

反应性：Agent 能够感知到其所在的外部环境(包括用户交互界面、实际的物理环境以及该环境中其它的 Agent 等)，并能够针对一些特定的事件做出相应的反应；

交互性：Agent 能够与其它对象(包括其它 Agent 或者用户等)以特定的语言进行各种各样的交互，也能够和其它的 Agent 一起协同工作，有效地完成各种层次的任务；

主动性：Agent 能够遵循其承诺采取主动行动，表现出面向目标(包括静态目标和动态目标)的行为。

根据 Agent 的这些特点，可以用图1来抽象地描述其行为。

利用 Agent 的这些固有属性能够使得 Agent 技术更适合于分布式网络环境下信息的设计和实现，这主要体现在：

(1)更好的主动性和适应性：Agent 能够以目标为导向，一旦接受某个目标，就尽可能地采用最佳的方法来实现该目标。并在实现目标的过程中，不断地感知所处环境的一切变

*)本文受国家自然科学基金(69873027)资助。张艳华 硕士生，主要研究移动代理技术。贺 红 副教授，主要研究智能路由算法。马绍汉 教授，博导，主要研究方向为组合优化算法等。

化，根据这些变化做出各种相应的调整，以适应动态变化的环境，并确保以最有力的模式实现最终目标。

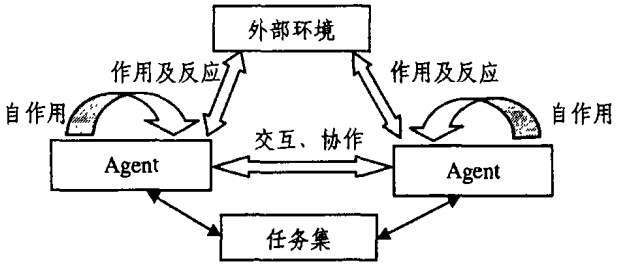

图1 Agent 的行为抽象

(2)更为灵活的交互和协作模式：Agent 之间有着语义更丰富的通信语言，以满足 Agent 在分布式网络环境下的交互；

(3)更为灵活有效的协作模式：Agent 之间还拥有更为灵活有效的协作模式以确保 Agent 之间能够通过相互协作，在分布式网络环境下高效完成相对复杂的任务。协作模式主要包括结构化组织模式、合约模式、多 Agent 规划模式和协商模式等。

除了这些基本属性，Agent 技术在各个研究领域的研究过程中派生出了很多可扩展的属性以及相应的 Agent 技术，如可移动属性和相应的可移动 Agent 技术，移动 Agent 具有根据任务目标，通信能力和服务器负载等因素，动态规划下步操作的能力。智能化路由能够很好地优化网络和计算资源，实现负载均衡，提高问题求解速度，避免资源访问的盲目性。同时 Agent 完成任务的效率和准确度很大程度上取决于路由策略的优劣。良好的路由策略能够动态地根据所处的网络环境的传输能力和服务器的负载状态等因素选择最佳移动路径和移动目标。

3 集成路由算法的网络模型

3.1 基本概念

定义1 对于赋权有向图 G(V,E,T)，其中 V={1,2,…,n}是图的结点集合，E 是图的弧的集合。对于从结点 i 到 j 的每一条弧 $e_{ij}\in E$ 有一个非负实数 $W(e_{ij},t)$叫做弧的权值，每一个结点处有一个非负实数 D(i,t)叫做结点的权值，T 为感兴趣的时间段区间。某时刻从结点 i 到 j 的网络花费为 $D(i,t)+W(e_{ij},t)$，这样的一个图称为一个时间依赖网络。

定义2 P(x, y,t)是时间依赖网络 G 中，在时刻 t 一条从结点 x 出发到结点 y 的路径，若从结点 x 沿路径 e_{xi}, e_{ij}, …, e_{ry}到达结点 y，则称 $P(x,y,t)=(x,i,j,\cdots,r,y,t)$是一条从结点 x 到结点 y 的有向路径，该路径的长度定义为 $L(x,y,t)=D(x,t)+W(e_{xi},t)+D(i,t)+W(e_{ij},t)+\cdots+D(r,t)+W(e_{ry},t)$，若在某一时间区间 Δt (Δt→0)内，从结点 x 到结点 y 有 n (n≥4)条路径，最短路径 SP 表示从结点 x 到结点 y 所有 n 条路径中长度最小的路径，我们记结点 x 到结点 y 的最短路径为 $P_{sp}(x,y,t)$。

