基于大数据的 GitHub 开源社区开源项目量化分析

叶培根[1,2] 毛建华[1] 刘学锋[1]
(1.上海大学通信与信息工程学院 上海 200000; 2.中科院上海高等研究院智慧城市研究中心 上海 201210)

摘 要: 通过挖掘 GitHub 开源项目开发过程中的数据,基于复杂网络和机器学习算法,量化分析出软件开发团队当前进展的相关指标。利用网络爬虫抓取项目数据进行复杂网络的相关分析,结果显示 GitHub 开源社区中的开发者网络具有小世界效应,一些项目的开发者网络具有无标度网络效应,网络聚集系数随着项目的新生会出现峰值随后趋于正常,并展示了基于时间序列的网络模块度以及其他表征网络特性指标的变化趋势。量化分析结果使管理者能够动态详实的掌握开发团队的情况,合理分配资源、安排开发任务,提高软件开发效率。

关键词: 开源项目;数据挖掘; 软件开发
中图分类号: TP31; TN915.09 **文献标识码**: A **国家标准学科分类代码**: 520.6099

Quantitative analysis of open source project in GitHub community based on big data

Ye Peigen[1,2] Mao Jianhua[1] Liu Xuefeng[1]
(1. School of Communication and information engineering, Shanghai University, Shanghai 200000, China;
2. Smart City Research Center, Shanghai Advanced Research Institute, Chinese Academy of Science, Shanghai 201210, China)

Abstract: By mining the data on GitHub open source project's developing progress, based on the complex network theory and machine learning, the paper analyses the relative index in the current progress of software developing group with quantitative way. The research uses data which acquire by internet worm from open source software(OSS) in GitHub to analysis with complex networks. The results show that the network of OSS developer has a small world effect, some of the project developer network has scale-free network effect, the clustering coefficient of the network will has a peak value with the new project then tend to be normal. The paper also shows the change of network modularity and other characteristics of network based on time series. Quantitative analysis results can enable the managers to acknowledge the situation of developing group dynamically so well that they can be reasonable to allocate the resource ,to arrange the developing task and to improve its efficiency.

Keywords: open source project; data mining; software development

0 引 言

随着互联网、传感器等技术的发展,数据逐步成为具有潜在指导意义的价值源,中国也逐步步入数据时代,在智慧城市、大数据、物联网等应用中,大量的传感器节点不断地向数据中心传递所采集的数据,从而形成了海量的并不断增长的异构数据流[1]。作为获取互联网信息的虚拟传感器,能够采集获取互联网上的数据,为当前网络热点的舆情分析,互联网金融、电商推荐,商业智能等应用的实现提供了数据基础。云计算以互联网为载体,利用虚拟化等手段整合大规模分布式可配置的网络、计算、存储、数据、应用等计算资源,使其以服务的方式提供给用户,满足数据量日益增长的计算模式[2]。云计算的出现更加推进了互联网虚拟传感器的发展,为时刻增长的海量数据提供了计算保障。

随着软件产品的不断发展,开源软件逐渐成为软件发展史上的一支新型力量,特别是进入 21 世纪,开源软件在软件进化史上一枝独秀,取得了令人瞩目的成就。无论从开发效率还是开发质量方面,成功的开源软件都丝毫不亚于传统开发过程下的商业软件。而由于开源软件的免费性,使用者即开发者等特性,优秀的开源软件在市场上的占有率已大大超过同类商业软件,对全球软件产业的格局产生了重大影响[3]。这些成功的开源软件并没有遵循传统的软件开发过程的基本理论,而由于其开源的充分共享性,我们可以很容易的获得其开发过程的数据。基于数据的开源

