ISSN 1673-9418 CODEN JKYTA8
Journal of Frontiers of Computer Science and Technology
1673-9418/2014/08(00)-0000-00
doi: 10.3778/j.issn.1673-9418.

E-mail: fcst@vip.163.com
http://www.ceaj.org
Tel: +86-10-89056056

基于 Spark 的并行图数据分析系统*

王虹旭[1+],吴斌[2], 刘旸[3]

1. 北京邮电大学 计算机学院, 北京 100876

A Parallel Graph Data Analysis System Based on Spark*

WANG Hongxu[1+], WU Bin[2], LIU Yang[3]

1. School of Computer Science, Beijing University of Posts and Telecommunications, Beijing 100876, China
+ Corresponding author: E-mail: 513196584@qq.com

WANG Hongxu, WU Bin, LIU Yang. A Parallel Graph Data Analysis System Based on Spark. Journal of Frontiers of Computer Science and Technology, 2014, 8(0): 1-000.

Abstract: A parallel data analysis system was built based on the computing platform of Spark. This system mainly aims at large-scale graph data analysis tasks, supports analysis applications of non-graph data, and integrates set of data analysis algorithms and non-graph data analysis algorithms. This paper describes the design and implementation of the system, as well as part of the parallel data analysis algorithms. Through tests of multiple scale of datasets, the system is proved to be more efficient at completing computing tasks comparing to the previous graph data mining system, and non-graph data also can be analyzed efficiently.

Key words: cloud computing; parallel algorithms; graph data analysis; data mining; social network analysis

摘　要: 提出了一种基于 Spark 计算平台的并行数据分析系统。系统以大规模图数据分析任务为主，并支持非图数据分析应用，集成了数据分析算法集合与非图数据分析算法集。详细阐述了该系统的架构设计，以及部分并行数据分析算法的设计与实现。通过多种规模的数据集测试，该系统相对于以往的图数据挖掘系统可以更高效的完成计算任务，而且也可以有效进行非图数据分析。

关键词: 云计算；并行算法；图数据分析；数据挖掘；社会网络分析

文献标志码：A　　**中图分类号**：TP311

1 引言

信息科学的高速发展,特别是物联网,云计算,社交网络,移动通信等技术的产生与普及,标志着大数据时代的到来。这些数据种类繁多,大致可以分为结构化数据和非结构化数据两类,其中图数据作为非结构化数据中最重要的数据类型,具有很强的表示能力和广泛的应用。传统应用如最优运输路线的确定、疾病爆发路径的预测、科技文献的引用关系等;新兴应用如社交网络分析、语义网络分析、生物信息网络分析等。

虽然图的应用和处理技术已经发展了很长时间,理论也日趋完善,但随着信息化时代的到来,各种信息以爆炸模式增长,导致图的规模日益增大,如何对大规模图进行高效处理,成为一个新的挑战。美国互联网中心指出,互联网的数据总量正在以每年膨胀 50%的速度增长,而且将近 90%的内容是最近两年才产生的。随着移动互联网技术的广泛应用,数以亿计的智能数字终端每时每刻都在产生新的数据。作为国内最大的网络交易平台,淘宝网已经有将近 5 亿注册用户数,每天有超过 6000 万用户的访问,在线的商品超过 8 亿件,随之而来的数以百万计的交易数据生成大约 50TB 的数据。近十几年来,随着互联网的普及和 web2.0 技术的推动,网页数量增长迅猛,据 CNNIC 统计,根据中国网络信息中心的统计,截止至 2013 年末,国内的网页总数达到了 1500 亿,相对上一年增长了 22%,而基于互联网的社交网络也后来居上,微博用户达到 3.08 亿,占网民总体数量的 54.7%,社交网站用户达到 2.75 亿,较上年增长了 12.6%,这些大规模的图数据,对图挖掘技术提出了更高的要求。

为了应对大规模图数据分析的挑战,我们提出了一种基于 Spark 的并行图数据分析平台。该平台采用 Spark 构建针对性的图计算引擎,也可以针对非图数据的并行分析任务调用通用计算引擎;采用了组件化的算法集成设计,可以方便扩展平台功能;能够存储和管理海量数据。

本文后续结构如下:第 2 章相关工作,介绍当前并行计算框架和数据分析平台的现状;第 3 章系统架构,说明各层的功能和特色;第 4 章介绍相关算法的原理与实现和系统所采用的工作流引擎以及动态组件更新技术;第 5 章,介绍系统性能测试结果以及和传统 MapReduce 平台的对比;第 6 章总结全文,说明后续研究工作。

