基于多路分块的 Pay-as-you-go 实体识别方法

孙琛琛 申德荣 寇 月 聂铁铮 于 戈
(东北大学计算机科学与工程学院 沈阳 110169)

摘 要 实体识别是数据集成和数据清洗的一个重要方面.针对 Pay-as-you-go 数据管理需求,本文提出一个基于多路分块的 Pay-as-you-go 实体识别方法.该方法不要求提供最优的分块或排序的键,并且可以直接找出脏数据集中冗余度最大的区域.分为两个阶段,初始化阶段和迭代阶段.在初始化阶段,初步地生成候选数据对象对,并按匹配可能性排序后加入到候选队列.在迭代阶段,每次选择候选队列队首的候选对(即最可能匹配的)来处理,并且根据实时的实体识别结果,动态地更新候选对的匹配可能性,调整候选队列.这样减少了无用的数据对象比较,使得实时的识别结果最优化.通过在真实数据集和合成数据集上的实验对比,说明本文提出的基于多路分块的 Pay-as-you-go 实体识别方法显著地优于已有工作中提出的方法.

关键词 实体识别;Pay-as-you-go;多路分块;候选对选择;数据集成;数据清洗
中图法分类号 TP391 **DOI 号** 10.11897/SP.J.1016.2019.01704

A Multi-Pass Blocking Based Pay-as-you-go Entity Resolution Approach

SUN Chen-Chen SHEN De-Rong KOU Yue NIE Tie-Zheng YU Ge
(*School of Computer Science and Engineering, Northeastern University, Shenyang* 110169)

Abstract Entity resolution (ER) is a key aspect of data integration and data cleaning, and is a necessary pre-processing step of data analytic and data mining. Traditional ER approaches take a whole dirty dataset as input, and output the complete ER result after a batch based process. However, nowadays many new applications emerge, demanding (nearly) real-time data analytic, but traditional batch based ER approaches cannot satisfy such requirements. For instance, a finance news feed tries to resolve as many companies and persons as possible within limited time, from financial data generated frequently. In order to fulfill such requirements, Pay-as-you-go ER tries to maximize number of resolved duplicate data objects given limited time (far shorter than overall running time). Pay-as-you-go ER is also called progressive ER, since it resolves data objects progressively. The more data objects an ER approach resolve within limited time, the higher its progressiveness is. The core challenge of Pay-as-you-go ER is to effectively select the most matchable object pairs for comparison with high priorities. Existing Pay-as-you-go ER solutions rely upon perfect blocking keys or sorting keys for pair selection. However, the best blocking/sorting keys cannot be got without deep domain knowledge and fully understanding of each dataset. It is impossible for common users. Worse still, perfect blocking/sorting keys do not always exist. We try to work out an effective Pay-as-you-go ER solution without perfect blocking/sorting keys. We resolve data objects progressively with multi-pass blocking. Multi-pass blocking

收稿日期:2018-07-04;在线出版日期:2019-03-14.本课题得到国家"九七三"重点基础研究计划基金项目(2012CB316201)、国家自然科学基金项目(U1435216,61672142,61472070,61602103)、国家重点研发计划项目(2018YFB1003404)资助.**孙琛琛**,博士研究生,主要研究方向为实体识别、数据质量.E-mail:dustinchenchen_sun@163.com.**申德荣**,博士,教授,博士生导师,中国计算机学会(CCF)高级会员,主要研究领域为分布式数据管理、数据集成.**寇 月**,博士,副教授,中国计算机学会(CCF)会员,主要研究方向为实体搜索、数据挖掘.**聂铁铮**,博士,副教授,中国计算机学会(CCF)会员,主要研究方向为数据质量、数据集成.**于 戈**,博士,教授,博士生导师,中国计算机学会(CCF)会士,主要研究领域为数据库、大数据管理.

results in blocking redundancy, which is helpful for computing match probabilities of candidate object pairs. Intuitively, the more blocks a pair shares, the more matchable the pair is. Yet different blocks usually offer different contributions to match probability of a pair, so it is necessary to evaluate the redundancy of each block. Meanwhile, blocking redundancy has to be eliminated efficiently before pair comparisons. We propose a blocking based Pay-as-you-go ER (BPER) approach. BPER utilizes multi-pass blocking instead of perfect blocking/sorting keys based techniques. Redundancy of each block is evaluated dynamically, and the evaluation result is called block credit. Block credits are used for real-time pair match probability computing. Also, an efficient graph based method is proposed to eliminate blocking redundancy. BPER consists of two stages: the initialization stage and the iterative stage. In the initialization stage, generate candidate data object pairs, and sort them according to match probabilities in a candidate queue. In the iterative stage, each time choose the front candidate pair (the most matchable pair) of the candidate queue for processing; dynamically update candidate pairs' match probabilities according to the real-time ER result, and then update the candidate queue. Candidate pairs resolution and block credits computation proceed interactively, and promote each other. As a result, the most matchable pairs are selected and resolved in real time. In such a way, the proposed ER approach reduces useless data object comparisons, and optimizes the real-time ER result. Finally, we experimentally evaluate the proposed Pay-as-you-go ER approach over real datasets and synthetic datasets. The experiment results show that BPER improves existing works greatly. We also evaluate the contribution of each component to progressiveness in BPER.

Keywords　entity resolution; Pay-as-you-go; multi-pass blocking; candidate pair selection; data integration; data cleaning

1　引　言

实体识别(Entity Resolution, ER)是数据集成和数据清洗的一个关键方面，是数据挖掘和数据分析的一个必要的预处理步骤[1-9]. 传统的实体识别方法将整个脏数据集作为输入，处理完成后输出识别结果[1,8]. 然而，当前出现了很多新的应用，要求(近似的)实时数据分析，传统的实体识别技术无法满足这一需求[10-12]. 在给定较短时间时(远小于识别完整个数据集所需的时间)，Pay-as-you-go 实体识别可以尽最大可能地优化识别结果，从而解决前面的需求. 例如，金融新闻的信息流应用通常很强调实时性，以便于客户及时地进行相应的金融业务处理. 股票市场的金融数据变化特别快，每隔一段时间就会生成新的数据；金融数据会涉及大量公司和个人的名字，而短时间内，不可能将这些数据全部识别出来. 信息流应用在发布金融新闻之前，需要在较短的时间内识别出大量的公司和个人的名字[10]. 为了解决这类应用需求，新的实体识别方法应该在给定的短时间内处理得到尽量多的重复(匹配)数据对象对(简称为“重复对”或“匹配对”).

定义 1.　Pay-as-you-go 实体识别. 相对于传统的实体识别，Pay-as-you-go 实体识别需要额外满足以下两个条件[12]：

(1) 早期的识别结果更好. 给定任意一个较短的运行时间 t，Pay-as-you-go 实体识别方法能够比传统的实体识别方法识别出更多的重复对. 时间段 t 远小于完全的实体识别运行时间.

(2) 相同的最终识别结果. 如果传统的实体识别方法和 Pay-as-you-go 实体识别方法都运行到自然终止，两者应该产生相同的识别结果.

本文将研究 Pay-as-you-go 实体识别，这类方法以 Pay-as-you-go 的形式逐渐优化实体识别结果，也称为渐近式实体识别. 在给定的短时间内，当一个实体识别方法识别出的数据对象越多，那么它的渐近性越高；反之，它的渐近性越低.

传统的实体识别方法可分为两类：基于聚类的方法(即无监督的，对应图 1 中传统方法 1)和基于分类的方法(即监督的，对应图 1 中传统方法 2). 如

图 1 所示，基于聚类的方法首先要计算所有候选数据对象对(简称为“候选对”)的相似度，然后再通过聚类算法得到识别结果，它需要在整个执行时间的后期才输出识别结果；在不考虑训练过程的前提下，基于分类的方法将逐个地处理候选对，在执行过程中逐步地输出识别结果，但它并没考虑候选对的优先级. 与前两类方法相反，Pay-as-you-go 方法花费较短时间进行预处理，估计候选对的匹配可能性，将候选对按优先级进行处理，从而在给定较短执行时间内识别出尽量多的重复对. 因此，如果只给定有限的执行时间，Pay-as-you-go 实体识别方法必将比传统的实体识别方法得到更多的重复对. 需要指出的是，如果运行到自然终止，Pay-as-you-go 实体识别方法可能比传统方法花费更长时间，因为预处理也有一定的时间开销.

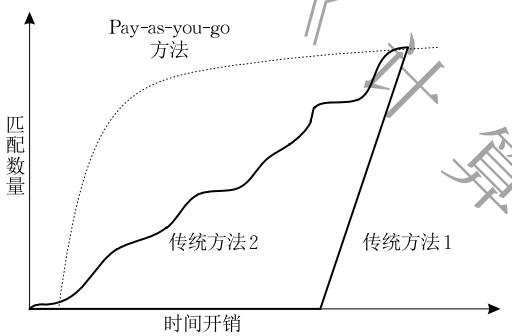


图 1 Pay-as-you-go 实体识别与传统方法对比

正如已有工作[10,12]所说，Pay-as-you-go 实体识别的核心是，根据候选对匹配的可能性，高效地选择最有可能匹配的数据对象对优先进行比较，从而保证高的实体识别渐近性. Whang 等人首次提出 Pay-as-you-go 实体识别的概念，并基于排序或分块技术提出三个 Pay-as-you-go 实体识别“线索(hints)”方法，这些方法的前提是给定了最优的排序或分块的键[10]. Papenbrock 等人提出了渐近式滑动窗口方法(Progressive Sorted Neighborhood Method，PSNM)和渐近式分块方法(Progressive Blocking，PB)，这两个方法都是基于排序的数据对象列表，即需要给定最优的排序键[12]. Papenbrock 等人的方法要优于 Whang 等人的方法，然而 PSNM 和 PB 方法都需要运行全局的滑动窗口，从而很有可能会降低渐近性. 比如，利用 PSNM 和 PB 方法来解决例 1 中的 Pay-as-you-go 实体识别任务. 如图 2 所示，将表 1 中的数据对象按照姓进行排列，得到一个从左到右的数据对象列表. 在该列表中存在两个区域：区域 a 是一个低冗余的区域，不存在重复数据对象；区域 b 是一个高冗余的区域，每两个数据对象都彼此重复($\langle r_1, r_4\rangle$，$\langle r_1, r_3\rangle$，$\langle r_4, r_3\rangle$)，一共有三个重复对. PSNM 和 PB 方法都要对图 2 中的数据对象列表进行从左到右的遍历，因此这两个方法都会先处理区域 a、然后才处理区域 b，推迟了重复对的发现，降低了渐近性. 由此可见，Pay-as-you-go 实体识别的一个大挑战是，如何准确定位出数据集中冗余度最高的区域. 另外，Whang 等人和 Papenbrock 等人的方法都假定最优的排序或者分块的键已经得到. 然而，找出最优的排序或分块的键，需要有丰富的领域知识以及对每个数据集的非常的熟悉程度. 普通的用户无法达到这样的要求，因此难以得到最优的排序或分块的键. 此外，一些数据集的最优的键并不存在. 如何克服键的选择问题是 Pay-as-you-go 实体识别的另一个挑战.