定义3 在一定时间间隔内，一个结点根据负荷的历史数据和当前的负荷，用相应的算法预测下一时刻本结点可能的负荷情况，计算出相应的数据，此过程称为预计算。

定义4 将结果数据从一个结点传播到其它结点，此过程称为广播。

3.2 反例

传统的 Dijkstra 最短路径算法在时间依赖网络中不能有效地求解最短路径问题，如图2：

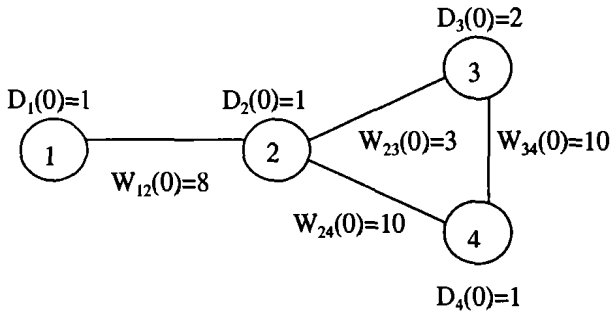

图2 违返 Dijkstra 最短路径算法的时间依赖网络中 t=0时的状态

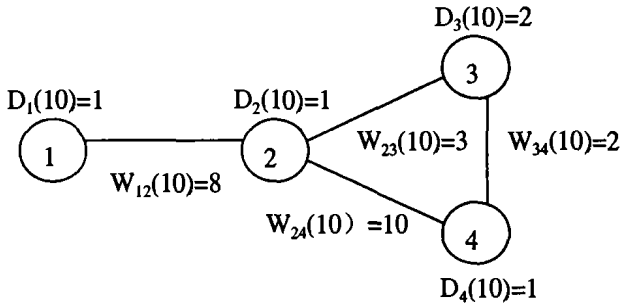

图3 违反 Dijkstra 最短路径算法的时间依赖网络中 t=10时的状态

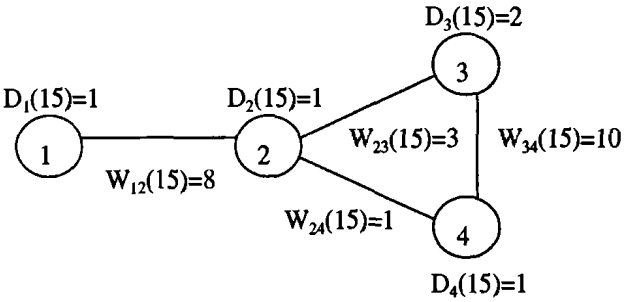

图4 违反 Dijkstra 最短路径算法的时间依赖网络中 t=15时的状态

如图2所示，当 t=0，行者从结点1到结点4的最优路径按照传统的 Dijkstra 方法是路径(1,2,4)，路径长度为20；当 t=10时，行者恰好到达结点2，如图3所示，此时求得从1到4的最短路径为(1,2,3,4)，路径长度为17，所以行者选择沿(2,3)边走到结点3；当 t=15时，行者到达结点3，这时网络状态又一次改变，如图4，但行者只能从3走到4，总得路径长度为25，但是在上述变化中路径(1,2,4)的长度为20，小于25。在这个例子中，使用传统的 Dijkstra 算法没有求得最短路。

3.3 网络模型

我们考虑的网络，是由定义1所定义的网络模型，此模型由一系列的结点构成，节点被物理连接在一起，每个结点都有一个权值，每条链路上也有权为距离，每条链路的距离通常由延迟、带宽等来确定。网络中的路由器相应地决定了数据包的传输路径，同时每个节点上均有一个自主决策和可交互的 Agent，它们以一定时间间隔，根据负荷的历史数据和当前的负荷用神经网络算法预测下一时刻本结点可能的负荷情况，计算出相应的数据以便找到最佳路径。例如：图5为简化的路由器拓扑结构。

当只考虑传输延迟(链路权值)时，则 H_1到 R_6的最短路为

$P_{sp}(H_1, R_6)=\langle H_1,R_1,R_4,R_6\rangle$，由于延迟包括排队延迟与传输延迟的总和，所以考虑到结点的权值与链路权值，则 H_1 到 R_6 的最短路 $P_{sp}(H_1, R_6)$ 有两条，分别是 $\langle H_1,R_1, R_6\rangle$ 与 $\langle H_1, R_1,R_3,R_6\rangle$，假设 $\langle H_1,R_1,R_6\rangle$ 的使用频率高于 $\langle H_1,R_1,R_3,R_6\rangle$ 的使用频率，则 H_1 到 R_6 的最短路为 $P_{sp}(H_1, R_6)=\langle H_1,R_1, R_3,R_6\rangle$，这样选取的最短路可以增大网络的吞吐量，减少不必要的网络拥挤。

