Pagerank 算法

实

验

报

告

离散实验报告

一，实验目的

（1）理解 Google 的 pregel 算法的运行方式；

（2）弄清楚 Google 的 pregel 算法与图论之间的关系；

（3）明白 Google 的 pregle 算法的图论知识的运用。

二，实验过程

通过使用互联网工具，搜集和 Google 的 pregel 算法有关的资料和文献以及一些人的关于这个算法发表的博客，然后对这些资料进行阅读和分析，同时根据资料中提及的此算法在实际生活中的应用，在利用互联网工具对这些应用进行了解；整个过程中都在了解和思考图论在 pregel 算法中是如何被运用的，以及在算法的实际应用中，实际问题又是如何被抽象成图，进行数据和信息的传递和处理。

三，实验结果

Google 的 Pregel 是一种分布式大型图处理计算框架。这个计算框架采用了 BSP 模式，即“计算”-“通信”-“同步”模式；将程序用一系列的迭代来描述，每一次迭代中，每一个顶点都能接受来自上一次迭代的信息，并将这些信息传给下一个顶点，并在此过程中修改自身的信息状态，以该顶点为出边的状态信息，或者改变整个图的拓扑结构，隐式的同步性使它对程序的确认变得简单易行（分布式相关的细节被一组抽象的 API 给隐藏）。

Pregel 计算框架由一系列的迭代组成，每一次迭代，计算框架都会调用户针对每一个顶点自定义的函数，这个过程是并行的（同步性的特点保证了实现算法时推算程序执行的语义变得简单）。用户自定义函数定义了一个顶点 V 和一次迭代 S 中需要执行的操作，该函数可以将前一轮迭代（S-1）发送给 V 的信息读入，并将该信息通过下一轮迭代（S+1）发送给另外的顶点，并在此过程中修改顶点 V 以及其出边的状态，消息通常通过顶点的出边发送，但一个消息可能会被发送到许多已知身份的特定顶点上去，可使用户只需要关注其本身的执行逻辑，分别独立的执行各自的处理过程，而其剩余的大量的复杂的事情都将由系统处理，此框架适合分布式化的实现，没有限制每次迭代的顺序，所有通信过程仅限于第 S 次迭代和第 S+1 次迭代之间。

Pregel 算法的优点是设计高效，可拓展和足够容错，并有在上千台计算节点的集群中得以实现的表现力很强，容易编程且很好的解决了为特定的图制定相应的式分布框架的重复性；基于已有的分布式平台进行操作的局限性；使用单机的图算法库对图本身的规模有限制，使用已有的并行图计算系统，对容错等大规模分布式计算系统中非常重要的一些方面支持不好等其他计算框架难于解决的问题。

Pregel 和图论之间的关系：在 pregel 的计算过程中，输入的是一个有向图，它以邻接表的方式存储，每一个顶点都有一个身份号作为其身份唯一标识，该有向图的每一个顶点都有一个边来描述其属性，该属性可以被修改，其初始值由用户定义，每一条有向边和其原顶点关联，并且也拥有一些由用户定义的属性和值，并且还记录了其目的顶点的身份。一个典型的 pregel 的计算过程：读取输入数据，并初始化该图，然后运行一系列迭代，每一次迭代都在全局的角度上独立运行，直到所有计算结束，输出结果。在每一次迭代中，顶点的计算时并行的，每一次执行用户定义的同一个函数，顶点可以修改其自身的信息状态，或者修改以它为顶点的出边的信息状态，或者改变整个图的拓扑结构，从前序迭代中接受信息，并传给后续迭代，对于边，则没有相应的运算运行其上。算法结束的关键是所有的顶点是否都已处于被计算过的状态；在迭代开始之前，图中所有顶点都被置于“活跃”状态，而每一个处于“活跃”的顶点都会在计算的执行中在某一次的迭代中被计算，顶点通过其自身的属性将其自身设置成“停止”，表示它已经不再“活跃”，这表示该顶点没有下一步计算需要进行，除非被额外触发其他运算，而 pregel 框架将不会在接下来的迭代中计算该顶点，除非该顶点收到一个其他迭代传送的消息。如果顶点接受到消息，该消息将该顶点重新置“活跃”，那么在随后的计算中在此“停止”其自身。整个计算在所有顶点都达到“非活跃”状态，并且没有消息在传送的时候宣告结束。整个 pregel 程序输出的是所有顶点输出的集合。一般来说 pregel 程序的输出是跟输入时同构的有向图，但是并非一定是这样，因为在计算中可以对边进行添加和删除。Pregel 将顶点和边在本地机器进行运算，而仅仅利用网络来传输信息，而不是传输数据。