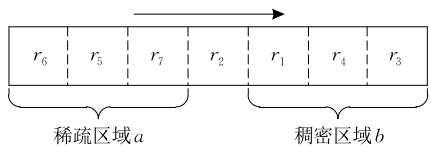


图 2 将表 1 中的记录根据姓进行排序，得到了从左往右的列表；该列表存在一个稀疏区域 $a(r_6, r_5, r_7)$和一个稠密区域 $b(r_1, r_4, r_3)$

表 1 样例脏数据集(包含 7 条个人记录，属性有姓名、年龄、工作和所在城市)

ID	姓名	年龄	工作	城市
r_1	John Young	29	Waiter	Poston
r_2	John Joung	29	Waiter	Boston
r_3	Jon Young	—	Waiter	Boston
r_4	John Young	29	Waiter	Boston
r_5	Bob Brown	27	Waiter	Austin
r_6	Jeff Allen	29	—	Boston
r_7	Will Green	29	Teacher	Boston

例 1. 如表 1 所示，有一个包含 7 条记录的样例数据集. 这是一个脏数据集，对应的真实识别结果是$\{\{r_1, r_2, r_3, r_4\}, \{r_5\}, \{r_6\}, \{r_7\}\}$. 当前要求渐近地识别这个脏数据集，也就是说，给定较短的运行时间，尽量识别出最多的重复记录对. 在随后的第 5 节，将利用本文提出的 Pay-as-you-go 实体识别方法逐步地来解决本例提出的问题.

本文将研究如何在最优的分块或排序的键未知的情况下，实现高效的 Pay-as-you-go 实体识别. Papenbrock等人利用一个属性并行(Attribute Concurrency，AC)技术将 PSNM 和 PB 方法拓展得到 AC-PSNM 和 AC-PB 方法，这两个新方法可以用于解决本文研究的问题[12]. 但是，AC-PSNM 和 AC-PB 方法要交替地处理 $|K|$(不同的排序的键的总数目)个按不同键排序的数据对象列表，并且没有

实时地定位到冗余度最高的区域，这样会严重影响渐近性.

Pay-as-you-go 实体识别的关键是，找出最可能匹配的候选对. 候选对选择的常用技术是分块[13-19]. 本文设定最优的分块的键是未知的，因此将利用多路分块来解决 Pay-as-you-go 实体识别的问题. 一方面，多路分块会产生大量的分块冗余；另一方面，可以利用分块冗余来估计候选对的匹配可能性. 直观来说，一个候选对同时出现在越多的块中，该数据对象对的匹配可能性越大. 如果一个数据对象对出现在一个块中，那么这个块是这个数据对象对的共现块；同一数据对象对，可能会出在多个块中，从而对应多个共现块. 通过不同的共现块来估计同一数据对象对，得到的匹配可能性很可能是不同的. 为了综合地估计一个数据对象对的匹配可能性，需要先评估该数据对象对的每个共现块. 同时，为了保证渐近性，在比较候选对之前，必须快速地去除掉分块冗余.

本文提出一个基于分块的 Pay-as-you-go 实体识别方法 BPER(Blocking based Pay-as-you-go Entity Resolution). BPER 可以精确地定位脏数据集中最冗余的区域，给予这样的区域内的候选对高优先级，从而解决了本节第 3 段提出的第一个挑战，保证了高渐近性. 随后第 5 节将通过例 2 到例 4 来示例说明，BPER 方法如何在给定的较短时间内识别出尽量多的重复对. BPER 利用多路分块来解决最优的分块键的选择问题. 动态地评估每个块的冗余度，然后利用块的冗余度来实时地计算候选对的匹配可能性. 候选对的比较和块冗余度估计交互地进行和彼此影响对方，从而在实体识别过程中实时地选择最可能匹配的数据对象对. 与此同时，为了保证渐近性，提出一个基于图的方法来快速地去除由于多路分块导致的分块冗余. 通过在两个真实的数据集和一组合成的数据集上的对比实验，证明 BPER 方法要优于已有方法.

本文的主要贡献总结如下：

(1) 提出两个块冗余度估计方法：基于静态的块信用度的块冗余度估计方法和基于动态的块信用度的块冗余度估计方法. 块冗余度估计对于 Pay-as-you-go 实体识别中的候选对的选择至关重要.

(2) 提出一个基于多路分块的 Pay-as-you-go 实体识别方法 BPER. 该方法基于有交叠(冗余)的分块，随着实体识别的进行，动态地估计候选对的匹配可能性. 同时，高效地去除分块冗余，保证高渐近性.

(3) 在两个真实的数据集和一组合成的数据集上评估了本文提出的 BPER 方法. 评估结果说明 BPER 方法要明显优于已有的 Pay-as-you-go 实体识别方法. 另外，分别测试了 BPER 方法的不同组成部分对渐近性的影响.

本文第 2 节介绍相关工作；第 3 节介绍准备工作，包括多路分块和方法概述；第 4 节介绍块冗余度估计，包括静态的块信用度、动态的块信用度和候选对的信用度；第 5 节详细阐述基于分块的 Pay-as-you-go 实体识别方法 BPER，包括 Pay-as-you-go 实体识别模型、初始化阶段和迭代阶段；第 6 节进行实验评价与分析，包括实验设置、综合测试和关键组成部分测试；最后，对全文进行总结.

2　相关工作

大部分已有的实体识别研究，要么聚焦于提高最终的识别结果的精确性，要么致力于降低实体识别运行的总时间. 近些年，随着数据增长得更快和数据演化得更频繁，Pay-as-you-go 数据管理方法逐渐流行，并得到研究者的重视[20-23]. 文献[23]提出了针对海量 Web 数据的 Pay-as-you-go 数据集成方法，文献[22]提出了面向数据空间系统的 Pay-as-you-go 数据清洗方法. 在实体识别研究领域，同样存在一些 Pay-as-you-go 的实体识别方法[10-12]，这些工作与本文的研究最相关.

Whang 等人最早研究了 Pay-as-you-go 实体识别，提出了三个启发式的 Pay-as-you-go 实体识别“线索”方法，这些方法需要与已有的实体识别方法结合使用[10]. 一个“线索”可以帮助传统的实体识别方法优先选择匹配可能性高的数据对象对来进行识别. 第一个“线索”方法，根据候选对的匹配的可能性生成一个排好序的数据对象对列表. 第二个“线索”方法，对数据集进行不同层级的划分，每一层的划分代表实体识别结果的一种可能. 第三个“线索”方法，根据某个数据对象与其他数据对象重复的可能性对数据对象进行排序，得到一个排序的数据对象列表. 所有的“线索”方法都生成固定的序列或划分，不能动态地调整，限制了其渐近性. 另外，这三个“线索”方法基于最优的排序或者分块的键，因此无法解决本文研究的问题. Papenbrock 等人提出一组 Pay-as-you-go 实体识别方法[12]. 渐近式滑动窗口方法(Progressive Sorted Neighborhood Method，PSNM)

扩展了传统的滑动窗口方法(Sorted Neighborhood Method,SNM),PSNM 方法迭代地将滑动窗口从小到大变化,同时通过预测函数来动态地改变识别顺序.渐近式分块方法(Progressive Blocking,PB)基于排序的数据对象列表与等距分块.PB 首先生成一个分块集合,然后逐渐地扩展这个分块集合.PSNM 和 PB 方法都是基于排序的数据对象列表.然而,在本文的问题设定中,最优的排序键是未知的.Papenbrock 等人提出两个基于多路技术的扩展方法:属性同步(Attribute Concurrency)的 PSNM(AC-PSNM)方法和属性同步的 PB(AC-PB)方法.AC-PSNM 和 AC-PB 方法交替地处理多个排序列表,并根据已有的识别结果动态地选择列表进行处理.这两个方法并不需要给出最优的排序键,因此可以解决本文提出的问题.然而,AC-PSNM 和 AC-PB 方法需要交替地处理同一数据集的不同排序列表,存在严重冗余,限制了其渐近性.Papenbrock 等人的方法是基于硬盘的,可以处理规模超出内存大小的数据集.Altowim 等人提出一个渐近的联合式实体识别方法[11].该方法定义了"收益-开销"模型,然后根据这样的模型生成联合式实体识别执行计划.Altowim 等人的方法针对多类型关联数据,如引文数据和电影数据,但不能用于处理单类型的数据集.

Pay-as-you-go 实体识别的核心是选择最可能匹配的数据对象对,以便在给定的较短时间内得到尽量多的重复对.已有的数据对象对选择方法通常有基于分块的方法和基于滑动窗口的方法[13-19],这些方法的目标都是降低总的实体识别时间.由此可见,不同的目标将 Pay-as-you-go 实体识别与传统的分块或滑动窗口方法区分开来.Papadakis 等人针对高度异构的数据空间提出一系列的分块技术,旨在不影响精确性的前提下减少实体识别总时间[16-17].Papadakis 等人的一些方法在一定程度上是 Pay-as-you-go 的,如分块调度、块删除、比较调度[16]和基于元的分块[17].

3 准备工作

3.1 多路分块

分块是实体识别中一种常见的数据对象对选择技术.本文将利用有交叠的分块来选择最可能匹配的数据对象对.直观来讲,两个数据对象共同出现在越多的块中,这两个数据对象匹配的可能性越大.

定义 2. 分块.给定一个脏数据集 $R=\{r\}$ 和一个分块的键 k,一个分块方法根据一个数据对象 r 的键值 $r.k$ 将 r 分到一个块 $b=d(r.k)$.$d(*)$ 是一个分配函数.分块结果集合 B 是不交叠的,$b_i \cap b_j = \varnothing$, $b_i, b_j \in B$.

特别地,如果一个数据对象在给定的键对应的属性值是缺失的话,该数据对象将不被分配到任何一个块中.

块中的一对数据对象称为候选数据对象对或候选对,记作$\langle r_i, r_j \rangle$,$r_i, r_j \in b$.$\langle r_i, r_j \rangle$和$\langle r_j, r_i \rangle$是相同的,称为对称性.如果一个候选对已经被识别过了,称为已识别的候选对;否则,就是未识别的候选对.一个块是一个数据对象的集合,块 b 的势(Cardinality)是 b 中不同数据对象对的总数目,记作 $\|b\| = \binom{|b|}{2} = (|b| \times (|b|-1))/2$,其中 $|b|$ 表示块 b 的对象个数.一个块 b 中所有候选对组成的集合,记作 $\Gamma(b)$,即 $|\Gamma(b)| = \|b\|$.在实体识别过程中,一个块 b 中所有已识别的数据对象对组成已识别的集合,记作 $\Xi(b) \subseteq \Gamma(b)$;所有未识别的数据对象对组成未识别的集合,记作 $(\Gamma(b) - \Xi(b)) \subseteq \Gamma(b)$.$\Xi(b)$中所有重复对组成的集合,记作 $\Xi^+(b)$;$\Xi(b)$中所有不重复对组成的集合,记作 $\Xi^-(b)$;$\Xi(b) = \Xi^+(b) + \Xi^-(b)$.

