ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.21, Supplement, December 2010, pp.224−237

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

不完全标记的多个并行事务踪迹的“剥离”*

满君丰[1+], 李长云[1,2], 文志诚[1], 温向兵[1]

[1](湖南工业大学 计算机与通信学院,湖南 株洲 412008)

[2](国防科学技术大学 计算机学院,湖南 长沙 410073)

Stripping Multiple Parallel Transaction Footprints with Incomplete Tokens

MAN Jun-Feng[1+], LI Chang-Yun[1,2], WEN Zhi-Cheng[1], WEN Xiang-Bing[1]

(School of Computer and Communication, Hunan University of Technology, Zhuzhou 412008, China)

(School of Computer Science, National University of Defense Technology, Changsha 410073, China)

+ Corresponding author: E-mail:mjfok@qq.com

Man JF, Li CY, Wen ZC, Wen XB. Stripping multiple parallel transaction footprints with incomplete tokens. ***Journal of Software***, **2010,21(Suppl.):224−237.** http://www.jos.org.cn/1000-9825/10024.htm

Abstract: Multiple parallel transactions in new-type distributed software result in that the events produced by them are randomly ranked. If the tokens of these events are incomplete or unavailable, they can't be distinguished to belong to which transaction. In this paper, the problem of stripping events with incomplete tokens is converted into maximum-weight perfect matching of bigraph system. If the transition time among these events is independently and identically distributed, all possible states (events) are separated into multiple cutsets, each of them becomes a bigraph system. The improved algorithm of maximum-weight perfect matching is adopted to finish respective matching, and then the matching results are spliced to gain the most possible footprint sequences produced by these transactions. Simulation experiments confirm that our method can effectively strip transaction footprints with incomplete tokens. Compared to traditional bigraph matching algorithm, the improved one has higher efficiency.

Key words: new-type distributed software, incomplete token, transaction footprint, bigraph matching, rank-maximal matching algorithm

摘 要: 新型分布式软件的多个并行处理事务导致每个事务产生的事件按随机排序,如果这些事件的标记不完全或不可用,将无法区分这些事件到底属于哪个事务.将不完全标记事件的“剥离”问题转化成偶图最大权重完全匹配问题.对于事件间的转换时间是独立等同分布的情况,将所有可能状态(事件)划分为若干个割集,每个割集构成一个偶图.在这些偶图系统中,采用所提出的偶图最大权重完全匹配的改进算法进行分散匹配,通过拼接匹配结果得到各事务产生的最可能踪迹序列.仿真实验证实:该方法可以有效实现不完全标记的事务踪迹的“剥离”;与传统的偶图匹配方法相比,改进算法具有更高的匹配效率.

关键词: 新型分布式软件;不完全标记;事务踪迹;偶图匹配;排列最大匹配算法

随着开放、动态、难控的新型分布式软件的发展,对其监管问题引起了高度关注,这主要有两个原因:一是

* Supported by the National Natural Science Foundation of China under Grant No.60773110 (国家自然科学基金); the Post-Doctoral Science Foundation of China under Grant No.20080440216 (博士后基金); the Natural Science Foundation of Hu'nan Province of China under Grant No.09JJ6087 (湖南省自然科学基金).

Received 2010-06-15; Accepted 2010-12-10

新型分布式软件在国民经济中起着越来越重要的作用;二是分布式软件系统的规模越来越庞大、功能越来越复杂,而人们对其可用性、可靠性和安全性等可信性质给予了更高的期望和要求[1].在新型分布式软件运行期,如何对监测获得的软件实体交互产生的事务踪迹进行有效诊断、分析和预测,实时调整被监测的软件实体,提高软件可信性,确保软件行为与预期相一致,这些已成为软件领域的热点问题.不完全标记的事务踪迹分析问题是上述问题的一个重要方面,本文针对该问题展开研究,并提出一套切实可行的新方案.

在新型分布式软件运行环境下,一个端到端的事务监测与分析系统包括 4 部分:(1) 发现进行交互的 IT 设施(artifact),如事务依赖的服务器或应用程序;(2) 在事务上下文环境中,模型化这些交互实体的相互关系;(3) 对这些交互实体进行监测来获得事务流的各状态,即事务踪迹状态;(4) 分析事务踪迹的可信性.依据信息和辅助信息的可用程度不同,上述问题有不同的挑战.一般情况下,可以通过 open-group ARM instrumentation[2]等工业标准标记事务状态的流转.但某些软件实体(特别是新型分布式环境下的软件实体)在交互过程中,产生的事件可能标记(token)不完全或不可用.如图 1 所示,除了状态 S_0,其他状态是不包含标记的.在该情况下,不可能准确无误地鉴别每个踪迹的唯一来源.如果是单一事务,且事务交互过程中,各事件按顺序发生,即遵循 FIFO(first-in-first-out)或 LIFO(last-in-first-out)原则,可以很容易将各事件拼接起来组成一个完整的事务踪迹.在分布式的多线程环境中,多个事务的并行处理导致每个事务的各事件按随机排序,如果这些事件标记不完全或不可用,将无法区分这些事件到底属于哪个事务,这将导致无法对交互行为进行实时分析和预测.这就需要寻找一种行之有效的方法将按随机排序的各事件进行“剥离”,即对各事件进行标记,然后将标记的结果拼接起来形成完整的事务踪迹,便于后续分析和预测.针对不完全标记的简单(无重复子踪迹)事务踪迹,如何寻找一种高效、准确的“剥离”技术成为解决问题的关键所在.

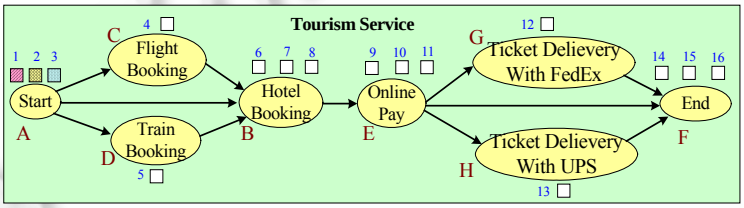


Fig.1 The instance of state transition model with incomplete tokens

图 1 不完全标记的状态转换模型实例

当事务执行一个事件(被表示成为模型中的一个状态)仅依赖于其输出(被表示成为另一个状态),不依赖于事务的过去交互历史,则这样的事务具有马尔可夫性,本文就在这种特殊情况下对不完全事务踪迹实施高效、准确地“剥离”.解决问题的思路是:在监测时根据事务踪迹的外在表现和内部变化,构建他们之间的对应关系;对监测获得的多个样本事务踪迹,分析他们的相关性,构建转移矩阵;通过跟踪事务实例集,可以发现这些事务间访问过的最相似的序列,并估计每个实例中各状态所花费的时间.当目前仅有一个事务踪迹,将简化为在单个实例中匹配踪迹中各状态问题.在本文提出的方法中,融入有关不同状态转换时间的统计信息,采用最大相似规则(Maximum-Likelihood Rule, MLR),以便所有踪迹被正确地匹配到产生它的事务上.对于两状态系统,在独立等同分布(Independent and Identically Distributed, I.I.D.)的转换时间下,优化的 MLR 将简化为偶图(Bipartite Graph 或 Bigraph)最大权重完全匹配问题.这将引发一个需要深入探索的问题:偶图匹配算法能否以合理的复杂度实现?只有有效解决该问题,才能高效、准确地“剥离”不完全标记事务踪迹,才能进行后续的分析与预测.

