



LGP-SA:分布式环境下基于模拟退火的大规模图划分算法

许金凤,董一鸿,王诗懿,何贤芒,陈华辉

(宁波大学信息科学与工程学院,浙江 宁波 315211)

摘要:针对大规模图数据的分布式计算,首先需要进行图划分。当前大规模图划分方法采用顶点转移策略来减少分区间的边割数以降低通信开销,但容易陷入局部最优,引入模拟退火的方法进行顶点转移后,极大地避免了局部最优的陷阱,也极大地防止了顶点无效转移,更好地降低了通信开销。对比实验显示,本算法划分大规模图的边割率有了极大的改进,并用 PageRank 算法验证了算法的有效性和可行性。

关键词:图划分; Giraph; 模拟退火; 大规模图; BSP

中图分类号: TP391

文献标识码: A

doi: 10.11959/j.issn.1000-0801.2016078

LGP-SA: Graph partition algorithm based on simulated annealing in large-scale graph processing

XU Jinfeng, DONG Yihong, WANG Shiyi, HE Xianmang, CHEN Huahui

College of Information Science and Engineering, Ningbo University, Ningbo 315211, China

Abstract: Distributed computing for large-scale graph data need to partition the graph firstly. The current methods of large-scale graph partitioning is to reduce the edge cut in order to lessen communication overhead by using vertex transfer strategies, but easily to fall into local optimum. Simulated annealing has a great probability to avoid the trap of local optimum and prevent vertices from invalid transfer which was introduced to transfer vertices. This method decreased communication overhead greatly. Comparative experiments show that the proposed algorithm has made a great improvement in reducing edge cut rates in large scale graph partition field. PageRank algorithm was also used to verify the effectiveness and feasibility of this method.

Key words: graph partition, Giraph, simulated annealing, large-scale graph, BSP

1 引言

近年来,随着互联网的普及,网络中用户规模的不断扩大,与之对应的网络图动辄有数十亿个顶点和上万亿条边。普通的计算机由于内存的限制无法正常处理,这给常见的图计算(如寻找连通分量、计算三角形和 PageRank)

带来了巨大挑战,解决这个问题的最好方法就是分布式计算。为了提高不同分区间的并行速度,需要使这些子图的规模均衡,而且通信开销要小^[1],因此,大规模图划分的工作就显得非常迫切和必要。已有的大规模图划分算法为了减少通信开销主要采用顶点转移策略,即根据顶点的邻居所在的分区将它转移到邻居数最多的那个分区上,达到最

收稿日期:2015-04-07;修回日期:2016-02-04

基金项目:浙江省自然科学基金资助项目(No.LY16F020003);国家自然科学基金资助项目(No.61572266)

Foundation Items: Zhejiang Provincial Natural Science Foundation of China (No.LY16F020003), The National Natural Science Foundation of China (No.61572266)



小边割的目的。然而,这种顶点转移策略容易陷入局部最优的陷阱,并没有尽量减少通信开销。

本文针对传统顶点转移策略的不足和局限性,提出了基于模拟退火的大规模图划分 (large-scale graph partition based on simulated annealing, LGP-SA) 算法。该算法在局部转移顶点的过程中引入模拟退火的思想,允许以一定的概率增加分区间的边割数,从而很大程度上避免了局部最优的陷阱,并且分析了顶点无效转移的情况,通过限制顶点转移次数来避免无效转移。本文在大规模图数据环境下实现了 LGP-SA 算法,它既保证了分区间负载均衡又考虑了图的内部结构。通过实验将 LGP-SA 算法和其他算法进行了比较,实验结果显示了 LGP-SA 算法的有效性和可行性。

2 相关工作与 Giraph 平台

2.1 图划分定义和相关工作

图划分定义:对于一个图 $G=(V,E)$, V 代表图中的顶点集, E 代表图中边集, $P=\{V_1, \dots, V_k\}$ 表示将图划分成 k 个部分,它是顶点的一个划分,其中 $V_i \neq \emptyset, V_i \cap V_j = \emptyset, \cup V_i = V, i, j=1, \dots, k, i \neq j$ 。

图划分方法需要满足两个主要原则:一是子图与子图之间相连的边数尽量小,即交互边数少;二是子图与子图的规模应当相差不大,即负载均衡。即使得:对于每个 V_i , $|V_i| \approx \frac{|V|}{k}, i=1, \dots, k$; 交互边数尽量少。

图划分算法在图数据处理系统中至关重要,然而它是一个 NP 难问题,集中式图划分算法已经研究了相当长一段时间。由于这些算法具有较高的时间复杂度,随着图数据规模的增大,它们处理时间增长得特别快,甚至超过图计算的时间。因此,目前大规模图划分算法基本都是采用简单的方法(如散列算法)或者启发式方法(如流图算法)。

从图划分目的上主要有两类,第一类是控制负载均衡,主要的算法有散列划分 (Giraph^[2]、Pregel^[3]等)、BHP 算法^[4]和 Mizan 算法^[5]等,这类算法没考虑图的内部结构,导致划分后各分区间边割数量大,因此子图间的通信开销大;第二类主要是控制分区间交互边,主要的算法有 BLP 算法^[6]、xdgp 算法^[7]、x-pergel^[8]和 gps^[9]等,这类图划分算法一般采用顶点转移策略来降低分区间的边割,也就是根据顶点的邻居所在的分区将它转移到邻居数最多的那个分区上,达到最小化边割的目的,从而减小通信开销,然而它们都是根据顶点的局部信息来进行顶点转移的,因此这种顶