收稿日期:2016-12

软件社区量化分析,不仅能揭示开源软件开发过程,其分析过程中所涉及到的方法更能够将当前进行的软件开发过程进行可视化的展示。

传统的软件开发活动主要是基于经验,进而制定相应的开发规划。但不同项目的不同环境(如开发人员的能力,开发团队的沟通效率等)导致不能对其差异进行准确的刻画,并且用传统的衡量软件开发进度的方法进行度量效果不佳,究其原因主要是软件开发过程具有智力密集性,人作为软件开发的核心动力始终是软件开发过程中无法回避,且灵活多变的一个关键因素。传统的定性研究对人的因素给出了很多具有深刻意义的结论,但由于缺少量化方法,这些结论只能由有经验的人员制定实施,人的主观作用较大,这些方法难以得到有效传播和普及。现在的软件开发项目较为复杂,需多个开发者共同完成。然而,开发团队的开发者过多需要协同。这种大型团队协同的需求引出了一个简单而重要的问题:相比于单个的开发者,怎样衡量一个团队中的开发者的绩效,或者换句话说,相比于单个开发者完成项目的时间,n 个开发者完成项目需要多少时间。这个问题显然是很重要的,不仅关系到项目管理,也与软件开发过程中的合理成本估算模型的开发有关。部分人想当然的认为一个团队中生产力是叠加的,即相比于单人的生产力,一个 n 人的团队会有 n 倍的生产力。然而,这种想当然的观点忽略了两个重要的因素:1)开发者间的协作会产生协同效应,其具体表现为一个团队的总生产力大于其成员单个的生产力。基于这种假设,团队平均单人生产力的增加能通过简单的增加开发人员而实现,即所谓总体大于部分之和。2)团队整体的协调与沟通是影响团队生产力的另一个消极因素,这种团队整体的增加能够影响生产力的现象在团队规模变的较大时更为明显,所以也成为了软件工程和项目管理方面的研究热点。协同问题能够导致当团队规模变大时团队单人成果输出降低,Maximilian Ringelmann,最早对团队规模增加会影响生产力进行研究。在软件工程项目管理方面,这种现象被称为“布鲁克斯定律”[4],即在延误的软件项目中增加人力会使项目更延误,Brooks 反对以增加人力的方式提高生产力,并比喻为“一个女人可以用 9 个月生出孩子,但 9 个女人无法在 1 个月内生出孩子”。

怎样提高项目绩效一直是工程研究的重要问题之一。研究针对相同经验的开发人员在生产力方面的不同差异。如果公司能够确定出生产力最高的开发人员,提升生产力靠后的开发人员,其整个团队生产力的提升将是一个巨大的优势。过去很难研究影响软件项目的生产力因素,也很难预测项目开发的成本。由于缺少对软件生产力的理解和衡量软件团队生产力的方法,使得对软件产品的成本估计经常陷入迷惑之中。而改进软件产品质量需要提高开发过程的团队生产力、精确地开发成本估算等。在软件企业,需要同时考虑生产力和质量,一个职业开发者不仅要快速的完成任务,也要保证质量。快速、高质量地完成任务是软件团队生产力的关键。

考虑到竞争的问题,软件企业需要逐步提高开发团队的生产力。因此也就必须知道哪些因素会影响生产力。开源社区的发展,为从数据层面剖析整个项目演化过程提供了一个新的思路。

随着开源软件社区的不断发展,开源社区不仅为开发人员提供了交流的平台,也潜在的积累了大量软件开发和应用数据,并随着软件的演化,数据也在不断的更新和增大。例如:版本控制系统 Git、CVS 和问题追踪系统的广泛应用积累了大量数据,记录了软件过程的演变[5],GitHub 开源社区代码库数量超过 1 600 万,如图 1 所示。

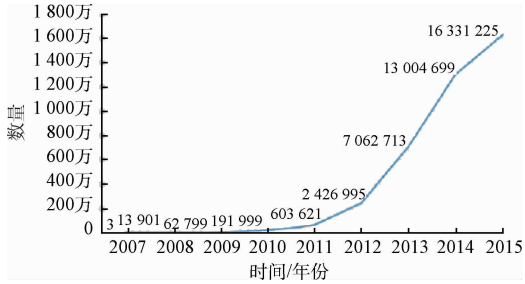

图 1 GitHub 代码库增长趋势[3]

这为软件开发活动的量化分析提供了良好的数据条件。使用开源社区软件开发过程中的数据除了可获得性之外还有其他特点,例如:1)传统软件开发过程的一手数据由于具有商业性质,数据具有保密性因而难以获取,绝大部分的研究学者使用的都是经过脱敏处理后的数据,这样可能会造成结果偏差。2)目前开源社区的研究主要是基于社会交换理论、软件开发理论等,存在一定的局限性[3]。文章试图从指导软件开发的方面对数据进行分析。