在偶图最大权重匹配问题上,很多学者已经进行了深入的研究,并有较成熟的算法,最快算法的运行时间复杂度为 $O(n(m+n\log n))$ [3]和 $O(\sqrt{n}m\log nC)$ [4],这里 C 是一个实例中的最大权重.前一个是强多项式算法,而后一个是弱多项式算法.简单应用前述算法到排列最大匹配问题上,不论在时间复杂度和空间复杂度上,都不能得到很好的效果,问题在于边的权重最大可能为 n^r,其输入大小并非为多项式.这两个算法假定数字间的计算操作(时间复杂的为 $O(C)$)按常量时间执行,而这并不符合实际情况,在 $O(n^r)$ 中对于数字计算花费时间为 $\Omega(r)$.因此,运行时间复杂度分别为 $O(rn(m+n\log n))$ 和 $O(r^2\sqrt{n}m\log n)$,所需空间都为 $O(rn)$.在文献[5]中,实现了针对权

重匹配的可伸缩算法,且独立于边的权重.在这种情况下,运行时间复杂度提高为 $O(r\sqrt{nm}\log n)$.在文献[6,7],作者提出了合并算法,它解决了排列最大化匹配问题,且时间复杂度为 $O(\min(n+r,r\sqrt{n})m)$,空间复杂度为线性.该算法不能解决最大基数排列的最佳匹配问题.为了有效实现排列最大化匹配,本文提出了一个能够解决排列最大化匹配问题的算法,且时间和空间复杂度符合文献[8]的要求.我们认为:该算法基于权重匹配消减,且更加简单和直观,在设计偶图匹配的高效算法上,本文进行了初步和有益的探索.

1 事务踪迹的标记模型

1.1 有关概念和定义

设 $f_X(x)$ 为连续随机变量 X 的概率密度函数(Probability Density Function, PDF), $\overline{F}_X(x) := P[X > x]$ 是它的互补积累分布函数(complementary cumulative distribution function,简称 CCDF).对于矩阵 A,设 $A(i,j)$ 代表 i^{th} 行 j^{th} 列的元素.设 π 代表排列矢量 $\{1,...,n\}$, $\log x$ 是 x 的自然对数, $|A|$ 是 A 的基数.对于集合 A 和 B,设 $A \setminus B = \{i, i \in A, i \notin B\}$.

对于无向图 $G(V,E)$,设 (i,j) 表示 i 和 j 间的边, $\deg(i)$ 是节点 i 的度.对于偶图 $G = \langle V_0 \cup V_1, E\rangle$,边由 0-1 二值矩阵 $A = [A(i,j)]$ 确定,这里 $A(i,j) = 1$ 表示存在一个从 $V_0(i)$ 到 $V_1(i)$ 的边.对于有向图,若存在一条从 i 到 j 的边,则 j 是 i 的直接后继,i 是 j 的直接前驱.i 的所有直接后继的集合用 $N(i)$ 表示,j 的直接前驱集合用 $P(j)$ 表示.有向无环图(directed acyclic graph,简称 DAG)至少有一个没有进入边的源点(起点)和至少一个没有离开边的汇点(终点).

下面给出半马尔过程(semi-Markov process,简称 SMP)的定义: $S_i, i = 0,...,N_s$ 代表事务踪迹序列的 i^{th} 状态, $T_{i,j}$ 代表从状态 S_i 到状态 S_j 的转换时间.如果访问过的状态序列是一条 Markov 链,事务过程被认为是半 Markov,转换概率矩阵用 $P = [P(i,j)]$ 表示,每一个转换时间 $T_{i,j}$ 是一个仅依赖于状态 S_i 和 S_j 的随机变量.这里认为:每个事务进展历程符合一般 SMP,每个转换时间 $T_{i,j}$ 从一个已知的 PDF $f_{T_{i,j}}$ (连续的时间间隔)抽取出来.这里考虑状态转换图是无循环的简单情况,这确保所有被处理的事务是单向的,没有事务在一个状态留下多于一个的踪迹.此外,还要求转换时间独立于系统负载,本文不考虑转换时间与负载相关的情况.

1.2 MLR

踪迹是指当一个事务进入模型中一个状态时,由应用程序日志创建的时间戳入口.既然用时间戳跟踪事务,需要对不同服务器上的时间进行同步.目前有很多解决该问题的方案,例如网络时间协议(network time protocol,简称 NTP).除了时间戳问题,踪迹还包含隶属与哪一个事务的、独一无二的标识符(identifier)或标记.为了讨论方便,在唯一起点状态 S_0 的踪迹被分配一个标记,这里假定对于监测日志记录中的踪迹没有丢失.本文考虑监测时的一般情形,即事务实例仍旧驻留在系统的不同状态,因此这些事务的所有踪迹没有最后产生.在这种情况下,跟踪事务将受到日志中各记录顺序先后的影响,这里假定写操作没有缓冲,确保从不同日志获得的记录能够按照正确时间顺序到达监测引擎.

事务间任何有效的匹配不能够共享同一踪迹,因为每一个事务产生一个独一无二的踪迹.因此,对于模型中每个状态 S_k ,任何有效的匹配可以用排列矢量 S_k 的集合表示.设 π_k 表示在状态 Y_k 中踪迹时间戳的矢量.当监测引擎按正确时间顺序接收到事务踪迹时, S_k 按升序排列,最新发生的事件排列在后面.根据排列矢量 Y_k ,用 π_k 表示 $Y_k^{\pi_k}$ 的排列.为了方便起见,在开始状态 Y_k 分配标记给踪迹 S_0 .因此,开始状态的排列矢量被设置为相等($\pi_0 = I$).换句话说,从 S_0 开始发现其他踪迹与 S_0 的关联关系.当事务转换时间 f_T 的联合 PDF 已知时,就能够将定量踪迹分析问题转化为求取所有事务实例能正确地匹配他们的事务踪迹(通过 MLR 最大化方法)的概率问题.因此,对于 N_s+1 个状态,事务的 MLR 踪迹分析简化为发现 N_s 个排列矢量的集合.

$$[\hat{\pi}_1^{ML}, ..., \hat{\pi}_{N_s}^{ML}] := \underset{\pi_1,...,\pi_{N_S}}{\arg\max} P(Y_1^{\pi_1}, ..., Y_{N_s}^{\pi_{N_S}} \mid Y_0) \quad (1)$$

解决公式(1)是 NP 难问题,本文试图解决其特殊情况:两状态系统,从而使解决公式(1)的问题变得简单.

2 两状态系统

这里考虑两状态模型,其作为更复杂模型的基础.对于该模型,下面将阐述用 MLR 将其在 I.I.D.转换时间下简化为偶图的完全匹配问题.