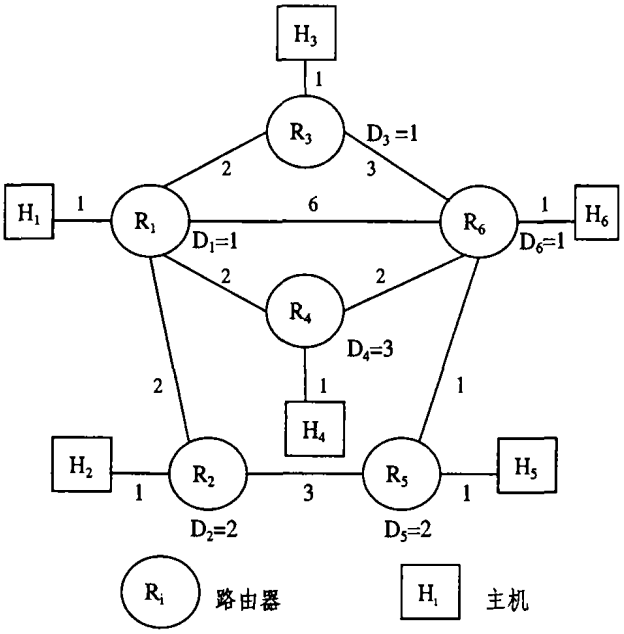

图5 简化的路由器拓扑结构

4 集成路由算法

4.1 算法思想

我们选取延迟作为主要的条件参数，借鉴神经网络算法在不超过2.0ms 就收敛到最短路，同时考虑到神经网络计算时间是 ns 级的，所以此算法能快速更新路由表，但考虑到神经网络代价高，所以结合一般的算法，并通过 Agent 的固有属性和扩展属性来有效地利用资源，提高网络服务的效率。

在网络的骨干路由器中，根据网络实时负载大小，交替使用实用反馈式神经网络算法与普通多项式算法，用预计算的方法求出时间依赖网络中两个结点的最短路并用移动 Agent 技术快速广播到相邻结点。

4.2 算法的关键技术

算法的关键技术在于算法转换的条件，此条件考虑了服务质量的控制要求，选定延迟和路由瓶颈带宽作为条件参数。一般地，在选择路由参数时，主要考虑3种计算参数：①可加性参数，如时延、时延抖动、跳数；②可乘性参数，如丢失率；③取小参数，如带宽。在 QoS 路由选择参数中，带宽与时延是最常用的。延迟包括排队延迟与传输延迟的总和。网络负载的轻重影响数据包在队列中的等待延迟，链路瓶颈带宽、输出链路的 MTU 决定数据包的传输延迟。

4.2.1 *静态负载条件下的参数* 设路由器有 M 个输入接口，N 个输出接口，其中输出接口 j 所连接的链路上的瓶颈带宽是 B_j，MTU 为最大传输单位，往接口 j 转发的数据包的最大延迟是 T_j。系统允许的最大延迟为 m_d，$\bar{b}$ 为一负载临界值向量，其中 $\bar{b}=(b_1,b_2,\cdots,b_j,\cdots,b_N)$，负载临界值 $b=\min(b_1,b_2,\cdots,b_j,\cdots,b_N)$。

B_j 用延迟表示：把 B_j 用传输10kb 的数据所用的时间量以毫秒为单位。例如，对于10Mbps 的以太网，传输10kbit 的数据包大约是1ms，因此 $B_j=1$。

最大延迟计算：当 M 个接口有数据包同时到达并且都经过 j 接口转发时，数据包经历的延迟最大 $T_j=M*MTU*B_j/10$。

显然，当 $T_j>m_d$ 时将违背延迟允许。因此 $b_j=10*m_d/(MTU*B_j)$，例如，当 $m_d=1ms$，$B_j=0.1$ 并且 $MTU=10kbits$，则 $b_j=10$。

4.2.2 *动态负载条件下的参数* 设路由器有 M 个输入接口，N 个输出接口，往接口 j 转发的数据包的最大延迟是 T_j，$Dis(i)(i=1,2,\cdots,N)$ 为第 i 个接口被用来传输数据包到目的地所经历的跳数，让通过接口 j 的链路为要选择的最短路径，Q_j 为在接口 j 的队列，$|Q_j|$ 为在队列 Q_j 中的数据包数量，则 $T_j=B*,(|Q_j|+Dis(j))$。其中 $Dis(j)$ 与 B_j 是静态的，可在建立路由表时计算出来，而 $|Q_j|$ 是动态的。