1 研究内容

研究过程主要分为两步:1)用爬虫程序在 GitHub 开源社区抓取了十几个较为成功的开源项目的数据,如:Hadoop、Apache、Tomcat、Docker、Mono、Rails、Redis 等。2)运用复杂网络分析和机器学习算法,计算出聚集系数,模块度,小世界网络图,无标度网络等相关量化因素。计算网络图中各节点参数,计算出各开发者在整个软件开发过程中的参与度,利用复杂网络的 3 个中心性指标对开发者核心地位进行排名,找到软件开发的核心人员;使管理者能够动态详实的掌握开发团队的情况,如沟通效率、协作程度,开发进度等,合理分配资源、安排开发任务,提高软件开发效率。

2 相关概念介绍

复杂网络定义:复杂网络由节点集 $V=\{v_1,v_2,v_3,\cdots,v_n\}$ 与边集 $E=\{e_1,e_2,\cdots,e_M\}$ 组成,其中若边是有向的则为有向网络,边是无向的则为无向网络,文章所用网络是无

向网络。

复杂网络 G 可定义为节点和连接这些节点的边的集合，公式表示为：

$G=(V,E)$

V 为网络中节点的集合，E 为边的集合。

复杂网络不同于随机网络和规则网络，它具有独特的统计特征，最典型的是“小世界效应”(small-world effect)和无标度特性(scale-free property)[6]. 如图 2 所示。

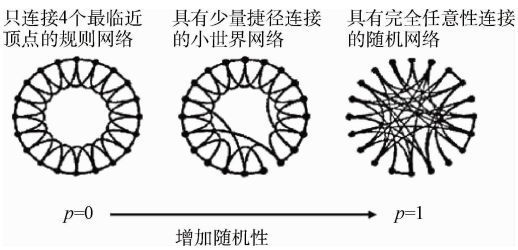

图 2　规则网络、小世界网络、随机网络关系

2.1　小世界网络

20 世纪 30 年代，匈牙利研究者 Karinthy 首次提出“小世界网络现象”的论述[7]。该研究者认为，世界上不论任何地区，任何地域，任何两个人平均可以通过一条由 5～6 位联系人组成的关系链接而互相建立关系，即著名的“六度分隔”理论。1967 年，美国哈佛大学社会心理学家斯坦利·米尔格伦(Stanley Milgram)明确提出小世界概念，米格尔伦做了一个随机人群中的信件流通实验。他以美国的威奇塔市和奥玛哈市为起点，将一些信件交给自愿的随机挑选的 296 名参加者，要求参与实验的志愿者通过自己的熟人关系网络以及亲信将实验所用信件传递到信封上指明的马萨诸塞州首府波士顿郊区收信人手里，通过解读其实验量化结果，在 296 位参与者中，有 64 位参与者成功投递到目标人信箱，而这成功投递的 64 个案例中，平均通过 5.3 次转接，这种数值上的结果与 20 世纪 30 年代匈牙利的研究者所做的假设相吻合，米尔格伦这次实验被认为是“六度分隔”理论一次成功的验证[8]。进一步阐述了在社交网络中，任意两个节点之间的“距离”是 6[4]。随着“六度分隔”理论被验证，越来越多的研究者加入到这一奇特的现象研究中来，基于此理论，1998 年，小世界网络由丁肯瓦茨(Duncan Watts)和斯蒂文·斯特罗加茨(Steven Strogatz)在《nature》上发表的《小世界网络的集体动力学》中首次被提出，他们将高集聚系数和低平均路径长度作为特征，提出了一种新的网络模型，一般就称作瓦茨-斯特罗加茨模型(WS 模型)，WS 模型由两种网络结构参数所决定：1)较短的网络平均距离。2)较大的网络聚集系数。这也是最典型的小世界网络的模型。

针对小世界网络现象的相关理论概念随着对复杂网络的研究的深入而出现。这里小世界网络中的“网络”一词从其研究目的上来说相当于离散数学中研究拓扑结构的图，即由一个网络中的顶点和它们之间所链接而成的边所构成。在网络理论中则采用另一套形式化的研究方法，用“节点”代替“顶点”，用“连结”代替“边”。

