面向超大规模的中文文本N-gram串统计

余一骄[1] 刘 芹[2]
(华中师范大学语言学系 武汉 430079)[1] (武汉大学计算机学院 武汉 430072)[2]

摘 要 中文文本统计软件Cici高效地实现了对超大规模中文文本语料N-gram串频次的统计与检索。通过统计不同规模中文语料库发现,当N等于6时,语料库中包含的不同N-gram汉字串数量最多。根据"句子"的平均长度和数量,可以准确估算语料库中包含的N-gram串数量。根据多数汉字串在语料库中出现频次低于10次的特点,提出对汉字串频次信息实现分段存储与排序,即对频次不超过10的汉字串独立存储,对频次高于10的汉字串进行分段排序与存储。对大规模中文文本应先进行分块统计,然后合并分块统计结果,建议分块规模约为20MB。

关键词 汉字,N-gram,语料库,排序
中图法分类号 TP391.1 **文献标识码** A

N-gram Chinese Characters Counting for Huge Text Corpora

YU Yi-Jiao[1] LIU Qin[2]
(Department of Linguistics, Central China Normal University, Wuhan 430079, China)[1]
(Computer School, Wuhan University, Wuhan 430072, China)[2]

Abstract Counting N-gram Chinese characters of huge text corpora is a challenge for Chinese information processing and Cici was developed to count huge Chinese text corpora efficiently. We found that the number of different Chinese strings is maximal when the length of strings is 6, and the number of strings can be estimated by the average length of sentences. Since most Chinese strings appear no more than 10 times in the corpora, the N-gram characters are stored in 13 separate files according to their frequency, and only highly used strings are sorted. This strategy speeds up the accounting process dramatically. Due to the limited physical memory, huge Chinese text corpora have to be divided into many blocks, whose size is suggested to be 20MB. Every block is counted separately, and then the block statistic results are merged together. We implemented the algorithm of accounting huge corpora efficiently in personal computer.

Keywords Chinese character, N-gram, Corpora, Sorting

1 前言

中文信息处理研究的诸多热点领域,如新词识别[1]、涉嫌恐怖主义网页过滤[2]、中文网页检索[3]等,都依赖对大规模中文文本进行N-gram汉字串频率统计。对句子"我爱钓鱼岛"而言,它包含的2-gram汉字串为"我爱"、"爱钓"、"钓鱼"、"鱼岛";3-gram串包括"我爱钓"、"爱钓鱼"、"钓鱼岛"。从汉语本体研究的角度来看,一些N-gram汉字串是词或短语,如"钓鱼"、"钓鱼岛"等,而多数的N-gram汉字串则不具有语言学研究价值,如"爱钓"。

汉语词汇和语法研究者对特定汉字串在大规模语料库中的出现频次十分关注,但他们却很难获得相关数据。目前汉语研究者大多通过查询网络语料库,来获取特定汉字串的使用频率。依赖通用语料库的语言学研究存在以下不足:第一,通用语料库仅提供单个汉字或二字串的频次及排序信息,不提供三字串、四字串及更长字串的频次信息[4]。第二,用户无权决定、更改语料来源。通用语料库坚持语料来源的广泛性、均衡性等原则,每条语料长度较短,且不会覆盖某个作家或某部作品的全部内容。而有些语言研究洽洽是要深入到某个作家、某个流派作家的作品以及特定体裁的文本中,以获取其语言特征。第三,语料规模偏小。即便是在语言学领域应用十分广泛的北京大学CCL中文语料库,它也仅包含4.77亿个汉字,1.06GB。其中CCL还包含397MB的古代汉语语料,因此现代汉语语料不到700MB[4]。汉语言、中文信息处理、文本挖掘等领域的研究者都期望对更大规模(GB级别)的现代汉语语料进行N-gram串频次统计分析[5]。第四,语料更新速度慢。通用语料库一旦开发完成,后期较少进行语料更新。新词自动发现、汉语词汇使用规律的历时考察,都需要对不同时期的语料,特别是最近一段时间的语料样本进行统计[6]。针对更新缓慢甚至是一成不变的语料库的统计,难以满足汉语研究者的需求。

2011年5月开始,我们深入地对华中师范大学语言学

到稿日期:2013-06-05 返修日期:2013-10-15 本文受教育部人文社会科学研究项目:逻辑推理与词义匹配相融合的中文网页语义检索技术研究(10YJA740120),湖北省教育厅人文社会科学研究项目:基于语义理解的中文网页检索方法研究(2010b032)资助。
余一骄(1978—),男,博士,副教授,主要研究方向为计算语言学、计算机网络,E-mail:yjyu@mail.ccnu.edu.cn;刘 芹(1978—),女,博士,副教授,主要研究方向为计算机网络。