2 相关工作

与本系统的相关工作主要可以分为并行计算模型与并行数据分析系统两方面。下面进行简单介绍。

2.1 并行计算模型

经过多年的研究与发展,研究人员提出并实现了多种并行计算模型。目前,在云计算环境中最广泛使用的是 MapReduce 模型[1],该模型将一个并行作业分成多个 map 任务和多个 reduce 任务,作业的执行分为 map 阶段和 reduce 阶段。在 map 阶段,每个 map 任务对分配给它的数据进行计算,然后将输出的数据映射到对应的 reduce 任务中;在 reduce 阶段,每个 reduce 任务对收集到的数据做进一步处理,最后得到输出结果。BSP(Bulk Synchronous Parallel)[2]模型是一种基于消息通信的并行执行模型。每个 BSP 作业由若干顺序执行的超步(super step)组成,每个超步对应一个迭代处理,并行任务按照超步组织,接受来自上一个超步的消息,执行本地计算并发送消息给下一个超步。此外,还有

Gather-Apply-Scatter（GAS）模型[3]，该模型将每个节点的程序分为三个概念性的阶段：收集（Gather），应用（Apply）和扩散（Scatter）。由 GraphLab 项目组提出，除了支持同步计算，还支持异步计算和动态计算，可以适应大多数图挖掘算法。

在实际应用中，Hadoop 在 MapReduce 模型的基础上引入了缓存和索引技术，降低了不必要的磁盘读写过程，改进了原有的任务调度模块，得到了广泛的应用。经过数年的版本更新，日前发布的 Hadoop MapReduce 框架更名为 MRv2 或者称为 Yarn。相对于以往的 Hadoop MapReduce，Yarn 将 JobTracker 的资源管理和任务调度/监控功能分为 ApplicationManager 和 ResourceManager 两个单独的组件。从而避免了 JobTracker 成为系统瓶颈，使开发人员可以根据需要定义 ApplicationManager，添加更多类型的编程模型。

BSP 模型方面，Google 推出了 Pregel[4]，在图分割，计算处理，消息通信优化，同步控制和容错管理方面提供了可行解决方案，是目前比较完善的针对大规模图处理应用的系统。Giraph[5]是一个开源的大规模图算法库，它基于 Hadoop 平台，只使用 MapReduce 模型的 map 任务，参考了 Pregel 机制，对 BSP 模型进行嵌套，开发人员可以根据需要对计算过程中的数据进行选择性的存储。Hama[6]同样是一个开源的并行处理平台，目前已经成为 Apache 的顶级开源项目（Apache Top-Level open source project）。它专注于采用 BSP 模型为机器学习和图算法领域的大规模，高迭代次数的计算应用提供处理平台。

Spark[7]是由 UC Berkeley AMPLab 开发的一个开源分布式集群计算系统，采用函数式编程语言 Scala 实现，同时支持 Java、Python 等语言接口，它克服了 MapReduce 在迭代式计算和交互式计算中表现的不足。Spark 的设计者引入了 RDD（Resilient Distributed Datasets）作为数据对象，可以很好的解决传统 MapReduce 模型在处理迭代计算时需要反复读写 HDFS 导致系统性能过低的问题。Spark 提供了多种 Transforms 和 Actions 操作，相对于 MapReduce 的 Map 函数和 Reduce 函数更为多样化，可以在很大程度上简化用户的编程。基于 Spark，伯克利实验室实现了 Shark、Bagel 等工具，以及 GraphX 图计算接口。GraphX[8]是一个运行在 Spark 框架下的图计算 API。它扩展了 Spark 中的 RDD 数据抽象，提出了 RDG（Resilient Distributed Graph）。根据图计算的一些基本运算方式，GraphX 编写了一系列基本操作，如 subgraph、joinVertices、mapReduceTriplets 等。GraphX 使用点分割方式代替常见的边分割方式对图数据进行划分，更符合大规模图数据的特点。

2.2 并行数据分析系统

主流的并行数据分析系统列举如下：Mahout 是由 Apache 基金会支持的项目之一，支持聚类，分类，模式识别，回归计算，数据降维等分析任务，但是对图算法的支持较少。PEGASUS[9]是一个基于 Hadoop 平台实现的开源大规模图数据分析系统。它的核心思想是将图挖掘的操作转换成为矩阵向量的计算过程，尽管它可以支持大规模的图数据计算，但是并非所有的图挖掘算法都能完整的转换为矩阵向量操作。Dryad[10]是由微软研究院开发的通用并行系统，它将数据挖掘操作中的计算与通信过程抽象为顶点与有向边，最后以有向图的形式进行优化和执行。BC-PDM[11]是由中国移动研究院开发的大规模数据分析系统，提供了可视化的数据挖掘操作和图挖掘操作过程。由于它以 Hadoop 平台为并