点转移策略是贪婪的,容易陷入局部最优。但它们的侧重点不同^[11]: BLP 算法侧重于顶点转移中采用线性规划函数来决定转移顶点数,每次转移顶点之前都需要建立线性规划函数,时间效率不高;xdgp 侧重点在于动态图,顶点随着时间而加入或删除,对图划分的效果影响不大,因此也说明了顶点转移策略能够很好地应用在动态图上;x-pergel 侧重点在于无效转移,它是利用时间代价来减少无效转移,代价很大。gps 侧重点在于负载均衡。

这些顶点转移策略^[6,9]判断一个顶点是否进行转移由它的局部信息决定,即将该顶点转移到邻居数最多的那个分区上。这种顶点转移策略存在两个不足:一是由于每次转移仅仅考虑该顶点的邻居信息,所以会陷入局部最优,对幂律图的效果尤其不好;二是因为它是贪婪算法,所以会出现无效转移,参考文献[8]提出了共享邻接表和每次只有一个 worker 转移顶点的方法解决这个问题,共享邻接表方法耗时,而每次仅有一个 worker 转移则收敛慢,迭代时间长。如图 1 所示,按照参考文献[6-9]的顶点转移策略,图 1(a)是一个原始图状态,按照传统顶点转移策略则顶点 4 会转移出去,因为它在本地分区的邻居为 0,而在非本地分区的邻居为 2,形成图 1(b)的状态,达到稳定,但这是一个局部最优状态,而全局最优状态(即边割数最少的状态)如图 1(c)所示。按照传统的顶点转移策略,只能达到图 1(b)

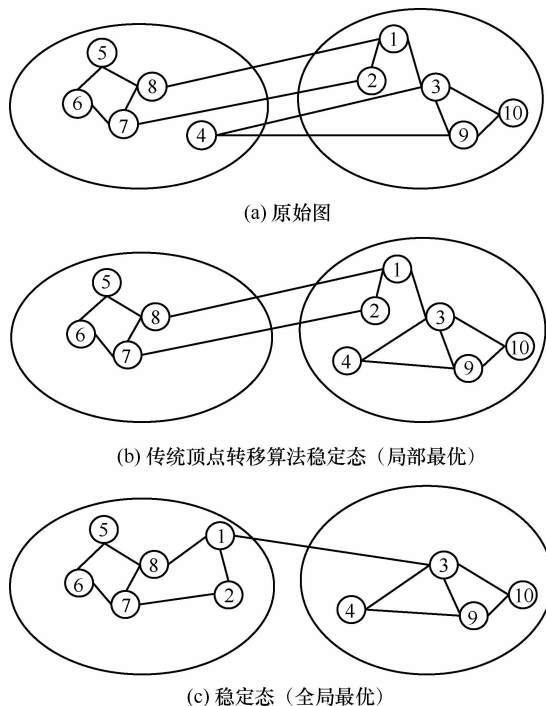


图 1 定点转移算法效果对比

的局部稳定状态,无法达到图 1(c)的全局稳定状态。而采用本文提出的基于模拟退火的大规模图划分算法将有很大概率达到图 1(c)的稳定状态。

对于传统顶点转移策略的第二个不足,无效转移分为两种,一种就是一个顶点转移数次之后又回到之前的分区中,这种无效转移称为“良性”无效转移,它肯定会发生,不可避免;第二种无效转移称为循环无效转移,即顶点相互来回转移,出现死循环,不会停止。如图 2 所示,顶点 1 和顶点 4 都会相互转移到对方的分区中,当然这种情况的顶点数不多,如图 2(b)中顶点 1 和顶点 4 又会进行转移,这样如此循环下去,不会结束。本文提出的 LGP-SA 算法主要针对局部最优进行解决,在一定程度上缓解了无效转移的产生。

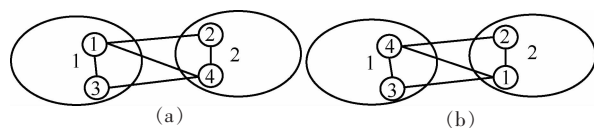
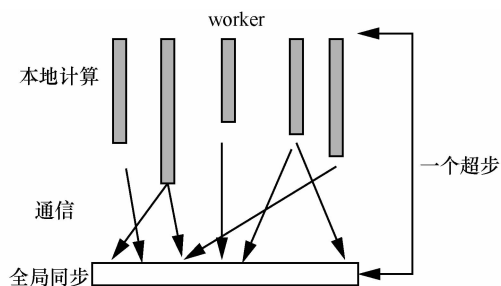


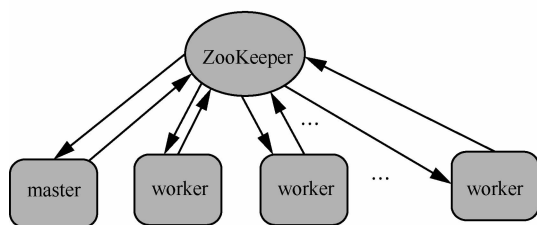
图 2 循环无效转移

2.2 云计算平台 Giraph

Giraph^[2]是 Pregel^[3]的开源,采用 BSP (bulk synchronous parallel)模型^[10],BSP 模型是 Valiant 在 1990 年提出的一种基于消息传递的并行执行模型,由一系列超步(superstep)组成,如图 3(a)所示,每个超步的最后均有个全局同步机制,它的优点就是可以避免死锁和数据竞争问题。超步与超步之间是串行执行的,而超步内部的本地计算是并行执行的,由 worker(工作机)的数目决定并行程度。每个超步包括本地计算、通信、全局同步 3 个