系、语言与语言教育部人文研究中心的众多汉语本体研究者进行了调研，获得了汉语研究者对 N-gram 频次统计分析的需求。2011 年 9 月至今，我们用 Java 开发了中文文本 N-gram 串统计和检索软件 Cici V1.0。Cici 支持 GB 级别的中文文本语料 N-gram 串统计与检索，如今它是一个自由、绿色软件。它已被华中师范大学汉语研究者使用了近两年，为汉语本体研究提供了直接的技术支持[7]。本文将讨论 Cici 中关于 N-gram 串频次统计的关键技术。

本文第 2 节介绍语料库 N-gram 串统计领域的相关研究；第 3 节介绍本文的语料来源和最大汉字串长度；第 4 节估算语料库中的 N-gram 汉字串数量以及不同的 N-gram 汉字串数量；第 5 节讨论 Cici 如何对汉字串频次信息进行分段存储与排序；第 6 节研究语料分块统计技术；最后对全文进行总结。

2 相关工作

中文和日文都是象形、表意文字。文献[8]提出先对日文字符串建立 Coincidence 表，然后通过计算字典序相邻字符串所拥有的共同前缀字符数量，来统计日文 N-gram 字符串频次。该统计方法的突出优点是：只需一次扫描待统计日文文本并生成 Coincidence 表，就能快速统计不同长度的 N-gram 串($n\leqslant256$)的频次。当 n 变化范围越大时，该算法的优势就越明显。该算法只用数组记录了每个字串的起始位置(终止位置统一设置为文档的结束)，因此其存储开销极小。使用 2 字节日文编码，若被统计日文字符为 S 个，则系统所需的存储空间为 7S 个字节。文献[9]将文献[8]所提出的日文统计算法应用到汉字 N-gram 串统计中，对规模为数十 MB 的中文语料进行统计，也获得了较好的效果。

文献[10]利用后缀阵列(Suffix Array)分别对包含 5 千万个英语词汇的英文文本、2.16 亿个日文字符的日语文本语料进行 N-gram 串频次统计。文献[11]介绍面向英文单词的 N-gram 串统计软件 N-gram Statistics Package 的功能，但未对该软件包的开发技术作深入讨论。从该文内容来看，该软件不适合较大规模的文本统计。

N-Gram Extraction Tool 是一个针对小规模汉语文本的 N-gram 串统计软件，但它不对频次统计结果进行排序[12]。该软件由于使用命令行方式进行人机交互，且缺乏汉字串的频次排序结果，因此较难为汉语研究者提供便捷的支持。

文献[13]仅统计了 GB2312-80 包含的 6763 个汉字，它以 2-gram 为基础，建立二级索引统计汉字 n-gram 频率。该方案存储空间较低，提高了针对某个 n-gram 汉字串的统计速度。文献[14]对大规模中文文本进行 1-gram 和 2-gram 统计。2-gram 统计与任意长度字串统计在实现方法上有显著差异。二字节 Unicode 编码包含 2 万多个汉字，在语料库中出现的汉字通常更少，完全可以用二维静态数组来记录语料中不同二字串的频次。也就是说，MB 级别和 GB 级别的中文二字串统计，在存储复杂性方面没有太大差异。但四字串、五字串及更长汉字串的频次统计，则不可能用简单的静态多维数组来实现。

文献[15]对规模为 20 亿字的汉语文本语料进行了二字串、三字串统计，并总结了二字串、三字串数量与语料规模的拟合公式。该文未对所用的统计工具和统计方法作详细说明。文献[16]对 200MB 的中文和英文文本进行了统计。比较中英文字串、词串统计结果发现，中英文 1-gram 和 2-gram 统计结果差异大，3-gram 和 4-gram 统计结果差异很小。

综上所述，目前较少有针对大规模中文文本 N-gram 串统计技术研究的论文，对用户自定义语料进行汉字串频次统计与排序的免费软件更少。

3 语料来源与统计对象

本文所使用的测试语料规模为 5.93GB，均来自因特网。根据对语料的加工程度，我们将其分为两类。

(1)规模为 1.08GB 的半自动采集语料。这部分语料将从互联网上手工采集的文本内容存为 TXT 文件。Cici 对这些 TXT 文本进行自动语料预处理，删除网站来源、版权、广告、目录等信息；然后，再经过语言学专业研究生人工检测，进一步删除乱码文字、英文字符等噪声信息。这部分语料包含 4 大类：中国现当代文学作品(包含现当代著名、知名作家的代表性作品，以及比较知名的文学作品)、政府公文(包含法律、政策、公文等)、新闻(包含人民日报、新华网等主流媒体的新闻报道)、网络小说(包括都市言情、军事、科幻灵异、玄幻修真、游戏竞技等子类)。4 类语料的规模分别为 312MB、94.3MB、92.2MB、609MB，共 1107.5MB，比例约为 3.4:1:1:6.6。本文对 1GB 以内的语料进行计算，主要针对这部分精确度较高的语料进行统计。