2.2　小世界网络的性质

在复杂网络中，有这样的一种特殊网络，既具有较大的平均聚集系数，同时兼有较小的平均路径，这种特殊拓扑结构的网络被称为小世界网络。由于小世界网络具有较高的集聚系数，它的结构中会有许多完全连接的“团”(团中任意两点间互联的完全任意性连接的随机网络)和类似于“团”的，只比“团”少些连接的类“团”小网络。另一方面，任两个结点大多会以至少一条短路径连接着。这是要求有小的最短路径长度平均值的结果。此外，小世界网络常连带地具有一些性质，不过这些性质并不是作为这类网络非有不可的。很典型的是这类网络常常会出现“枢纽”(与很多节点都相连的节点)。

2.3　无标度特性

无标度网络中大部分节点所连边较少，小部分数量的节点连接大量的边这一特性。这种网络的特征在普通随机网络中现象不明显，普通随机网络的度(即节点所连接边的条数)分布满足泊松分布，大多数节点的度都会在泊松分布的峰值附近，这个峰值被称为网络的特征度数[8]，大量节点的度分布在特征度数附近，而无标度网络的度是满足幂律分布，不存在特征度数，公式如下：

$$p(n) \propto n^{(-a)}, a \in R^{+} \tag{1}$$

式中：n 表示的是节点的度，a 为标度指数，无标度网络作为特殊的一种网络结构，区别于其他随机网络或规则网络，具有较强的异质性，其各节点之间的连接状况(度数)具有严重的不均匀分布性：网络中少数称之为 Hub 点的节点拥有极其多的连接，而大多数节点只有很少量的连接。少数 Hub 点对无标度网络的运行起着主导的作用。从广义上说，无标度网络的无标度性是描述大量复杂系统整体上严重不均匀分布的一种内在性质[8]。在 GitHub 开源社区开发者团队中，此现象说明团队中具有关键人物，其重要性远大于其他开发者。

2.4　复杂网络相关背景

刻画一个拓扑结构或网络可以用以下最基本的 3 种特征量来描述：度分布、聚集系数、平均最短距离。

2.4.1　度

表示与节点 v 相连接的其他节点数之和，即这个节点所拥有的边数，在有向图中，节点的度又分为出度和入度。节点的度可以反映局部中心性，关于节点的度的数学定义：在网络 $G=(V,E)$ 中，v 是节点集 V 中一个节点，即 $v \in V$，节点 v 的度 $Degree(v)$，

$$Degree(v)=\sum_{i \neq v \in V} L(i,v) \tag{2}$$

其中，$L(i,v)=\begin{cases}1, & i \text{ 和 } v \text{ 直接相连}\\ 0, & i \text{ 和 } v \text{ 不直接相连}\end{cases}$

2.4.2 度分布

一个网络的形态可以用节点的度分布来描述,从直观上来讲,度分布即度为 k 的节点的个数与整个网络所有节点的个数之比,也可以理解成度为 k 的节点在网络中所占的比重。从统计学的角度,在网络中随机选择一个顶点,这个顶点的度刚好为 k 的概率即为度分布 $p(k)$。

网络节点度分布的两种典型分布分别是泊松分布和幂律分布。

泊松分布:

$$p(k) \sim \frac{e^{-y} < k >^{-k}}{k!} \tag{3}$$

式中:$<k>$表示的是泊松分布中 p 达到峰值时的值,即所提到的特征度数,在开发者网络中的实际意义为度为峰值的节点的数目最多,并且大多数节点的度与泊松分布的峰值相近,只有少部分节点除外,这类少部分节点或者拥有较多的边,或者拥有较少的边,其取值并不徘徊在峰值附近。

幂律分布:$p(k) \sim k^{-c}$

式中:c 为标度指数,在双对数坐标的展示形式下幂律分布的形态接近为一条斜率小于 0 的直线,此特性即为无标度网络中的无标度特性。与泊松分布相区别,幂律分布不存在特征度数,即无标度的特性[8]。拥有此特征的网络中大多数节点的度都较小,只有少量节点的度数较大,真实世界中有许多关系网络满足此特征,如计算机科学文献数据库 DBLP 中的作者-论文网络[9],大多数作者所著论文有限,合作共著人数都不会太多,而少量高产作者却有很多的论文数量,一般来说,度较大的节点在整个网络中的核心程度也较大。