行计算引擎，对高迭代次数的分析算法性能较差，所以难以满足大规模图数据分析的性能需求。BC-BSP[12]同样是由中国移动研究院开发的项目，旨在提供一种改进过基于内存的 BSP 并行计算模型平台，它通过精心设计的数据加载与数据分发机制达到扩展平台数据处理规模的目的。PGM 是由中兴通信有限公司研发的基于 Hama 的大规模图数据分析系统，集成了丰富的图挖掘算法，由于采用了 BSP 并行计算模型，图挖掘性能较好，但是对于非图数据的挖掘支持较少。MBGM 平台为了弥补 PGM 平台对非图数据计算支持不好的缺陷，提出了采用 Hadoop 作为大规模非图数据的计算引擎，完成数据的 ETL 和构建图结构的相关计算。

3 系统架构

首先，由于现实的大数据往往不是直接以图形式呈现的，需要进行数据的清洗，提取，以及转换为图数据的操作，因此有必要在大规模图数据挖掘平台中包含非图数据的计算处理能力；其次，一个平台能够完成对图数据和非图数据的挖掘任务，有助于节约集群资源，简化平台管理；最后，采用两种并行计算框架分别完成图数据计算和非图数据计算需要开发人员了解两种程序设计模式，影响系统的二次开发能力。因此，我们提出了采用 Spark 并行计算框架作为系统的非图数据处理引擎，采用 Spark 的图计算接口 GraphX 作为系统的图数据计算引擎，解决了上述问题。系统的架构介绍如下。

如图 1 所示，本系统包含：提供并行存储和并行计算环境的并行平台层，提供图数据分析核心能力的算法层，提供平台组件更新，工作流调度算法和管理模块的逻辑层，以及提供用户使用本平台的界面层。

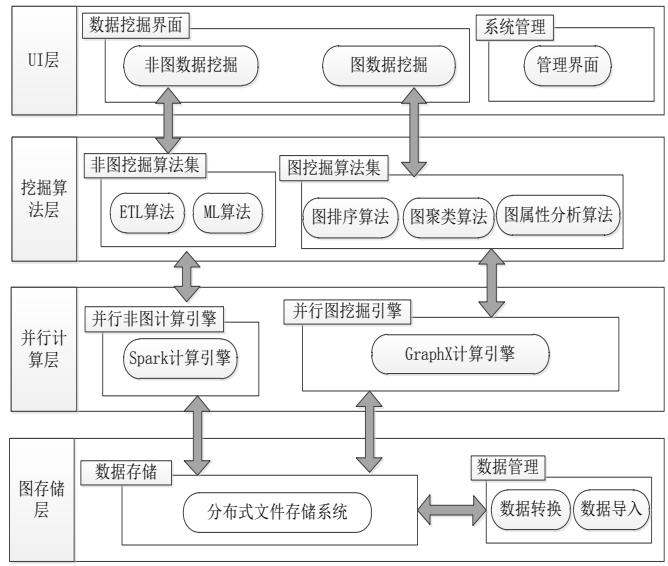

Fig.1 System architecture

图 1 系统架构图

3.1 图存储层

主要提供大规模图数据的存储以及数据获取功能。为了支持大规模数据分析，我们采用分布式文件系统存储任务。数据管理模块包括数据转换和数据导入两个子功能，负责完成非图数据和图数据之间的转换以及将外部数据导入平台。

3.2 并行计算层

该层的目的是为图查询提供执行能力支持，主要分为两个部分，并行图计算引擎和并行非图计算引擎。前者主要完成图数据挖掘的并行计算，采用 GraphX 进行实现。由于图挖掘算法的在并行执行时的特殊性，我们利用 GraphX 分别实现 BSP 并行计算模型和 GAS 并行计算模型分别执行同步并行计算和异步并行计算。并行非图计算引擎为 Spark 并行计算引擎，负责完成非图数据的分析任务，如数据转换，图数据 ETL 计算等任务。

3.3 挖掘算法层

图算法层分为图挖掘算法集和非图挖掘算法集两部分。其中非图挖掘算法集可以大致分为数据 ETL 算法和机器学习（Machine Learning，ML）算

法集两部分。前者为平台的数据预处理算法集，包括常用的数据清洗，数据转换等算法，以及针对图数据设计的预处理算法，如非图数据向图数据的格式转换，节点-邻边格式和节点-节点格式的转换，数据分隔符转换，去除网络中自环边等。我们也探索该平台架构的并行机器学习算法执行能力，目前完成了 LDA[13]（Latent Dirichlet Allocation）算法的并行实现。图挖掘算法集主要分为节点排序算法，图聚类算法和图属性分析算法，具体实现包括 PageRank 算法，LPA[14](Label Propagation Algorithm)算法，HITS[15]算法，节点亲近度算法，单源最短路径算法，图聚集系数算法[16]的并行实现。