2.1 预备知识

图 2 展示了两状态系统,这里的状态 S_0 和 S_1 的踪迹 Y_0 和 Y_1 通过一个未知的排列矢量 π 相联系.

$$Y_0(j)=Y_1(\pi(j))-T(j),\ 1\leqslant j\leqslant|Y_1| \tag{2}$$

这里,$T(j)$ 是 $Y_0(j)$ 的转换时间,在状态 S_0 第 j^{th} 个踪迹,根据 π ,$Y_1(\pi(j))$ 是排列 Y_1 的第 j^{th} 元素.踪迹 $Y_0(j)$ 和 $Y_1(\pi(j))$ 是由同一事务产生的.这里有 $|Y_0|\geqslant|Y_1|$,因为有些事务可能依然驻留在状态 S_0 .在状态 S_0 观察到的实例数量为

$$Cnt(S_0)=|Y_0|-|Y_1| \tag{3}$$

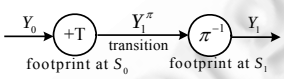


Fig.2 Finding permutation vector π given Y_0 and Y_1

图 2 给定 Y_0 和 Y_0 情况下发现排列 π

对于状态 S_0 的一批踪迹,当 $Cnt(S_0)=0$ 后 S_1 方开始,是一个完全批,即所有事务已经退出系统,否则批是部分的.容易理解:有效匹配发生在同一批次的踪迹而不能跨批次.因此,(Y_0,Y_1) 代表一个在单一批次中的踪迹集合.

下面考虑非参数转换时间的情况,即转换时间 PDF 未知,由于偶然性 PDF 支持度的下界是零.我们可能知道 PDF 支持度的上界△,即 $0\leqslant T\leqslant\Delta$;如果未指定,其界限可能是无限的.由于时间戳的存在,一些匹配 π 可能违反了偶然性和界限约束.为了有效地发现匹配数量,定义了一个偶图 $G=\langle V_0\cup V_1,E\rangle$,这里 $V_i(j)$ 表示在状态 S_i 的第 j^{th} 踪迹,E 中的边表示 S_0 和 S_1 间的有效通信.因此,当两个踪迹间的时间戳的差异满足 PDF 边界时,为 E 添加一条边.既然转换时间的实际 PDF 是未知的,该边是无权重的.给定二部邻接矩阵 A,在同一个批次中,唯一的有效匹配数量用 $N_B(|Y_0|,|Y_1|,A)$ 表示,这提供了跟踪每个实例的较精确方法.对于一个完全的批(即 $|Y_0|=|Y_1|=n$),通过二部邻接矩阵 A, $N_B(n,n,A)$ 由下面公式给出:

$$N_B(n,n,A)=perm(A):=\sum_{\pi}\prod_{i=1}^{n}A(i,\pi(i)) \tag{4}$$

这里,该公式指所有 $\{1,...,n\}$ 排列实例 π 的和.因此,踪迹上的时间戳减少了有效匹配的数量.针对真值转换模式,既然这里至少存在一个完全匹配,有 $1\leqslant perm(A)\leqslant n!$.对于一个完全偶图,上限是可以达到的,即在同一批次,所有实例从 S_0 离开后到达状态 S_1 .

对于部分批次,一些在 S_1 的踪迹还没有产生,因此,完全偶图匹配是不可行的.对于没能按正确顺序达到的踪迹,任何最大基数的偶图匹配都是一个有效匹配.然而,当踪迹按正确顺序达到,对于仍有驻留在 S_0 的踪迹,这改变了偶图的结构.给定 $|Y_1|=k<|Y_0|=n$, $Y_1(k)$ 时间后,这里有 $n–k$ 个实例没有转换离开,大多数最近的踪迹的时间戳在 S_1 .通过添加哑节点的 $n–k$ 个等同实例拷贝,用 $V_1(\delta)$ 表示.如果 $Y_1(k)-Y_0(k)<\Delta$,即没有超过界限,则在 $V_1(\delta)$ 和任何节点 $V_0(i)$ 添加边.既然所有哑节点是等同的,在该偶图中一些完全匹配也相等,部分批次的唯一匹配为

$$N_B(n,k,A)=\frac{perm(A)}{(n-k)!} \tag{5}$$

既然添加哑点 $V_1(\delta)$ 的拷贝得到的排列是相等的,当 $n=k$ 时,其简化为公式(4).

计算公式(5)的 $perm(A)$ 是一个 NP 难问题,这里求助于近似地确定界限.因此,批的创建避免了添加不需要

的边,从而边的稀疏构建和匹配能够跨不同批而独立进行.

2.2 优化跟踪

在两状态系统中,当事务转换时间 $T=[T(j)]$ 的联合 PDF f_T 已知,公式(1)中的 ML 匹配简化为

$$\hat{\pi}^{ML}(Y_0,Y_1;f):=\arg\max_{\pi} P(Y_1^{\pi}\mid Y_0)=\arg\max_{\pi} f_T[Y_1^{\pi}-Y_0] \tag{6}$$

对于 n 个踪迹的一个完全批次,$|Y_1|=|Y_0|=n$.在批中所有 n 个踪迹被正确地匹配的概率通过 MLR 进行最大化:

$$P^{ML}(Y_0,Y_1;f)=\frac{f_T[Y_1^{\hat{\pi}^{ML}}-Y_0]}{\sum_{\pi} f_T[Y_1^{\pi}-Y_0]} \tag{7}$$

这里,ML 排列 $\hat{\pi}^{ML}$ 的相似度用所有有效排列 π 的相似度之和来规范化.既然至少有一个排列有正的相似度(真模式),公式(7)是严格为正值的.

对于转换时间 $T=[T(j)]$ 的一般的联合 PDF f_T,需要对所有排列矢量 π 进行查询,其复杂度可能是指数的.这里进行简化:所有实例的转换时间是服从 PDF f_T 的 I.I.D.,对于一个 (Y_0,Y_1) 批,ML 规则简化为

$$\hat{\pi}^{ML}(Y_0,Y_1;f)=\arg\max_{\pi}\prod_{1\leqslant\pi(i)\leqslant k} f_T[Y_1(\pi(i)-Y_0(i)]\prod_{k<\pi(i)\leqslant n}\bar{F}_T[Y_1(\pi(k)-Y_0(i)] \tag{8}$$

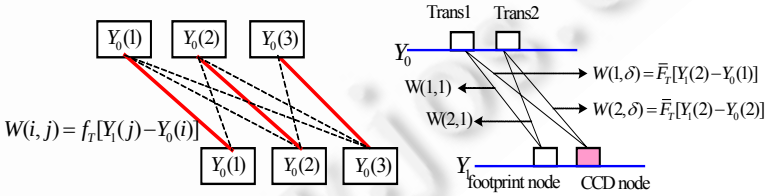
这里,$\bar{F}(t)=P[T>t]$ 是 CCDF.对于偶图 $G=\langle V_0\cup V_1,E\rangle$,有额外节点 $V_1(\delta)$ (后面简称为 CCDF 节点),对于边(i,j),现在分配一个权重 $W(i,j)$:

$$W(i,j):=\begin{cases}-\log f_T[Y_1(\pi(i)-Y_0(i)], & j\leqslant k \quad (9a)\\ -\log \bar{F}_T[Y_1(\pi(k)-Y_0(i)], & j=\delta \quad (9b)\end{cases}$$

这里假定踪迹按正确顺序达到.当踪迹没有按正确顺序达到,将不添加 CCDF,边的权重仅有(9a)决定.最大权重完全匹配为

$$\pi^*(n,k,W):=\arg\max_{\pi}\sum_{i=1}^{n}W(i,\pi(i)) \tag{10}$$

最大权重匹配的例子如图 3 所示,图 3(a)表示一个完全批,图 3(b)表示一个部分批.因为 n 和 m 相继被消减,则批的创建使偶图匹配可以有效实现.在下面定理中,给出了 MLR π^{ML} 和匹配概率 P^{ML}.对于 n 和 m,文献[9]给出了最大权重完全匹配的时间复杂度为 $O(n(m+n\log n))$.这是一个指数级的复杂度,当 n 和 m 值很大时,最大权重匹配花费的时间也较长,这很难适用于时效性要求很高的不完全标记事务踪迹的“剥离”.第 4 节提出了优化的最大权重匹配算法,该算法在线性空间复杂度情况下具有高效的 $O(\min(n+r,r\sqrt{n})m)$ 时间复杂度.假定在一个匹配中有 10 000 个顶点,10 000 条边,边级分割 r 为 200,采用文献[9]则时间花销约为 1.02×10^9,而采用本文提出的算法时间花销约为 1.02×10^8,时间花销仅为文献[9]算法的约 1/10.