(2)规模为 4.87GB 的全自动获取、去噪网页文本。这部分语料的原始网页为 35GB，主要来自百度论坛。自动抽取网页中的文本数据，将其转存为 TXT 文档。这部分语料大多是用户在网络上发的帖子以及生活新闻。由于数据量大，且网页布局格式复杂，这部分数据存在较多噪声。这部分语料用来测试 Cici 对汉字编码格式(如 Unicode、GBK 等)自动判定的准确率，以及对大规模中文文本的统计效率。

被统计汉字串越长，计算复杂性和空间复杂性就越高。确定最长的汉字串长度时，既要充分考虑汉语本体研究的需要，也要考虑程序的计算复杂性。中文文本 N-gram 串频次统计主要是为了挖掘关于汉语词的规律。汉语词大多是单字词、二字词、三字词和四字词(主要是成语)。《现代汉语词典(第五版)》所列词汇的平均长度为 2.3 个汉字[17]。但现代汉语中部分专属名词比较长，如“中华人民共和国”、“华中师范大学”等。上述 1.08GB 现代汉语语料库约有 4.86 亿个汉字，它所包含的不同十字串数量、频次信息、百分比如表 1 所示，其中对小数点后第二位的数据进行了四舍五入处理。

表 1 大规模语料库中的十字串数量、频次及百分比

频次	1	2	3	4
字串数量	93702716	4787652	524226	202487
字串百分比	94.16%	4.81%	0.53%	0.20%
频次百分比	86.04%	8.79%	1.44%	0.74%
频次	5	6	7	8
字串数量	87040	60138	33719	24244
字串百分比	0.09%	0.06%	0.03%	0.02%
频次百分比	0.40%	0.33%	0.22%	0.18%
频次	9	10	11-100	>100
字串数量	15855	13136	66075	1572
字串百分比	0.02%	0.01%	0.07%	0.00%
频次百分比	0.13%	0.12%	1.25%	0.34%

由表 1 可知：在该语料库中，94.16%的十字串仅出现一

次，而出现 11－100 次的十字串占全体不同的十字串的比率仅为 0.0664%；出现 101－1000 次的十字串占全体不同十字串的比率为 0.0016%；出现 1000 次以上的十字串居然只有 29 个，大多是“人民代表大会常务委员”、“社会主义市场经济体制”之类的字串。汉语词是频次较高的汉字串，既然高频十字串很少，那么更长的 N-gram 汉字串频次统计结果对汉语研究者的帮助就更小。根据表 1 所列的统计结果，并征求汉语本体研究者的建议，Cici 最终只对长度为 1－10 的汉字串进行频次统计。

为了简便起见，本文用实际测量的统计时间来描述不同问题的计算时间复杂性。全文所有实验均使用同一台笔记本电脑来完成，该机软硬件配置信息如下：Win7 操作系统、AMD4 核 CPU（因没有采用并行处理技术，Cici 运行时其实只使用了其中的一个核）、8GB 内存。

4 汉字串数量

汉语研究者不仅关注特定汉字串在语料库中的出现频次，还关注不同汉字串在语料库中的频次排序。我们期望一次性获得全体汉字串的频次信息，然后对其中的高频汉字串进行深入考察。因此，在获得全体汉字串的出现频次后，还需根据频次大小对汉字串排序。排序算法的复杂度与待排序元素个数相关，高效排序算法的时间复杂度为 $O(slogs)$，其中 s 为待排序元素数量。根据频次排序汉字串，此处的 s 为不同的汉字串数量。如果语料库中出现了 M 个不同的汉字，则长度为 N 的 N-gram 汉字串可能达到 M^N 个。汉字串数量随字串长度增加呈指数增长，给字串统计带来了巨大挑战。

本文对 1.08GB 的语料进行统计，长度为 2－10 的不同汉字串数量如表 2 所列。当 $N=6$ 时，不同的 N-gram 汉字串最多，超过 1.80 亿个。当 $N<6$ 时，不同的 N-gram 汉字串数量随着串长增加而增大；但 $N>6$ 时，不同的 N-gram 汉字串数量却随着串长增大而降低。

表 2 大规模语料库中的不同长度字串数量

字串长度	2	3	4
字串数量	4029779	43697074	116935223
字串长度	5	6	7
字串数量	169111110	180725570	166956581
字串长度	8	9	10
字串数量	144484470	120936642	99518860

为了排除特定语料对不同汉字串数量的影响，本文设计了 6 组不同规模的中文语料：100MB、200MB、300MB、400MB、500MB、600MB，然后观察不同规模语料库中不同汉字串数量与字串长度 N 的关系。图 1 示出在这 6 组被测试语料中，都出现了当 N 大于 6 时不同的汉字串数随着 N 增大而减少的现象。这说明表 2 中所描述的数据特征具有普遍性。