(a) BSP模型



(b) Giraph通信流程

图 3 Giraph 的模型结构和通信流程

阶段,是一个高扩展性的交互图形处理系统。Giraph 的基础结构由 ZooKeeper、JobTracker 和 TaskTracker 组成,其中 ZooKeeper 是用来实现同步的,而 JobTracker 就是 master 节点(主节点),TaskTracker 是 slave 节点(从节点),每个 slave 节点里面又可以包含多个 worker,如图 3(b)所示。

Giraph 中的同步是通过 ZooKeeper 来控制的, ZooKeeper 分布式服务框架是 Apache Hadoop^[11]的一个子项目,它主要是用来解决分布式应用中经常遇到的一些数据管理问题,在 Giraph 中主要实现的是选择 master 和超步结束后的同步。除了超步结束后的同步,还有很多协调 worker 和 master 的同步工作,例如检查 worker 是否健康、协调数据刚开始的分片等。master 主要是指导 worker 工作,包括为 worker 分配数据、协调同步、收集聚集值、检查 worker 是否健康;worker 主要执行本地计算、发送、接收消息。Giraph 避免了 MapReduce 模型频繁的读写磁盘和数据混乱,其独有的全局同步机制,使迭代处理更加方便灵活,更适用于大规模图处理。

3 基于模拟退火的大规模图划分算法

3.1 LGP-SA 算法描述

现有的大规模图划分算法存在分区间边割数无法达到全局优化的缺陷,没有尽量减小通信开销。本文将模拟退火算法引入顶点转移策略中,提出了在分布式环境下的 LGP-SA 算法,使边割率达到近似全局最优的效果。在顶点转移的过程中,不仅接受边割数下降的顶点转移,也以一定的概率接受导致边割数上升的顶点转移,使得分区间的总边割数能逃离局部最小,从而减少了分区间的边割,降低了算法的通信开销。并且分析了顶点无效转移的情况, LGP-SA 算法虽然一定程度地避免了顶点的无效转移,但是没有从根本上避免顶点无效转移,所以本文通过限制顶点转移次数,来进一步避免无效转移。

模拟退火算法^[12]是根据熔融金属中粒子的统计力学与复杂的组合最优化问题的求解过程的相似性提出的,是一种在一个大的搜寻空间内寻找最优解的通用概率演算法。它的搜索过程引入了随机因素,根据 Metropolis 准则以一定的概率接受一个比当前解差的解,因此有很大可能会跳出这个局部的最优解以达到全局最优解。正是由于 Metropolis 准则,模拟退火算法才能够成为全局寻优的算法。具体流程如图 4 所示。

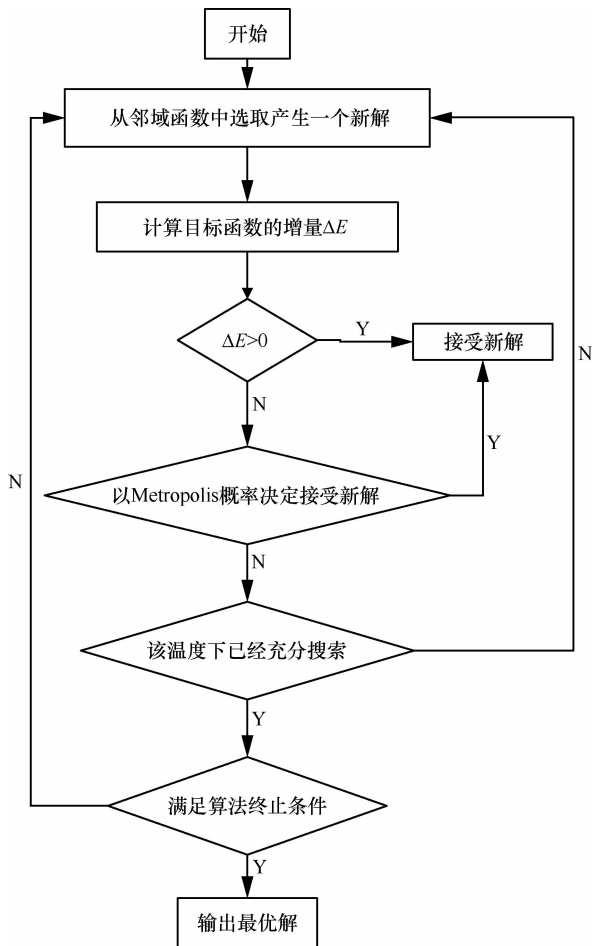


图4 模拟退火算法流程

图5是模拟退火算法例子,就像爬山一样,当前处于C的位置,根据贪婪算法,会找到局部最高点A,就会停止搜索,因为A点无论向哪个方向小幅度移动都不能得到更优的解。如果应用模拟退火算法则会以一定的概率接受到B的移动,最终就很有可能达到最高点D,跳出局部最优解A。

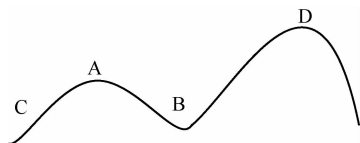


图5 模拟退火爬山例子

本文中传统顶点转移策略导致局部最优的根本原因就是顶点每次都会朝着减少边割数的方向走,没有加入随机因素,所以相对全局优化来说,性能肯定低。因此本文将模拟退火算法与传统顶点转移策略相结合,既接受边割数减少,又以一定概率接受边割数增加,由于它以一定的概率接受比当前情况要差的解,所以也一定程度地缓解了局

部最优,从而达到比较好的全局优化。将模拟退火算法应用到顶点转移策略上,主要目的就是最大化地减少分区间的边割,所以可以将边割数作为目标函数 E 的因子,将顶点在其他分区上的边数和顶点在本地分区上的边数之差作为 ΔE 。