3.4 UI 层

主要负责为用户提供系统管理，数据管理，算法调用和参数设置，以及结果显示等基本功能。通过 BS 模式的前端设计，能够便捷使用平台的各项功能，完成数据分析任务。

4 核心算法及工作流引擎介绍

由于算法较多，且篇幅有限，本章选取了非图数据挖掘算法中的 LDA 算法和图挖掘算法中的 LPA 算法进行介绍。

4.1 基于 Gibbs 采样的并行 LDA 算法

LDA，是一种主题模型，它首先由 Blei 等人于 2003 年提出，目前在文本挖掘领域包括文本主题识别、文本分类以及文本相似度计算方面都有应用。LDA 可以将文档集中每篇文档的主题按照概率分布的形式给出。同时它是一种无监督学习算法，在训练时不需要手工标注的训练集，需要的仅仅是文档集以及指定主题的数量即可。

我们采用 Gibbs 采样对 LDA 进行求解，但是它的复杂度比较高，对于规模大的数据集，效率仍然比较低。因此我们提出了一个新的并行算法。我们算法的核心思想是降低通信和同步的时间开销。但是这样我们算法的迭代次数会急剧增加，为了克服这个缺点，我们采用 Spark 实现基于 Gibbs 采样的并行 LDA 算法。

Table1 Related symbol of LDA

表 1 LDA 算法相关符号

符号	描述
X	数据集
$\boldsymbol{X_p}$	第 p 个子集
K	主题数
D	文档数目
W	单词数
$\boldsymbol{N_d}$	文档主题数矩阵
$\boldsymbol{N_w}$	单词主题数矩阵
$\boldsymbol{N_d^i}$	$\boldsymbol{N_d}$的第 i 部分
$\boldsymbol{N_w^j}$	$\boldsymbol{N_w}$的第 j 部分
$\boldsymbol{N_k}$	分配给主题 k 的单词数
$\boldsymbol{N_k^p}$	在第 p 个处理器上的$\boldsymbol{N_k}$的拷贝
$\widetilde{\boldsymbol{N_k^p}}$	采样过后$\boldsymbol{N_k}$与$\boldsymbol{N_k^p}$的差值

表 1 描述了 LDA 算法中我们使用的符号，我们的算法伪代码如下：

LDA 在 Spark 平台的实现

```
1.  FOR i<iteration DO
2.    FOR j< number of sub-datasets DO
3.      FOR each processor p in parallel DO
4.        Initialize  N~_k^p = {0}  and  N_k^p = N_k
5.        Simple  z ∈ X_p  With N_d^i , N_w^j , N_k^p
          and Update  N_d^i, N_w^j, N_k^p,  N~_k^p
6.      END FOR
7.      Shuffle  N_d^i ,  N_w^j  and update  N_k = N_k + Σ_p N~_k^p
8.    END FOR
9.  END FOR
```

为了在 Spark 平台上实现此算法，有两个问题要解决。一方面，在采样过程中，需要操作 X_p，N_d，N_w三个 RDD，但是在 RDD 超过 2 个时，Spark 性能会受到很大影响。我们采用如下方式解决这个问题：把X_p，N_d，N_w合并成为一个 RDD 进行采样，在采样完成之后，再把这个 RDD 划分回三个 RDD。另一方面，RDD 的划分方式默认是通过 hash 值来划分的，但是它不能满足我们的要求。因此我们实现了一个新的划分方法，首先我们使用指定的 *ID* 作为划分 *ID*，然后通过一个本地函数来确认 *X* 的划分。

4.2 并行 LPA 算法

LPA 算法指的是标签传播算法，这是一种具有较高效率的社团结构分析算法。

LPA 算法的基本思想是，一个节点所处的社团应该由其邻居节点所在的社团所决定。对于任意一个节点，如果它的所有邻居中，多数节点属于某一个社团，那么该节点自身也应该属于这个社团。

基于这一基本思想，LPA 算法初始时为每一个节点赋予一个标签，每次迭代每一个节点都在所有的邻居中寻找出现次数最多的那个标签作为自己新的标签。重复这一标签传播的过程，直到所有节点的标签不再变化或迭代次数已达到事先设定的值为止。当算法结束迭代之后，所有具有相同标签的节点就成为一个社团。

LPA 算法的主要步骤如下：

过程 1. 为网络中每一个节点赋予一个独一无二的初始标签，代表迭代开始前每个节点自身为一个独立的社团。

过程 2. 对于每一个节点，按照随机的顺序，遍历其所有邻居节点，选择其中出现次数最多的标签（当有多个标签出现次数同样多时，随机进行选择），将其设定为该节点新的标签。

过程 3. 重复第二步，直到所有节点的标签均不再改变或达到预先设定的迭代次数为止。

通常，LPA 算法的迭代次数不会很高，在 10 到 20 次之间即可得到较好的效果。

Spark 对于迭代运算效率更高。首先提供它提供了一套支持 DAG 图的分布式并行计算的编程框架，减少多次计算之间中间结果写到 HDFS 的开销，其次 Cache 机制支持需要反复迭代计算或者多次数据共享，减少数据读取的 IO 开销，而且它使用多线程池模型来减少 task 启动开稍，shuffle 过程中避免不必要的 sort 操作以及减少磁盘 IO 操作。所以我们在 Spark 平台上实现 LPA 算法。