语料库中包含的二字串总数多，但因串长太短，且二字串搭配规律性较强导致重复出现比率高，因此不同的二字串数量较少。为何不同的十字串数量少于不同的六字串数量则值得进一步探究。为了解释清楚表 2 和图 1 中的规律，以下通过计算被统计语料中“句子”的平均长度来回答该问题。由两个相邻标点符号之间的汉字串，或阿拉伯数字、英文字母之间的汉字串被称为“句子”。例如：“09:00 视频直播 NBA 季后赛西部决赛Ⅲ-灰熊 vs 马刺”，将分解为“视频直播”、“季后赛西部决赛”、“灰熊”、“马刺”4 个“句子”。显然，此处的“句子”与现代汉语中的句子不是同一概念。

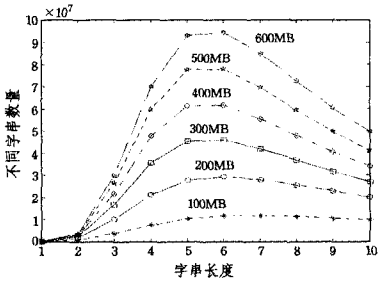

图 1 语料库规模与不同汉字串数量的关系

句子长度越短，包含的十字串数量就越少。以“我爱美丽的钓鱼岛”为例，它包括 3 个六字串、2 个七字串、1 个八字串，没有十字串。上述 1.08GB 语料库包含 56804758 个“句子”，平均长度为 8.56 个汉字；长度大于等于 10 的“句子”有 20577836 个，这些“句子”的平均长度为 14.3 个汉字；长度大于等于 6 的“句子”有 38513678 个，这些“句子”的平均长度为 11.1 个汉字。长度为 14.3 的汉字串可以产生 5.3 个十字串，$5.3\times20577836\approx1.09$ 亿；长度为 11.1 的汉字串可以产生 6.1 个六字串，$6.1\times38513678\approx2.34$ 亿。而程序统计结果显示：1.08GB 的语料包含 234330033 个六字串，包含 108903341 个十字串。考虑到计算中的舍入误差等因素，1.09 亿、2.34 亿分别与以上通过独立统计十字串、六字串数量所获得的 1.089 亿、2.343 亿极为接近。这表明通过“句子”平均长度来估算 N-gram 字串数量的方法是很准确的。

6 字以上的汉字串除了一些较长的专属名词及相关搭配如“人大常委委员会”、“根据中华人民共和国刑法”等外，其它汉字串的重复出现次数很低。在上述 1.08GB 的语料中，每个六字串平均出现 1.3 次，每个十字串平均出现 1.1 次。由于语料中包含的六字串总数约为十字串总数的 2.15 倍，即使六字串平均出现次数略高于十字串平均出现次数，不同的六字串数量也依然高于不同的十字串数量。当 N 大于 6 时，N-gram 汉字串重复出现次数低，因此语料库包含的 N-gram 串的数量对不同的 N-gram 汉字串数量就有重要的影响。由图 1 以及以上分析还可知：$N>10$ 时，其 N-gram 汉字串频次排序复杂度低于 $N=6$ 时的频次排序复杂度。

采集语料需要大量的人力、物力，因此语料库规模在短期内较难高速增长。不妨考察在语料规模呈现线性增长的情况下，不同的 N-gram 汉字串的数量变化规律。仍以图 1 实验中的 6 组语料作为被统计对象，它们所包含的不同二字串、三字串、四字串、五字串总数量变化情况如图 2 所示。

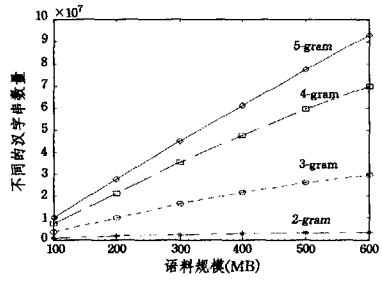

图 2 不同的汉字串数量与语料规模的关系

图 2 表明，第一，当语料规模增大时，不同字串数量显著增大。100MB 测试语料包含 715933 个不同的二字串、10045952个不同的五字串；600MB 测试语料包含 3244459 个不同的二字串、93072161 个五字串。上述语料规模增大 6 倍，不同的二字串数量增加了 4.53 倍，不同的五字串数量则增加了 9.26 倍。第二，当 $2 \leqslant N \leqslant 5$ 时，N 越大，不同的 N-gram 汉字串数量增长速度越快。

5 基于频次分布特征的汉字串频次排序与存储

获得全体汉字串的频次后，需要根据频次大小对这些不同的汉字串排序；利用文献[8]中的算法来统计不同长度的 N-gram 字符串，需要根据 Unicode 编码排序并生成 Coincidence 表。二者都需要执行大规模排序操作，但其采用的优化方法并不一致，以下将分别讨论。