然而,最优的分块的键不易找出,甚至有时候是不存在的.对于一个数据集,一些键能够探测出一些重复数据对象,而另外一些键可能找出别的重复数据对象.特别地,给定的一个分块的键,一些数据对象的对应属性值可能是缺失的.多路分块能够提高实体识别精确性,不必进行最优的分块键的选择[14,18].多路分块给予重复数据对象更多机会,以便每个重复对至少出现在一个块中;同时,多路分块使得同一候选对可能同时出现在多个块中,即块之间存在交叠,导致了分块冗余.分块冗余可以用于估计候选对的匹配可能性.如果一个候选对出现在一个块中,称这个块是这个候选对的共现块.如果两个数据对象共同出现在了较多的共现块中,那么这两个数据对象很有可能是匹配的.

定义 3. 多路分块.给定一个脏数据集 $R=\{r\}$ 和一个分块的键的集合 $K=\{k_i \mid 0 \leq i \leq |K|\}$,多路分块的结果集合是 $B_m = B_1 \cup B_2 \cup \cdots \cup B_{|K|}$.$B_m$是一个有交叠的块集合,每个数据对象最多可能出现在 $|K|$ 个不同块中.

候选对匹配的可能性来源于该候选对所在的共现块(一个或多个),然而不同的共现块给同一候选

对带来的匹配可能性是不同的. 第 4 节将提出块冗余度估计方法，进而计算候选对匹配的可能性. 本文中，块的冗余度和块集合的冗余（也称为分块冗余）是不同的概念. 块的冗余度是指某一块中重复对相对于该块的势（即候选对总数目）的比例. 分块冗余则是指，同一候选对出现在块集合内多个块中，即不同的块有交叠. 单路分块生成没有冗余的块集合，而多路分块会生成有冗余的块集合.

3.2　方法概述

Pay-as-you-go 实体识别的核心问题是，在实体识别过程中如何优先选择最有可能匹配的候选对进行识别. 整体而言，BPER 方法以多路分块为基础，通过块冗余度来估计候选对的匹配可能性. 具体地，BPER 方法提出块信用度来估计块的冗余度；基于块信用度，又提出候选对信用度来计算候选对的匹配可能性（第 4 节）. BPER 方法根据候选对的信用度对所有的候选对进行排序；在实体识别过程中，根据实时的匹配结果更新候选对信用度，动态地调整候选对排序，每次都选择排序最靠前的候选对进行识别；因此，BPER 方法实时地选择最可能匹配的候选对来进行识别，保证了高渐近性.

图 3 描绘了 BPER 方法的整体工作流程. 虚线绘制的图形表示数据和数据结构，实体绘制的图形表示操作. 带箭头的虚线线条表示数据和操作的交互，带箭头的实线线条表示操作的先后顺序. 总体而言，BPER 方法由两个阶段组成：初始化阶段和迭代阶段（第 5 节）.

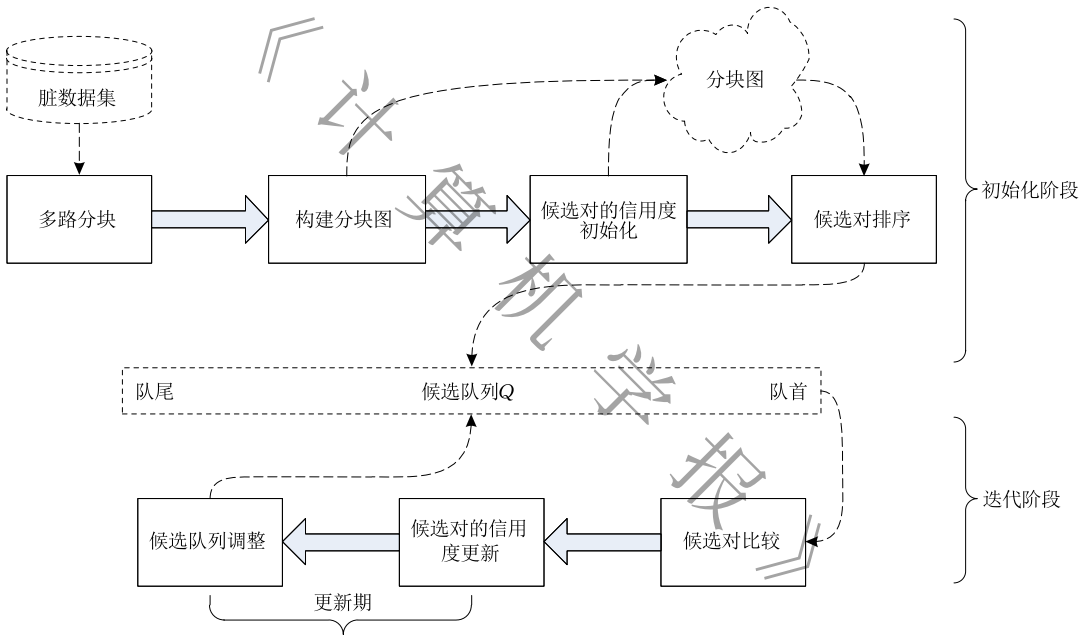


图 3　本文提出的 Pay-as-you-go 实体识别方法的工作流程

初始化阶段为迭代阶段准备候选对. 首先，利用多路分块构建有交叠的块集合，并通过构建分块图来快速去除冗余的数据对象对；接着，初始化候选对的信用度，将候选对的信用度作为分块图中相应的边的权重；最后，将所有候选对根据其信用度进行排序，并依次加入到一个优先队列 Q 中，称 Q 为候选队列.

迭代阶段是 BPER 的核心，包括三个迭代的步骤. 首先，从候选队列 Q 队首取得一个候选对，调用一个实体识别匹配函数来处理该候选对；接着，根据前一步的结果，对一部分候选对的信用度进行重新计算；最后，根据重新计算的候选对的信用度，调整候选队列. 将候选对的信用度重新计算和候选队列的调整合起来称为更新期. 上述迭代过程一直持续，直到时间预算用完或者候选队列为空.

本文聚焦于实体识别的渐近性，关注核心是候选对的识别顺序，而实体识别匹配函数的准确性与识别顺序是正交的，因此将匹配函数当作黑盒[12]. $m(\langle r_i, r_j \rangle)$ 表示一个实体识别匹配函数，如果 $\langle r_i, r_j \rangle$ 匹配，$m(\langle r_i, r_j \rangle)$ 返回真；否则，$m(\langle r_i, r_j \rangle)$ 返回假. 本文假定 $m(\langle r_i, r_j \rangle)$ 的精确性为 1，即永远返回正确的识别结果.

4　块冗余度估计

本节将介绍如何估计一个块的冗余度. 不同块

的冗余度通常是不同的.用块信用度来估计块的冗余度.块冗余度的估计是候选对的信用度的计算的基础,如图3所示,涉及初始化阶段的候选对的信用度初始化和迭代阶段的候选对的信用度的更新.首先,定义块信用度;然后,提出两种块信用度;最后,介绍如何利用共现块的信用度来计算候选对的信用度,即匹配可能性.

定义4. 块信用度.给定一个块b,块b的信用度是,块内未识别的候选对从该块获得的匹配可能性,

$$\sigma(b)=\Pr(m(\langle r_i,r_j\rangle)=\text{true}|\langle r_i,r_j\rangle\in(\Gamma(b)-\Xi(b))) \tag{1}$$

4.1 静态的块信用度

本小节提出静态的块信用度(Static Block Credit,SBC),只需要在进行实体识别之前进行一次计算.如果一个块中包含的数据对象越多,那么该块内的任意一个候选对匹配的可能性越小.较大的块中的候选对所拥有的信息量很小,不足以保证该候选对的匹配.比如,一个个人信息数据集(美国)中,很可能许多人来自同一个州,而没有那么多人拥有相同的姓.平均来讲,通过键"州"生成的块要比通过键"姓"生成的块要大.后者块内的候选对匹配的可能性要平均大于前者块内的候选对匹配的可能性.基于上述想法,静态的块信用度给予小块更高的优先级.

定义5. 静态的块信用度.给定一个块b,该块的静态的信用度与它的势成负相关,

$$\sigma_s(b)=1/\|b\| \tag{2}$$

4.2 动态的块信用度

静态的块信用度的一个缺点是,如果数据集中真实存在一些大的簇(即描述相同实体的数据对象组成的集合),那么给予所有大块较小的信用度并且一成不变是不合理的.为了克服这个缺点,可以在实体识别过程中动态地调整块的信用度,称之为动态的块信用度(Dynamic Block Credit,DBC).直观来讲,如果一个块中当前的重复数据对象所占的比例比较大,那么该块的真实的冗余度很可能是比较高的,即该块的信用度很可能是比较大的.简言之,给定一个块,根据当前的实体识别结果计算出的当前的冗余度可以估计该块真实的冗余度.基于这样的思想,提出动态的块信用度.

定义6. 动态的块信用度.给定一个块b,该块的动态的信用度与该块当前的匹配的集合大小成正相关,与该块的势成负相关,

$$\sigma_d(b)=(|\Xi^+(b)|+1)/(\|b\|+1) \tag{3}$$

与静态的块信用度相似,动态的块信用度给予小块较大的初始信用度.初始时,所有候选对都还未被识别,$\sigma_d(b)=1/(\|b\|+1)$,与静态的块信用度非常相近.随着实体识别的进行,如果一个块内被识别出一些重复对,那么该块的动态的块信用度会变大,这样就克服了静态的块信用度的缺点.动态调整过程中,动态的块信用度同样偏好小块.分析式(3),分母是平方级的,而分子是一次方级的,因此,大块需要获得大量的重复对才能拥有较大的动态的块信用度,而小块只需要相对较少量的重复对就可以拥有相同的动态的块信用度.

4.3 候选对的信用度

本小节将介绍如何利用共现块的信用度来估计候选对的匹配可能性.有交叠的块集合中的分块冗余可以用来估计候选对的匹配可能性.简单来讲,如果一个候选对共同出现在越多的块中,那么该候选对匹配的可能性越大.然而,不同的共现块给相同的候选对提供的匹配可能性通常是不同的.定义候选对的信用度来估计候选对的相似性,即匹配的可能性.

定义7. 候选对的信用度.给定一个候选对$\langle r_i,r_j\rangle$,它的信用度估计该候选对的匹配可能性.候选对的信用度是将该候选对的共现块提供的匹配可能性进行聚集,并利用键总数进行归约,

$$\sigma(\langle r_i,r_j\rangle)=\Big(\sum_{r_i,r_j\in b,\ b\in B}\sigma(b)\Big)\Big/|K| \tag{4}$$

特别地,如果某些数据对象在一个键对应的属性值缺失,那么这些数据对象在本次分块中将不被分到任何一个块中.因此根据式(4),包含这样的数据对象的候选对无法从按该键分块的结果集合中获得块信用度.

5 基于多路分块的Pay-as-you-go实体识别

本节详述基于多路分块的Pay-as-you-go实体识别方法BPER.首先,5.1小节提出一个两阶段的Pay-as-you-go实体识别模型;然后,5.2小节介绍初始化阶段;最后,5.3小节介绍迭代阶段.

5.1 Pay-as-you-go实体识别模型

如图4所示,整个Pay-as-you-go实体识别过程划分为两个连续的阶段:初始化阶段和迭代阶段.

