基于Hadoop平台的并行中文句法分析研究

刘胜久[1] 李天瑞[1] 贾 真[1] 珠 杰[1,2]
(西南交通大学信息科学与技术学院 成都 610031)[1]
(西藏大学计算机科学系藏文信息技术研究中心 拉萨 850000)[2]

摘 要 作为自然语言理解研究重点的句法分析一直受到人们的关注。针对现今句法分析方法效率低、准确度不高的问题,借助云计算计算能力强的优势,探讨了在云计算平台上实现并行中文句法分析的方法。利用公开的语料库及开源的句法分析工具在搭建的Hadoop云计算试验平台上实现并行中文句法分析,实验结果及理论分析均证实了所设计的基于Hadoop平台的并行句法分析方法的可行性、有效性与稳定性。

关键词 云计算,Hadoop,并行,句法分析
中图法分类号 TP391 **文献标识码** A

Research on Parallel Chinese Syntactic Analysis Based on Hadoop Platform

LIU Sheng-jiu[1] LI Tian-rui[1] JIA Zhen[1] ZHU Jie[1,2]
(School of Information Science and Technology, Southwest Jiaotong University, Chengdu 610031, China)[1]
(Department of Computer Science, Tibetan University, Lhasa 850000, China)[2]

Abstract As one of research focuses of natural language processing, syntactic analysis has received much attention. Aiming at low efficiency and mediocre accuracy of current syntactic analysis methods, we investigated a parallel method of Chinese syntactic analysis on the cloud computing platform with the advantage of the computing power of cloud computing. We realized the parallel Chinese syntactic analysis on the built Hadoop cloud computing test platform with open corpus and source parser. Both experimental results and theoretical analysis confirm the feasibility, effectiveness and stability of the proposed method of parallel parsing based on Hadoop platform.

Keywords Cloud computing, Hadoop, Parallellism, Syntactic analysis

1 引言

自计算机诞生以来,就有用计算机实现自然语言理解的设想,并逐渐形成了一个新的学科——计算语言学。信息时代对自然语言处理提出了新的更高的要求。由于中文汉字之间没有空格,对中文的自然语言处理首先需要进行分词及词性标注,即所谓的词法分析。计算机是以处理印欧语系为基础的,对印欧语系的自然语言处理具有较好的支撑能力,但作为汉藏语系的汉语与印欧语系差别很大,能够处理印欧语言的计算机面对汉语汉字却显得无能为力。姚天顺等指出"汉语分析是个极其复杂的问题,非常不利于计算机处理"[1]。

在中文词法分析方面,中科院[2]、哈尔滨工业大学、北京语言大学、北京航空航天大学[3]、天津海量信息技术有限公司,以及其他高校与科研机构均进行了相应的研究并推出了不同的中文词法分析工具,其在准确率与效率方面均达到了较高的水平。如北京语言大学现代汉语通用分词系统(GPWS v3.5)分词总体准确率超过98%,分词速度超过60万字/秒[1);中科院的ICTCLAS汉语分词系统分词精度达到98.45%,分词速度达到500kB/s[2);海量信息技术有限公司推出的智能分词系统准确率达到99.7%,分词速度达到2000万字/分钟[3)。实际上这3个系统都在不同领域得到了广泛的应用。

句法分析是对词法分析后的结果进行句子结构分析,一般情况下是构建句法依存树、句法分析树或语法树,为后续进一步更深层次的研究奠定基础。早期对中文句法分析的研究是直接套用国外的句法分析方法,如概率上下文无关文法

到稿日期:2013-05-15 返修日期:2013-07-21 本文受国家自然科学基金(61175047,61262058,61152001),中国科学院自动化研究所复杂系统管理与控制重点实验室开放课题(20110102)资助。

刘胜久(1988—),男,博士生,主要研究方向为数据挖掘与知识发现等,E-mail:liushengjiu2008@163.com;李天瑞(1969—),男,教授,博士生导师,主要研究方向为数据挖掘与知识发现、粗糙集与粒计算等;贾 真(1975—),女,博士生,讲师,主要研究方向为信息抽取、内容安全等;珠 杰(1973—),男,博士生,副教授,主要研究方向为藏文信息处理、数据挖掘等。