在同一长度的字符串及频次信息统计完毕后，必须根据频次大小，对不同的汉字串进行排序。当语料规模高达数百 MB 时，该操作无论是在内存空间分配还是计算速度方面都存在较大挑战。若采用简单的二维表结构来完成不同的汉字串频次排序，需要存储汉字串及该汉字串的频次。不妨以表 2 中的六字串频次排序为例进行分析：1.81 亿个不同的六字串，存储每个六字串需 12 个字节，存储其频次需要 4 字节（Java 为整型分配 4 个字节）。仅存储这两项资源，就需要近 3GB 的内存。目前许多 PC 机难以满足单个应用程序 3GB 及以上的内存需求。另外，执行数以亿计的字符串交换操作，需要消耗大量时间。

考察大规模汉语语料库 N-gram 汉字串频次统计结果不难发现：绝大多数汉字串仅出现 1 次、2 次或 3 次，汉字串的频次分布与随机数分布有显著差异。表 3 列出了 1.08GB 语料库中包含不同的二字串、三字串、四字串（这是观察汉字词最关键的字串长度）信息。其中，“比率”是该频次的 N-gram 汉字串在该长度、不同 N-gram 汉字串中所占的百分比。例如，仅出现一次的二字串为 1283083 个，不同的二字串为 4029779 个，因此表 3 的第二行第三列为 1283083/4029779=31.84%。

表 3 规模为 1.08GB 语料中的二、三、四字串的频次分布比率

串长	频次	比率	频次	比率	频次	比率
2	1	31.84%	2	13.33%	3	7.56%
	4	5.17%	5	3.78%	6	2.91%
	7	2.35%	8	1.98%	9	1.66%
	10	1.43%	11－100	20.22%	101－1000	6.42%
3	1	56.33%	2	15.47%	3	6.86%
	4	4.06%	5	2.66%	6	1.92%
	7	1.44%	8	1.14%	9	0.91%
	10	0.76%	11－100	7.40%	101－1000	0.97%
4	1	72.30%	2	13.26%	3	4.65%
	4	2.46%	5	1.47%	6	1.01%
	7	0.72%	8	0.54%	9	0.42%
	10	0.34%	11－100	2.64%	101－1000	0.19%

从表 3 可见，90%以上的三字串、四字串的频次不超过 10；50%以上的二字串、三字串、四字串的出现频次不超过 3。尽管语料规模较大，但出现 1000 次以上的汉字串却很少。频次超过 1000 的二字串为 53748 个，仅占 1.33%；出现 1000 次以上的三字串为 32500 个，占 0.074%；出现 1000 次以上的四字词串为 8378 个，占 0.00072%。通过统计多种不同规模的汉语语料库可知，表 3 所列汉字串频次分布极不均匀的特征在不同规模的语料库都普通存在。

Cici 首先浏览汉字串频次统计结果（该结果是根据汉字串的 Unicode 编码为序来存储），把频次为 1、2、3、4、5、6、7、8、9、10 的字串，分别写入到 10 个文件中（在文件中按字符串的 Unicode 编码序存储）；把频次在 11－100 的汉字串，写入到一个文件中；把频次在 101－1000 的汉字串，写入到一个文件；把频次高于 1000 的汉字串，写入到一个文件。然后对记录频次大于 10 的汉字串频次信息的 3 个文件，分别进行内排序。若语料库规模持续增大，还可以对 101－1000 次之间的字串进行进一步分割，如把频次为 101－500 的字串存放在一个文件，把 501－1000 的字串放入到一个文件。该方法巧妙地利用汉字串频次差异大的特征，实现了快速的汉字串频次排序。

汉字串频次分段存储与排序能显著地加快汉字串频次排序速度。不妨根据表 2 和表 3 中所列数据，以对四字串频次排序为例来计算该方法的加速比。其中 116935223 是不同的四字串数量；308249 是频次在 11－100 之间的四字串数量；219895 是频次在 101－1000 之间的四字串数量；8378 是频次大于 1000 的四字串数量；log 是以 2 为底的对数。

$$Speedup=\frac{116935223\log 116935223}{3082419\log 3082419+219895\log 219895+8378\log 8378}$$

$$=\frac{3133996618}{5620531+3902356+109185}$$

$$=325$$

加速比计算结果表明，分段存储与排序确实能起到比单方面改进排序算法或数据结构更好的优化效果，并且它的执行过程很简单。特别是当语料库规模达到 GB 级别时，同一长度的汉字串存储文本文件将达到数 GB。对 GB 规模的文本文件的管理是不方便的。而分段存储则显著减小了文本文件的规模，更有利于查询与检索统计结果。本文第 6 节还将给出频次排序在整个 N-gram 串中的排序时间实测结果，以及它占整个统计时间的百分比。