判断一个顶点是否进行转移,首先要看这个顶点的邻居顶点信息,如果邻居顶点在非本地分区的数目大于本地分区的数目,就将其无条件转移过去,否则依据式(2)和式(3)计算它的转移概率,来决定它是否进行转移。

(1)目标函数:

$$E(S0)=n_j, E(S1)=\max(n_1, n_2, \dots, n_{j-1}, n_{j+1}, n_n) \quad (1)$$

(2)能量差:

$$\Delta E=E(S1)-E(S0)-r \quad (2)$$

(3)转移概率:

$$P=\exp\left(\frac{\text{能量差}}{\text{温度值}}\right)=\exp\left(\frac{\Delta E}{T}\right) \quad (3)$$

对于 w_j 分区的顶点 v , $S1$ 是顶点 v 转移到非本地分区, $S0$ 是顶点不进行转移, n_j 是本地分区的邻居数, n_i 是顶点在非本地分区 w_i 的邻居数,其中 $E(S1)$ 是顶点在非本地分区 w_i 中的边数(w_i 是顶点在非本地分区中具有最多邻居数的分区), $E(S0)$ 是顶点在本地分区 w_j 中的边数, r 是防止两个分区中邻居数相等时设置的一个阈值,是一个常数。

采用随机数据生成器来产生一个随机数 $\alpha \in [0,1]$,如果 $P>\alpha$ 则接受这个状态并改变当前状态,否则保持当前状态不变。随着温度变化,顶点 v 以 P 的概率接受向不比当前边割数少的方向移动,以逃离局部最优。当初始温度很高时,概率 P 就会很大。这时顶点转移是无序的,逃离局部最优的可能性最大;当温度 T 渐渐下降时,顶点的转移概率也会变小;顶点转移数量就会减少,最终当 T 很小的时候,转移概率就会接近于0,顶点只会朝着边割率减少的方向转移,此时就演变成传统顶点转移策略。开始时顶点转移相对较多,随着温度的下降,顶点逐渐趋于传统顶点转移时的状态,即不接受比较差的解。采用了模拟退火算法的顶点转移策略既可以一定程度地逃离局部最优,还能在温度下降到很小时和传统方法保持一致,接受好的顶点转移,从而控制逃离局部最优的可能。

3.2 LGP-SA 算法的实现

本文主要分析了传统顶点转移策略的缺点,即局部最优,正是由于它仅仅考虑顶点的局部信息,只要在它仅仅

考虑顶点局部信息基础上添加一个随机扰动,就能使得它能够比之前的算法更多地减少分区间的边割,当然这个扰动一定要有理论依据,然而本文算法也不一定能够达到全局最优,因为如果达到全局最优,边割率虽然降低到极限,但是其转移顶点数、收敛时间等额外开销使得全局最优的代价太大,本文算法是达到一个近似全局最优的性能。实验也证明了此方法的有效性和可行性。

(1)与传统模拟退火的区别

大规模图数据环境下,根据模拟退火算法来决定顶点是否转移,在贪心算法(传统顶点转移策略)里添加了一定的随机性,使得性能比传统顶点转移策略更优。传统的模拟退火算法参数设置以初始温度足够高、降温速度足够慢为原则^[13],来逃离局部最优。对于小规模的数据,这样的设置能够达到很好的效果,但是在大规模图数据环境下这样的参数设置会需要很多次迭代,导致转移顶点数过大,这样即使能够达到很好的性能,但是它带来的开销会远远超过其最终带来的性能。本文采用启发式模拟退火算法,以尽量使转移顶点的数量少为原则,这也正是与传统模拟退火算法不同的特点,通过设置初始温度和降温函数,使得在大规模图情况下也会达到很好的效果。其中初始温度不宜过高,如果初始温度过高,则概率过大,转移顶点数过多,导致开销大。降温参数也不宜过大,本文采用快速降温函数,使得算法既能够逃离比较差的局部最优,又能够达到比较好的效果。

(2)负载均衡

负载均衡是大数据并行计算非常重要的原则,每个超步需要等待所有的分区执行完之后才会进入下一个超步。如果分区的负载相差很大,负载多的分区执行时间长,而负载少的分区执行时间短,就会产生大量的等待和空闲时间浪费,计算效率低。LGP-SA算法的负载均衡是通过在超步内限制分区内数据量和顶点转移数目来进行控制的。每个分区的数据量为 S ,用公式表示为: $\frac{V}{\lambda \cdot k} < S < \frac{V \cdot \lambda}{k}$,其中 S 为每个分区中的顶点数范围, V 是图中所有的顶点数, k 是分区数,平衡因子 λ 。初始迭代时顶点转移数目较多,可能会出现性能瓶颈,可以设置每个分区里最大转移顶点数,如果超过这个最大转移顶点数就不再转出顶点。

(3)无效转移

无效转移的根本原因在于只考虑顶点局部信息,如果

不仅仅考虑顶点的局部信息,还增加顶点转移的随机性,也就是允许以一定概率朝着增加边割数的方向转移,会在很大程度上缓解无效转移。虽然加入随机因素之后,无效转移的顶点数会大部分减少,但是还是存在顶点无效转移。对于循环无效转移,实验设置如果该顶点转移的次数超过一个阈值 β ,就认为这个顶点是循环无效转移,以后将不再对这个顶点进行转移,有效地避免了无效转移。这个阈值如果设置过大,则与没有控制无效顶点转移差不多,效率没有什么改进,如果设置过小,一定程度上影响了模拟退火算法的性能。因此,本文采用一个折中的阈值。

