# GraphFlow:基于状态更新的实时图计算系统

# 摘要

# 引言

[什么是图数据]

图是计算机科学中常用的一类数据结构，它能够很好的表达了数据之间的关联性。现实世界中有很多数据都可以抽象成图数据，例如Web网页之间的链接、社交人物之间的互动以及买卖双方的交易都可以抽象成彼此关联而形成的图。而随着互联网的快速发展，图数据的总量也在急剧增加。如截至2014年第一季度Facebook包含了12.3亿个活跃用户，每个用户平均好友130个；web链接图顶点数达到Ｔ级，边的个数达到Ｐ级[1]。

[图计算模型的运用]

因为图数据能够很好的表达数据之间的关联性和聚集情况，因此针对图数据表达的关联关系可以挖掘出很多有用信息。比如，通过为购物者之间的关系建模，就能很快找到口味相似的用户，并为之推荐商品；在社交网络中，通过传播关系发现意见领袖。图算法及相关的处理框架已经广泛运用在社交分析、商品推荐、舆论监测、欺诈检测等各个领域。

[图计算模型面临的挑战]

处理这些海量动态的图数据也对现有的图计算模型提出了挑战。一方面，这种超大规模的图数据很难一次性的全部导入内存进行处理，即使能够借助外存一批一批的处理图数据，也使得计算延迟显著增加；另一方面，这些数据又是动态变化，实时更新的，现有的图计算模型要能够在这种动态的数据集上进行增量计算。

[传统的解决方案]

现有的成熟的图计算系统如Google Pregel[2], Spark GraphX[3], 这些图计算框架都采用了分布式的集群和BSP（Bulk Synchronized Processing）[4]消息同步机制来处理图数据。然而这些系统都是在静态的图数据结构上进行的离线批量处理。即每次针对整体的图进行计算，当图动态变化时，需要在变化后的整个图上重新计算一遍。这使得用户等待周期长，无法满足实时计算的要求，也浪费了系统资源。[5]

因此针对现有的图数据不断变化的情况，提出了很多直接在动态图上进行计算的方法。针对这类动态图计算的问题，大致可以分为两类：估计计算和准确计算。对于估计计算，大部分的算法是希望通过采样的方式来降低时间和空间复杂度，并通过特定的采样方法来减少真实值和实测值之间的差距。 如文献[6][7][8]研究了在流图上如何通过设计采样规则来估计图中三角形数目，使得占用的空间少而且估计的结果在给定的误差范围内；文献[9][10][11] 通过将原始的图信息转化为简单的数据结构来保存图中元素，使得消耗内存量远远小于原图。同时结构随数据流不断进行更新。虽然估计算法能够在一定程度上节约了内存和计算开销，但其估计的误差在实际的生产环境中往往变得不可控制，文献[12]指出，针对大体量的无法全部载入内存的图数据，近似算法的错误率在95%-133%之间。对于准确计算，现有的KineoGraph[13]和IncGraph[5]提出采用增量计算的模型进行实时计算，然而其计算模型都是基于增量消息进行的计算，而且更新都是串行执行的，实时性有限。SpecGraph[14]虽然在上述两个增量模型的基础上有所改进，提出了并发更新的实时图计算模型，然而该模型中假设顶点的状态只依赖于顶点当前接收的信息，而无需关心顶点之前的旧状态。这种假设使得系统的适用性差，很多算法其实不仅跟定点接收消息有关，还跟顶点的旧状态有关，因此模型的表达能力有限。

针对动态图计算的实时性要求，本文在上述已有的研究基础之上，提出了**基于状态更新的动态图计算模型**，能够在原有图状态的基础上，并发的计算增量信息对状态的影响，而无需在整个图上重新计算，同时通过控制更新影响范围来提高并发性，实现状态的并发更新。为此，本文的主要工作有以下3点：

（1）分析现有的图计算的特点，抽象出在流式场景下图算法的典型特征。从影响范围，迭代次数和更新顺序三个维度分析了5个典型的图算法：节点度分布（DD, Degree Distribution），三角形数目（TC, Triangle Count），单源点最短路径（SSSP, Single Source Shortest Path），连通子图（CC, Connected Components），PageRank（PR）。