4.3 工作流引擎及动态组件更新技术

系统逻辑层采用了工作流引擎结合 OSGI[17]的架构。系统的工作流引擎由我们自主开发，是在现有一些成熟框架基础上结合系统的需求开发的一套新的机制。由于大规模图数据的处理过程的稳定性与多种因素有关，如并行存储系统，网络通信情况，算法本身局限性等，所以一般的工作流引擎难以满足应用需求。本工作流引擎除提供流程执行和流程暂停之外，增加了流程运行错误自动挂起和重运行机制，解决了流程运行失败之后必须重新开始计算的问题，与并行平台的容错机制相结合，大大提高了系统运行的稳定性。另外，流程执行采用了线程池的优化机制，防止了由于运行流程过多而导致系统崩溃的问题。

在组件动态更新方面，平台结合了 OSGI 组件，可以实现动态修改、添加和删除算法组件的操作，而不需要修改系统后台代码以及重启 web 容器。OSGI 充当一个服务容器，所有的组件和操作引擎都以插件的形式注入并注册服务。另外，采用 JNDI

作为 OSGI 容器和 web 容器的桥梁，web 只需通过 JNDI 即可获取 OSGI 提供的服务 ,从而使本平台具备了真正的组件动态更新能力。

5 系统性能

5.1 图挖掘算法

本文对平台进行了性能测试，将平台部署在 9 台华为 RH2285 2U 服务器上 ,其中 8 台为工作节点，一台为主节点。配置为 Intel(R) Xeon(R) CPU E5530 @ 2.40GHz 双处理器，内存为 48G，4T 硬盘。测试数据采用按照点边比例 1:8 随机生成的图数据，分为 10 万条边，50 万条边，100 万条边，500 万条边，1000 万条边和 2000 万条边 6 组，对图分析算法分别进行测试。测试结果见表 2。需要说明的是由于生成数据为邻接表形式。

另外，进行了 1000 万条边数据的 MapReduce 算法性能对比。MapReduce 测试平台部署在 30 个节点集群上，Map 和 Reduce 个数都设置为 60 个，配置为 Intel(R) Xeon(R) CPU E5504 @ 2.00GHz 双处理器，内存为 4G，1T 硬盘。对比测试结果见图 2。

可以看出，本平台的图数据分析性能均远远高于 MapReduce 平台 ,说明本平台更加适合大规模图数据的分析和挖掘。

Table2 Tests result of algorithm performance

表 2 算法性能测试结果

数据集（万边）	特征向量中心性	入度统计	出度统计	PageRank	网络密度	LPA	Kmeans	互惠系数	HITS
10	16	13	15	17	21	14	9	19	23
50	25	16	20	18	27	26	19	28	50
100	28	22	24	22	28	38	22	39	73
500	88	65	69	40	29	75	27	562	269
1000	174	77	80	118	36	186	85	920	930
2000	653	202	228	287	63	289	156	2020	1926

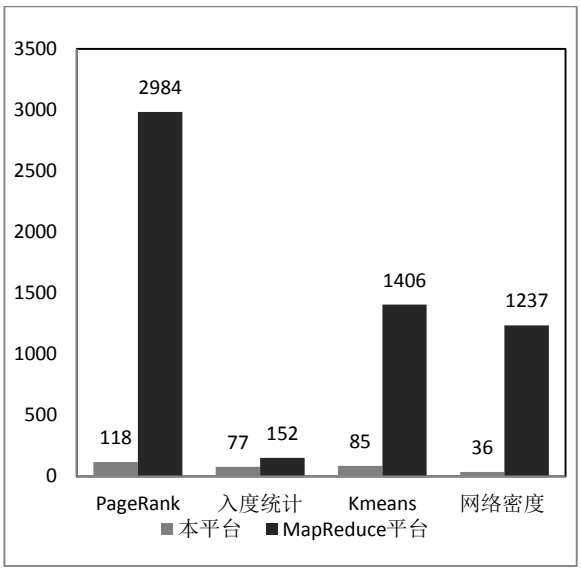

Fig.2 Performance comparison

图 2 算法性能对比

5.2 LDA 算法

我们使用来自 UCI 机器学习库[**http://archive.ics.uci.edu/ml/datasets/Bag+of+Words**]的三个文本数据集：KOS 博客文章， NIPS 论文和纽约时报的新闻文章。这三个数据集涵盖了不同的规模，内容和文件的平均长度。KOS 数据集是最小的，纽约时报的数据集是比较大的，而 NIPS 数据集则是中等大小。三个数据集的特性示于表 3。