Pregel 利用编程语言所写的应用程序接口中最重要的是用户编写的一种函

数被允许查询当前顶点的信息，其边的信息，并发送信息到其他的顶点，当然也可以得到当前顶点的值或者修改当前顶点的值。同时还可以通过对边定义方法来获得和建立以该顶点为起点的边对应的值。这种状态的修改时立时生效的，由于这种可见性是仅限于修改顶点的时候，所以在对不同的顶点进行并行的数据访问时不存在数据的竞争关系。顶点的值和其对应的边的值是仅有的在迭代之间不断变换。将由计算框架管理的图状态限制在一个单一的顶点值或边值的这种做法，可以使得估算计算轮次，图分布化以及错误恢复变得简单。

下面以一个具体的计算任务来作为 Pregel 图计算模型的实例进行介绍，这个任务要求将图中节点的最大值传播给图中所有的其他节点，下图是其示意图，图中的实线箭头表明了图的链接关系，而图中节点内的数值代表了节点的当前数值，图中虚线代表了不同次迭代之间的消息传递关系，同时，带有斜纹标记的图节点是不活跃节点。

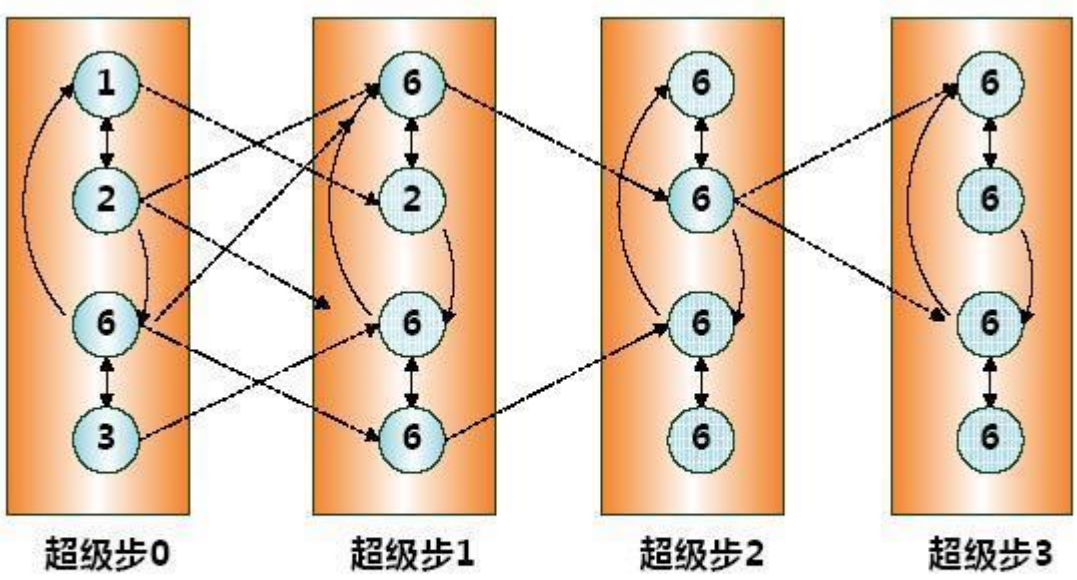

图 14-14　Pregel 传播最大值

从图中可以看出，数值 6 是图中的最大值，在迭代开始之前，所有的节点都是活跃的，系统执行用户定义的函数 *F*(*vertex*)：节点将自身的数值通过链接关系传播出去，接收到消息的节点选择其中的最大值，并和自身的数值进行比较，如果比自身的数值大，则更新为新的数值，如果不比自身的数值大，则转为不活跃状

态。具体运行过程如下：

（1）第 0 个超级步中，每个节点都将自身的数值通过链接传播出去；

（2）系统进入第 1 个超级步，执行 $F(vertex)$函数，第一行和第四行的节点因为接收到了比自身数值大的数值，所以更新为新的数值 6。第二行和第三行的节点没有接收到比自身数值大的数，所以转为不活跃状态。在执行完函数后，处于活跃状态的节点再次发出消息；

（3）系统进入第 2 个超级步，第二行节点本来处于不活跃状态，因为接收到新消息，所以更新数值到 6，重新处于活跃状态，而其他节点都进入了不活跃状态；

（4）Pregel 进入第 3 个超级步，所有的节点处于不活跃状态，所以计算任务结束，这样就完成了整个任务，最大数值通过 4 个超级步传递给图中所有其他的节点。

Pregel 主要应用于最短路径，图遍历，pagerank 计算，二部图最大匹配问题等；以 pagerank 计算为例来说：PageRank 是 Google 用于用来标识网页的等级（或网页重要性）的一种方法，是 Google 用来衡量一个网站的好坏的重要标准，PageRank 既考虑到入链数量对网页的影响，也参考了网页质量因素，两者相结合获得了更好的网页重要性评价标准。对于某个互联网网页 A 来说，该网页 PageRank 的计算首先基于以下两个基本假设：