(a) Complete batch matching (b) Partial batch matching including CCDF events

Fig.3 Simplifying MLR to weight matching

图 3 简化 MLR 为权重匹配

定理 1(I.I.D.事务). 在一个两状态系统中,对于 I.I.D.事务,转换时间由给定的 PDF f_T 确定,踪迹按正确顺序到达,MLR 由公式(10)中的最大权重完全匹配给出,在一(n,k,W)批中,根据 MLR,所有踪迹被正确匹配的概率为

$$P^{ML}(n,k,W)=\frac{(n-k)!\exp(W^*)}{perm[\exp(W)]} \tag{11}$$

这里, $\exp(W)=\exp(W(i,j))$,W 由公式(9)给出, W^* 为最大权重匹配值.对于踪迹没有按正确顺序达到的情况.仅基于公式(9a)求边权重.这里可以看到:针对 I.I.D.事务,MLR 的复杂性被大大消减.

3 排列最大匹配的改进算法

3.1 预备知识

设 $V\mapsto\mathbb{Z}_{\geqslant 0}$ 为一个由 G 的顶点定义的潜在功能.对于每条边 $e=(v,w)\in E$ 表示其权重为 $c(e)$,同时定义了与 π 相关的消减权重为 $\bar{c}(e)=\pi(v)+\pi(w)-c(e)$.此外,如果 $\bar{c}(e)=0$,那么称这种情况为边的紧缩.对于 c 来说,如果所有边 $e\in E$ 满足 $\bar{c}(e)\geqslant 0$,我们说 π 是一个可行的潜在功能.如果在 G 中有一个匹配 M 对于每条边 $e\in E$ 满足 $\bar{c}(e)=0$,且对于所有顶点 $v\in V$ 有 $\pi(v)=0$ (他们是 M 中自由顶点),我们说可行的潜在功能 π 是优化的.

既然对于任何匹配 M 有 $c(M)=\sum_{e=(v,w)\in M}c(e)\leqslant\sum_{e=(v,w)\in M}\pi(v)+\pi(w)\leqslant\sum_{v\in V}\pi(v)=\prod$,则定义 $\prod=\sum_{v\in V}\pi(v)$.此外,既然任何自由顶点的潜在功能为 0,则对于优化的潜在功能 π 和相应匹配 M 有 $c(M)=\prod$.

本文提出的排列最大匹配的改进算法使用 Kao 等人提出的分解理论[10].对于整数 $h\in[1,C]$ (这里,C 是边的最大权重),将 G 分解成两个更加简单的子图 G_h 和 G'_h :(1) G_h 由边 $(u,v)\in G$ 组成,以便 $c(e)\in[C-h+1,C]$.一条边 $e\in G_h$ 有权重 $c_h(e)=c(e)-(C-h)$;(2)设 π_h 为针对优化 G_h 的潜在功能.如果 $\pi_h(u)+\pi_h(v)-c(e)<0$,则一条边 $e=(v,w)\in G$ 属于 G'_h .在这种情况下, G'_h 中的边 e 有权重 $c_h(e)=c(e)-\pi_h(u)-\pi_h(v)$.

定理 2(偶图边分解). 针对上述 G, G_h 和 G'_h ,设 $mwm(G)$ 代表 G 中最大权重匹配的权重,那么 $mwm(G)=mwm(G_h)+mwm(G'_h)$.

如果 π_h 和 π'_h 是针对 G_h 和 G'_h 的优化潜在功能,则 $\pi_h+\pi'_h$ 是 G 的一个潜在优化功能[10].

3.2 边的分解

考察排列最大化匹配问题的一个实例和权重匹配消减的实例,G 的边有形如 $1,n,n^2,...,n^{r-1}$ 的权重,用定理 2 将其分解并用递归方法解决它.递归的实质是最大基数匹配.

选择 $h=n^{r-1}-1$,那么 G_h 包含 G 的边,其权重范围为 $[2,n^{r-1}]$, G_h 中每条边有权重 $c_h(e)=c(e)-1$.这样,图 G_h 包含排列最高的 $r-1$ 条 G 的所有边,这些边的权重为 $n-1,n^2-1,...,n^{r-1}-1$.图 G'_h 更加简单,假定 π_h 是 G_h 的优化潜在功能, G'_h 仅包含负数权重被消减后的 G 的边,这样边自然被分成两类:(1) 边 $e=(v,w)\in E$,这里 $c(e)=1$ 且 $\pi_h(u)+\pi_h(w)=0$,这样的边在 G'_h 中代价为 1;(2) 边 $e=(v,w)\in E$,这里 $c(e)>1$ 且 $\pi_h(u)+\pi_h(w)-c(e)<0$,由于 G_h 中 π_h 的可行性, $\pi_h(u)+\pi_h(w)-c_h(e)\geqslant 0$.这里有 $c_h(e)=c(e)-1$,可以推导出 $\pi_h(u)+\pi_h(w)-c(e)\geqslant -1$,且所有 G'_h 中的边代价为 1.这些恰是 G_h 中拥有紧缩 π_h 的边.

分解导致两个子图(如图 4 所示).既然所有边的权重是 1, G'_h 中的子问题是最大基数匹配计算.另一方面,图 G_h 有形如 $n-1,n^2-1,...,n^{r-1}-1$ 权重的边.后续将展示这些权重的优化潜在功能可针对权重 $1,n,n^2,...,n^{r-2}$ 进行.这样,在 G_h 中的子图就是用 $r-1$ 排列计算得到的最大匹配,这可以在 $T(r-1)$ 时间内通过递归方法解决,这里 $T(r)$ 是用 r 个不同排列解决一个实例的时间.

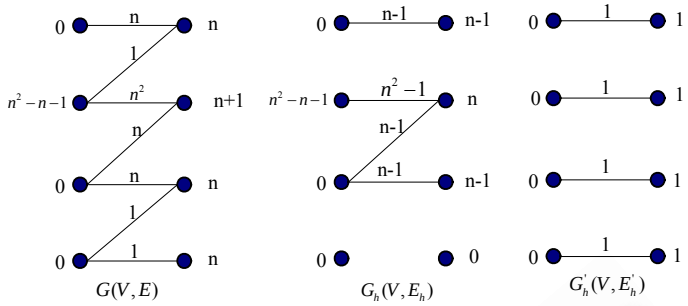


Fig.4 The edge decomposition instance of Bigraph

图 4 偶图边的分解实例

额外需要关注的问题是算法执行过程中观察计算操作所花费的代价,既然潜在的功能 π_h 花费代价达 $O(n^r)$.在这方面考虑针对优化潜在功能 π 的如下表示:(1) 节点 V^0 集包含所有节点 $v\in V$ 且 $\pi(v)=0$;(2) 边 E^0 集包含所有边 $e=(v,w)\in E$,这些边是关于 π 的紧缩边,即 $\bar{c}(e)=\pi(u)+\pi(w)-c(e)=0$.