Table3 Related parameters of datasets

表 3 数据集的相关参数

数据集	KOS	NIPS	NYT
文档总数 D	3,430	1,500	300,000
词汇表大小 W	6,909	12,419	102,660
单词总数 N	467,714	1,932,365	99,542,125

我们通过两种方式来测量性能。首先，我们比较 LDA（在单一处理器上采用标准 Gibbs 抽样）和我们的分布式算法 Spark-LDA，我们通过计算两个小的数据集 KOS 和 NIPS 的 perplexity 值[13]来完成。其次，我们使用所有这些数据集测试 Spark-LDA 的加速效果。进行试验的主机配置均是 Intel Xeon 2.00GHz 的处理器和 8GB 的内存。LDA 是在一台计算机上执行，而 Spark-LDA 是在拥有 25 台电脑的机群上进行，其中包括 1 个主节点和 24 个工作节点。

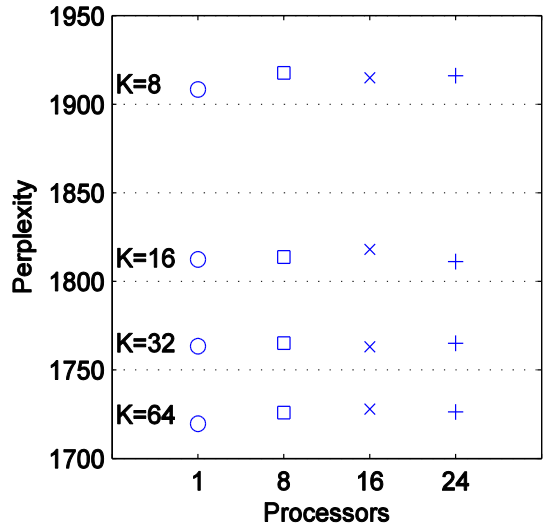

Fig.3 Tests result of KOS's perplexity

图 3 KOS 的 perplexity 值测试结果

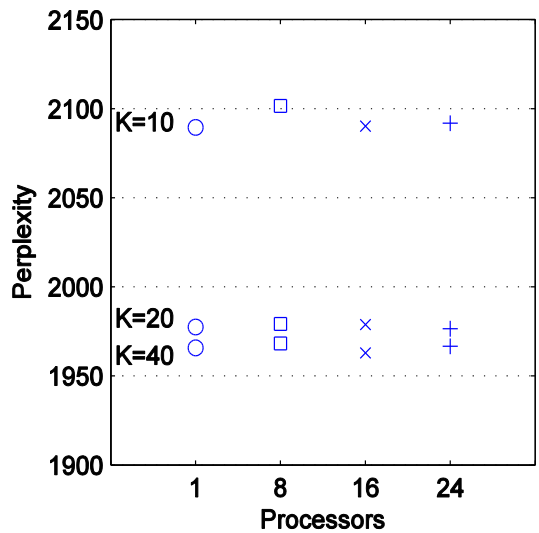

Fig.4 Tests result of NIPS's perplexity

图 4 NIPS 的 perplexity 值测试结果

图 3 和图 4 显示了对于数据集 KOS 和 NIPS 的 perplexity 值测试的结果。我们使用标准的 LDA 计

算 P=1 时的 perplexity 值，使用 Spark-LDA 计算 P=8,16,24 时的 perplexity 值。该图清楚地表明，对于不同数目的主题，LDA 和 Spark-LDA 计算的 perplexity 值之间没有显著差异。这表明，我们的算法与标准的 LDA 具有同样的预测能力。我们还使用了 Spark-LDA 计算 K=128,256 时 KOS 的 perplexity 值以及 K=80,160 时 NIPS 的 perplexity 值，实验结果给出了同样的结论。由于随着主题数量的增加，perplexity 值也在增加，这使得图变得不清晰，所以我们仅在图 5 和图 6 中展示了部分结果。