1) http://democlip.blcu.edu.cn:8081/gpws/

2) http://ictclas.org/

3) http://www.hylanda.com/product/fenci/

(Probabilistic Context-Free Grammar,简写为 PCFG)、短语结构语法(Phrase Structure Grammar,简写为 PSG)、依存语法(Dependency Grammar)、树邻接语法(Tree Adjoining Grammar)、链语法(Link Grammar)、词汇功能语法(Lexical Functional Grammar)、范畴语法(Categorial Grammar)、扩充转移网络(Argumented Transition Network)、功能合一语法(Functional Unification Grammar)等[4],并针对中文的实际,将其改进后直接用于中文句法分析,实际的句法分析效果也各有千秋,互有长短。

1.1 中文句法分析

对于句法分析而言,效率是一个至关重要的因素。句法分析对句子的长度极为敏感,随着长度的增加,句法分析的效率以及准确率均会受到严重的影响。Eisner 于 1996 年提出的最大生成树算法(Projective Maximum Spanning Tree),其复杂度为句子长度的三次方[5]。一般情况下比较中文句法分析的准确率也只是在句子长度不超过 40 的宾州中文树库测试集上进行的。PCFG 尽管效率较高,但其基于统计的句法分析思想,效率随语料库的扩大而急速降低。在可接受的句法分析准确度下最大限度地提高句法分析的效率是一个亟待解决的重要问题。中科院 ICTPROP 汉语句法分析系统通过引入结构上下文条件,在一定程度上突破了 PCFG 的上下文无关假设,使得分析结果正确率有了明显提高[6]。哈尔滨工业大学的中文依存句法分析系统通过对大规模依存树库的统计学习,建立了一个词汇化的概率分析模型,并充分利用句子中的结构信息,在分析算法上使用一个确定性的搜索算法,在线性时间内对句子进行解码,使分析结果的准确率和运行的时空效率都达到了较高的水平[7]。

对自然语言的句法分析研究尚在进一步深入,句法分析的准确率也不断得到提升[8]。对中文而言,层级分类[9]、结构下文及词汇信息[10]也逐步引入到句法分析中,准确率已超过 90%。但在同等条件下,准确率的提升是以效率的下降为代价的,例如哈尔滨工业大学的中文依存句法分析系统的中文依存句法分析每秒只能处理 500 个句子[4)]。这是由于复杂的句法分析方法利用了过多的理论、方法与技术,导致中文句法分析的效率越来越低。

由于自然语言表达的多样性及其固有的歧义问题,中文句法分析的准确度及效率远远落后于词法分析,对开放领域的海量数据处理更是乏力,导致包括语料库构建在内的中文自然语言理解方面的研究滞后于英语等其他语言,这也是迄今尚没有与 Ask、DBpedia[11] 等实用的智能搜索引擎类似的中文智能搜索引擎的重要原因。

1.2 MapReduce 技术

云计算的出现为数据密集型、计算密集型、存储密集型、安全密集型等其他类型的计算提供了一个新的解决方案[12]。以 Google 云计算技术为基础的 Hadoop 开源云计算在应用方面得到了极为广泛的应用。其中 Google 定义的并行程序设计模式——MapReduce 是目前面向海量数据处理最为成功的技术,在云计算的应用方面大部分的研究均是基于 MapReduce 计算模式的[13,14]。

在 Hadoop 云计算平台上,计算是通过 Map-Reduce (Fork-Join)模型实现的。即先通过 Map,将数据近似均衡地分配到不同的计算机上进行计算,再通过 Reduce 将不同计算机上计算得到的结果进行归约,并将计算结果返回给用户。实际中若数据量过大,对每一个 Map 的结果,即 split(分片),还存在类似于 Reduce 的中间操作,即 Combine。其采用键值对(Key-Value)方式存储数据,对相同的键对应的数据集合进行处理。由于 Hadoop 提供有 Map、Combine、Reduce 3 个计算阶段,可以针对实际计算的特点在不同的阶段对数据进行处理。

