## CAGNET：降低GNN的通信（分布式）

### 1 Motivate：

#### 1.1 mini-batch

目的：内存占用较小、通过提高运算强度实现高性能和良好收敛性的平衡

但是在基层迭代之后，mini-batch就会依赖整个图数据——邻域爆炸，破坏了减少内存占用的目标。

#### 1.2 sampling

为了解决邻域爆炸这种问题，researchers采用基于复杂采样的方法，通过减少考虑的k-hop邻域的数量来减少内存占用。然而，采样算法存在近似误差。在这里，我们使用集群或超级计算机的聚合内存来训练GNN，而无需mini-banch。

#### 1.3 解决

大多试验表明，full gradient descent在性能上与mini-batch相同，基于采样的方法可能会降低准确率。在此基础上提出减少通信的分布式算法，虽适用于全梯度下降，但是稍作修改即可在mini-batch设置下运行。

分布式GNN训练的主要挑战是通信成本和负载均衡。本文关注最小化通信成本。

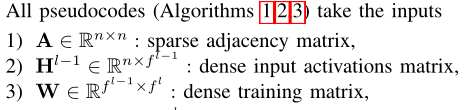
#### 1.4 相关工作（背景知识——原文翻译）

近年来，人们深入研究了DNN中的并行问题。分为数据并行和模型并行。数据并行性在每个进程的内存中复制DNN模型，并且只对数据进行分区。数据并行可以分为样本并行（也称为批处理）和域并行。在卷积神经网络（CNN）的特殊情况下，域并行通常被称为空间并行。另一方面，模型并行性明确地划分了模型。在常见情况下，每个DNN层可以划分为所有进程，层可以按其原始顺序计算。或者，层间管道并行性可以用于计算的某些部分，但不是全部。对于CNN，可以利用滤波器和通道形式的进一步并行度作为模型并行的特例。

各种GNN训练系统中的数据分区细节很简单。ROC提倡一种专门的图分割方法，并表明它比随机顶点和边分割具有更好的可扩展性。AliGraph提到它实现了基于图分区的方法和2D分区方法，但没有给出任何细节或提供通信成本分析。

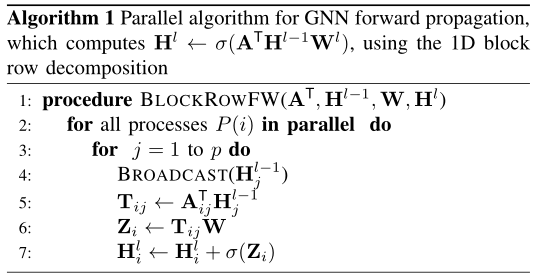
### 2 主要工作

理论上，分布式内存并行GNN训练算法在所有进程中消耗O(nf L + nnz(A))。（f：feature长度；L：layers数量）

下面所谓nD算法，就是将A分为不同的块，使用不同的机器去实现矩阵乘法。

#### 2.1 1D算法



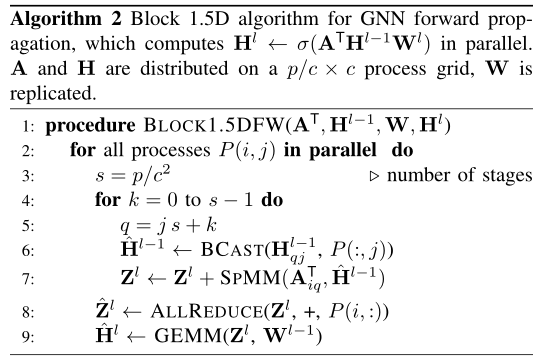
AT、HL按行分割，即1D。W在每个进程中完全复制。这个过程就是SpMM

激活函数这部分没有通信

这里使用Broadcast在SpMM中移动密集的feature矩阵。移动稀疏矩阵的替代方法在实践中会产生类似的通信成本，因为GNN中的密集特征矩阵的大小（以字节计）与它们所运行的图大致相同。

由于我们移动密集矩阵并保持稀疏矩阵静止，因此可以将图和超图分区工具用作预处理，以启发式地最小化通信。

#### 2.2 1.5D算法

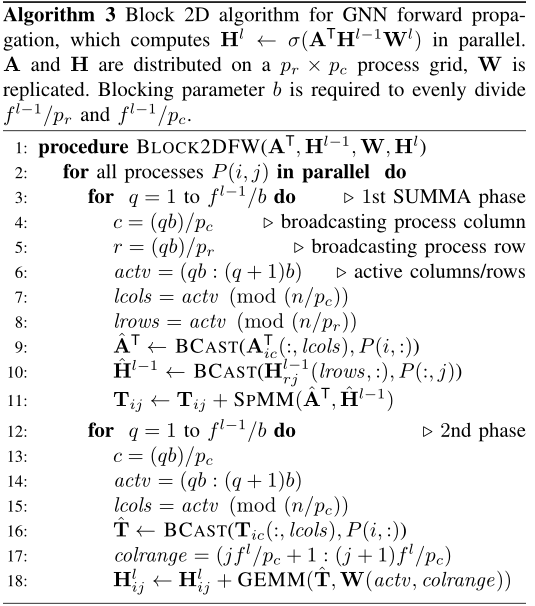


c：Chunk

减少广播的矩阵的Size，来避免通信。在我看来，就是将A分块，这样每次广播的时候不比广播所有的H，只需要广播本块附近的一部分H，减少通信的同时可以不丢失计算的准确率。因为本块A进行SpMM时不会改变其>2-hop邻居的embedding。（个人理解，有点难说实话）

下面2D、3D算法也类似，就是划分矩阵A来减少计算。

#### 2.2 2D算法



#### 2.3 3D算法

文中只是描述了算法，但是未给伪代码。核心思想是在2D基础上进一步拆分邻接矩阵A，从而达到减少通信的效果。