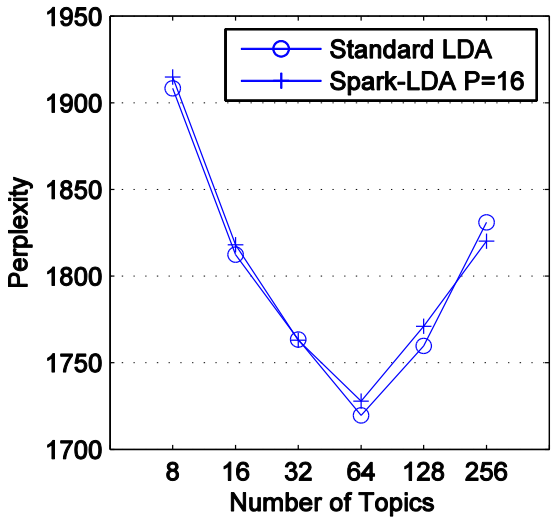

Fig.5 Changing process of KOS's perplexity

图 5 KOS 的 perplexity 值变化过程

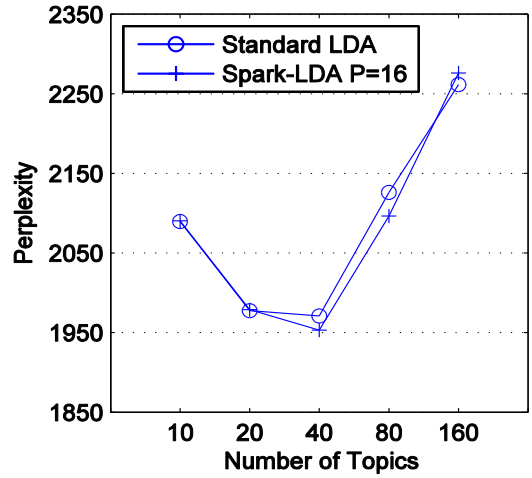

Fig.6 Changing process of NIPS's perplexity

图 6 NIPS 的 perplexity 值变化过程

我们对两个小的数据集（KOS，NIPS）和一个大的数据集（NYT）分别进行了加速实验。加速比被定义为$\frac{1}{(1-S)+S/P}$，其中 S 为抽样时间的百分比和 P 是处理器的数目。其结果如图 7 所示。该图清楚地表明，随着数据集大小的增加加速比值也在增加。这种现象反映了我们的算法对大数据集的表现较好，但对于小的数据集效果不显著。

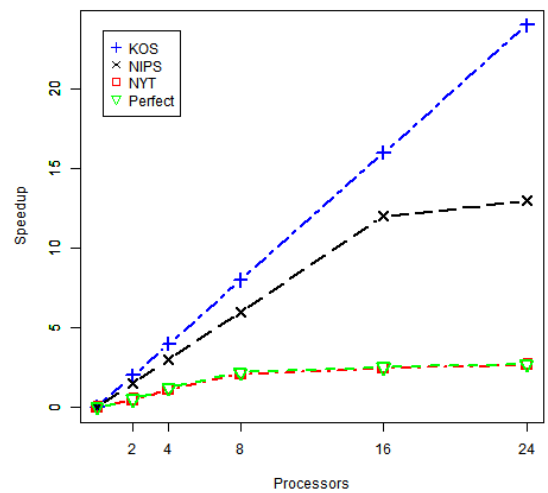

Fig.7 Changing process of speedup

图 7 加速比的变化过程

为了分析该现象的原因，我们接着测试了每次迭代中采样时间的比重，结果显示在图 8 中。随着

Spark-LDA 中处理器数目的增加，在每个分区中的单词的数目变少了，且采样的执行时间减少了。在我们的实验中，对于 KOS（K = 8，P = 24），在每次迭代中采样大约花费 40ms。但是，启动作业消耗恒定的时间，所以采样时间的比例变小。这个过程说明为什么对于小的数据集 Spark-LDA 的表现并不好。对于大型数据集，这种现象也发生，但在我们的实验中它还在可接受的范围内。

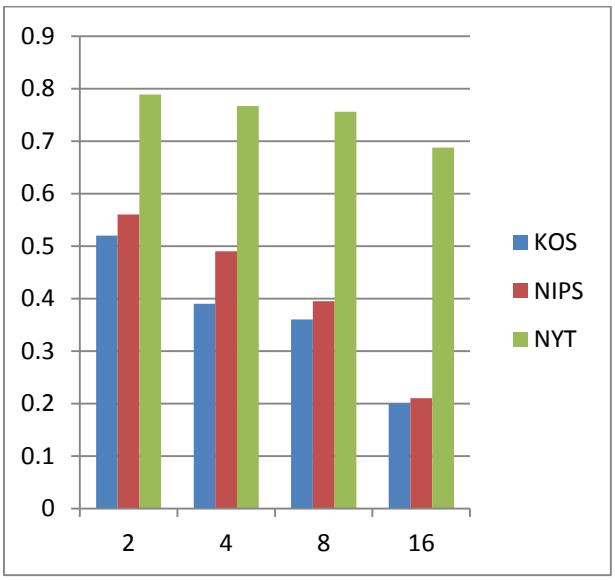

Fig.8 Ratio of the sampling time of each iteration

图 8 每次迭代采样时间的比例

6 总结

本文从系统架构，系统特色，系统性能等多个角度，全面介绍了一款基于 Spark 计算模式的并行图数据分析系统。本系统融合图排序算法，图聚类算法，图属性分析算法，涵盖了图数据分析的常用技术；针对并行计算平台只依靠平台进行容错控制的缺陷，重新设计了具有任务重跑等功能的工作流引擎，提高了系统的稳定性；由于目前图算法研究发展迅猛，还提供了动态组件更新支持，使系统可以适应不断发展的图数据分析需求，最后通过实验证明本平台在图数据分析方面性能优异，效率和性能都超过传统的 MapReduce 技术的平台，是一个前沿且注重实用的实践。下一步的工作包括：扩大图分析算法的种类，探索新的并行计算技术，扩展或优化平台的并行计算引擎，进一步提高系统性能。

References:

[1] Dean J, Ghemawat S. MapReduce: simplified data processing on large clusters[J]. Communications of the ACM, 2008, 51(1): 107-113.