将云计算技术应用于自然语言理解以解决自然语言理解面临的一些难题是当前研究的一大热点。文献[15]对 MapReduce 近年来在文本处理各个方面的应用进行了分类总结和整理。文献[16]利用 Hadoop 框架下的 MapReduce 编程模型实现了句群相似度并行计算方法,并通过实验验证了该算法的稳定性和处理大量数据的可行性。本文拟尝试利用云计算计算能力强的优势,在通用的 Hadoop 云计算平台下采用 MapReduce 计算模型实现并行中文句法分析以提高句法分析的效率,同时从理论上定量地分析并行句法分析相对于串行句法分析的优势。

2 基于 Hadoop 平台的并行中文句法分析框架

在中文句法分析领域,为了提高分析的效率与准确率,不同的科研机构提出了不同的方法,包括提供多种句法分析方法供用户选择、通过训练选取更合适的模型参数、针对不同领域的需求采用不同的句法分析方法等。以 PCFG 及 Factored 两种最常用的句法分析方法为例,PCFG 的主要优势在于分析速度,而 Factored 的主要优势在于分析准确率。同时有多种不同的句法分析方法占主导地位,一方面说明了句法分析的重要性,另一方面也反映了句法分析的困难程度。国内的复旦大学、国外的斯坦福大学[17]以及其他高校与科研机构均提供有各种通用或专用的句法分析工具。

词法分析主要分析语句中词语的物理关联类似,而句法分析主要分析语句中词语的逻辑关联,它将词法分析后的语句处理为结构化的依存分析树或句法分析树,其分析过程可以视为从待分析语句到句法分析结果的函数 *Par*,完全可以利用 Hadoop 云计算框架下的 MapReduce 模型实现并行中文句法分析。

Hadoop 整个计算过程有 Map、Combine、Reduce 3 个计算阶段,理论上可以在上述任一阶段进行句法分析。在实验中我们发现,在 Map 或 Combine 阶段进行句法分析时计算节点的失败率较高,存在较多的回退计算,且由于输入中文语句集合可能存在重复的语句,导致在 Map 或 Combine 阶段进行句法分析存在不必要的冗余计算等,我们采用在 Reduce 操作中调用句法分析函数 *Par* 实现在此阶段进行句法分析。于是,可以得到 Hadoop 云计算平台下基于 MapReduce 计算模型的并行中文句法分析框架,如图 1 所示。

图中 Seg 表示分词操作;Par 表示句法分析操作;Map(·)、Combine(·)、Reduce(·)分别表示在各自对应的阶段进行的 Map、Combine、Reduce 处理操作;S 表示原始语句集合;S′、S″表示在不同阶段经过去重处理后的语句集合;S′、

4) http://ir.hit.edu.cn/phpwebsite/index.php?module=pagemaster&PAGE_user_op=view_page&PAGE_id=147