引入模拟退火后的顶点转移策略的LGP-SA算法主要步骤和流程(如图6所示)如下。

步骤1 初始图:散列划分初始状态、温度初始值 T 以及降温函数 $T(m)$;

步骤2 对于 w_j 分区中的顶点 v ,根据顶点 v 的邻居信息和式(2)计算能量差;

步骤3 如果 $\Delta E > 0$,则无条件接受 $S1$ 。否则,以一定概率 P 接受 $S1$;

步骤4 控制该温度的迭代次数 L ,若小于迭代次数 L 则转步骤2,否则进行步骤5;

步骤5 利用降温函数 $T(m)$,计算下一个温度值;

步骤6 控制温度最小值,如达到温度最小值则退火过程结束,否则转步骤2。

引入模拟退火算法之前,对于图1(a)中的顶点1和顶点2,按照传统顶点转移策略是不转移的,在引入模拟退火策略之后,这两个顶点可能发生转移,随着迭

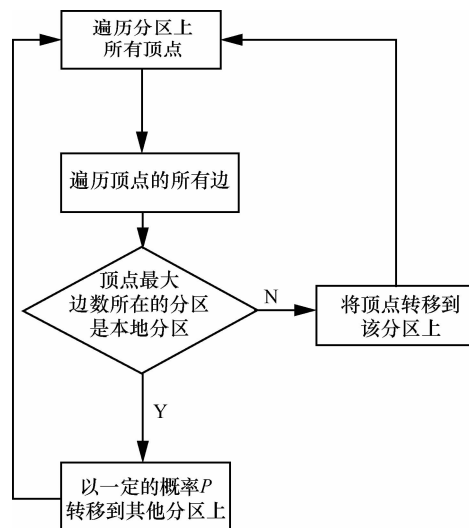


图6 一次超步流程



代次数的增加,这个转移概率会越来越小。只要有一个顶点转移出去,另一个顶点也会跟着转移出去,从而达到图 1(c)中的全局最优状态,因此有很大概率能够达到全局优化。

4 实验

4.1 实验设置

实验在 32 台 PC 组成的集群上进行,其中 1 台作为 master 节点,其余 31 台作为 slave 节点。每台 PC 配置相同:CPU 为 Intel Core i3 3.4 GHz,8 GB 内存,操作系统为 CentOS 6.5,Hadoop 版本为 0.20.203,JDK 版本为 1.6,Giraph 版本为 1.0。数据集使用见表 1,本文数据集均来自真实数据 snap,其中前两个数据集是幂律图,其他都是普通图。

将在 Giraph 上实现传统顶点转移策略的算法称为大规模图划分算法 (large-scale graph partition algorithm, LGP),也是本文的比较算法。实验首先根据实验效果和大规模图数据特点,设置了关于模拟退火的一些参数,使得边割率和运行时间达到一个比较好的临界点;其次,验证了理想情况下基于模拟退火算法的顶点转移策略的有效性,即在不控制负载均衡的情况下将 LGP-SA 算法与 LGP 算法进行比较,观察边割率变化情况;由于图处理系统仅仅降低通信开销是不够的,还需要保证负载均衡,因此本

文再通过控制负载均衡比较这两种算法对于减少边割率的影响;最后,通过运行 PageRank 算法,对比每个超步的运行时间来验证其效果。因为图规模不同,所以边割数差异也很大,导致效果不太容易观看,本文的性能是用边割率来衡量的,边割率就是本地迭代的边割数除以刚开始的边割数,将边割数归一化成为边割率 r 。

$$r = \frac{\text{最终的边割总数}}{\text{开始某种划分策略的边割总数}} \quad (4)$$

本实验刚开始采用散列划分,刚开始的边割数就是散列划分产生的边割数。

4.2 实验结果和分析

4.2.1 LGP-SA 算法参数设定

在大规模图环境下,应当以尽量减少顶点转移数目为原则,表 2 显示了不同的初始温度对最终边割率的影响。本实验采用快速降温的策略,降温函数为: $T' = T/k$, T 是上个超步的温度值, T' 是本超步的温度值, k 是一个自增参数,每次需要降温时就会增加 1。取初始温度 T 为 10、30、50、70、90 时各运行 10 次的结果。观察表 2 温度对边割率的影响,温度设置为 30 比较折中。后面实验的初始温度都取为 30。看表 2 发现温度 T 达到 30 之后,边割率虽然在减小,但是减小的程度很少,但是它带来的顶点转移概率、顶点转移数目、顶点转移时间都相对大幅增加,因此将温度设置为 30 是比较优化的选择。

表 1 数据集介绍

图	顶点数/个	边数/个	来源	类别
Ego-Facebook	4 039	88 234	Facebook 匿名社交圈	幂律图
Email-Enron	36 692	367 662	Enron 电邮通信网络	幂律图
Ca-HepTh	9 877	51 971	协作网络	非幂律图
Loc-Brightkit	58 228	214 078	基于 Brightkite 位置的在线社交网络	非幂律图
Loc-Gowalla	196 591	950 327	基于 Gowalla 位置的在线社交网络	非幂律图
as-Skitter	1 696 415	11 095 298	网络拓扑图	非幂律图

表 2 温度对边割率的影响

图	温度				
	$T=10$	$T=30$	$T=50$	$T=70$	$T=90$
Ego-Facebook	0.234 8	0.181 4	0.184 9	0.180 3	0.184 5
Email-Enron	0.351 4	0.274 5	0.264 7	0.254 5	0.243 5
Ca-HepTh	0.251 5	0.187 8	0.186 4	0.173 4	0.165 4
Loc-Brightkit	0.214 5	0.172 4	0.171 8	0.164 2	0.157 5
Loc-Gowalla	0.364 1	0.145 4	0.124 8	0.148 7	0.134 5
as-Skitter	0.406 2	0.101 2	0.097 8	0.101 5	0.109 1