起初我们的路由算法为多项式算法，当 $B_j.(|Q_j|+Dis(j))>B_i*Dis(i)(i=1,2,\cdots,N,i\neq j)$ 转化为实用反馈式神经网络算法，寻找最短路。当 $B_j.(\sum_{j=1}^{n}|Q_j|+Dis(j))\leqslant\delta*B_i*Dis(i)$ 我们的路由算法转化为多项式算法，这里 δ 为0到1之间的一个数。例如取 δ 为0.8，此时网络负载相对减少，用一般的路由算法消耗的网络资源降低。

4.3 加权系数的影响

考虑到网络中的链路的使用频率，我们每隔一定时间，用过去一年的历史数据，分析每条链路的利用率，取其平均值作为此链路的使用频率权系数，当有不只一条链路满足要求时，我们选取权系数最小的作为被选最短路，这样能更好地改善上面的算法。

4.4 算法的描述

在此算法中取时间间隔 $\Delta t=20ms$，网络中路由器的个数为 K，每个路由器的输出接口个数为 N。在一个时间间隔中，一个路由器上的路由算法如下：

```
step1:置初值 j:=1;
          |Q|:=0;
step2:if j≤n then
        T1[j]:=Bj * Dis(j);
        T2[j]:= Bj * (|Qj|+Dis(j));
        |Q|:=|Qj|+|Q|;
      if j接口被用作当时的输出端口 then
        FLAG[j]:=1
        else
          FLAG[j]:=0
          else
          goto step4
step3:j:=j+1;
step4:BIG:=max(T1[1],T1[2],T1[3],…,T1[n]);
      LIT :=min(T1[1],T1[2],T1[3],…,T1[n]);
step5:for j:=1 to n
      if FLAG[j]=1 then
      goto step6;
      else
      continue
step6:if T2[j]>BIG then
      procedure program1 /* 用实用反馈式神经网络算法求两点之
          间的最短路,并将数据包传播出去;*/
      else if Bj * |Qj|+T1[j]<0.8 * LIT then
      stop program1 /*停止使用实用反馈式神经网络算法*/
      procedure program2 /*,使用一般的多项式算法,求两点之间
          的最短路,并将数据包传播出去;*/
```

结束语 路由策略是计算机网络的核心技术，而人工智能尤其是 Agent 技术在路由中的应用正待进一步深入研究。为了提高网络服务的效率，本文给出了节点加权的网络模型，提出基于智能 Agent 系统的集成路由算法的思路，并给出了具体的实现算法。该算法因其良好的实用特性和智能决策能力而优于其它算法。这样一个模型要用到实际的计算机网络中，还需要做更加深入的工作，这也是我们下一阶段的研究内

容。

参考文献

1 Jia Weijia, Xuan D, Zhao W. Integrated Routing Algorithms for Anycast Messages. Appear in IEEE Communication Magazine

2 TAN Guo-Zhen, GAO Wen. Shortest Path Algorithm in Time-Dependent Networks. CHINESE J. COMPUTERS, 2002, 25(2)

3 FENG Jing, MA Xiao-Jun, GU Guan-Qun. Research of Network Model Adapt to QoS Routing Mechanism. CHINESE J. COMPUTERS, 2000, 123(8)

4 黄晓斌,李琦. Agent 技术在地理信息领域中的作用. 计算机科学, 2002, 29(9)

5 HE Xiao-Yan, FEI Xiang, LUO Jun-Zhou, WU Jie-Yi. A Scheme for QoS-Based Routing Using Genetic Algorithm in Internet. CHINESE J. COMPUTERS, 2000, 23(11)

6 WANG Ming-Zhong, Xie Jian-Ying, ZHANG Jing-Yuan. Strategy of Constructing Minimum Cost Multicast Routing Tree with Delay and Delay Variation Bounds. CHINESE J. COMPUTERS, 2002, 25(5)

7 顾尚杰,薛质编著. 计算机通信网基础. 电子工业出版社, 2000

(上接第17页)