定义8. 初始化阶段.初始化阶段依次包括多路分块、分块图构建、候选对信用度初始化和候选对

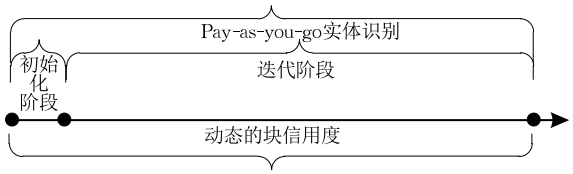


图 4　Pay-as-you-go 实体识别阶段划分

排序. 本阶段的输入是一个脏数据集，输出是候选队列 Q，该队列是一个优先队列，包含了所有候选对，并按候选对信用度排序.

定义 9.　迭代阶段. 迭代阶段包括三个迭代的步骤：候选对比较、候选对信用度更新和候选队列调整. 本阶段的输入是候选队列 Q，逐渐地输出识别结果.

Pay-as-you-go 实体识别的关键是，在实体识别过程中找出最可能匹配的候选对，因此需要将候选对根据其信用度进行非升序排序. 在初始化阶段，BPER 方法初始地对候选对进行排序；在迭代阶段，根据实时的实体识别结果，动态地调整候选对顺序. 其中，候选对的信用度基于块信用度. 由第 4 节可知，动态的块信用度随着实体识别的进行，动态地更新块信用度，这使得它比静态的块信用度可以更精确地估计块冗余度. BPER 方法将默认地采用动态的块信用度来计算候选对的信用度.

5.2　初始化阶段

作为一个预备步骤，初始化阶段为迭代阶段提供初始排序的候选对. 初始化阶段依次包括多路分块、分块图构建、候选对的信用度初始化和候选对排序.

5.2.1　分块图

因为多路分块会产生大量的分块冗余，所以提出一个高效的、基于分块图的分块冗余去除方法. 这里的分块冗余是指分块集合中多个块有交叠，而不是单一个块中的重复数据对象.

定义 10.　分块图. 给定一个多路分块的结果集合 B_m，存在一个带权、无向图 $G=(V,E)$，称为分块图. V 是结点集合，任意结点 $v\in V$ 对应 B_m 中的一个数据对象. E 是边集合，对于任意边 $e(v_i,v_j)\in E$（记作 e_{ij}），数据对象 v_i,v_j 至少共同出现在 B_m 中一个块内. 边 e_{ij} 的权重 w_{ij} 是候选对 $\langle v_i,v_j\rangle$ 的信用度. 本文中 r 和 v 都可以表示数据对象，R 和 V 都可以表示数据集.

算法 1 描述了如何构建分块图. 外层循环遍历给定的多路分块的结果集合（行 5），在内层循环遍历每个块（行 6），并作如下处理（行 7 到 15）. 如果一个候选对 $\langle v_i,v_j\rangle$ 对应的边 e_{ij} 不存在，生成新边 e_{ij}，并将 e_{ij} 加入到边的集合 E（行 7、14）；如果结点集合 V 不包含数据对象 v_i 或 v_j，则将未包含的数据对象加入到结点集合中（行 8 到 13）. 最后，得到一个分块图 $G=(V,E)$.

算法 1.　BlockingGraphConstruction.

```
1. 输入：多路分块集合 B_m
2. 输出：分块图 G=(V,E)
3.  V={}; //结点集合
4.  E={}; //边集合;
5.  For b∈B_m
6.     For ⟨v_i,v_j⟩∈Γ(b)
7.        If e_ij !∈E
8.           If v_i !∈V
9.              V=V∪{v_i};
10.          Endif
11.          If v_j !∈V
12.             V=V∪{v_j};
13.          Endif
14.          E=E∪{e_ij};
15.       Endif
16.    Endfor
17. Endfor
18. 输出分块图 G;
```

算法 1 复杂度分析：将未消除冗余的多路分块集合 B_m 的候选对集合表示为表示为 BP_{mR}，那么算法 1 的复杂度为 $O(|BP_{mR}|)$.

通过遍历分块图的边生成候选对集合，每条边对应的两个数据对象对应一个唯一的候选对. 根据 4.3 节中的定义 7 计算候选对的信用度，然后根据信用度来给分块图赋权重. 通过构建分块图，可以快速去除掉多路分块带来的分块冗余. 接下来，通过例 2 来举例说明如何进行多路分块以及如何利用分块图来去除分块冗余.

例 2.　续例 1. 对于表 1 中的脏数据集，分别把姓名、年龄、工作和城市作为键进行多路分块，得到如下结果集合，

$$B_m=B_{\text{surname}}\cup B_{\text{age}}\cup B_{\text{job}}\cup B_{\text{city}},$$

$$B_{\text{surname}}=\{b_{s1}=\{r_1,r_3,r_4\},b_{s2}=\{r_2\},$$
$$b_{s3}=\{r_5\},b_{s4}=\{r_6\},b_{s5}=\{r_7\}\}$$

$$B_{\text{age}}=\{b_{a1}=\{r_1,r_2,r_4,r_6,r_7\},b_{a2}=\{r_5\}\},$$

$$B_{\text{job}}=\{b_{j1}=\{r_1,r_2,r_3,r_4,r_5\},b_{j2}=\{r_7\}\},$$

$$B_{\text{city}}=\{b_{c1}=\{r_2,r_3,r_4,r_6,r_7\},$$
$$b_{c2}=\{r_1\},b_{c3}=\{r_5\}\}.$$

上述块集合 B_m 中共有 33 个候选对，存在冗余.

比如，候选对$\langle r_1, r_4\rangle$同时出现在块b_{s1}，b_{a1}和b_{j1}中.利用算法1来构建分块图，从而去除B_m的分块冗余.如图5所示，得到B_m对应的分块图G_B，图G_B中的每条边对应一个唯一的候选对.由图5可知，去除掉分块冗余后，候选对数目从33降至19.

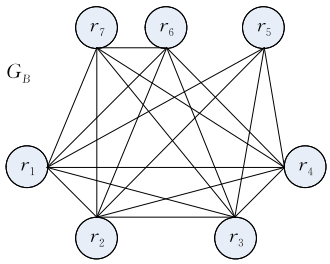


图5 例2中的块集合对应的分块图G_B

5.2.2 初始化阶段

算法2描述了初始化阶段.首先，利用一组键K对给定的脏数据集V进行多路分块，生成一个带冗余的块集合B_m(行3).然后，调用算法1构建B_m对应的分块图，以此来去除B_m的分块冗余(行4)；计算B_m中块的信用度，调用定义6中式(3)来计算每个块b的DBC，此时重复对的集合规模$|\Xi^{-}(b)|$为0(行5～7).接着，调用定义7中式(4)，利用块信用度初始化候选对$\langle v_i, v_j\rangle$的信用度，进而给分块图中对应边e_{ij}赋权重w_{ij}(行8～10).最后，将所有候选对加入到候选队列Q，并按照信用度非升序排列(行11).候选队列Q是一个优先队列，通过堆(Heap)来实现，它的插入开销为$O(\log n)$，初始构建(包含排序)的开销为$O(n\log n)$.例3将说明，如何初始化块信用度和候选对的信用度，以及如何对候选对排序.

算法2. Initialization.

1. 输入：脏数据集V，一组分块键K
2. 输出：候选队列Q
3. 利用V和K构建一个带冗余的块集合B_m；
4. G=BlockingGraphConstruction(B_m)；
   //调用算法1构建分块图$G=(V,E)$
5. For $b\in B_m$
6. $\sigma_d(b)$；//初始化块b的信用度
7. Endfor
8. For $e_{ij}\in E$ //e_{ij}对应候选对$\langle v_i, v_j\rangle$
9. $w_{ij}=\sigma(\langle v_i, v_j\rangle)$；
10. Endfor
11. 根据初始的候选对的信用度，将候选对加入到一个候选队列Q；
12. 输出候选队列Q；

算法2复杂度分析：构建块集合B_m的开销为$O(K\times|V|)$；用$|B_m|$表示B_m中块的数目，初始化块信用度的开销为$O(|B_m|)$；将消除冗余后的多路分块集合B_m的候选对集合表示为BP_{mC}，实际就是分块图的边的集合E，那么初始化候选对的信用度的开销为$O(|E|\times K\times\log|B_m|)$；将所有候选对在候选队列$Q$中按初始信用度排序的开销为$O(|E|\times\log|E|)$；那么算法2的复杂度为$O(K\times|V|+|B_m|+|BP_{mR}|+|E|\times(K\times\log|B_m|+\log|E|))$.

例3. 续例2.利用算法2处理表1中的脏数据集，得到初始的块信用度和候选对的信用度，如下，

块信用度：$\sigma_d(b_{s1})=1/4$，$\sigma_d(b_{a1})=1/11$，$\sigma_d(b_{j1})=1/11$，$\sigma_d(b_{c1})=1/11$.

候选对的信用度(非升序)：$\sigma_d(\langle r_1, r_4\rangle)=0.108$，$\sigma_d(\langle r_3, r_4\rangle)=0.108$，$\sigma_d(\langle r_1, r_3\rangle)=0.085$，$\sigma_d(\langle r_2, r_4\rangle)=0.068$，$\sigma_d(\langle r_6, r_7\rangle)=0.045$，$\sigma_d(\langle r_2, r_3\rangle)=0.045$，$\sigma_d(\langle r_1, r_2\rangle)=0.045$，…

根据上述候选对的信用度，后续应该先处理$\langle r_1, r_4\rangle$或$\langle r_3, r_4\rangle$.根据真实识别结果$\{\{r_1, r_2, r_3, r_4\}, \{r_5\}, \{r_6\}, \{r_7\}\}$可知，这两个候选对都是重复的.由此可见，初始的候选对的排序是有效的.

5.3 迭代阶段

迭代阶段是BPER方法的核心部分.迭代阶段主要由三个迭代步骤组成：候选对比较、部分候选对的信用度更新和候选队列调整；在此过程中，将逐渐地输出最新被识别出的重复对.将部分候选对的信用度更新和候选队列调整合称为更新期.针对更新期，提出两个更新策略：朴素的更新策略和延迟的更新策略.另外，还将利用数据对象匹配的传递性来提高渐近性.

5.3.1 迭代阶段

算法3描述了迭代阶段.算法3在更新期采用了朴素的更新策略(行5、6)，也可以采用5.3.2小节中提出的延迟的更新策略.从候选队列Q队首取一个候选对$\langle v_i, v_j\rangle$，用实体识别匹配函数对$\langle v_i, v_j\rangle$进行比较(行4、5).如果$\langle v_i, v_j\rangle$被判定为重复的，那么执行如下操作：首先，执行更新操作(行6)，根据当前识别结果，更新块信用度、候选对信用度以及调整候选队列顺序，详情见算法4；其次，如果已经存在重复对象对$\langle v_i, v_k\rangle$(或$\langle v_k, v_j\rangle$)，那么Look-around函数将直接处理$\langle v_k, v_j\rangle$(或$\langle v_i, v_k\rangle$)，从而提高渐近性(行7)，此函数利用了数据对象匹配的传递性，将在5.3.3小节中介绍.特别地，算法3在调用Look-around函数(算法6)时，需要对算法6进行微小修改，即在算法6

第 8、9 行之间调用算法 4，执行更新操作. 重复上述过程(行 4～9)，直到时间预算用完或候选队列 Q 为空. 在此过程中，实体识别结果被逐渐地输出. 例 4 将说明 BPER 方法如何迭代地、动态地选择最可能匹配的候选对来进行识别.