2.4.3 平均最短距离

平均最短距离是体现一个网络拓扑结构的重要属性,在图论以及复杂网络中都是衡量网络性能质量的的重要指标之一,从宏观上来讲,平均最短距离的大小能够表征网络性能,较大的平均最短距离表明网络性能差,平均最短距离越小网络性能越好,其数学定义如下:

$$L = \frac{1}{N(N-1)} \sum_{v \in V} \sum_{w \neq v \in V} d(v,w) \tag{4}$$

式中:w,v 是网络 $G = (V, \{Edge\})$ 中的节点,$d(v,w)$ 是节点 w 和 v 的最短路径距离。当网络不连通,$d(v,w)$ 为无穷大,此公式失效。

现实世界中的很多网络都具有节点数很多但平均最短距离较小的特点,如互联网节点、科学引文网络等。在万维网只需几步点击即可到达你想访问的任何网站链接,社会关系网络中任何两个人建立联系所需步数不超过 6。这种节点数目庞大,平均最短距离小的现象即“小世界效应”[10]。

2.4.4 聚集系数

聚集系数是表征网络中节点聚集程度的参数,聚集系数越大表明网络越具有“抱团”聚集的趋势[7],聚集系数越小表明网络聚集程度越分散,其公式如下:

$$Y(v) = \frac{2Edge(v)}{Degree(v)[Degree(v)-1]} \tag{5}$$

式中:$Edge(v)$ 指的是节点 v 与其相邻节点之间实际存在的连接边数,聚集系数是一个介于 0~1 的数值。当聚集系数等于 0 时,节点 v 与所有相邻节点互不联系(其实已无相邻节点),当聚集系数等于 1 时,整个网络中的任意两个节点都有边相连接。

3 实验设计与结果分析

首先用爬虫程序在 GitHub 开源社区抓取了十几个较为成功的开源项目的数据,如:Hadoop、Apache、Tomcat、Docker、Mono、Rails、Redis 等。未来还会在 StackOverFlow 问答社区抓取相关项目的运营数据,如:使用相关软件后参与讨论的人数、问题提问以及回答数等,进而都能够计算出该项目活跃度等指标。运用复杂网络分析和机器学习算法,计算出聚集系数、模块度、小世界网络图、无标度网络等相关量化因素[8]。计算网络图中各节点参数,计算出各开发者在整个软件开发过程中的参与度,利用复杂网络的 3 个中心性指标对开发者核心地位进行排名,找到软件开发的核心人员;使管理者能够动态详实的掌握开发团队的情况,如沟通效率、协作程度、开发进度等、从而合理分配资源、安排开发任务,提高软件开发效率。其工作流程如图 3 所示。