$=n/4$时的改进算法(即动态二表算法)也进行了实验。实验结果如表1示。本实验中使用512M 内存、1.0G Intel Celeron CPU 的计算机,程序代码使用 Turbo C 编写,实验中规定程序可以使用的存储空间不超过256M,如果运行时间超过10000秒,则认为该实例无法求解,在表1中该实例下用符号"----"表示。从表1可见,无论 ε 取1还是 $n/4$,改进算法的综合性能都明显优于文[9]中二表算法的性能。应该指出的是,改进算法中 ε 可以根据实际计算环境进行调整这一特点对于求解该问题是十分重要的,因为当待求解实例的变量数较多时,囿于求解时间和存储空间的限制,在计算机上便无法对这些实例求解,如对表1中变量数为60的实例,二表算法和动态二表算法都不能对其求解,但改进算法则只需将 ε 置为7,便可在规定的时间界内求解此类实例。

表1 改进算法与原文算法的性能比较

问题		原文算法		改进算法(ε 可变)		改进算法($\varepsilon=n/4$)	
变量数	是否有解	空间	时间(秒)	空间	时间(秒)	空间	时间(秒)
30	否		0.21		0.02		0.10
30	否	512kB	0.22	512kB	0.03	6.40kB	0.11
30	否		0.23		0.03		0.12
40	否		10.86		0.82		5.41
40	否	24MB	10.94	24MB	0.87	60.1kB	5.64
40	是		8.08		0.79		4.76
50	否		—		78.43		398.7
50	是	768MB	—	256MB	37.46	322kB	104.9
50	是		—		52.52		217.4
60	否				4117		—
60	是	16GB	—	256MB	3465	2.42MB	—
60	是		—		2173		—

注:当变量数 $n=30$、40时 $\varepsilon=1$,$n=50$、60时,ε 分别等于4和7。

结论 本文基于动态二表算法,使用4个非平衡的子表动态产生二表算法中2个排序表中子集和元素的方法,提出一种求解子集合问题的改进算法。算法允许使用 $O(2^{n/2-\varepsilon})$ 的存储空间($1\leqslant\varepsilon\leqslant n/4$),在 $O(\varepsilon(2^{n/2}))$ 的时间内求解子集和问题。算法可视计算环境和问题实例的规模,通过调整 ε 在1到 $n/4$ 内的取值,有效求解此前不能求解的子集合实例。特别地,当 $\varepsilon=n/4$ 时,Schroeppel 和 Shanir 的动态二表算法[6]可成为本文算法的一种特例。若如 Schroeppel 和 Shanir[6]所断言:"$T=O(2^{n/2})$ 是顺序求解子集和问题的时间下界",那么本文算法将是求解此问题的最优算法。将本文算法与现有求解算法在理论和数值实验上进行的对比分析表明,本文算法明显改进了此前该领域的研究结果。

参考文献

1 Garey M R, Johnson D S. Computers and intractability: A guide to the theory of NP-Completeness. San Francisco: W. H. Freeman and Co, 1979

2 Chor B, Rivest R L. A knapsack-type public key cryptosystem based on arithmetic in finite fields. IEEE Trans. Inform. Theory, 1988, 34 (5): 901～909

3 Laih C-S, Lee, J-Y, Harn L, Su Y-K. Linearly shift knapsack public-key cryptosystem. IEEE J. Selected Areas Commun, 1989, 7 (4): 534～539

4 Horowitz E, Sahni S. Computing partitions with applications to the knapsack problem. J. ACM, 1974, 21(2): 272～292

5 Schroeppel R, Shamir A. A $T=O(2^{n/2})$, $S=O(2^{n/4})$ algorithm for certain NP-complete problems. SIAM J. Comput, 1981, 10 (3): 456～464

6 Ferreira A G. A parallel time/hardware tradeoff $T\cdot H=O(2^{n/2})$ for the knapsack problem. IEEE Trans. Comput, 1991, 40(2): 221～225

7 Lou D C, Chang C C. A parallel two-list algorithm for the knapsack problem. Parallel Computing, 1997, 22: 1985～1996

8 Chang H K-C, Chen J J-R, Shyu S-J. A parallel algorithm for the knapsack problem using a generation and searching technique. Parallel Computing, 1994, 20(2): 233～243

9 Cornuejols G, Dawande M. A class of hard small 0-1 programs. 6th International IPCO Conference. Lectures Notes in Computer Science 1998, 1412: 284～293

10 Aardal K, Bixby R E, Hurkens C A J, Lenstra A K, et al. Market split and basis reduction: towards a solution of the Cornuejils-Dawande instances. In: 7th Intl. IPCO Conf. Lecture Notes in Computer Science, 1999, 1610: 1～16