Si′、Si″表示不同计算节点上的 split 中不同处理阶段的语句集合。

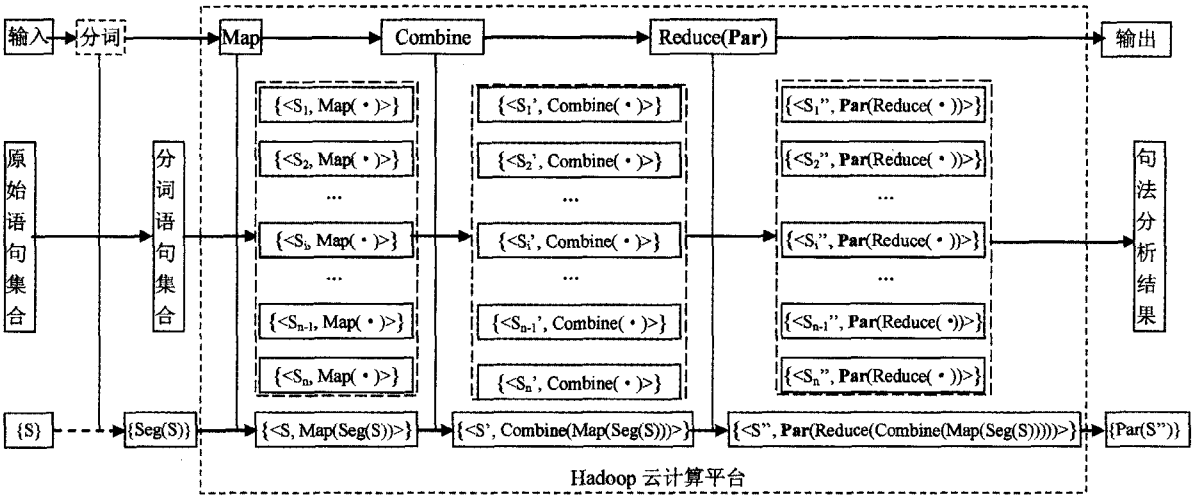

图 1 Hadoop 云计算平台下并行中文句法分析框架

原始中文语句经过词法分析阶段的分词操作，得到分词后的结果，在 Hadoop 云计算平台上的 Reduce 阶段进行句法分析，最后得到句法分析后的结果。实际中，若中文语句已经经过词法分析的分词处理或已有分词后的中文语句，可直接将分词后的语句作为输入，即可略去分词操作。

3 基于 Hadoop 平台的并行中文句法分析实验

在中文语料库方面，北京大学计算语言学研究所和富士通研究开发中心有限公司制作了《人民日报》1998 年全年 2600 万汉字的人民日报标注语料库，其应用较广。这里选取其中 1 月份的标注语料库进行实验。首先是剔除标注信息，经过处理，得到以“。”、“?”、“!”及“……”等结尾的中文语句共 37740 条。分别随机选取 1000、2000、3000、……、36000 条语句进行实验，共 36 组。

在开源句法分析工具方面，复旦大学及斯坦福大学均提供有开源的中文句法分析工具，这里选用斯坦福大学提供的中文句法分析工具进行并行句法分析。另外，由于斯坦福大学提供的开源句法分析工具共有 xinhuaPCFG、xinhuaFactored、xinhuaFactoredSegmenting、chinesePCFG、chineseFactored、chineseFactored7 等 6 种中文句法分析方法。其中 xinhuaPCFG 是基于新华社的语料库采用 PCFG 进行中文句法分析，应用较广，比较适合此标注语料集，于是本文选取 xinhuaPCFG 句法分析方法来进行实验。

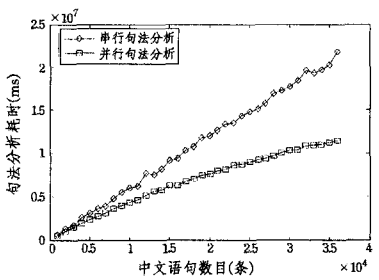

图 2 串行及并行中文句法分析效率对比图

我们用 10 台 DELL PowerEdge R710(Xeon E5506/12G/600G)服务器搭建了 Hadoop 云计算试验平台，其中 1 个主节点、9 个从节点，主节点与从节点通过千兆以太网相连接。所用的 Linux 内核版本为 2.6.18-164.el5，Java 版本为 1.7.0_01，Hadoop 版本为 1.0.3，使用斯坦福大学提供的开源句法分析工具版本为 stanford-parser-2012-03-09，其他参数均为默认配置。实验的目的是比较在 1 台服务器及 Hadoop 云计算试验平台上的中文句法分析效率。我们得到的句法分析结果如图 2 所示。

利用 IBM SPSS Statistics 20 统计分析软件进行回归分析，得到的回归方程为：

$$\begin{cases} T_{Linux}=100429.930+590.680Num, R^2=0.999 \\ T_{Reduce}=150087.278+459.032Num-0.004Num^2, R^2=0.998 \end{cases}$$