算法 3.　Iteration.

1. 输入：分块集合 B_m，候选队列 Q，实体识别时间预算 BGT
2. 输出：Pay-as-you-go 实体识别结果
3. Repeat
4. 　从 Q 队首取一个候选对$\langle v_i, v_j\rangle$；
5. 　If $m(\langle v_i, v_j\rangle)==$true //$m(*,*)$为实体识别匹配函数
6. 　　Update(B_m, Q, $\{\langle v_i, v_j\rangle\}$)；//调用算法 4
7. 　　Look-around($\langle v_i, v_j\rangle$)；//调用算法 6
8. 　　输出本轮识别结果；
9. 　Endif
10. Until (执行时间$>BGT$) or (候选队列 Q 为空)

算法 3 复杂度分析：此处分析一轮迭代的复杂度. 一次实体识别匹配函数调用的开销记作 $O(C_m)$；Look-around 函数的开销为 $O(La)$，后文 5.3.3 节将给出具体分析；更新操作的开销为 $O(C_{\text{update}})$，将在算法 4 的介绍中给出具体分析；那么一次迭代的复杂度为 $O(C_m+La+C_{\text{update}})$.

算法 4 描述了更新操作. 根据最新识别的重复对的集合 S，找出最新识别出的重复对所在的块，称为受影响的块 b_{af}，受影响的块的集合记作 $B_{\text{af}}=\{b_{\text{af}}\}$(行 3～7). 由于这些受影响的块中被识别为重复数据对象的比例提高了，根据 S 利用 DBC 定义来更新这些受影响的块的信用度(行 8～10). 这些受影响的块所包含的未识别的候选对称为受影响的候选对，利用新的块信用度来更新这些受影响的候选对的信用度，B_{af} 中未识别的候选对的集合记作 BP_{afu}(行 11～15). 根据新的候选对的信用度来调整候选队列 Q(行 16).

算法 4.　Update.

1. 输入：分块集合 B_m，候选队列 Q，最新识别的重复对的集合 S
2. 输出：更新的分块集合 B_m，更新的候选队列 Q
3. $B_{\text{af}}=\{\}$；
4. For $\langle v_i, v_j\rangle\in S$
5. 　$B_{ij}=$findAffectedBlocks($\langle v_i, v_j\rangle$)；
6. 　$B_{\text{af}}=B_{\text{af}}\cup B_{ij}$；
7. Endfor
8. For $b_{\text{af}}\in B_{\text{af}}$
9. 　$\sigma_d(b_{\text{af}})$；//更新块 b_{af} 的信用度
10. Endfor
11. For $b_{\text{af}}\in B_{\text{af}}$
12. 　For $\langle v_i, v_j\rangle\in(\Gamma(b_{\text{af}})-\Xi(b_{\text{af}}))$
13. 　　$w_{ij}=\sigma(\langle v_i, v_j\rangle)$；//更新候选对$\langle v_i, v_j\rangle$的信用度
14. 　Endfor
15. Endfor
16. 根据上述更新结果调整候选队列 Q；

算法 4 复杂度分析：查找受影响块的集合 B_{af} 的开销为 $O(|S|\times K\times\log|B_m|)$；块信用度更新的开销为 $O(|B_{\text{af}}|)$；每个受影响的候选对对应的更新块的数目平均值为 $K_{\text{af}}\leq K$，那么未识别的候选对的信用度更新的开销为 $O(|BP_{\text{afu}}|\times K_{\text{af}}\times\log|B_m|)$；当前候选队列的规模为 E_{now}，候选队列调整的开销为 $O(|BP_{\text{afu}}|\times\log E_{\text{now}})$；那么算法 4 的复杂度为 $O(|S|\times K\times\log|B_m|+|B_{\text{af}}|+|BP_{\text{afu}}|\times(K_{\text{af}}\times\log|B_m|+\log E_{\text{now}}))$.

例 4.　续例 3. 利用算法 3 处理从例 3 得到的候选队列. 为了更直观地说明动态的块信用度对渐近性的作用，本例将略去算法 3 中的 Look-around 函数(行 7). 逐轮观察 BPER 的迭代阶段. 表 2 呈现了算法 3 处理表 1 中脏数据集的前 6 轮迭代. 根据定义 7 中式(4)和表 2，第 2～6 轮迭代中，候选对按信用度非升序排列如下(每一轮队首的候选对被处理，将不出现在下一轮)：

第 2 轮，$\boldsymbol{\sigma_d(\langle r_3, r_4\rangle)=0.193}$，$\sigma_d(\langle r_1, r_3\rangle)=0.170$，$\sigma_d(\langle r_2, r_4\rangle)=0.114$，$\sigma_d(\langle r_1, r_2\rangle)=0.091$，$\sigma_d(\langle r_2, r_3\rangle)=0.068$，$\sigma_d(\langle r_6, r_7\rangle)=0.068$，$\sigma_d(\langle r_2, r_7\rangle)=0.068$，…

第 3 轮，$\boldsymbol{\sigma_d(\langle r_1, r_3\rangle)=0.256}$，$\sigma_d(\langle r_2, r_4\rangle)=0.159$，$\sigma_d(\langle r_1, r_2\rangle)=0.114$，$\sigma_d(\langle r_2, r_3\rangle)=0.114$，$\sigma_d(\langle r_6, r_7\rangle)=0.091$，$\sigma_d(\langle r_2, r_7\rangle)=0.091$，$\sigma_d(\langle r_4, r_7\rangle)=0.091$，…

第 4 轮，$\boldsymbol{\sigma_d(\langle r_2, r_4\rangle)=0.182}$，$\sigma_d(\langle r_1, r_2\rangle)=0.136$，$\sigma_d(\langle r_2, r_3\rangle)=0.136$，$\sigma_d(\langle r_6, r_7\rangle)=0.091$，$\sigma_d(\langle r_2, r_7\rangle)=0.091$，$\sigma_d(\langle r_4, r_7\rangle)=0.091$，…

第 5 轮，$\boldsymbol{\sigma_d(\langle r_1, r_2\rangle)=0.182}$，$\sigma_d(\langle r_2, r_3\rangle)=0.182$，$\sigma_d(\langle r_6, r_7\rangle)=0.136$，$\sigma_d(\langle r_2, r_7\rangle)=0.136$，$\sigma_d(\langle r_4, r_7\rangle)=0.136$，…

第 6 轮，$\boldsymbol{\sigma_d(\langle r_2, r_3\rangle)=0.227}$，$\sigma_d(\langle r_6, r_7\rangle)=0.159$，$\sigma_d(\langle r_2, r_7\rangle)=0.159$，$\sigma_d(\langle r_4, r_7\rangle)=0.159$，…

根据真实识别结果$\{\{r_1, r_2, r_3, r_4\}, \{r_5\}, \{r_6\}, \{r_7\}\}$，前 6 轮中每轮都识别出一个重复对. 因此，假如实体识别预算设定为 6 次数据对象对比较的话，本文提出的 BPER 方法可以在预算范围内识别出

所有的重复对，说明 BPER 方法的渐近性非常高. 接下来，深入分析算法执行细节. 前两个被识别的候选对$\langle r_1, r_4\rangle$和$\langle r_3, r_4\rangle$获得高优先级，因为(1)它们有三个共现块，不比其它任何候选对少；(2)块b_{s1}初始时具有最大的块信用度. 此外，一些候选对是在动态更新过程中获得了高优先级. 比如，$\langle r_6, r_7\rangle$不是重复的，而$\langle r_2, r_3\rangle$是重复的. 初始时，$\sigma_d(\langle r_6, r_7\rangle)=0.045$，$\sigma_d(\langle r_2, r_3\rangle)=0.045$，两者具有相同的优先级；然而，到第 5 轮迭代时，$\sigma_d(\langle r_6, r_7\rangle)=0.136$，$\sigma_d(\langle r_2, r_3\rangle)=0.182$，因此，$\langle r_2, r_3\rangle$获得更高的优先级，将先被识别. 尽管$\langle r_2, r_3\rangle$和$\langle r_6, r_7\rangle$都有两个大小为 5 的共现块(分别为块$b_{j1}$和$b_{c1}$，块$b_{a1}$和$b_{c1}$)，但是第 5 轮迭代时，块$b_{j1}$的信用度比$b_{a1}$和$b_{c1}$的信用度要大. 块$b_{j1}$中有 4 个被识别出的重复对，而$b_{a1}$和$b_{c1}$中都只有 2 个被识别出的重复对. 总而言之，本文提出的 BPER 方法可以在给定较小的时间预算内，尽量多地识别出重复对，具有高渐近性，可以高效地解决 Pay-as-you-go 实体识别任务.

表 2 算法 3 处理表 1 中脏数据集过程中，块信用度和候选队列队首的候选对

迭代轮数	$\sigma_d(b_{s1})$	$\sigma_d(b_{a1})$	$\sigma_d(b_{j1})$	$\sigma_d(b_{c1})$	队首候选对
1	1/4	1/11	1/11	1/11	$\langle r_1, r_4\rangle$
2	2/4	2/11	2/11	1/11	$\langle r_3, r_4\rangle$
3	3/4	2/11	3/11	2/11	$\langle r_1, r_3\rangle$
4	1	2/11	4/11	2/11	$\langle r_2, r_4\rangle$
5	1	3/11	5/11	3/11	$\langle r_2, r_3\rangle$
6	1	3/11	6/11	4/11	$\langle r_1, r_2\rangle$
…	…	…	…	…	…

5.3.2 延迟的更新策略

如算法 3 中 5～6 行所示，在更新期，朴素的更新策略每当识别出一个重复对，就要执行一次更新操作. 更新操作涉及三部分：受影响的块信用度更新、受影响的候选对的信用度更新和候选队列调整. 一方面，更新操作可以提高一些候选对的优先级，从而帮助提高渐近性；另一方面，频繁的更新操作会带来较大的开销，导致渐近性降低. 由此可见，合理的更新策略对于渐近性非常重要. 显然，朴素的更新策略频繁地执行更新操作，因此是低效的.

为了提高渐近性，提出一个延迟的更新策略. 根据式(3)和(4)，只有识别出新的重复对时，才有必要执行更新操作. 当连续地识别出重复对时，不需要停下来执行更新操作，因为此时的渐近性是最高的；当连续地识别出不重复数据对象对时，没必要停下来执行更新操作，因为不影响块信用度. 因此，更新操作只需要发生在连续地识别出重复对之后. 算法 5 描述了基于延迟的更新操作的迭代阶段. 算法 5 与算法 3 的不同之处在于，更新操作只发生在连续地识别出重复对之后、遇到第一个不重复数据对象对的时候(行 7～20). 算法 5 的复杂度分析与算法 3 类似，但更新操作次数更少，因此开销更小.