以上两个集合能够指导匹配 M 以 $O(\sqrt{n}m)$ 时间来构建,以便任何被匹配的边是紧缩的且每个自由顶点有 0 个潜在优化.以上隐含了这样一个问题:这样一个匹配与优化潜在功能有相同的代价,因此是自优化的.在该算法中,处理这两个集合并维护所有元组对应的某些优化潜在功能不变.

3.3 排列问题

设 G 为边权重为 $1,n,n^2,...,n^{r-2}$,设 V^0 和 E^0 为两个表示优化潜在功能的集合.这部分将展示针对边权重为 $n-1,n^2-1,...,n^{r-1}-1$ 的同一集合.更确切地讲,存在优化的潜在功能以便这两个集合是其表示.

假定已经解决了以 $T(r-1)$ 时间计算边代价为 $1,n,n^2,...,n^{r-2}$ 的子问题,有优化潜在功能 π (由 V^0 和 E^0 表示).为了得到针对边代价 $n-1,n^2-1,...,n^{r-1}-1$ 的优化潜在功能,第 1 步是得到一个针对边代价 $n,n^2,...,n^{r-1}$ 的优化潜在功能.如下定理阐述了该问题.

定理 3(潜在功能优化). 针对边代价为 $1,n,n^2,...,n^{r-2}$ 的图 G,设 M 为 G 的一个最大权重匹配,且 $\pi:V\mapsto\mathbb{Z}_{\geqslant 0}$ 是证明了其优化性的潜在功能,那么潜在功能 $n\pi$ 证明了针对边代价为 $n,n^2,...,n^{r-1}$ 图 G 的匹配 M 是优化的.

设 π' 为通过定理 3 中 $n\pi$ 得到潜在功能.从 π 的可行性看,对 π' 的定义和潜在功能的集成将得到定理 4.

定理 4(潜在功能性质). 对于任何节点 $v\in V$,或者 $\pi'(v)=0$,或者 $\pi'(v)\geqslant n$;对于任何边 $e\in E$,或者 $\bar{c}(e)=\pi'(u)+\pi'(w)-c(e)=0$,或者 $\bar{c}(e)\geqslant n$.

同一集合 V^0 和 E^0 是针对新权重的新优化潜在功能 π' 的表示.下一步就是潜在功能 π'' 的构建,而 π'' 是针对权重 $n-1,n^2-1,...,n^{r-1}-1$ 的优化.优化潜在功能(optimal potential function,简称 OPF)算法构建了这样的潜在优化.

算法 1. OPF 算法.

输入:针对边代价为 $n,n^2,...,n^{r-1}$ 的优化潜在功能 π'.

输出:针对边代价为 $n-1,n^2-1,...,n^{r-1}-1$ 的优化潜在功能 π''.

```
1.  Begin
2.    While G= 有 π'(v)=0 的顶点
3.      设 A_v 和 B_v 分别为 G= 中从 v 可达到的偶数和奇数路径;
4.      For  w∈A_v ,Set  π''(w)=π'(w);
5.      For  w∈B_v ,Set  π''(w)=π'(w)-1;
6.      Delete A_v 和 B_v;
7.    End While
8.    If  G= 不为空
9.      设 A∪B 为 G= 的可保留集;
```

10. For 每个 $w\in A_v$, Set $\pi''(w)=\pi'(w)$;
11. For 每个 $w\in B_v$, Set $\pi''(w)=\pi'(w)-1$;
12. End If
13. End.

定义 1(等同子图). 对于边代价 c 为 $E\mapsto\mathbb{Z}_{>0}$ 的图 $G(V,E)$,其潜在功能为 $\pi:V\mapsto\mathbb{Z}_{\geqslant 0}$,设等同子图 $G_{=}(V,E_{=})$ 为边集 $E_{=}=\{e=(u,v)\in E:\ \bar{c}(e)=\pi(u)+\pi(v)-c(e)=0\}$ 的图.

3.4 合并问题

下面着重解决的问题是:(1)图 G'_h 和一个优化潜在功能 π'_π,其通过最大基数匹配计算获得;(2)图 G_h、集合 V^0 和 E^0 表示一个优化潜在功能 π_h.

合并两个方案需要添加两个潜在功能 π_h 和 π'_h.添加操作通过改变基于潜在功能 π'_h 的 V^0 和 E^0 而隐含地被执行.更新 V^0 需要检查每个顶点 $v\in V$ 是否有 $\pi_h(v)=\pi'_h(v)=0$,而 $\pi_h(v)=\pi'_h(v)=0$ 可以通过检查 $\pi'_h(v)=0$ 和 $v\in V^0$ 来完成.更新 E^0 将稍微复杂些.

(1) 对于 G 中 $c(e)=1$ 的每一条边 $e=(v,u)$,即 $e\in E_r$,需要检查是否 $\pi_h(v)+\pi'_h(v)+\pi_h(u)+\pi'_h(u)=1$.在第 4.3 节已经阐述:如果一个顶点 $v\in V$ 有 $\pi_h(v)>0$,那么 $\pi_h(v)>n-2$,因此可以检查: 1) $\pi'_h(v)+\pi'_h(u)=1$ (为常数时间的操作,因为 π'_h 是多项式界限的); 2) $\pi_h(v)=\pi_h(u)=0$ 可以通过检查 $\{v,u\}\in V^0$ 实现.

(2) 对于其他边,需要检查 $\pi_h(v)+\pi'_h(v)+\pi_h(u)+\pi'_h(u)=c(e)$.通过 π_h 的可行性,可知 $\pi_h(v)+\pi_h(u)\geqslant c(e)-1$.此外,对于一个边 e, $\pi_h(v)+\pi_h(u)\neq c(e)-1$,可知 $\pi_h(v)-1+\pi_h(u)-1-(c(e)-1)\geqslant n-1$ (对于边权重被减 1 的,当转换成一个新的潜在功能时,其为最坏情况,两个终点通过减少 1 获得他们潜在功能).因此,这样的边不能被紧缩.

这里得到如下结论:如果一条边有 $\pi_h(v)+\pi_h(u)=c(e)-1$,且属于 E^0,如果 $\pi'_h(v)+\pi'_h(u)=1$,那么它将保持紧缩.算法的最后输出是优化潜在功能的集合 V^0 和 E^0.

在这些集合中,一个排列最大匹配可以通过执行一个最大基数匹配计算来构建.设 $G_{=}(V,E^0)$ 为最后等同子图.创建一个新图 $G^{\alpha\beta}$,其包括 $G_{=}$ 两个拷贝 $G^{\alpha}(V,E^{\alpha})$ 和 $G^{\beta}(V,E^{\beta})$.对于顶点 $v\in V$,设 v^{α} 和 v^{β} 为新图中的两个拷贝.那么,如果 $v\in V^0$ 包括边 (v^{α},v^{β}).最后在 $G^{\alpha\beta}$ 中发现一个最大基数匹配 M,匹配 $M\cap E^{\alpha}$ 是原图的最大权重匹配.