（2）根据这些流式图算法的特点，建立基于状态更新的增量图计算模型，该模型能够丰富的表达图的计算过程，并且采用并发更新的方式来快速计算结果。

（3）在上述建立的状态更新的动态图计算模型基础上实现了XXX 4个典型的算法，并且从正确性、实时性和扩展性三个层面对算法进行评估，结果表明：

# 基于状态更新的增量图计算模型

## 流式图数据

所谓流式图数据是指图的数据（包括图的顶点、图的边、图顶点的值和图边的权重）不再是静态的存储在文件或数据库中，而是以流的形式源源不断的添加到系统中。因此系统中的图是随着时间而动态变化的。常见的流图模型有两种[14]，一种是Cash Register Model：流中的每一项仅仅是数据集中一项，通过流的方式不断的扩充数据集。另一种是Turnstile Model : 在该模型中，我们有一个初始化为空的集合D，流中的数据由两项组成，一项是数据集的某一项，另一项是一个标志位，可以对集合D进行动态改变。例如，流图中的每一项为(x, U)，如果U为＋，就将x加入D，如果U为－，就将x从D删除。如下图所示，在一个管道中，图的每条边按照一定顺序流入系统中，其中+表示增加一条边，-表示删除这条边，对应这些边的变化，图的结构和状态也在随着不断变化。在本文中，我们考虑的是边的Turnstile Model，即图数据是按照边的添加和删除来进行组织的。

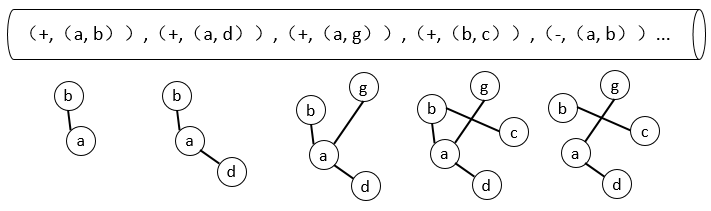


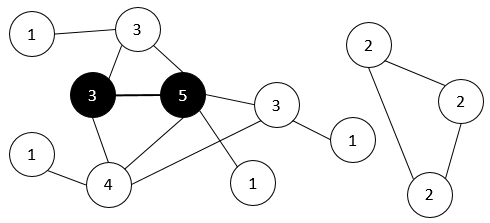
Figure 1 流式图数据

## 图算法分析

相比较传统的静态图数据，流式图数据的图算法有很大不同。下面选取了Degree Distribution、Triangle Count，Single Source Shortest Path和PageRank这四种典型的图算法来分析在边不断添加的情况下各个算法的特点。

（1）Degree Distribution

Degree Distribution是统计无向图中各个节点的度。如图2展示了在流式场景下如何统计节点的度。图2中的每个圆圈表示一个节点，圆圈之间的连线表示一条边，圆圈内部的数字表示当前时刻节点的度。在某一时刻节点的度分布情况如图2(a)所示，当增加一条边（图2b中标黑的两个节点之间增加一条边）后，这条边对应的源顶点和目标顶点的度分别增加1.增加该条边所带来的影响如图2b所示。由此可见，增加的这条边只影响了这条边的源点和目标点。

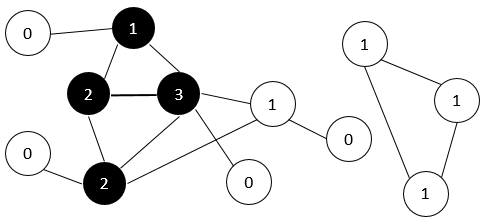
 

（a）某一时刻节点的度分布情况 （b）增加一条边后节点的度分布情况

Figure 2 节点度分布

（2）Triangle Count

Triangle Count是用来统计无向图中的不同三角形的数目。该算法在复杂网络分析、链接标签和推荐等多个领域中都是非常基础重要的度量，也是一些诸如复杂网络、聚集系数等图运算中的基本方法。图3展示了在流式场景下如何统计三角形的数目。图3中节点内的数字表示该节点所拥有的三角形的数目。图3a表示在某一时刻三角形的分布情况，当在三角形数目为0和1的节点之间增加一条边时，它会使得这两个节点的公共邻接点的三角形的数目分别增加1，而这两个节点的三角形数目增加N,N为公共邻接点的数目。由此可见，增加的这条边不仅影响了这两条边，还影响了这两条边的邻接点。

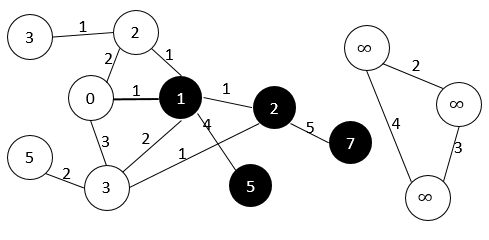


（a）某一时刻三角形分布情况 （b）增加一条边后三角形分布情况

Figure 3 三角形数目分布

（3）Single Source Shortest Path

Single Source Shortest Path算法是在有向图或无向图中，给定一个源点，求解这个源点到图中其它各个点的最短路径问题。最短路径问题是图论算法中的经典问题，也是诸如路径规划、物流规划、GPS导航、社交网络等现实世界中许多应用的基本问题。[15]图4展示了流式场景下的单源点最短路径问题解决方法。图4反应了在流式场景下的最短路径求解问题。图中边上的数字表示这两个节点之间的距离，节点内的数字表示当前时刻源点到该节点的最短距离，数字为0的节点为源点。图4（a）反应了某一时刻源点到各个节点的最短路径情况，图4（b）表示新增一条边（节点内数字为0的点和节点内数字为3的点新增了一条边，边上的权重为1）之后的情况。由图4（b）可见，新增的这条边会将影响沿着某条路径传播下去（即图中的0->1->2->7和图中的0->1->5），如图中新增的权重为1的这条边，会将原来值为3的节点更改为1，并且该节点的后续节点的值都会被更改。