算法 5. IterationByLazyUpdate.

```
1.  输入:分块集合 B_m,候选队列 Q,实体识别时间预
        算 BGT
2.  输出:Pay-as-you-go 实体识别结果
3.  counter=0;
4.  S={}; //最新识别出的重复对的集合
5.  Repeat
6.    从 Q 队首取一个候选对⟨v_i,v_j⟩;
7.    If m(⟨v_i,v_j⟩)==true
8.      S=S∪{⟨v_i,v_j⟩};
9.      counter++;
10.     R=Look-around(⟨v_i,v_j⟩);
11.     S=S∪R;
12.     counter+=|R|;
13.     输出本轮识别结果;
14.   Else
15.     If (counter>0)
16.       update(B_m,Q,S);
17.       counter=0;
18.       S={};
19.     Endif
20.   Endif
21. Until(执行时间>BGT) or(候选队列 Q 为空)
```

如图 6 所示，对于队列Q_1，当连续地将p_1，p_2和p_3识别为重复对后，识别出一个不重复数据对象对p_4；然后，利用新识别出的重复对p_1，p_2和p_3执行更新操作；最终，相比于朴素的更新策略，节省了 2 次更新操作. 对于队列Q_2，连续地将p_5，p_6和p_7识别为不重复数据对象对后，连续地识别出两个重复对p_8和p_9，直到识别出不重复数据对象对p_{10}，才执行更新操作；最后，相比于朴素的更新策略，节省了 1 次更新操作.

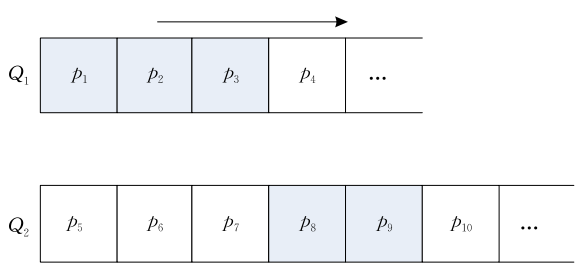


图 6 两个样例候选队列(阴影的方块表示重复对，白色的方块表示不重复数据对象对)

5.3.3　传递性

作为一个理论性质，传递性对于实体识别非常重要[8-9,12,24]. 尽管传递性在工程实践中不总是有效的，但它对于提高实体识别的渐近性非常有用. BPER 方法将传递性实现为一个 Look-around 函数，包括两种方式：(1) 传递闭包(Transitivity Closure，TC). 当识别出重复对$\langle v_i, v_j\rangle$和$\langle v_j, v_k\rangle$时，那么直接将$\langle v_i, v_k\rangle$判定为重复对，节省了一次数据对象对比较；(2) 传递预测(Transitivity Prediction，TP). 当识别出重复对$\langle v_i, v_j\rangle$和$\langle v_j, v_k\rangle$时，那么接下来直接比较$\langle v_i, v_k\rangle$，而不是候选队列的下一个队首数据对象对. 第一种实现方式可以极大地提高实体识别的渐近性，但可能损失一定的精确性，因为这种方式略过了数据对象对比较而直接判定为重复. 第二种实现方式对渐近性的提高要低于第一种实现方式，但不会损失精确性，因为第二种实现方式只是根据传递性优化了识别顺序，并没有略过数据对象对比较. 算法 6 描述了第二种实现方式. 给定一个数据对象v_i，v_i的匹配集包含v_i和所有已识别出与v_i重复(匹配)的数据对象，记作$ms(v_i)$. 至于第一种实现方式，只需在算法 6 的第 7 行直接将$\langle v_k, v_l\rangle$判定为重复.

算法 6.　Look-around.

```
1.  输入：一个最新识别出的重复对⟨v_i, v_j⟩，匹配集合
        ms(v_i)和 ms(v_j)
2.  输出：新的重复对集合 R
3.  R={}; //新的重复对集合
4.  For v_k∈ms(v_i)
5.    For v_l∈ms(v_j)
6.      If ⟨v_k, v_l⟩!=⟨v_i, v_j⟩
7.        If m(v_k, v_l)==true
8.          R=R∪{⟨v_k, v_l⟩};
9.        Endif
10.     Endif
11.   Endfor
12. Endfor
13. 输出新的重复对集合 R;
```

算法 6 复杂度分析：Look-around 的复杂度为$O(|ms(v_i)|\times|ms(v_j)|)$.

6　实验评价

6.1 节介绍实验设置，包括实验环境、数据集、评价指标和方法设置. 6.2 节与已有工作进行对比，分别在两个真实的数据集上和一组合成的数据集上进行测试；6.3 节对 BPER 的关键组成部分进行测试，包括更新期测试、块信用度测试和传递性测试.

6.1　实验设置

6.1.1　实验环境

本文所有实验通过 Java 实现. 处理器为 Intel(R) Core(TM) i7-2600、主频 3.4GHz、8 核；内存为 8GB；操作系统为 Microsoft Windows 7 Ultimate，64 位；运行环境为 Java 1.7.

6.1.2　数据集

本节将采用两个真实的数据集和一组合成的数据集来评价本文提出的 BPER 方法. (1) CDDB 数据集[12-13]. 该数据集包括 9763 条音乐 CD 记录，抽自 freeDB① 网站，其中有 299 个重复记录对. CD 记录的属性有艺术家、标题、分类、流派、额外信息、年份和音轨；(2) Citeseer 数据集. 该数据集包括 90 000 条引文记录，由作者抽取自 Citeseer② 网站. 通过对该数据集运行蛮力(brute force)的实体识别方法，得到 7631 个重复记录对作为真实的识别结果. 引文记录的属性有标题、作者、地址、会议和日期；(3) UIS 合成数据集. 利用 UIS 数据生成器[14,25]生成具有特定数据特征的个人信息数据集. 合成数据集的详细信息将在 6.2.2 小节中详细介绍. 个人信息记录的属性包括社保号、姓名、地址、城市、州和邮编.

6.1.3　评价指标

与文献[12]类似，本文的研究目标是在准确性(用来衡量方法的 effectiveness)不变的情况下，提高识别速度(用来衡量方法的 efficiency). 在本文的实验评价中，匹配函数$m(\langle r_i, r_j\rangle)$作为黑盒使用，除 6.3.3 节外，$m(\langle r_i, r_j\rangle)$直接查询正确的匹配结果(Ground Truth，GT)，如果 GT 中包含了$\langle r_i, r_j\rangle$，则返回真，否则返回假，即精确性为 1. 在 6.3.3 节，给定$\langle r_i, r_j\rangle$，$m(\langle r_i, r_j\rangle)$以 0.94 的概率返回正确的匹配结果.

定义 11.　渐近性质量. 给定脏数据集 R 中有 N 个重复对，权重函数 $w(t)$是关于时间 t 的函数，周期性识别结果函数 $r(t)$返回的是时间段$(t-1, t]$内识别出的重复对的数目，T 为评价时间，那么实体识别方法的渐近性质量是[12]：

$$Q(T)=1/N\times\sum_{t=1}^{T}(w(t)\times r(t)) \tag{5}$$

① http://www.freedb.org/
② http://citeseerx.ist.psu.edu/

渐近性质量的值域为[0,1].周期性识别结果函数 $r(t)$本质上是抽样.给定 n 个实体识别方法,它们在相同的运行环境和同一数据集上,运行时间为 T_i $(0\leq i\leq n)$,那么评价时间为 $T=\min\{T_i|1\leq I\leq n\}$.权重函数 $w(t)$是单调递减函数.本实验采用经济权重函数[12],如式(6)所示,其中 BGT 是实体识别时间预算.初始时,$w(1)=1$,此时权重最大;随着时间推移,权重将逐渐降低;最终,当时间预算用完后,权重变为 0,不再变化.可以发现,权重函数会惩罚后来被识别出的重复对,这与 Pay-as-you-go 实体识别的定义是一致的.

$$w(t)=\max(1-(t-1)/BGT,0) \tag{6}$$

为了帮助读者更好地解读实验结果,后文将通过曲线描绘实时的实体识别结果的召回率.

6.1.4 方法设置

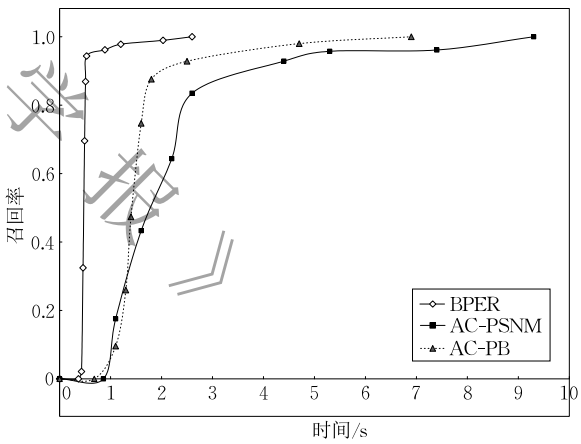
对于 CDDB 数据集,选择艺术家、标题、年份和第一音轨作为多路分块的键.对于 Citeseer 数据集,选择标题、第一作者、会议和年份作为多路分块的键.对于合成的个人信息数据集,选择社保号、姓名、城市和邮编作为多路分块的键.除了 6.3.2 小节,都采用动态的块信用度;6.3.2 小节还采用了静态的块信用度.除了 6.3.3 小节,都采用完美的实体识别匹配函数(精确性为 1);6.3.3 小节采用不完美的匹配函数(精确性为 0.94),用来测试传递性对渐近性的影响.除了 6.3.3 小节,Look-around 函数都采用传递闭包(TC)的实现方式;6.3.3 小节还采用了传递预测(TP)的实现方式.除了 6.3.1 小节,在更新期都采用延迟的更新策略;6.3.1 小节还采用了朴素的更新策略.

6.2 综合测试

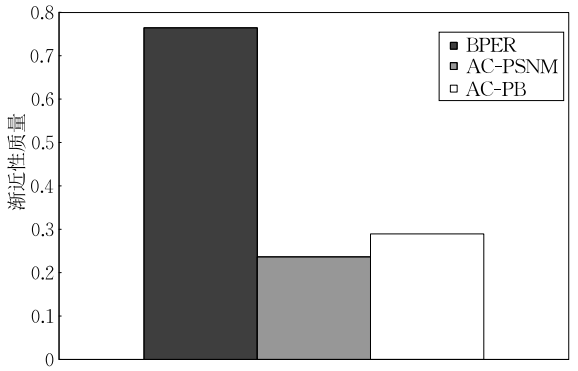
Papenbrock 等人基于排序的数据对象列表,提出一组 Pay-as-you-go 实体识别方法[12].渐近式滑动窗口方法(Progressive Sorted Neighborhood Method,PSNM)扩展了传统的滑动窗口方法(Sorted Neighborhood Method,SNM)[14].渐近式分块方法(Progressive Blocking,PB)基于排序的数据对象列表和等距分块.为了降低最优键选择的难度,Papenbrock 等人提出两个基于多路技术的扩展方法:属性同步(Attribute Concurrency)的 PSNM 方法(AC-PSNM)和属性同步的 PB 方法(AC-PB).在本文的设定中,最优的分块或排序的键未知,因此将本文提出的 BPER 方法与 AC-PSNM 和 AC-PB 方法进行比较.Papenbrock 等人的方法是基于硬盘的,BPER 是基于内存的.在实验对比时,AC-PSNM 和 AC-PB 方法将都在没有内存限制的条件下运行.AC-PSNM 方法的窗口设为 20,窗口区间参数设为 1;AC-PB 方法的块大小设为 5.