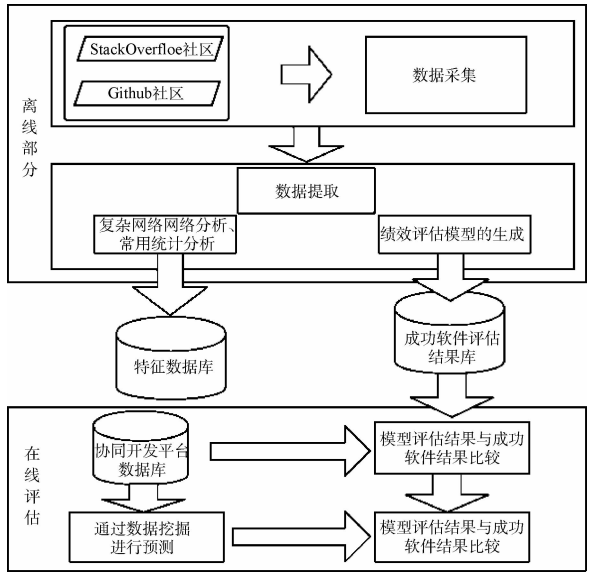

图 3 工作流程

通过 SPSS 数据分析软件,对从 GitHub 开源社区爬取的数据进行分析,部分分析结果如图 4 所示。

此示例展示了采用聚集系数并基于时间序列来描述图或网络中的节点之间结集成团的程度。从图中可以看出,一个较为成功的开源软件在开源社区初始发布时,会在短

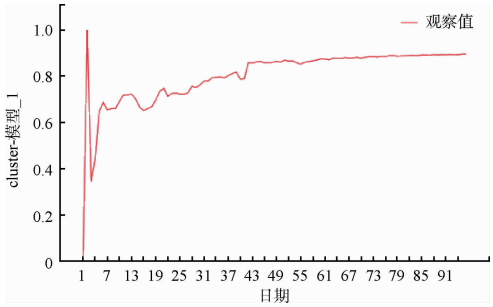

图 4　聚集系数时间序列

时间内吸引大量的参与者，在达到峰值之后，随后部分的参与者关注此项目后可能由于发现不是自己感兴趣的项目等原因，又会有较大部分参与人员流失，在经历最低值后，此项目随着时间推移其聚集程度又缓慢上升，直到到达较平衡的增长趋势。从此图的聚集系数变化趋势可大致反映出一个较成功的开源社区中项目参与者的变化形势。

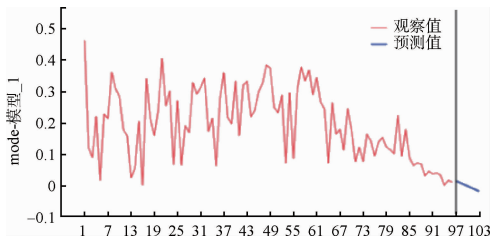

图 5　模块度时间序列

此实例展示了基于时间序列的模块度变化趋势，并对未来一段时间内进行预测。

模块度也称模块化度量值，是目前常用的一种衡量网络社区结构强度的方法

模块度值的大小主要取决于网络中结点的社区分配 C，即网络的社区划分情况，可以用来定量的衡量网络社区划分质量，其值越接近 1，表示网络划分出的社区结构的强度越强，也就是划分质量越好。因此可以通过最大化模块度来获得最优的网络社区划分。

根据此图模块度值随时间变化的趋势来看，模块度随项目的发展起伏变动，上下起伏的原因尚需研究，可能会有多种原因，如一个新的软件缺陷的出现使得若干开发者聚集在一起成为一个时间段内的小集合团体，或者某需求需要完善使得多个开发者集中此部分的开发等。随着一个项目的迭代周期进入尾声，其模块度值随时间逐渐下降。

GitHub 开发者小世界网络，此网络选取了 24 个相互间有交流的开发者，即 24 个节点，边代表两节点间有交流。在此图中有 3 个节点的度数较多，可知在此小网络中这 3 人为软件开发中的活跃者。另外，由于形成了小世界网络，此 24 节点间的关系较为紧密。

GitHub 开发者无标度网络，其典型特征是在网络中的大部分节点只和很少节点连接，而有极少的节点与非常多

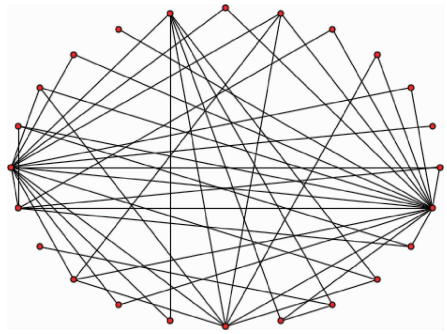

图 6　某项目团队小世界网络

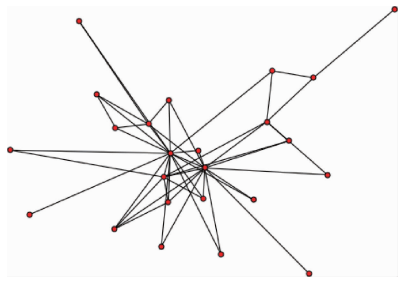

图 7　某项目团队无标度网络

的节点连接。这种关键的节点（称为“枢纽”或“集散节点”）的存在使得无尺度网络对意外故障有强大的承受能力，但面对协同性攻击时则显得脆弱[10]。现实中的许多网络都带有无尺度的特性，例如因特网、金融系统网络、社会人际网络等部分维度时间序列图：