算法 2. 排列最大匹配算法.

输入:有边分割的 $E_1,E_2,...,E_r$.

输出:集合 V^0 和 E^0.

1. Begin
2. If $r=1$
3. 计算 G 中的匹配 M 和最优的 π;
4. 基于 π 计算 V^0 和 E^0 并返回他们;
5. Else
6. 针对 $G(V,E\setminus E_r)$ 和 $E_1,E_2,...,E_{r-1}$,递归地处理实例;
7. 设 V^0 和 E^0 为结果;
8. 设 $E^{+}=\{e=(v,u)\in E_r:(v,u)\in V^0\}$;
9. 形成无权重的 $G'_h(V,E^{+}\cup E^0)$ 和在 G'_h 发现潜在优化功能 π'_h;
10. Set $E^0=\{e=(v,u)\in E^{+}\cup E^0:\pi'_h(v)+\pi'_h(u)=1\}$;
11. Set $V^0=\{v\in V^0:\pi'_h(v)=0\}$;
12. Return V^0 和 E^0;
13. End If
14. End.

4 SMP 模型

上述两状态系统表示只有进入和退出点的高层模型.当观测到更多的系统状态,如图 1 的子过程包括:订火车票、订飞机票、订宾馆、在线支付、送票等,两状态系统能够被扩充为多状态系统.假定每个事务的转换时间组成一个多状态的 SMP,不同事务的状态转换相互独立,这里考虑针对事务的所有可用踪迹的 ML 匹配.

既然踪迹一般仅携带进入一个状态的时间戳,甚至事务访问过的状态集不能被确切跟踪.因此,需要发现可用踪迹间的最相似匹配,并用排列实例的 ML 序列表示:

$$[\hat{\pi}_1^{ML},...,\hat{\pi}_{N_s}^{ML}] := \arg\max_{\pi_1,...,\pi_{N_S}} P(Y_1^{\pi_1},...,Y_{N_s}^{\pi_{N_s}} \mid Y_0) \tag{12}$$

通过转换, $\pi_0 = I$,从开始状态 S_0 到所有结束状态,通过对排列矢量的 ML 序列进行强制搜索,找到可能的踪迹.利用半马尔可夫属性(事务时间仅依赖于当前状态和下一个状态),将上述过程简化为

$$P(Y_1^{\pi_1},...,Y_{N_s}^{\pi_{N_s}} \mid Y_0) = \prod_{m=1}^{N_s} P(Y_m^{\pi_m} \mid \bigcup_{j\in P(m)} Y_j^{\pi_j}) \tag{13}$$

然而,发生在任何两个时期的事件可能彼此不连接,因为 S_k 和 S_l 的直接前驱 $P(k)$ 和 $P(l)$ 的集合可能不相交.这意味着不能用公式(13)直接匹配踪迹.因此,需要构建高层状态,包括多个局部化踪迹运动的模型状态,从而能够在这些高层状态中执行独立的匹配.最后,定义状态 $(B_m)_{m\geq 0}$ 的分割,以便分割中的任何两个状态不共享同一直接前驱.

$$P(S_k) \cap P(S_l) = \varnothing, \ \forall S_k \in B_m, S_l \in B_j, m, j \neq 0 \tag{14}$$

设开始状态 $B_0 = S_0$,既然认为状态转换图是 DAG,公式(13)中的分割可以被定义.因此,可以重写公式(13)为

$$P(Y_1^{\pi_1},...,Y_{N_s}^{\pi_{N_s}} \mid Y_0) = \prod_{m>0} P\left(\bigcup_{S_k\in B_m} Y_k^{\pi_k} \mid \bigcup_{S_l\in P(B_m)} Y_l^{\pi_l} \right) \tag{15}$$

在每个阶段,对于偶图系统 $(P(B_m),B_m)$,其开始状态为 $P(B_m)$,结束状态为 B_m .这些偶图系统是不相交的;一个状态不能在两个系统中扮演同样角色,即通过定义,所有 B_m 是彼此不相交的集合(分割),在公式(15)中,要求集合 $P(B_m)$ 是不相交的.因此,这里能够在这些偶图系统中实现非集中的匹配.这意味着:在每个偶图系统中,踪迹的信息和模型参数(如转换时间 PDF)仅局部需要.在所有的偶图中完成了匹配,通过拼接偶图匹配结果得到公式(1)中的排列矢量 π_i^{ML} ,由事务产生的踪迹的最可能序列被构建出来.下面给出分割 (B_m) 的一些属性.

定理 5(分割 $(B_m)_{m\geq 0}$ 的属性).对于无循环转换图的 SMP 和状态分割 $(B_m)_{m\geq 0}$,如果公式(13)成立,在有限递归次数内,下列属性成立:

$$N(P...(N(P(B_m)))) \subset B_m, \ m>0 \tag{16}$$

$$P...(N(P(B_m))) \subset P(B_m), \ m>0 \tag{17}$$

本文采用文献[10]提出的偶图状态划分算法,其算法描述如下:

算法 3. 偶图状态划分算法.

输入:拥有 DAG G 的 SMP 和状态 S 的集合.

输出:割集 B_m .

```
1.  Begin
2.      P、N 分别为 G 中直接前驱和后继;
3.      Set B_0 ← S_0 (开始状态), m ← 1;
4.      While ⋃_{i=0}^{m-1} B_i ≠ S do m−1
5.          对于那些 S_i ∈ S \ ⋃_{i=0}^{m-1} B_i, Init B_m ← S_i;
6.          Repeat
7.              Prev_B_m ← B_m;
8.              B_m ← N(P(B_m));
9.          Until B_m = Prev_B_m
10.         m ← m+1;
```

11. End While
12. Return $(B_m)_{m\geqslant 0}$;
13. End.

从定理 5 可知:在 $P(B_m)$ 中的任何状态的直接后继在 B_m 中,且 B_m 中任何状态的直接前驱在 $P(B_m)$ 中.因此,在 $(P(B_m),B_m)$ 中的状态集合表示完全的一步向前和向后的可达性.使用定理 5 的属性,由公式(16)和公式(17)产生唯一的划分 (B_m).这里,算法能够在有限时间内终止,因为状态转换图是无循环的,事务经过每个状态仅一次.