6 语料分块

当语料规模达到 GB 级别时，中文文本 N-gram 频次统计在个人电脑上需花费数小时的计算时间；所需内存空间会达到数 GB；统计结果高达数十 GB。大规模中文文本 N-gram 串统计属于计算密集型、数据密集型应用，须分而治之。Cici 对大规模语料 N-gram 串频次统计与排序的操作步骤如图 3 所示。

步骤 1 将数 GB 的语料分割为多个子块；

步骤 2 利用文献[8]中提出的 N-gram 串统计算法，对每个子块进行不同长度汉字串统计，并将块内统计结果按 Unicode 编码大小排序，并存储到硬盘中；

步骤 3 将全部子块中不同长度汉字串的频次信息进行合并，合并结果保存在硬盘中；

步骤 4 对同一长度汉字串频次信息文件进行扫描，按本文第 5 节所述方法，将频次不超过 10、11－100、101－1000、大于 1000 次的汉字串分别写入到独立的文件中；

步骤 5 对频次大于 10 的汉字串按频次排序。

图 3 分块统计大规模中文语料的 N-gram 汉字串频次

步骤 2 生成的块内汉字串频次信息是根据汉字串 Unicode 编码排序存储，而不是按频次大小排序。其原因是：第一，利用文献[8]提出的算法生成 Coincidence 表时，已根据汉字串 Unicode 码值进行从小到大排序，若存储结果也按 Unicode 编码排序方式存储，则可以利用 Coincidence 表的排序结果，而无需再进行排序处理；第二，对在多个子块内出现的汉字串，步骤 3 需把该串在各个块中的频次累加求和。如果各子块的统计结果是按 Unicode 编码进行排序的，那么合并两个子块的频次统计结果则是将两个有序表合并。如果块内统计结果不是按 Unicode 编码来排序存储，例如是按频次大小来排序，则执行两个块间频次统计结果合并是属于无序表合并操作。无序表合并时间复杂性为 $O(s_1 \times s_2)$，其中 s_1 是表 1 的长度，s_2 是表 2 的长度。当 s_1 和 s_2 为数千万或数亿时，$O(s_1 \times s_2)$的时间复杂度会计算很长时间，从而导致用户不乐于接受中文文本 N-gram 统计软件。

根据 Unicode 编码次序排序汉字串是生成 Coincidence 表的关键，定理 1 分析了块大小对步骤 2 计算效率的影响。其中 m 是分块数量；S 是被统计语料所包含的总汉字数；s_i 是第 i 个语料分块包含的汉字数。特别值得指出的是：此处的排序操作不是依据频次信息进行的，而是依据每个汉字到所在“句子”的最后一个字符所生成字符串（语料中包含多少个汉字符，就有多少个汉字串）的 Unicode 编码进行排序。若是依据频次对不同汉字串排序，几乎一定会出现相同的汉字串在不同语料分块中重复出现的情况，即 $\sum_{i=1}^{m} d_i > d$，其中 d 是全体语料中出现的不同汉字串数量，d_i 是第 i 个语料分块所包含的不同汉字串数量。

定理 1 $S\log S > s_1\log s_1 + s_2\log s_2 + \cdots + s_m\log s_m$，其中 $\sum_{i=1}^{m} s_i = S$，且 $s_i \geqslant 1 (i=1,2,\cdots,m)$。

证明：由于

$$S\log S = \log S^S = \log S^{(s_1+s_2+\cdots+s_m)} = \log(S^{s_1} S^{s_2} \cdots S^{s_m}) \quad (1)$$

且

$$s_1\log s_1 + s_2\log s_2 + \cdots + s_m\log s_m = \log(s_1^{s_1} s_2^{s_2} \cdots s_m^{s_m}) \quad (2)$$

而 $S > s_i \geqslant 1$，因此

$$S^{s_1} S^{s_2} \cdots S^{s_m} > s_1^{s_1} s_2^{s_2} \cdots s_m^{s_m} \quad (3)$$

将式(3)代入式(1)和式(2)有

$$\log(S^{s_1} S^{s_2} \cdots S^{s_m}) > \log(s_1^{s_1} s_2^{s_2} \cdots s_m^{s_m})$$

因此定理 1 得证。

从定理 1 可知：子块规模越小，块内排序所需计算开销越小，但过多的分块将导致块间统计结果合并操作变慢；反之，分块越大则块内统计速度越慢，但块间合并次数减少，步骤 3 的执行速度会增快。由此可知，需要在步骤 2 和步骤 3 之间取得权衡。