本文采用快速降温函数, 需要设置温度迭代步长, 见表3。温度迭代步长 L 同样要依据减少顶点转移数的原则, 因此温度迭代步长不宜过长, 使得模拟退火既能够快速收敛, 也能够达到很好的效果。下面实验采用 $L=(1, 3, 5, 7, 9)$, $L=7$ 以上的收敛次数太长, 会导致额外开销很多, 因此都不对其进行考虑。由表3可知只要 L 达到3之后边割率就会很稳定, 后面变化并不是很明显, 因此本实验中取 $L=3$ 。

表3 温度步长对边割率的影响

图	温度迭代步长				
	$L=1$	$L=3$	$L=5$	$L=7$	$L=9$
Ego-Facebook	0.211 2	0.174 9	0.163 5	0.160 3	0.164 5
Email-Enron	0.316 5	0.274 7	0.264 5	0.274 5	0.253 5
Ca-HepTh	0.221 5	0.176 4	0.184 5	0.173 4	0.164 8
Loc-Brightkit	0.214 5	0.161 8	0.157 8	0.164 2	0.163 5
Loc-Gowalla	0.234 1	0.144 8	0.148 8	0.148 7	0.134 5
as-Skitter	0.186 2	0.107 8	0.114 5	0.094 1	0.099 1

LGP-SA 算法与 LGP 算法对不同图的最终出现无效转移顶点数目变化如图7所示, 由图7可知 LGP-SA 算法一定程度地改进了无效顶点转移的数目, 因为造成无效顶点转移现象的根本原因是转移过程中只考虑顶点的局部信息, 而 LGP-SA 算法加入了一定的随机因素, 致使顶点无效转移数目降低。本文的 LGP-SA 算法不仅很大程度减少了无效顶点转移数, 还通过控制阈值(LGP-SA- β)的方式进一步减少无效顶点转移数。

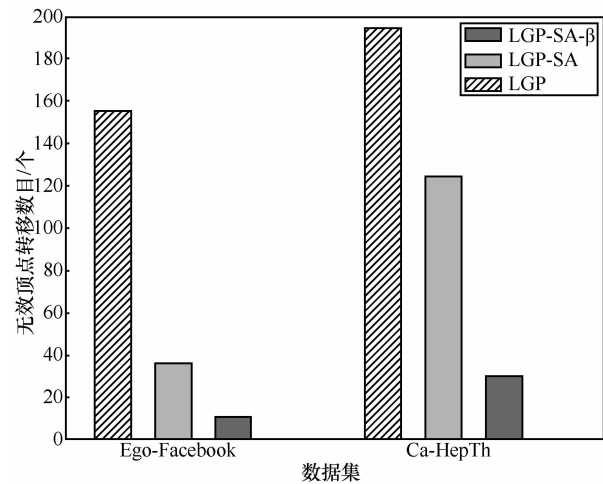


图7 不同图的最终无效顶点转移数目变化对比

4.2.2 不考虑负载均衡时的边割率

LGP-SA 算法与传统转移顶点算法的最终目的都是降

低分区间的边割数, 本文首先将 LGP-SA 算法与传统转移顶点算法相比较, 通过比较它们减少的边割数就可以知道算法的有效性。在不控制负载均衡的情况下, 观察两种算法的效果差异。如图8所示, 前两列显示的幂律图, 其他列都是普通图, 由图8可知 LGP-SA 算法对于这两类图的效果都很明显, LGP-SA 算法比 LGP 算法好, 特别是非幂律图。而幂律图的效果和其他图相比较要差一点。因为幂律图本身就很倾斜, 所以不控制复杂均衡的情况下, 幂律图的分区负载差异会很大, 实验中也正是如此。

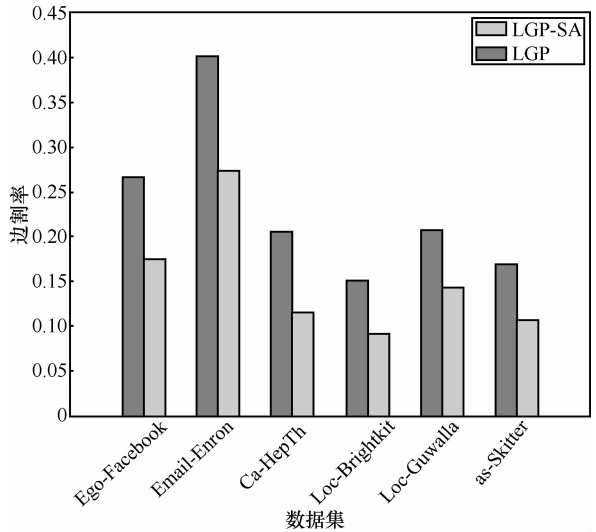


图8 边割率对比

图9表示 LGP-SA 算法与传统 LGP 算法每个超步的边割率变化。由于 LGP-SA 算法在传统 LGP 算法不转移顶点的时候, 以一定的概率转移该顶点, 而一开始的时候顶点的转移概率高, 所以开始的时候边割可能会增加, 这也正是模拟退火算法逃离局部最优的必经之路, 之后顶点转

