**读后感**

1. MapReduce: Simplified Data Processing on Large Clusters

MapReduce是利用Map函数处理一个输入key/value pair集合来产生一个具有相同key值key值的value值输出的中间key/value pair集合，使用Reduce函数合并所有相同key值的value值的编程模型, 也是一个处理和生成超大数据集的算法实现模型。MapReduce这个模型能够处理因为数据量大而将计算分布在成百上千的主机上产生的并行计算、分发数据、错误处理等问题，它将并行计算、容错、数据分布、负载均衡等复杂的细节封装在一起，也能让那些并没有并行计算和分布式处理系统开发经验的程序员有效利用分布式系统的丰富资源。

MapReduce模型可根据当前的具体环境而采用不同的实现方式，如：小型的共享内存方式的机器、大型NUMA架构的多处理器的主机、大型的网络连接集群。MapReduce架构的程序能够在大量的普通配置的计算机上实现并行化处理，通过Map调用的输入数据自动分割为M个数据片段的集合，然后Map调用被分布到多台机器上执行，用户程序在机群中创建大量的副本(一个主副本程序master，其它的是worker程序)，master分配M个Map任务和R个Reduce任务，而worker程序则负责处理Map或Reduce任务，读取相关的输入数据片段解析出key/value pair后传递给用户自定以的Map函数生成并输出中间的key/value pair，将通过分区函数将其缓存在内存中的R个区域，之后周期性的写入到磁盘中，master会记录其在磁盘上的位置并传递给Reduce worker, R个区域的主机则会从 Map worker所在的主机读取数据，进行相同key值的数据聚合排序，这个过程由Reduce worker 完成，当所有的任务完成之后，master唤醒用户程序，用户对MapReduce的输出进行返回。任务处理过程中用到了成千上百的机器，会出现一定的问题，所以master还负责周期性的ping每个worker,监测机器故障；这个过程中产生的存储问题则是通过GFS文件系统来管理存储在机器的本地磁盘上节省网络带宽问题。我们用计算大约1TB的数据进行特定的模式匹配和排序来衡量MapReduce的性能。

MapReduce编程模型将并行处理、容错处理、数据本地化、负载均衡等问题在一个库里面方便我们使用，能够解决网络搜索服务、排序、数据挖掘和机器学习等各个方面，可以在一个数千台计算机组成的大型集群上部署，让我们可以更加好的利用计算资源和解决大量计算问题。

二、The Google File System

Google GFS 文件系统， 一个面向大规模数据密集型应用的、 可伸缩的分布式文件系统。GFS 虽然运行在廉价的普遍硬件设备上，但是它依然了提供灾难冗余的能力，为大量客户机提供了高性能的服务。

GFS文件系统的特点：Google GFS文件系统重新审视了传统文件系统，设计出了完全不同的设计思路，首先针对组件频繁失效的问题，将持续的监控监控、错误侦测、灾难冗余以及自动恢复机制集成在GFS中，为了解决这个问题设计了快速恢复和复制机制，快速恢复通过创建Chunk和Master副本的方式；其次由于文件巨大，假设条件和参数，I/O操作和Block的尺寸都经过重新考虑，为了保持数据的完整性每个用于存储数据的Chunk服务器都使用Checksum来检查保存的数据的完整性，并通过闯将；然后是由于海量文件的访问模式，采用在文件末尾追加数据，而不是覆盖原有数据，通过记录追加操作实现了生产者-消费者队列模式；最后应用程序和文件系统API协同设计，提高系统的灵活性。

为了解决上述问题，Google针对不同的应用部署了多套GFS集群，一个集群包含一个单独的Master节点、多台Chunk服务器，能够同时被多个客户端访问。Master节点管理所有的文件系统的元数据，Chunk服务器用于将GFS存储的文件的以Linux文件形式保存到本地磁盘中，用Chunk标识数据和管理数据，GFS客户端代码以库的形式链接到客户程序中，客户端代码实现了GFS文件系统的API接口函数、应用程序与Master节点和Chunk服务器通讯、以及对数据进行读写操作，控制流是在Mastr服务器处理，而数据流在Chunk服务器和客户端处理，这样的方式能够将中心的Master服务器的负担降到最低。

结合于传统的文件系统，和更加详细和全面的考虑，Google设计出了一个交互更可靠和稳定、具有自己容错和诊断的机制，能够在大量的并发读写操作时提高很高的合计吞吐量GFS文件系统，提高了文件系统整体性能和灾难冗余的能力。

三 、FastSGG: Efficient Social Graph Generation Using a Degree Distribution Generation Model

该文的目标是学习一个概率分布p（G）在一个简单图的基础之上。给定一个简单图G=(V,E)和一个图中节点的序列π。然后定义一个映射，表示可能的邻接矩阵，和图中节点的序列。用π去隐式的表示这个邻接矩阵中的节点的序列。其中，对于无向图来说，他的邻接矩阵是对称的所以就可以用下三角矩阵来代替整个邻接矩阵。然后在该方法中，每一步生成一行或者一个块中的数据。生成一个序列向量。其中：第i行数据被填充为零。然后该过程一直会持续到最大次数。最开始输入是一个简单图，然后同一个颜色的节点属于同一个类别也就是同属于一个块。下三角矩阵是一个邻接矩阵，其中的深色方格就代表节点之间是有边。然后一个块之中有2行，也就是把2行分为一个块，其中是可以通过参数自己设置的。然后加入一个新的块也就是其中的粉红色的节点，然后在新添加的节点之后的图通过图递归注意力网络，最后生成节点之间每条边的概率，最后根据概率生成新的图。其中的L是指每一条边的概率的集合的一个序列向量。

该模型每一次都生成一个块中的B行数据。生成图的步骤数为：节点个数N/每块中的行数向上取整。循环神经网络是一个标准的建模模型对于序列结构的数据，但是其中也有很多的缺点，比如说两个节点在图的拓扑结构中距离很近，但是在节点的序列结构中却相距很远，这是不合理的。为了解决这个问题，作者利用了GNN而不再是RNN。GNN能够更加好的探索节点之间的关系，这些都是RNN做不到的。通过一个权值参数W和一个偏执项b，将序列向量L量化成了一个节点数据h。因为是上一层输入过来的节点信息h，所以i要小于t。特别的，当时第0层的时候，也就是最开始，我们设置h=0。这样通过上面的式子我们就将一个序列向量量化成了一个节点数据，然后，就可以输入GNN，进行计算了。在经过将边的序列信息量化成节点信息之后，就可以输入图神经网络了。其中hir是指节点i在第r步中的隐层表示，mijr是指节点i到节点j的信息向量。hir一把是指对于hir节点信息的增强。其中，如果节点i在已经生成的图结构之中，那么xi的值为0。aijr是指节点i和节点j之间边的注意力权重。然后，在得到以上的信息之后通过一个循环神经网络GRU，将以上信息进行整合，得到最终的新更新的节点信息hir+1。

在经过R轮信息传递之后，获得了节点的最终表示hiR，然后模型的边的概率通过一个伯努利混合模型生成。权重和参数通过输入节点表示信息到多层感知机中计算得到，然后利用伯努利混合模型，最终得到边的概率分布。最后根据边的概率分布得到最后生成的图。