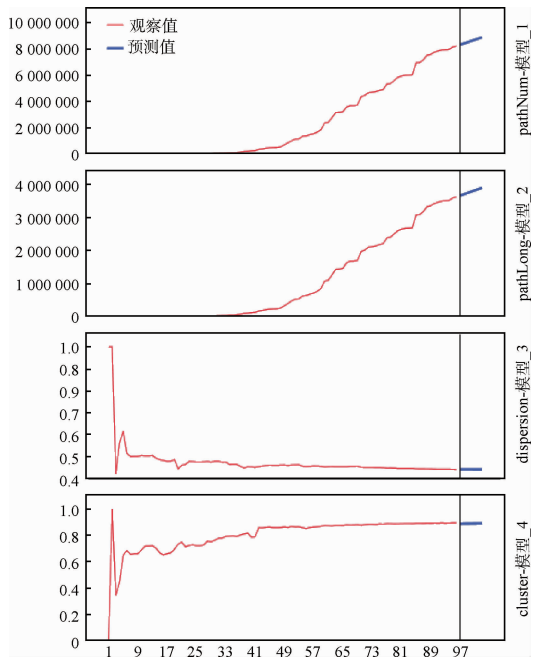

图 7　其他时间序列

上图依次是开发者网络路径数量走势及预测图，路径

长度走势及预测图,离散度走势及预测图,聚集系数走势及预测图,通过对当前开发者网络的刻画和预测,能够对开源项目的发展形成一个量化认知并对未来此开发项目的进展进行了预测。

4 结 论

通过对 GitHub 开源社区中较为成功的若干个开源项目演化过程中的数据进行分析,对开源软件的发展做了可视化的展示,并对未来一段时间的波动趋势进行预测。在实际应用中,可通过此方法对当前进行的软件开发过程的数据进行提取,分析,其可视化的量化结果可使管理层对当前软件开发进度进行知悉,并对决定下一步软件开发方向的指导提供决策支持。但当前研究仍有不少需要改进之处,如所选取的成功开源项目过少,如何定义以及选取成功的开源项目,如何能将多个项目联合进行分析等。未来,根据基于开源社区的开源项目中所遗留的数据,通过异构图[9]、复杂网络分析、机器学习等手段,能够将软件开发团队绩效进行评估并预测,真正将数据提炼为价值,更好地指导软件开发过程的进行。其次,当前数据都是通过爬虫程序由人工指定项目进行数据挖掘而获取信息,下一步工作要能够实现数据的过程感知,即在软件开发的过程中每当进入新的迭代期出现新的数据,能够立即自动的获取,实时的监控软件开发过程,充分利用数据的时效性,动态地展示当前工程进度并进行预测,为管理层提供第一时间的决策支持。

参考文献

[1] 丁治明, 刘奎恩. 海—云计算数据管理技术[J]. 金融电子化, 2013(2):36-39.

[2] 李晓辉. 云计算技术研究与应用综述[J]. 电子测量技术, 2011, 34(7):1-4.

[3] 周明辉,张伟,尹刚.开源软件的量化分析[J].中国计算机学会通讯,2016(2):24-26.

[4] SCHOLTES I, MAVRODIEV P, SCHWEITZER F. From aristotle to ringelmann: a large-scale analysis of team productivity and coordination in open source software projects [J]. Empirical Software Engineering, 2016, 21(2):642-683.

[5] 曾进群. 开源社区结构与行为及其特点研究[D]. 广州:华南理工大学, 2013.

[6] 周涛, 柏文洁, 汪秉宏,等. 复杂网络研究概述[J]. 物理, 2005, 34(1):31-36.

[7] 杜海峰 李数茧,MARCUS W F,等.小世界网络与无标度网络的社区结构研究[J]. 物理学报,2007, 56(12):6886-6893.

[8] 黄祥伟. 基于开发者合作网络的软件质量研究[D]. 武汉:中南民族大学, 2013.

[9] HAN J. Mining heterogeneous information networks[C]. ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2010:1-1.

[10] MOCKUS A. Amassing and indexing a large sample of version control systems: Towards the census of public source code history[C] International Working Conference on Mining Software Repositories, MSR. 2009:11-20.

作者简介

叶培根,工学硕士,主要研究方向为数据挖掘和软件工程中的数据智能等。

E-mail:yepg@sari.ac.cn

毛建华,工学博士,副教授,主要研究方向为移动地理信息系统、语义地理信息系统、三维地理信息系统与智慧城市、激光雷达遥感与应用。

E-mail:mjh@shu.edu.cn

刘学锋,工学博士,教授,主要研究方向为传感器网络数据服务、空间信息处理与应用、遥感图像处理与应用。

E-mail: lxf02@shu.edu.cn