[2] Gerbessiotis A V, Valiant L G. Direct bulk-synchronous parallel algorithms[J]. Journal of parallel and distributed computing, 1994, 22(2): 251-267.

[3] Low Y, Gonzalez J, Kyrola A, et al. Graphlab: A new framework for parallel machine learning[J]. arXiv preprint arXiv:1006.4990, 2010.

[4] Malewicz G, Austern M H, Bik A J C, et al. Pregel: a system for large-scale graph processing[C]//Proceedings of the 2010 ACM SIGMOD International Conference on Management of data. ACM, 2010: 135-146.

[5] Avery C. Giraph: Large-scale graph processing infrastruction on Hadoop[J]. Proceedings of Hadoop Summit. Santa Clara, USA:[sn], 2011.

[6] Seo S, Yoon E J, Kim J, et al. Hama: An efficient matrix computation with the mapreduce framework[C]//Cloud Computing Technology and Science (CloudCom), 2010 IEEE Second International Conference on. IEEE, 2010: 721-726.

[7] Zaharia M, Chowdhury M, Franklin M J, et al. Spark: cluster computing with working sets[C]//Proceedings of the 2nd USENIX conference on Hot topics in cloud computing. 2010: 10-10.

[8] Xin R S, Gonzalez J E, Franklin M J, et al. Graphx: A resilient distributed graph system on spark[C]//First International Workshop on Graph Data Management Experiences and Systems. ACM, 2013: 2.

[9] Deelman E, Singh G, Su M H, et al. Pegasus: A framework for mapping complex scientific workflows onto distributed systems[J]. Scientific Programming, 2005, 13(3): 219-237.

[10] Isard M, Budiu M, Yu Y, et al. Dryad: distributed data-parallel programs from sequential building blocks[C]//ACM SIGOPS Operating Systems Review. ACM, 2007, 41(3): 59-72.

[11] Yu L, Zheng J, Shen W C, et al. Bc-pdm: Data mining, social network analysis and text mining system based on cloud computing[C]//Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2012: 1496-1499.

[12] Bao Y, Wang Z, Gu Y, et al. BC-BSP: A BSP-based parallel iterative processing system for big data on cloud architecture[C]//Database Systems for Advanced Applications. Springer Berlin Heidelberg, 2013: 31-45.
[13] Blei D M, Ng A Y, Jordan M I. Latent dirichlet allocation[J]. the Journal of machine Learning research, 2003, 3: 993-1022.
[14] Raghavan U N, Albert R, Kumara S. Near linear time algorithm to detect community structures in large-scale networks[J]. Physical Review E, 2007, 76(3): 036106.
[15] Kleinberg J M. Authoritative sources in a hyperlinked environment[J]. Journal of the ACM (JACM), 1999, 46(5): 604-632.
[16] Barrat A, Weigt M. On the properties of small-world network models[J]. The European Physical Journal B-Condensed Matter and Complex Systems, 2000, 13(3): 547-560.
[17] Alliance O S G. Osgi service platform, release 3[M]. IOS Press, Inc., 2003.

WANG Hongxu was born in 1985. He is currently a master candidate at the School of Computer Science , Beijing University of Posts and Telecommunication, China. He is interested in data mining, complex network, and cloud computing.

王虹旭(1985–)，男，河北邢台人，北京邮电大学在读硕士研究生，主要研究领域为数据挖掘，复杂网络,云计算。

Bin Wu was born in 1969. He received his PhD degree from the Institute of Computing Technology, Chinese Academy of Science, Beijing, in 2002. He is a senior member of CCF. He is current a professor at the School of Computer Science, Beijing University of Posts and Telecommunication, China. His research interests are in data mining, complex network, and cloud computing. He has published more than 100 papers in refereed journals and conferences.

吴斌(1969–)，男，湖南长沙人，2002 年于中科院计算技术研究所获得博士学位，现为北京邮电大学教授，博士生导师，CCF 高级会员，主要研究领域为数据库，数据挖掘,复杂网络与复杂系统。发表论文 100 多篇。

LIU Yang was born in 1984. He received his BS degree in computer science from Henan University of Technology in 2007. He is currently a PhD candidate at the School of Computer Science, Beijing University of Posts and Telecommunication, China. He is interested in social network analysis, data mining, and cloud computing.

刘旸(1984–)，男，河南郑州人，北京邮电大学在读博士研究生，主要研究领域为社会网络分析，复杂网络,云计算。