现在为每个偶图系统 $(P(B_m),B_m)$ 指定节点和边的权重.根据公式(3),定义 $Cnt(P(B_m))$ 作为观测时驻留在 $P(B_m)$ 的实例数量.

$$Cnt(P(B_m))=|Y_{P(B_m)}|-|Y_{B_m}| \tag{18}$$

根据公式(9),对于任何状态 $S_k\in B_m$, $S_l\in P(B_m)$,且从 PDF $f_{T_{k,l}}$ 提取的 I.I.D.转换时间,在同一批次内 S_k 的第 i^{th} 踪迹和在 S_l 的第 j^{th} 踪迹间边的权重为

$$W(i,j;S_k,S_l):=\log[f_{T_{k,l}}(Y_l(j)-Y_k(i))],\ \forall S_k\in P(S_l),\ 1\leqslant i,j\leqslant|Y_k|,\ 1\leqslant j\leqslant|Y_l| \tag{19}$$

对于部分批次,有 $Cnt(P(B_m))>0$,有一些实例仍旧驻留在 $P(B_m)$.当踪迹按正确顺序到达时,为每个状态 $S_k\in P(B_m)$ 定义 CCDF 事件,当在 S_k 踪迹数量比它的直接后继数量多时,即 $Cnt(S_k)=|Y_k|-\sum_{l\in N(k)}|Y_l|>0$.事务对应第 i^{th} 踪迹仍旧驻留在 S_k 的概率:

$$P_{ccdf}(i,S_k):=\sum_{l\in N(k)}\bar{F}_{T_{k,l}}[Y_l(|Y_l|)-Y_k(i)] \tag{20}$$

这里 $Y_l(|Y_l|)$ 是状态 S_l 最近产生的踪迹.在该情况下,对应于状态 S_k 的 CCDF 边权重为

$$W(i,\delta_k;S_k):=P_{ccdf}(i,S_k),\ \ \forall 1\leqslant i\leqslant|Y_k| \tag{21}$$

CCDF 节点 δ_k 的 $Cnt(S_k)$ 个等同拷贝被添加到偶图中.

定理 6(SMP 中的 MLR). 给定一个无循环状态转换图和 I.I.D.事务的 SMP,MLR 由偶图系统 $(P(B_m),B_m)$ 的非集中的最大权重匹配给出,这里分割 (B_m) 由算法 3 给出,每个偶图的边权重由公式(19)和(21)给出,排列最大化的权重匹配由算法 2 给出,所有被正确跟踪的事务的概率是所有偶图系统 ML 概率结果。

上述论述表明:优化 MLR 能够最大化地将事务踪迹与产生他们的实例进行匹配,而采用改进后的排列最大化的权重匹配算法可以有效提高匹配效率.

如图 5 为事务踪迹与产生他们的实例的 MLR 匹配例子,用文献[10]的算法对其进行偶图划分得到: $B_0=\{A\}$, $B_1=\{B,C,D\}$, $B_2=\{E\}$ 和 $B_3=\{F,G,H\}$ 四个划分,({A,C,D},{B,C,D}),({B},{E})和({E,G,H},{F,G,H})组成 3 个偶图系统.图 6 是偶图匹配示意图,匹配是通过与偶图系统中所有状态联合完成的,但独立地跨系统和批是允许的,匹配结果被最后拼接.

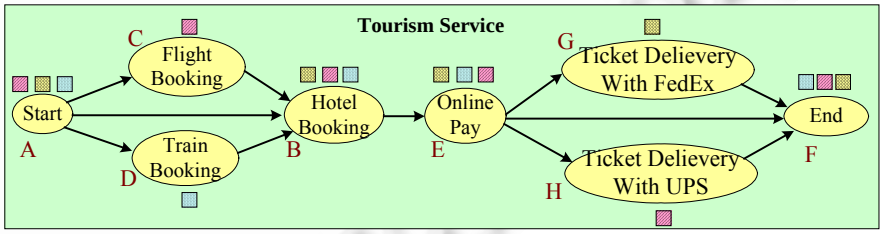


Fig.5 The MLR matching example of transaction footprint and instances that produce them

图 5 事务踪迹与产生他们的实例的 MLR 匹配例子

因此,从定理 6 中可知,偶图系统中的 MLR 可以被分解成非集中匹配,其复杂性由偶图的数量和每个偶图的规模决定;如果数量很大,在不同状态的事务不能够有很大的交错.然而,在最坏情况下,可能只有一个分割 $B_1=S\setminus S_0$,包含除 S_0 的所有状态,例如所有状态为 S_0 的直接后继.另一方面,对于 N_s+1 个状态,最好情况是线性转换,即有 N_s 个偶图.线性图是树的特例,树中偶图系统的数量等于树中非结束状态数,每个偶图系统包含一

个非终止状态和它的直接后继(孩子节点).

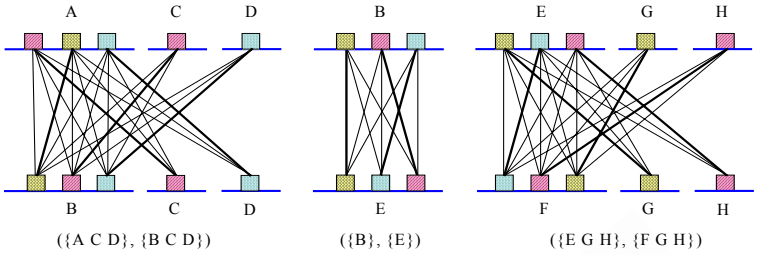


Fig.6 The diagram of bigraph matching

图 6 偶图匹配示意图

5 仿真实验

5.1 事务踪迹“剥离”的准确性测试

为验证本文提出的不完全标记行为踪迹“剥离”方法的准确性,用实际监测到的带标记事务踪迹数据作为测试数据(这里采用图 5 的实际数据),比较“剥离”方法的计算结果是否与真实情况相符.为了便于计算,这里只考虑状态间转换的时间,忽略在各状态停留的时间.为了便于说明,这里对各状态产生的踪迹进行了标号(图 1).事务踪迹在各状态的时间戳见表 1.

Table 1 The footprints and correspondence timestamps

表 1 事务踪迹在各状态的时间戳

编号	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
时间戳	0.000	0.105	0.396	0.210	3.485	1.192	0.899	5.116	2.095	5.898	1.608	4.096	2.066	6.586	3.215	6.118

表 2 是偶图({A,C,D},{B,C,D})的所有可能事务踪迹及权重之和.从表 2 可以看出:该偶图的所有可能踪迹序列有 18 种,而只有 4 种踪迹序列算出权值和,因为其他踪迹序列的时间戳之差为负值,这样的情况在实际中不可能发生.在 4 种踪迹序列中,2 号踪迹的权重和最大,这与图 5 的实际踪迹序列完全相符,说明采用本文提出的方法可以有效实现不完全标记事务踪迹的“剥离”.表 3 是偶图({B},{E})的所有可能事务踪迹及权重之和.从表 3 可以看出:该偶图的所有可能踪迹序列有 6 种,而只有 2 种踪迹序列算出权值和,其中 1 号踪迹的权重和最大,这与图 5 的实际踪迹序列相符.

Table 2 All footprints and weight sum in Bigraph 1

表 2 偶图 1 的所有踪迹及权重和

序号	踪迹序列	权重和
1	1−4−6, 2−7, 3−5−8	10.8677
2	1−4−7, 2−6, 3−5−8	11.5921
3	2−4−6, 1−7, 3−5−8	10.7532
4	2−4−6, 3−7, 1−5−8	11.4001
…	……	——

Table 3 All footprints and weight sum in Bigraph 2

表 3 偶图 2 的所有踪迹及权重和

序号	踪迹序列	权重和
1	6—9, 7—11, 8—10	3.9788
2	6—11, 7—9, 8—10	3.8297
…	……	——

下面对部分踪迹尚未产生情况(即含部分批)的事务踪迹“剥离”的准确性进行测试.这里测试图 5 中踪迹 6、9、12 和 16 尚未产生的情况.对于未产生踪迹的状态的 CCDF 边权重由公式(9b)计算得到.表 4 是偶图({A,C,D},{B,C,D})的所有可能行为踪迹及权重之和,表 5 是偶图({B},{E})的所有可能行为踪迹及权重之和.行为踪迹“剥离”结果与图 5 的实际踪迹序列依然相符.