6.2.1 基于真实数据集的对比测试

图 7 呈现了三个方法在 CDDB 数据集上的对比.图 7(a)是实时的召回率,图 7(b)是渐近性质量.如图 7(a)所示,BPER 方法识别的速度要比 AC-PSNM 和 AC-PB 方法都要快很多;此外,AC-PB 的识别速度比 AC-PSNM 快.比如,BPER 经过 0.53 s 识别出 94.36%的重复对,而 AC-PSNM 经过 4.4 s 识别出 92.83%,AC-PB 经过 2.5 s 识别出 93.52%.如图 7(b)所示,AC-PSNM 的渐近性质量是 BPER 的 30.94%,AC-PB 的渐近性质量是 BPER 的 37.86%.总之,在 CDDB 数据集上,BPER 的渐近性明显优于 AC-PSNM和 AC-PB 方法.AC-PSNM 和 AC-PB 方法要交替地处理 $|K|$(多路排序的键的总数)个排序队列,存在大量冗余,因此,渐近性受影响.对于 CDDB 数据集,BPER 总共只需要 2.6 s 完成识别,而 AC-PSNM 和 AC-PB 分别需要 9.3 s 和 6.9 s.



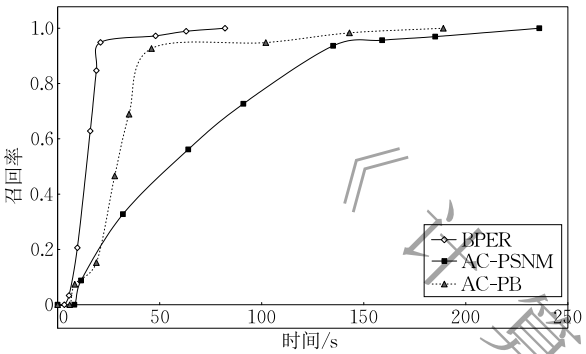
(a) 实时召回率



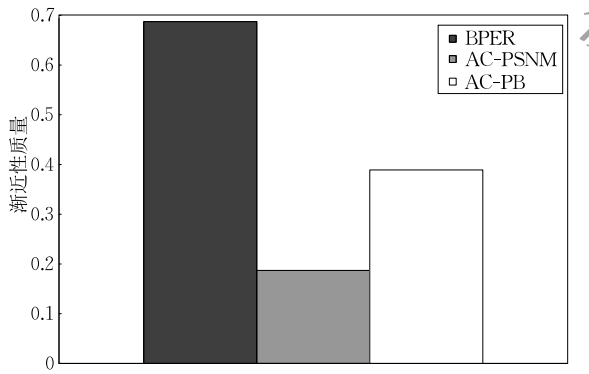
(b) 渐近性质量

图 7 在 CDDB 数据集上与已有工作对比

图 8 呈现了三个方法在 Citeseer 数据集上的对比，实验结果的对比情况与在 CDDB 数据集上相似. 如图 8(a)所示，BPER 方法识别的速度要比 AC-PSNM 和 AC-PB 方法都要快许多. 比如，BPER 用 21 s 识别出 94.87%的重复对，而 AC-PSNM 经过 135 s 识别出 93.63%，AC-PB 经过 46 s 识别出 92.66%. 根据图 8(b)可知，AC-PSNM 的渐近性质量是 BPER 的 27.19%，AC-PB 的渐近性质量是 BPER 的 57.36%. 综上所述，如同在 CDDB 数据集上，在 Citeseer 数据集上，BPER 的渐近性同样显著优于 AC-PSNM 和 AC-PB 方法.



(a) 实时召回率



(b) 渐近性质量

图 8　在 Citeseer 数据集上与已有工作对比

6.2.2　基于合成数据集的对比测试

本小节将用合成数据集来测试三种 Pay-as-you-go 实体识别方法在不同数据重复度分布下的表现. 生成两个个人信息数据集，两者大小相同但包含的重复数据对象数目不同. 第一个合成数据集，有 50 000 条记录，包括 1000 条重复记录对，记作 PSN-low；第二个合成数据集，有 50 000 条记录，包括 5000 条重复记录对，记作 PSN-high. 如图 9 所示，在 PSN-low 数据集上，AC-PSNM 的渐近性质量是 BPER 的 30.47%，AC-PB 的渐近性质量是 BPER 的 54.87%；在 PSN-high 数据集上，AC-PSNM 的渐近性质量是 BPER 的 20.87%，AC-PB 的渐近性质量是 BPER 的 63.82%. 由此可见，AC-PSNM 在稀疏的数据集上的渐近性比在稠密的数据集上要高一些；AC-PB 在稠密的数据集上的渐近性比在稀疏的数据集上要高一些. 但是无论在稠密的数据集上还是稀疏的数据集上，BPER 的渐近性都要远高于已有的两个方法.

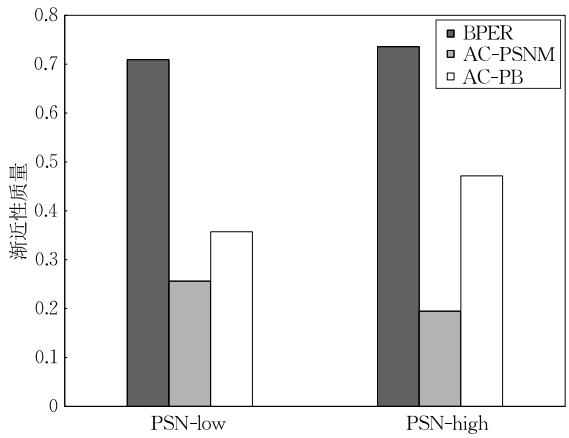


图 9　在合成数据集上与已有工作对比

6.3　关键组成部分测试

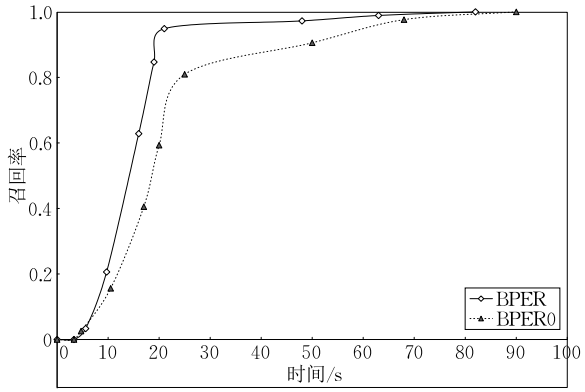
本小节将测试 BPER 方法的关键组成部分对渐近性的影响，将在 Citeseer 数据集上进行测试. 表 3 详细描述了本小节将用到的不同版本的 BPER 方法，SBC 表示静态的块信用度，DBC 表示动态的块信用度；NUS 表示朴素的更新策略，LUS 表示延迟的更新策略；TC 表示传递闭包，TP 表示传递预测，分别对应传递性的第一种和第二种实现方式；PM 表示完美的实体识别匹配函数(精确性为 1)，IPM 表示不完美的匹配函数(精确性小于 1).

表 3　关键组成部分测试用到的方法的详细情况

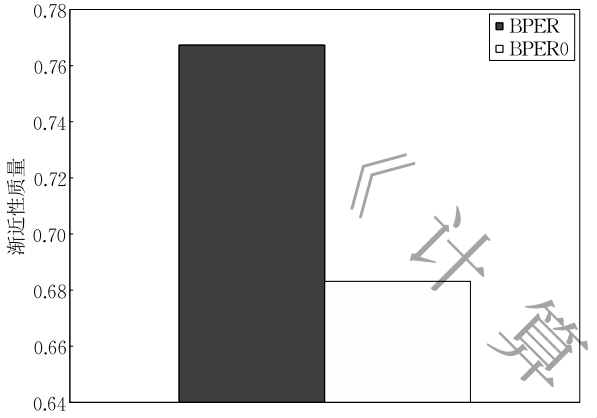
方法	块信用度	更新策略	传递性	匹配函数
BPER0	DBC	NUS	TC	PM
BPER1	SBC	LUS	TC	PM
BPER2	DBC	LUS	TP	IPM
BPER3	DBC	LUS	TC	IPM
BPER4	DBC	LUS	—	IPM
BPER	DBC	LUS	TC	PM

6.3.1　更新期测试

BPER0 采用朴素的更新策略，BPER 采用延迟的更新策略. 如图 10(a)所示，BPER 识别的速度要比 BPER0 快. 如图 10(b)所示，BPER 的渐近性质量是 76.73%，BPER0 的渐近性质量是 68.31%，BPER 的渐近性质量比 BPER0 高 12.33%. 分析原因，相比于朴素的更新策略，延迟的更新策略节省了一部分更新操作，从而提高了渐近性. 由此可见，延迟的更新策略是一个更好的更新策略.



(a) 实时召回率



(b) 渐近性质量

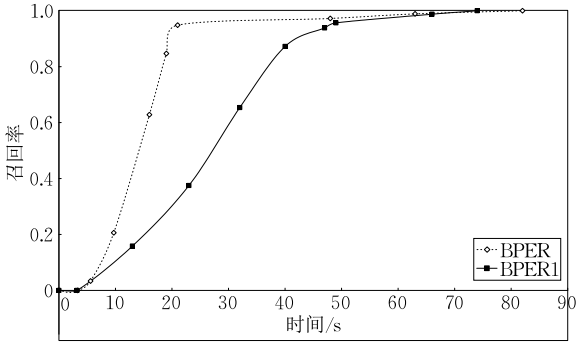
图 10 在 Citeseer 数据集上的更新期测试

6.3.2 块信用度测试

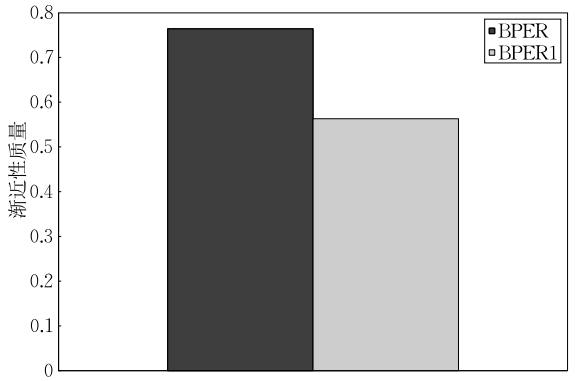
如表 3 所示,BPER1 方法采用静态的块信用度(SBC),BPER 方法采用动态的块信用度(DBC).根据图 11(a),BPER 识别的速度比 BPER1 明显快一些.如图 11(b)所示,BPER 的渐近性质量比 BPER1 高 26.31%.上述实验结果说明,基于动态的块信用度的方法(BPER)比基于静态的块信用度的方法(BPER1)的渐近性更高.静态的块信用度只是在实体识别进行前进行一次计算;而动态的块信用度根据实时的实体识别结果进行动态地调整,因此动态的块信用度能更准确地选择匹配可能性高的候选对,保证了高渐近性.