我们通过大规模用户测试发现，若子块规模设置为 40MB 及以上，Cici 在物理内存较小（不超过 2GB）的计算机上运行，块内统计过程中易出现内存溢出。为了保证 Cici 能在汉语研究者较低配置的计算机上运行，推荐子块规模不大于 30MB。不妨将子块规模设置为 5MB、10MB、20MB、25MB、30MB，然后对 150MB 的语料库进行统计。记录不同子块规模设置下的整体统计时间、块内统计时间、子块统计结果合并时间、频次分段存储与排序时间，图 4 描述了该实验的结果。注意：此处一次性地统计了 1-gram、2-gram、3-gram、…、10-gram 汉字串的频次信息。

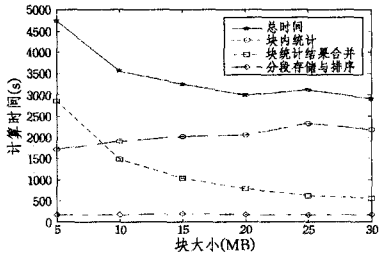

图 4　对 150MB 语料分块统计的计算时间

当子块大小从 5MB 变化到 30MB 时，块内统计所耗费时间随着子块规模增大而逐渐增多，该实验结果与定理 1 一致；子块结果合并耗费时间随着子块规模增大而迅速减少；频次信息分段存储与排序所需时间变化幅度很小；整个 N-gram 串统计操作耗费时间随着子块规模增大而减少。

语料库规模为 150MB 时，除语料块大小设置为 5MB 外，块内统计在 N-gram 串频次统计中耗费时间最多。观察程序运行记录，当子块规模为 5MB 时，150MB 的语料被分割成了 31 个子块（5MB 是最大块限制，实际上每个子块略小于 5MB，从而分成了 31 个子块）。系统需要执行 30 次子块合并操作，每次合并后都需要把合并结果写入到硬盘。硬盘读写速度远低于内存读写速度，频繁执行外存储读写操作，导致子块统计结果合并速度慢。由此可见，分块数量对总体计算时间有较大影响。

当语料规模扩大时，语料分块数会不可避免地增加，因此子块统计结果合并操作耗费的计算时间会进一步增大。为了探讨语料规模对块内统计、子块统计结果合并、频次分段存储与排序所耗费时间的影响，以下将子块大小固定设置为 20MB，维持其它软硬件环境不变，重新统计本文第 4 节采用的 100MB，200MB，300MB，400MB，500MB，600MB 语料库。记录每次统计的总体时间，以及 3 个子操作所耗费时间，计算出 3 类操作占总统计时间的百分比，如图 5 所示。

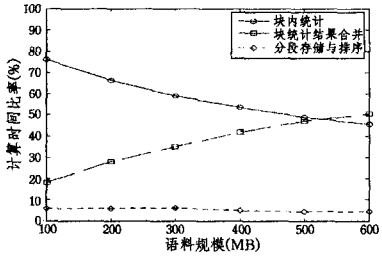

图 5　语料规模与语料统计时间百分比

从图 5 可知：随着语料库规模不断扩大，块内统计耗费时间所占百分比不断降低，子块统计结果合并操作耗费时间所占百分比不断增大。当语料库规模达到 500MB 时，二者已经十分接近；当语料规模达到 600MB 时，子块统计结果合并操作耗费时间占总时间的 50.2%，而块内统计时间仅占 45.5%。在对 1.08GB 语料的统计中，块内统计耗费 16928s，占总时间的 29.58%；子块统计结果合并操作耗费 38697s，占

总时间的 67.65%；分段存储和排序耗费 1586s，占总时间的 2.78%。这更加凸显了图 5 所示的时间比率变化趋势。由图 5 以及 1.08GB 语料库统计实验可知，对 GB 规模的语料统计，块统计结果合并是耗费时间最多的操作。

图 4 说明当语料库规模较小时，块内统计所占时间百分比高，因此要重点优化块内统计操作的相关算法；图 5 说明当语料规模达到数 GB 时，研究者须尽力优化子块统计结果合并算法。

图 6 显示出块内统计、块统计结果合并、频次信息分段存储与排序所耗费的时间都随着语料库规模的增大而增大，但三者的增长速度差异明显。统计 100MB 的语料库耗时 1693s，统计 600MB 语料库耗时 18628s。语料规模增长 6 倍，N-gram 串频次统计耗时增加 11.00 倍。完成 100MB 语料库块内统计耗时 1290s，完成 600MB 语料库块内统计耗时 8477s，增长 6.57 倍。完成 100MB 语料库子块统计结果合并操作耗时 308 秒，完成 600MB 语料库耗时 9360s，增长 30.39 倍。完成 100MB 语料库频次分段存储与排序操作耗时 95s，完成 600MB 语料库耗时 791 秒，增长 8.33 倍。