Table 4 All footprints and weight sum in Bigraph 1

表 4 偶图 1 的所有踪迹及权重和(含部分批)

序号	踪迹序列	权重和
1	1－4－6, 2－7, 3－5－8	10.9853
2	1－4－7, 2－6, 3－5－8	11.2876
3	2－4－6, 1－7, 3－5－8	11.1052
4	2－4－6, 3－7, 1－5－8	10.8574
…	……	——

Table 5 All footprints and weight sum in Bigraph 2

表 5 偶图 2 的所有踪迹及权重和(含部分批)

序号	踪迹	权重和
1	6—9, 7—11, 8—10	3.3656
…	……	——

上面通过一组实验证实本文提出的方法对完全批和部分批的不完全标记事务踪迹都可以有效实现“剥离”.下面选取 500 组数据测试该算法的准确性.从图 7 的测试效果分析,本文提出方法在事务踪迹“剥离”上准确率可达 89%,误判率约为 11%.图 8 是排列最大匹配系统的运行界面,通过该系统可以直观、有效地完成匹配工作.

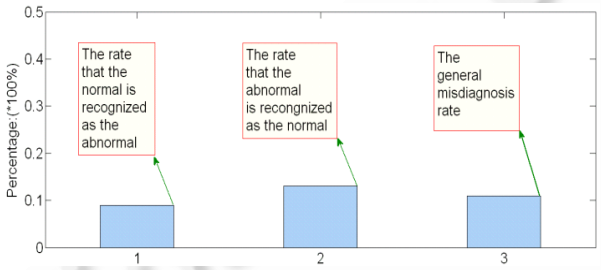


Fig.7 The misdiagnosis rate of stripping algorithm

图 7 事务踪迹“剥离”的误判率

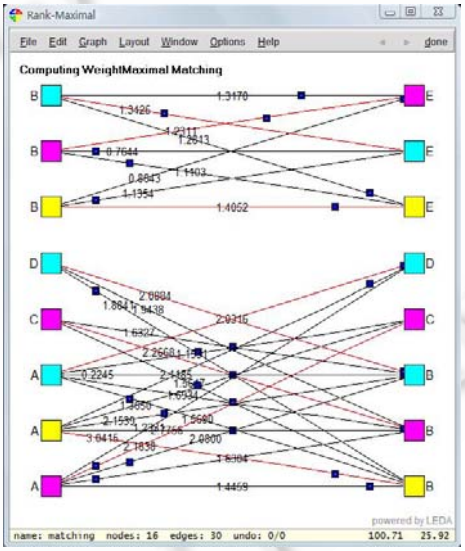


Fig.8 The running interface of rank-maximal matching system

图 8 排列最大匹配系统运行界面

5.2 算法性能对比测试

假定偶图节点概率服从正态分布,下面比较传统的最大权重匹配算法与本文提出的改进算法的性能.在仿真试验中,将偶图节点不断增加到 100,统计两种匹配算法花费时间(图 9).从图 9 可知:在相同的实验环境下,随着偶图节点数的增加,本文提出的最大权重匹配的改进算法性能优于经典的最大权重匹配算法 10 倍以上,能够满

足新型分布式软件环境下对不完全标记的事务踪迹进行有效“剥离”的要求.

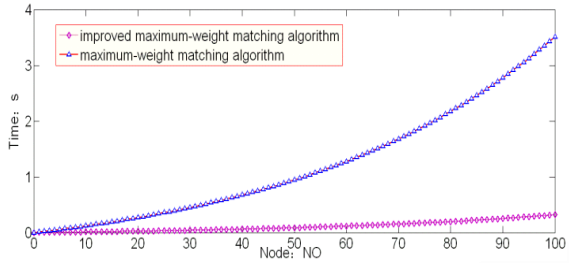


Fig.9 The contrast test of algorithm performance

图 9 算法性能对比测试

6 结束语

本文讨论了以往研究人员很少关注的不完全标记的多个并行事务踪迹的“剥离”问题.解决问题的思路是:如果事务的转换时间组成一个多状态的 SMP,不同事务状态的转换相互独立,将这些事务状态分割后形成若干个偶图;在各偶图内进行最大权重的完全匹配;当所有偶图完成了匹配,通过拼接偶图匹配结果得到“剥离”后的、添加了标记的事务踪迹.为了适应新型分布式软件环境下对不完全标记事务踪迹“剥离”效率的要求,本文对偶图最大权重完全匹配算法进行了改进.通过仿真实验证实:采用偶图最大权重完全匹配算法可有效“剥离”不完全标记的事务踪迹;最大权重完全匹配的改进算法在线性空间复杂度前提下具有更低的时间复杂度,其效率上较传统算法有显著的提高,更适合新型分布式软件环境下对时效性要求很高的不完全标记事务踪迹的“剥离”.

本文只讨论了事务开始状态有标记而其他状态无标记的特殊情况,而事务中部分有标记的“剥离”问题将更为复杂,我们后续的研究准备采用网络中的最大流算法解决该问题.

References:

[1] Li CY, Man JF, Wang ZB, Wen XB. Research on interactive behavior analyzing in new-type distributed software system. In: Jamshidi M, ed. Proc. of the Automation and Control. San Antonio: TSI Press, 2010. 513−517.

[2] Application Response Measurement. The Open Group. 2007. http://www.opengroup.org/tech/management/arm/

[3] Ahuja TL, Magnanti JB. Network Flows: Theory, Algorithms, and Applications. New Jersey: Prentice Hall, 1993. 1−35.

[4] Gabow HN, Tarjan R. Faster scaling algorithms for network problems. SIAM Journal of Computing, 1989,18(5):1013−1036.

[5] Goldberg AV, Kennedy R. An efficient cost scaling algorithm for the assignment problem. Mathematical Programming: Series A and B, 1995,71(2):153−177.

[6] Irving R, Kavitha T, Mehlhorn K, Paluch K. Rank-Maximal matchings. ACM Trans. on Algorithms, 2006,2(4):602−610.

[7] Irving R, Kavitha T, Mehlhorn K, Paluch K. Rank-Maximal matchings. In: Munro I, ed. Proc. of the 15th Annual ACM- SIAM Symposium on Discrete Algorithms. Society for Industrial and Applied Mathematics, 2004. 68−75.

[8] Mann HB, Ryser H J. Systems of distinct representatives. Journal of Mathematical Analysis and Applications, 1970,32(1):1−4.

[9] Anandkumar A, Bisdikian C, Agrawal D. Tracking in a spaghetti bowl: Monitoring transactions using footprints. In: Liu Z, ed. SIGMETRICS 2008. New York: ACM, 2008. 133−144.

[10] Monnot J, Toulouse S. The path partition problem and related problems in bipartite graphs. Operations Research Letters, 2007, 35(5):677−684.

满君丰(1976－),男,黑龙江海伦人,博士,副教授,CCF 会员,主要研究领域为可信软件技术,软件工程.

李长云(1971－),男,博士,教授,CCF 高级会员,主要研究领域为可信软件技术,软件工程.

文志诚(1972－),男,博士,副教授,主要研究领域为可信软件技术,软件工程.

温向兵(1983－),男,硕士生,主要研究领域为可信软件技术.