二者的耗时增长情况与语句数目的关系即为对应方程的一阶导数，即有：

$$\begin{cases} \Delta T_{Linux}=T_{Linux}'=590.680 \\ \Delta T_{Reduce}=T_{Reduce}'=459.032-0.008Num \end{cases}$$

对应的 $Sizeup$ 为 T_{Linux} 与 T_{Reduce} 之比，即有：

$$Sizeup=\frac{T_{Linux}}{T_{Reduce}}=\frac{100429.930+590.680Num}{150087.278+459.032Num-0.004Num^2}$$

分析计算表明，$Sizeup$ 的一阶导数 $Sizeup'$ 恒大于 0，即 $Sizeup$ 越来越大。

从上述分析可以看出，在 1 台服务器上，串行中文句法分析耗时随语句数目的增加而线性增加，增长率恒定；在 Hadoop 云计算平台上，并行中文句法分析耗时随语句数目的增加而亚线性增加，增长率变缓；$Sizeup$ 随语句数目的增加而大幅增长，虽然未能提高句法分析的准确率，但极大地提高了句法分析的处理速度，与其他同类算法相比有一定的优势。实验结果及理论分析均证实了所设计的并行句法分析方法的可行性、有效性与稳定性。

结束语 本文针对中文句法分析效率低的问题，借助云计算计算能力强的优势，研究了在云计算平台实现并行中文句法分析的方法。针对 Hadoop 云计算平台的实际，提出了相应的并行中文句法分析方法，并在搭建的云计算试验平台上进行了实验，实验结果与理论分析均证实了所提方法的可行性与有效性。今后的研究工作之一是如何进一步借助云计算计算能力强的特点来提高中文句法分析的效率及准确率，同时研究海量的中文语句句法分析的原理与方法，以期进一步提高并行中文句法分析的效率等。

（下转第 115 页）

[C]//Proceedings of the BCS-IRSG 22nd annual colloquium on information retrieval research. 2000:57-66

[3] Radlinski F, Joachims T. Query chains: learning to rank from implicit feedback[C]//Proceedings of the eleventh ACM SIGKDD international conference on knowledge discovery in data mining. 2005:239-248

[4] Jansen B J, Spink A. How are we searching the World Wide Web? A comparison of nine search engine transaction logs[J]. Information Processing & Management, 2006, 42(1):248-263

[5] Spink A, Park M, Jansen B J, et al. Multitasking during Web search sessions[J]. Information Processing & Management, 2006, 42(1):264-275

[6] Lau T, Horvitz E. Patterns of search: analyzing and modeling Web query refinement[C]//Proceeding UM'99—Proceeding of the Seventh International Conference on User Modeling. Springer-Verlag New York, 1999:119-128

[7] He D, Göker A, Harper D J. Combining evidence for automatic web session identification[J]. Information Processing & Management, 2002, 38(5):727-742

[8] Ozmutlu H C, Cavdur F. Application of automatic topic identification on excite web search engine data logs[J]. Information Processing & Management, 2005, 41(5):1243-1262

[9] Shen X, Tan B, Zhai C. Implicit user modeling for personalized search[C]//Proceedings of the 14th ACM International Conference on Information and Knowledge Management. 2005: 824-831

[10] Jones R, Klinkner K L. Beyond the session timeout: automatic hierarchical segmentation of search topics in query logs[C]// CIKM'08. 2008:699-708

[11] 张磊,李亚楠,王斌,等. 网页搜索引擎查询日志的 Session 划分研究[J]. 中文信息学报,2009,23(2):54-61

[12] Lucchese C, Orlando S, Perego R, et al. Identifying task-based sessions in search engine query logs[C]// Proceedings of the Fourth ACM International Conference on Web Search and Data Mining. 2011:277-286

[13] Lui Y, Agichtein E. On the Evolution of the Yahoo! Answers QA Community[C]//the ACM SIGIR International Conference on Research and Development in Information Retrieval. Singapore, 2008:737-738