（1）数量假设：在网络图模型中，如果一个页面节点接收到的其他网页指向的入链数量越多，那么这个页面越重要。

（2）质量假设：指向页面 A 的入链质量不同，质量高的页面会通过链接向其他页面传递更多的权重。所以越是质量高的页面指向页面 A，则页面 A 越重要。

利用以上两个假设，在开始计算之前 PageRank 算法刚赋予每个网页相同的重要性得分，通过迭代递归计算来更新每个页面节点的 PageRank 得分，直到得分稳定为止。 PageRank 计算得出的结果是网页的重要性评价，这和用户输入的查询是没有任何关系的，即算法是主题无关的。

PageRank 的计算步骤如下：

（1）在初始阶段：网页通过链接关系构建起网络图，每个页面设置相同的 PageRank 值，通过若干轮的计算，会得到每个页面所获得的最终 PageRank 值。随着每一轮的计算进行，网页当前的 PageRank 值会不断得到更新。

（2）在一轮中更新页面 PageRank 得分的计算方法：在一轮更新页面 PageRank 得分的计算中，每个页面将其当前的 PageRank 值平均分配到本页面包含的出链上，这样每个链接即获得了相应的权值。而每个页面将所有指向本页面的入链所传入的权值求和，即可得到新的 PageRank 得分。当每个页面都获得了更新后的 PageRank 值，就完成了一轮 PageRank 计算。

再拿 pregel 利用随机匹配思想解决二部图匹配（给定一个二分图 G，M 为 G 边集的一个子集，如果 M 满足当中的任意两条边都不依附于同一个顶点，则称 M 是一个匹配）问题为例：每个图节点维护一个二元组：（'Left/Right'，匹配节点 ID），'Left/Right'指明节点是二部图中的左端节点还是右端节点，以此作为身份识别标记。二元组的另一维记载匹配上的节点 ID。

算法运行经过以下四个阶段。

阶段一：对于二部图中左端尚未匹配的节点，向其邻接节点发出消息，要求进行匹配，之后转入非活跃状态。

阶段二：对于二部图中右端尚未匹配的节点，从接收到的请求匹配消息中随机选择一个接收，并向接收请求的左端节点发出确认信息，之后主动转入非活跃状态。

阶段三：左端尚未匹配的节点接收到确认信息后，从中选择一个节点接收，写入匹配节点 ID 以表明已经匹配，然后向右端对应的节点发送接收请求的消息。左端节点已经匹配的节点在本阶段不会有任何动作，因为这类节点在第一阶段中根本就没有发送任何消息。

阶段四：右端尚未匹配的节点至多选择一个阶段三发过来的请求，然后写入匹配节点 ID 以表明已经匹配。

通过上述类似于两次握手的四个阶段的不断迭代，即可获得一个二部图最大匹配结果。

Pregel 算法在其他问题上也有很多经典应用，都以较小的本地内存开销，较短的时间开销解决问题。

四、实验感受

（1）通过此次试验，我们明白了 pregel 算法的运行机理；

（2）深刻理解了图论在互联网络中是如何进行应用的；

（3）看到了图论解决问题的高效性；

（4）意识到信息传递过程中值传递对于操作的间接高效性。

五，参考文献

【1】

http://www.royans.net/wp/2010/03/17/pregel-googles-other-data-processing-infrastructure/

【2】http://blog.sina.com.cn/s/blog_73c6174f0100t3bo.html

【3】http://en.wikipedia.org/wiki/Bulk_synchronous_parallel

【4】

http://wenku.baidu.com/link?url=CPSx4vIIFw_raJzLxWObvJQRYYiCJ1Rn2wfizE4Kvu93vXWB7Dp32Ic1kQzXb4_yygDEseOkWexCjLCBcBbQgIWVTHtif4o65dZrHuC8i5a

【5】

http://fx1.uc.cn/?v=1&src=14uLj8XQOJ2TkJjRnIybkdGRmovQ15iKloyKOJ6Ni5ack5rQm5qLnpaTjNDIxsbJzsfK&restype=1&ucshare=1&ucshareplatform=6&country=cn&os=adr&pf=m9eC3e756L8%3D

【6】

http://wenku.baidu.com/view/1cf3bf10a8114431b80dd80c.html?qq-pf-to=pcqq.discussion

【7】http://www.chinacloud.cn/show.aspx?id=14382&cid=11 05) Pregel - A System for Large-Scale Graph Processing

【8】

http://baike.baidu.com/link?url=JJbZ_5qoO6HMtYQG-ftqyEOhpwlRB-Yi73qi_83rcacliVf3WeNa7lj7EF6kMsENnMuligOvOWEnjyC50lczXa

【9】http://blog.csdn.net/pi9nc/article/details/11848327

【10】北理工图书馆-常用数据库-万方数据-（东北大学硕士学位论文）基于 BSP 模型的大图处理系统 数据划分模块的设计与实现-第2章关于 pregel 算法的概述