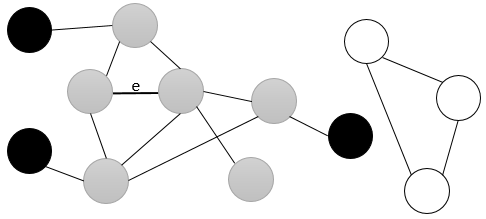
 

（a）某一时刻源点到各个节点的最短路径 （b）增加一条边后源点到各个节点的最短路径

Figure 4 单源点最短路径

（4）PageRank

PageRank算法[16]是基于网页链接分析来计算各个网页的重要程度。假设网页A引用了网页B，那么A就将一定的分数贡献给了B，该分数（即PageRank值，简称PR值）就是网页的重要程度的体现。而网页之间是相互引用的，因此经过若干次的迭代之后，网页的得分会趋于稳定，这个分数就反应了该网页的重要程度。图5a展示了某一时刻各个节点的PR值，当新增一条边e之后，图5b展示了这条边带来的影响：灰色的节点表示新增的这条边直接影响这些节点的值，随后灰色的节点又将这些影响继续往外传播给黑色节点，经过若干次的迭代之后各个节点的PR值保持稳定，算法运行结束。由此可见，对于PageRank算法，当图中新增一条边时，这条边会影响这条边所在的连通子图内的所有节点。

（a）某一时刻各个节点的PR值 （b）增加一条边后各个节点的PR值

Figure 5 PageRank

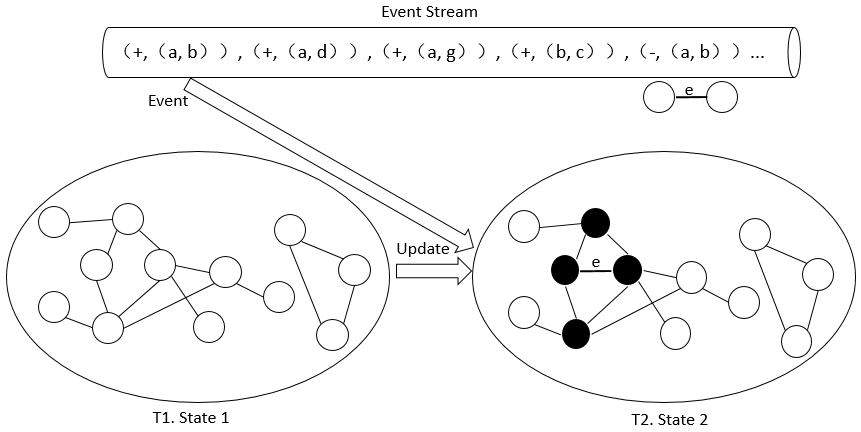
（5）总结

通过针对以上四种算法的分析，当新增一条边时，我们从影响范围、计算次数两个维度来分析各个算法的特点。影响范围是指新增加的这条边可能会影响到哪些节点，而计算次数是指这种影响是否会被计算多次，例如在DD算法中，新增加一条边只需要将这条边的源点和目标点对应的度数加1，这种运算只需要执行一次，而对于PR算法，新增加一条边时，这条边的源点和目标点的输出贡献将会发生变化，因此会首先影响他们的所有邻接点，这些邻接点在下一次的传播中会继续他们的邻接点，经过多次迭代计算之后各个节点的PR值会趋于稳定，在迭代计算的过程中，每个节点可能参与多次计算。

Table 1 图算法特征表

|  |  |  |
| --- | --- | --- |
|  | 影响范围 | 计算次数 |
| DD | 影响新增这条边的源点和目标点 | 被影响的节点只参与计算一次 |
| TC | 影响新增这条边的源点和目标点，以及这两个点的公共邻接点 | 被影响的节点只参与计算一次 |
| SSSP | 以这条边的某个节点为起点，沿着某条路径往其他节点传播影响 | 被影响的节点可能会参与计算多次 |
| PR | 影响这条边的源点和目标点所在的整个连通子图内的所有节点 | 被影响的节点一般会参与计算多次 |

## 基于状态更新的增量图计算模型



# GraphFlow系统

# 系统评测

# 相关工作

# 结束语