[14] Nam K K, Ackerman M S. Question in, Knowledge in?: a study of naver's question answering community[C]// Proceedings of CHI'09. Boston, MA, 2009:779-788

[15] Rodrigues E M, Frayling N M. Socializing or knowledge sharing?: characterizing social intent in community question answering[C]// Proceedings of CIKM 2009. Hong Kong, China, 2009:1127-1136

[16] Liu Q L, Agichtein E, Dror G, et al. Predicting web searcher satisfaction with existing community-based answers[C]// Proceedings of SIGIR'11. Beijing, China, 2011

[17] Jiang D, Pei J, Li H. Mining Search and Browse Logs for Web Search: A Survey[J]. ACM Transactions on Computational Logic, 2013, 4(4):1-42

[18] Wikipedia. Uniform resource locator[EB/OL]. http://en.wikipedia.org/wiki/Uniform_resource_locator

[19] Hassan A, Jones R, Klinkner K L. Beyond DCG: User behavior as a predictor of a successful search[C]// Proceedings of the third ACM international conference on Web search and data mining. 2010:221-230

[20] Liu Y, Bian J, Agichtein E. Predicting information seeker satisfaction in community question answering[C]// Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. 2008:483-490

(上接第 90 页)

参 考 文 献

[1] 姚天顺,朱靖波,张琍,等. 自然语言理解———种让机器懂得人类语言的研究(第 2 版)[M]. 北京:清华大学出版社,2002

[2] 俞鸿魁,张华平,刘群,等. 基于层叠隐马尔可夫模型的中文命名实体识别[J]. 通信学报,2006,27(2):87-94

[3] 曹勇刚,曹羽中,金茂忠,等. 面向信息检索的自适应中文分词系统[J]. 软件学报,2006,17(3):356-363

[4] 刘挺,马金山. 汉语自动句法分析的理论与方法[J]. 当代语言学,2009,11(2):100-112

[5] Mark A. Paskin, Cubic-time Parsing and Learning Algorithms for Grammatical Bigram Models [R]. Technique report, 2001

[6] 熊德意,刘群,林守勋. 融合丰富语言知识的汉语统计句法分析[J]. 中文信息学报,2005,19(3):61-66

[7] 李正华,车万翔,刘挺. 基于柱状搜索的高阶依存句法分析[J]. 中文信息学报,2010,24(1):37-41

[8] Harper M P, Huang Zhong-qiang. Chinese Statistical Parsing [M]// Joseph Olive, John McCary, Caitlin Christianson, eds. Handbook of Natural Language Processing and Machine Translation. Reston Virginia, Defense Research Projects Agency, 2011:90-102

[9] 代印唐,吴承荣,马胜祥,等. 层级分类概率句法分析[J]. 软件学报,2011,22(2):245-257

[10] 陈功,罗森林,陈开江,等. 结合结构下文及词汇信息的汉语句法分析方法[J]. 中文信息学报,2012,26(1):9-15

[11] Bizera C, Lehmannb J, Kobilarova G, et al. DBpedia-A Crystallization Point for the Web of Data [C]// Proceedings of Web Semantics: Science, Services and Agents on the World Wide Web. 2009:154-165

[12] 罗军舟,金嘉晖,宋爱波,等. 云计算:体系架构与关键技术[J]. 通信学报,2011,32(7):3-21

[13] 于戈,谷峪,鲍玉斌,等. 云计算环境下的大规模图数据处理技术[J]. 计算机学报,2011,34(10):1753-1767

[14] Bahga A, Madisetti V K. Analyzing Massive Machine Maintenance Data in a Computing Cloud [J]. IEEE Transactions on Parallel and Distributed Systems, 2012, 23(10):1831-1843

[15] 李锐,王斌. 文本处理中的 MapReduce 技术 [J]. 中文信息学报,2012,26(4):9-20

[16] 宁可为,王炜,李园伟. 基于 Hadoop 的句群相似度计算 [J]. 计算机系统应用,2010,19(12):59-63

[17] http://www-nlp.stanford.edu/software/lex-parser.shtml[EB/OL]