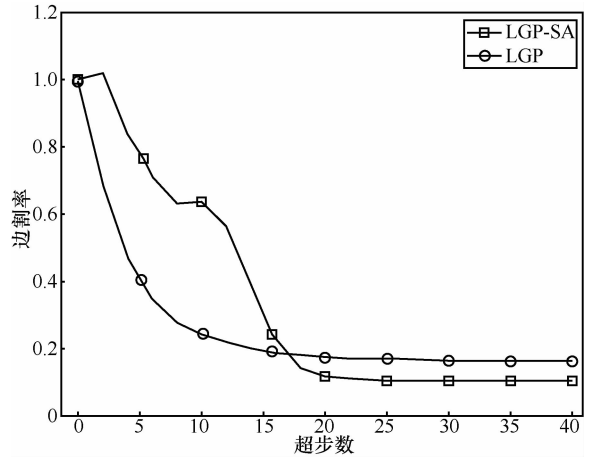


图9 每个超步边割率的变化比较



移概率随着迭代次数增加而降低,最终和传统算法一致。但是它最终下降的最低点比LGP算法更低。这就是刚开始以概率转移的好处。

4.2.3 负载均衡下的边割率

图10和图11用的数据都是as-Skitter。虽然不控制负载均衡情况下LGP-SA算法的效果很好,但是图划分的原则之一就是控制负载均衡,所以接下来的实验都是在控制负载均衡的条件下进行的。对于一个图分别对它进行控制平衡与不控制平衡的比较,发现控制平衡因子的算法效果没有不控制平衡因子效果好,因为控制负载均衡就一定地限制了转移顶点的自由,因此效果会变差,但是LGP-SA算法仍然比LGP算法好。图11是每个超步的边割率变化情况,图9和图11进行比较得知,控制负载均衡的边割率收敛得慢一点。

图12是从不同规模的图来验证算法有效性。其中Ego-Facebook和Email-Enron都是幂律图,而其他的都是非

幂律图,由图12可以看出幂律的平均边割减少得没有非幂律图多。因为幂律图本身就是很倾斜的,不管图怎么划分,它都会导致分区间的边割数较多。而其他图算法比较都差不多,都比传统方法好。

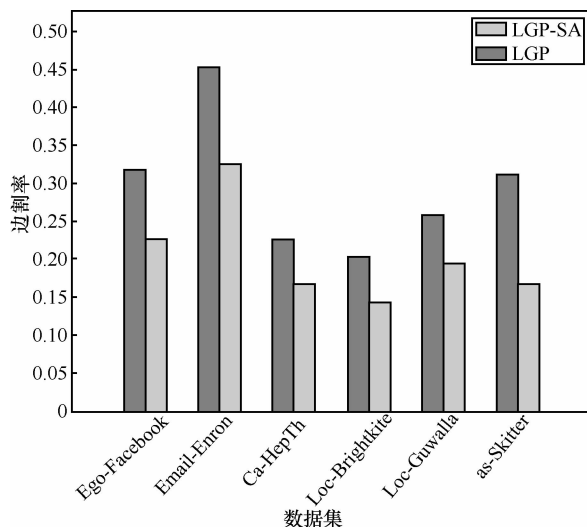


图12 不同图的边割率

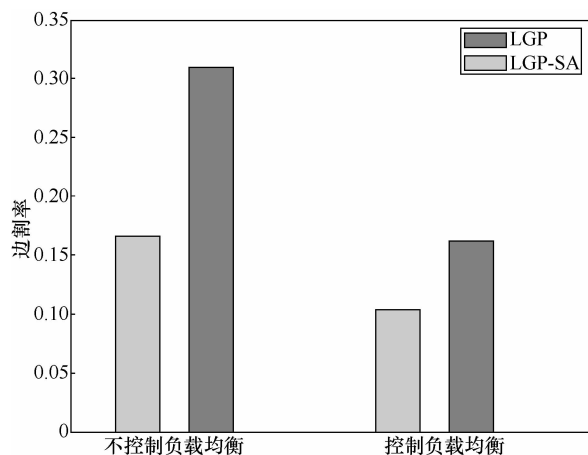


图10 控制平衡因子对边割率的变化情况比较

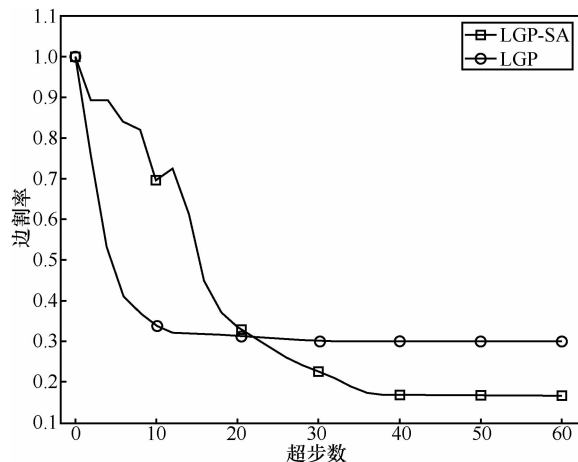


图11 总边割率变化

4.2.4 PageRank 算法的运行结果

判断系统性能最有说服力的是运行时间,将PageRank算法运行到该系统上,来证明算法的可行性。图13是运行PageRank算法的超步运行时间图,初始划分是散列划分。刚开始的时候由于LGP-SA方法转移的顶点数比传统方法多,所以响应时间比LGP算法时间长,随着温度的降低,顶点转移概率降低,边割数也会渐渐趋于稳定状态,通信开销也会趋于稳定。大概在超步第50个的时候LGP算法就会追平散列划分的时间。在第60个超步的时候LGP-SA算法就会追平LGP算法的运行时间。随着迭代次数的增加,算法的效果会越来越好。