6.3.3 传递性测试

如表 3 所示,BPER2 采用传递预测(TP)现实传递性,BPER3 采用传递闭包(TC),而 BPER4 没有利用传递性.不同于之前的测试,本小节所有方法的实体识别匹配函数是一个不完美的匹配函数,其精确性为 94%.精确性是 F 指数,即准确率与召回率的调和平均数.由图 12(a)可知,方法 BPER2 和 BPER3 的识别速度都要比方法 BPER4 快,因为传递性可以帮助选择匹配可能性高的候选对.BPER3

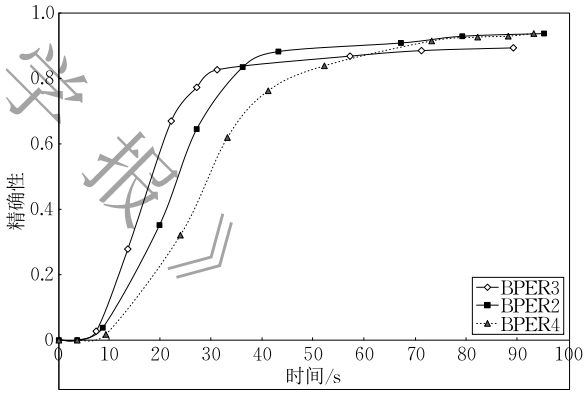


(a) 实时召回率

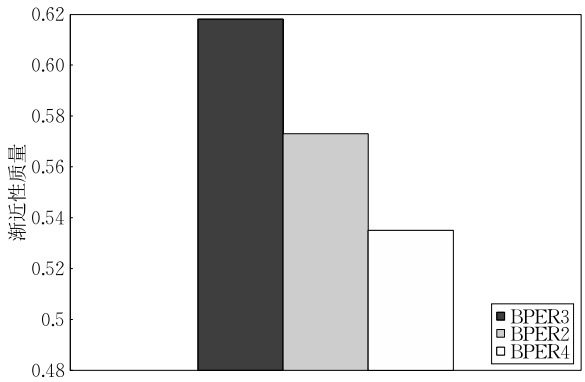


(b) 渐近性质量

图 11 在 Citeseer 数据集上的块信用度测试



(a) 实时召回率



(b) 渐近性质量

图 12 在 Citeseer 数据集上的传递性测试

的识别速度比 BPER2 快，因为：BPER3 以传递闭包的方式实现传递性，不需要进行数据对象比较，就直接判定数据对象对为匹配；BPER2 以传递预测的方式实现传递性，只是优化识别顺序，并没有减少数据对象比较. 从另一角度来看，BPER3 的最终精确性要低于 BPER2 和 BPER4，因为传递闭包会导致假真(false positive)的重复对出现. 根据图 12(b)，BPER3 的渐近性质量比 BPER2 高 7.85%，比 BPER4 高 15.51%；BPER2 的渐近性质量比 BPER4 高 7.10%. 综上所述，传递性可以提高实体识别的渐近性. 在工程实践中选择哪一种传递性实现方式，取决于用户更关注哪方面，精确性(传递预测)或者渐近性(传递闭包).

7　结束语

本文提出一个基于多路分块的 Pay-as-you-go 实体识别方法 BPER. BPER 方法不要求最优的排序或分块的键，因此普通用户可以很容易地使用. 其核心思想是，利用多路分块生成的分块冗余计算候选对的匹配可能性，并随着实体识别的进行，根据实时的识别结果，动态地调整候选对的匹配可能性. 因此，BPER 能够实时地选择最可能匹配的候选对进行识别，保证了高渐近性. 通过在真实数据集和合成数据集上的实验评估，证明本文提出的方法的渐近性显著地超过已有工作.

在未来的研究工作中，(1)将关注分块属性对数据对象相似度计算的影响，以及这种联系对实体识别效率和渐近性的影响；(2)将探索更加具有理论支撑和保证的块信用度和候选对信用度.

致　谢　非常感谢各位审稿人提出的质疑、意见和建议，使作者对本文的核心问题有了更深的认识和思考，帮助作者完善了本文！

参　考　文　献

[1] Naumann F, Herschel M. An introduction to duplicate detection. Synthesis Lectures on Data Management, 2010, 2(1): 1-87

[2] Sun Chen-Chen, Shen De-Rong, et al. A related data oriented joint entity resolution approach. Chinese Journal of Computers, 2015, 38(9): 1739-1754(in Chinese)
(孙琛琛，申德荣等. 面向关联数据的联合式实体识别方法. 计算机学报，2015，38(9)：1739-1754)

[3] Sun Chenchen, Shen Derong, et al. A genetic algorithm based entity resolution approach with active learning. Frontier of Computer Science, 2017, 11(1): 147-159

[4] Mudgal S, Li H, Rekatsinas T, et al. Deep learning for entity matching: A design space exploration//Proceedings of the ACM 2018 International Conference on Management of Data. Houston, USA, 2018: 19-34

[5] Galhotra S, Firmani D, Saha B, et al. Robust entity resolution using random graphs//Proceedings of the ACM 2018 International Conference on Management of Data. Houston, USA, 2018: 3-18

[6] Chi Y, Hong J, Jurek A, et al. Privacy preserving record linkage in the presence of missing values. Information Systems, 2017, 71(11): 199-210

[7] Wang H, Ding X, Li J, et al. Rule-based entity resolution on database with hidden temporal information. IEEE Transactions on Knowledge and Data Engineering, 2018, 30(11): 2199-2212

[8] Elmagarmid A K, Ipeirotis P G, Verykios V S. Duplicate record detection: A survey. IEEE Transactions on Knowledge and Data Engineering, 2007, 19(1): 1-16

[9] Benjelloun O, Garcia-Molina H, et al. Swoosh: A generic approach to entity resolution. The International Journal on Very Large Data Bases, 2009, 18(1): 255-276

[10] Whang S E, Marmaros D, Garcia-Molina H. Pay-as-you-go entity resolution. IEEE Transactions on Knowledge and Data Engineering, 2013, 25(5): 1111-1124

[11] Altowim Y, Kalashnikov D V, Mehrotra S. Progressive approach to relational entity resolution. The Very Large Data Bases Endowment, 2014, 7(11): 999-1010

[12] Papenbrock T, Heise A, Naumann F. Progressive duplicate detection. IEEE Transactions on Knowledge and Data Engineering, 2015, 27(5): 1316-1329

[13] Christen P. A survey of indexing techniques for scalable record linkage and deduplication. IEEE Transactions on Knowledge and Data Engineering, 2012, 24(9): 1537-1555

[14] Hernández M A, Stolfo S J. Real-world data is dirty: Data cleansing and the merge/purge problem. Data Mining and Knowledge Discovery, 1998, 2(1): 9-37

[15] McCallum A, Nigam K, Ungar L H. Efficient clustering of high-dimensional data sets with application to reference matching//Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Boston, USA, 2000: 169-178

[16] Papadakis G, Ioannou E, Palpanas T, et al. A blocking framework for entity resolution in highly heterogeneous information spaces. IEEE Transactions on Knowledge and Data Engineering, 2013, 25(12): 2665-2682

[17] Papadakis G, Koutrika G, Palpanas T, et al. Meta-blocking: Taking entity resolution to the next level. IEEE Transactions on Knowledge and Data Engineering, 2014, 26(8): 1946-1960

[18] Kenig B, Gal A. MFIBlocks: An effective blocking algorithm for entity resolution. Information Systems, 2013, 38(6): 908-926

[19] Fisher J, Christen P, Wang Q, et al. A clustering-based framework to control block sizes for entity resolution// Proceedings of the 21st ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Sydney, Australia, 2015: 2791

[20] Franklin M, Halevy A, Maier D. From databases to dataspaces: A new abstraction for information management. ACM Sigmod Record, 2005, 34(4): 27-33

[21] Halevy A, Franklin M, Maier D. Principles of dataspace systems//Proceedings of the 25th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems. Chicago, USA, 2006: 1-9

[22] Jeffery S R, Franklin M J, Halevy A Y. Pay-as-you-go user feedback for dataspace systems//Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data. Vancouver, Canada, 2008: 847-860

[23] Madhavan J, Jeffery S, Cohen S, et al. Web-scale data integration: You can only afford to pay as you go//Proceedings of the 3rd Biennial Conference on Innovative Data Systems Research. Asilomar, USA, 2007: 342-350

[24] Draisbach U, Naumann F, Szott S, et al. Adaptive windows for duplicate detection//Proceedings of the IEEE 28th International Conference on Data Engineering. Arlington, USA, 2012: 1073-1083

[25] Hassanzadeh O, Chiang F, Lee H C, et al. Framework for evaluating clustering algorithms in duplicate detection. The Very Large Data Bases Endowment, 2009, 2(1): 1282-1293

SUN Chen-Chen, Ph. D. candidate. His research interests include entity resolution and data quality.

SHEN De-Rong, Ph. D., professor, Ph. D. supervisor. Her research interests include distributed data management and data integration.

KOU Yue, Ph. D., associate professor. Her research interests include entity search and data mining.

NIE Tie-Zheng, Ph. D., associate professor. His research interests include data quality and data integration.

YU Ge, Ph. D., professor, Ph. D. supervisor. His research interests include database and big data management.

Background

Entity Resolution (ER) is a key aspect of data integration and data cleaning, and is a necessary preprocess step for data mining and data analytic. Traditional ER approaches process a whole dirty dataset, and output the final ER result in the end. Nowadays a number of (nearly) real-time data analytic applications emerge, but traditional ER approaches cannot fulfill them. Given a short time range, Pay-as-you-go ER tries its best to optimize ER results, competent for the aforementioned new applications. For instance, financial news streams are (nearly) real-time such that clients deal with their financial business in time. The stock markets generate data frequently, including lots of company names and person names. But it is impossible to resolve all of them in limited time. To fulfill such requirements, Pay-as-you-go ER tries to resolve as many records as possible, given a short time range. The existing Pay-as-you-go ER approaches either cannot select the most potential candidate pairs or rely upon the best blocking or sorting keys, which limits their progressiveness or usability.

We propose a multi-pass blocking based Pay-as-you-go ER (BPER) approach. The core of Pay-as-you-go ER is to select the most potential candidate pairs. The proposed ER approach is able to locate the most redundant regions of dirty datasets and process them first. BPER estimates the redundancy of each block dynamically, and ranks candidate pairs via probabilities of matching in real time. The redundancies of blocks and resolution of candidate pairs interact each other iteratively. Initially BPER utilizes a weighted blocking graph to kill redundancies of a block set, and ranks all candidate pairs in a queue Q. In the iterative period, BPER selects the most potential candidate pair to resolve, and re-ranks Q according to the latest resolution result. The iterative process continues until the budget runs out or Q is empty. BPER does not require the best blocking keys, and tries best to select the most potential candidate pairs. Therefore BPER outperforms existing Pay-as-you-go ER approaches.

This work is supported by the National Basic Research 973 Program of China under Grant (2012CB316201), the National Natural Science Foundation of China under Grants (U1435216, 61672142, 61472070, 61602103), and the National Key Research and Development Project under Grant (2018YFB1003404).