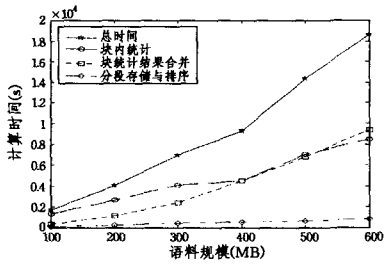

图 6　N-gram 串统计时间与语料规模的关系

根据图 6 可推测：对更大规模(大于 1GB)的语料库，根据频次信息分段存储与排序不同汉字串、按 Unicode 编码次序来存储块内统计结果的策略对提高 N-gram 串统计效率将起到更为突出的效果。

结束语　我们根据汉语本体研究人员的语言统计需求，开发了易用、高效的中文文本 N-gram 串统计和检索软件 Cici。与已有的中文文本统计软件相比，Cici 具有统计超大规模语料库以及语料自动预处理能力，特别是它允许汉语研究者自定义语料的灵活性，使它更能满足那些有着独特统计需求的汉语研究者。

未来 Cici 将朝以下 3 个方面进行改进：1)对块排序结果合并过程进行算法优化，尽量减少合并过程所需时间。2)从汉字串统计向词串统计发展。Cici v1.0 目前已具备汉语自动分词与词性标注功能，但标注正确性还有待提高。我们计划进一步提高 Cici 的自动分词、词性标准的准确率，最终实现支持 N-gram 词汇统计与检索。3)利用计算机处理器的多核提高语料统计速度。对大规模中文语料进行分块统计，很适合进行多核并行处理。目前个人电脑配置的 CPU 大多是 4 核以上。Cici V1.0 采用传统串行程序设计，操作系统只分配一个 CPU 核对其进行计算，而其它核却处于闲置状态。有效利用个人电脑的多个 CPU 核并行统计，能加快统计速度。

参 考 文 献

[1] Sun Xiao, Huang De-gan, Song Hai-yu, et al. Chinese new word identification: a latent discriminative model with global features [J]. Journal of Computer Science and Technology, 2011, 26(1): 14-24

[2] Zeng D, Wei Dong-hua, Chau M, et al. Domain-specific Chinese word segmentation using suffix tree and mutual information [J]. Information System Frontier, 2011, 13: 115-125

[3] 余一骄，刘芹. 基于语义的中文网页检索[J]. 计算机科学，2012，39(8)：89-97

[4] CCL 语料库[OL]. http://ccl.pku.edu.cn:8080/ ccl_corpus

[5] 宋柔. 对外汉语教学中的信息资源和信息处理[M]. 北京：北京大学出版社，2008

[6] 邹嘉彦，邝璃儿，路斌，等. 汉语共时语料库与追踪语料库：语料库语言学的新方向[J]. 中文信息学报，2011，25(6)：38-45

[7] 罗琭昕. 用统计的方法看“京派”与“海派”小说语言风格差异[J]. 现代语文：学术综合版，2012(4)：137-141

[8] Nagao M, Mori S. A New Method of N-gram Statistics for Large Number of n and Automatic Extraction of Words and Phrases from Large Text Data of Japanese [C]// Proceedings of the 15th International Conference on Computational Linguistics. 1994: 611-615

[9] 张民，李生，赵铁军. 大规模汉语语料库中任意 n 的 n-gram 统计算法及知识获取方法 [J]. 情报学报，1997，16(1)：27-34

[10] Yamamoto M, Churcht K W. Using Suffix Arrays to Compute Term Frequency and Document Frequency for All Substrings in a Corpus [J]. Computational Linguistics, 2001, 27(1): 1-40

[11] Banerjee S, Pedersen T. The Design, Implementation, and Use of the N-gram Statistics Package [C] // Proceedings of CICLing 2003. 2003: 370-381

[12] N-Gram Extraction Tools [OL]. http:// homepages.inf.ed.ac.uk/lzhang10/ngram.html

[13] Zhang Wei, Yang Lin-cong, Sun Xing-ming, et al. An Effective Method of Arbitrary Length N-gram Statistics for Chinese Text [J]. International Journal of Digital Content Technology and its Applications, 2011, 5(3): 143-155

[14] Jun Da. A corpus-based study of character and bigram frequencies in Chinese e-texts and its implications for Chinese language instruction [C] // Proceedings of the 4th International Conference on New Technologies in Teaching and Learning Chinese. Beijing: Tsinghua University Press, 2004: 501-511

[15] Zhang Hong, Xu Bo, Huang Tai-yi. Statistical Analysis of Chinese Language and Language, Modeling Based on Huge Text Corpora [C] // Proceedings of ICIM 2000. Berlin: Springer-Verlag: 279-286

[16] Yang S, Zhu Hong-jun, Ariel A, et al. N-gram Statistics in English and Chinese: Similarities and Differences [C] // Proceedings of International Conference on Semantic Computing 2007. Washington: IEEE Computer Society, 2007: 454-460

[17] 王惠. 词义·词长·词频—《现代汉语词典》(第 5 版)多义词计量分析[J]. 中国语文，2009(2)：120-130