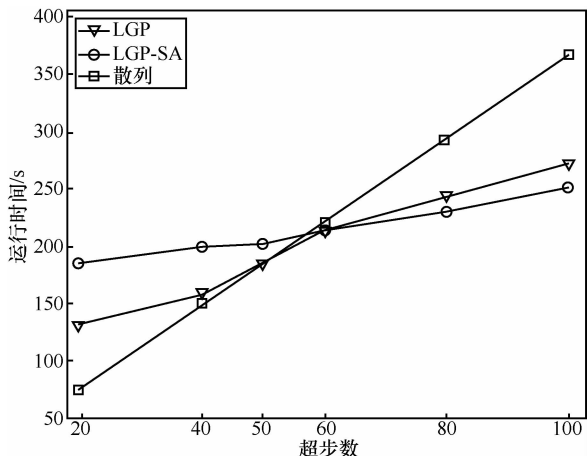


图13 PageRank 每个超步的运行时间

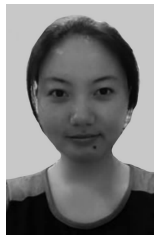
5 结束语

随着大规模图数据的出现,图划分也极具挑战性,传统的集中式划分算法已经处理不了大规模图数据,因此涌现了很多分布式并行框架,如Pregel、Giraph、GraphLab^[14,15]、Spark等。本文首先对Pregel的开源框架Giraph进行了分析,针对传统分布式图划分算法中采用转移顶点来降低分区间的边割,分析了这种方法的不足,由此提出了基于模拟退火的LGP-SA算法,并通过实验验证了其有效性和可行性。由于强制性控制负载均衡会使LGP-SA算法效率效果变差,希望接下来对这方面进行优化改进。

参考文献:

- [1] 许金凤,董一鸿,王诗懿,等.大规模图数据划分算法综述[J]. 电信科学,2014,30(7):100-106.
XU J F, DONG Y H, WANG S Y, et al. Summary of large-scale graph partitioning algorithms [J]. Telecommunications Science, 2014, 30(7):100-106.
- [2] CHING A. Giraph: Large-scale graph processing infrastructure on Hadoop [C] // Hadoop Summit 2011, Santa Clara, CA, USA. [S.l.:s.n.], 2011.
- [3] MALEWICZ G, AUSTEN M H, BIK A J C, et al. Pregel: a system for large-scale graph processing[C]//2010 ACM SIGMOD International Conference on Management of data, June 6-10, 2010, Indianapolis, Indiana, USA. New York: ACM Press, 2010: 135-146.
- [4] 周爽,鲍玉斌,王志刚,等. BHP: 面向BSP模型的负载均衡Hash图数据划分[J]. 计算机科学与探索, 2014, 8(1): 40-50.
ZHOU S, BAO Y B, WANG Z G, et al. BHP:BSP model oriented Hash graph data partition with load balancing [J]. Journal of Frontiers of Computer Science & Technology, 2014, 8(1): 40-50.
- [5] KHAYYAT Z, AWARA K, ALONAZI A, et al. Mizan: a system for dynamic load balancing in large-scale graph processing[C]// The 8th ACM European Conference on Computer Systems, April 15-17, 2013, Prague, Czech. New York: ACM Press, 2013: 169-182.
- [6] UGANDER J, BACKSTROM L. Balanced label propagation for partitioning massive graphs [C]// The 6th ACM International Conference on Web Search and Data Mining, February 6-8, 2013, Rome, Italy. New York: ACM Press, 2013: 507-516.
- [7] VAQUERO L, CUADRADO F, LOGOTHETIS D, et al. xDGP: a dynamic graph processing system with adaptive partitioning[J]. Eprint Arxiv, 2013(9).
- [8] BAO N T, SUZUMURA T. Towards highly scalable pregel-based graph processing platform with x10 [C]//The 22nd International Conference on World Wide Web Companion, May 13-17, 2013, Rio de Janeiro, Brazil. Geneva: International World Wide Web Conferences Steering Committee, 2013: 501-508.
- [9] SALIHOGLU S, WIDOM J. GPS: A graph processing system[C] //The 25th International Conference on Scientific and Statistical Database Management, July 29-31, 2013, Baltimore, Maryland, USA. New York: ACM Press, 2013.
- [10] VALIANT L G. A bridging model for parallel computation[J]. Communications of the ACM, 1990, 33(8): 103-111.
- [11] WHITE T. Hadoop: The Definitive Guide[M]. Cambridge: O'Reilly Media, Inc., 2012.
- [12] RUTENBAR R. Simulated annealing algorithms: an overview[J]. IEEE Circnit and Devices Magazine,1989: 19-26.
- [13] KUMAR V. Algorithm for constraint satisfaction problem: a survey[J]. AI Magazine, 1992,13(1): 32-44.
- [14] LOW Y, GONZALEZ J, KYROLA A, et al. GraphLab: a new framework for parallel machine learning[C] // The 26th Conference on Uncertainty in Artificial Intelligence (UAI'10), Jul 8-11, 2010, Catalina Island, California, USA. [S.l.:s.n.], 2010: 340-349.
- [15] LOW Y, BICKSON D, GONZALEZ J, et al. Distributed GraphLab: a framework for machine learning and data mining in the cloud[J]. Proceedings of the VLDB Endowment, 2012, 5(8): 716-727.

[作者简介]



许金凤(1990-),女,宁波大学硕士生,主要研究方向为大数据、数据挖掘。

董一鸿(1969-),男,博士,宁波大学教授,主要研究方向为大数据、数据挖掘和人工智能。

王诗懿(1989-),女,宁波大学硕士生,主要研究方向为大数据、数据挖掘。

何贤芒(1981-),男,宁波大学讲师,主要研究方向为大数据、数据挖掘、隐私保护。

陈华辉(1964-),男,博士,宁波大学教授,主要研究方向为数据流与数据挖掘。