HARP：做network embedding的时候保持网络高阶的结构特征，具体的做法是通过将原网络图进行合并，合并为多个层次的网络图。这个算法应该可以说是一个通用处理框架，可以用于改进现有算法的效果

一、Network embedding

1. DeepWalk:

思路就是对网络应用了word2vec的SkipGram模型。先应用随机游走得到一系列的网络中有序的节点序列，这些节点序列类似于文本中的句子，将这些“句子”跑SkipGram模型，从而得到“句子”每个“单词”的向量表示。

Deepwalk的随机游走过程事实上是对网络进行采样的过程，将网络中的节点通过随机游走的方式表示出来，两个节点联系越紧密，在一个随机游走过程中共现的可能性越大，反之若两个节点根本不连通，则一个随机游走过程是不可能将两个节点共现。因此deepwalk能很好的将网络的连接情况进行表达，且实验证明在网络规模较大时具有很高的效率。

2.Line

提出了一阶相似度与二阶邻近度的概念，基于这两个邻近度，提出了优化函数，得到的最优化结果即为每个节点的向量表示。

该方法的优化过程可以理解为基于两个假设：

a.直接相连的节点表示尽可能相近（一阶邻近度），如图中6,7。文中两个节点的联合概率表示其一阶邻近度：

b.两个节点公共的邻居节点越多，两个节点的表示越相近（二阶邻近度），如图中5,6。文中用两个节点的条件概率表示其二阶邻近度：

3. node2vec

在deepwalk的基础上进行了优化。deepwalk中的随机游走过程，实际是就是一种简单的深搜过程，每次随机随出一个与当前节点直接相连的节点作为后继节点，这种方法虽然能够保证采样到网络中的全局信息，但是对于该节点为中心的局部信息往往不能很好的进行采样。

node2vec是一个参数控制的随机游走。例如图所示的情况，v为随机游走的当前节点，它的前驱节点为t，这时与其相连的节点的类型有三种：一种是t，v的前驱节点；第二种是x1，不仅与v相连，还与其前驱节点相连，向第二种节点游走，则为广搜的过程；第三种是x2、x3，不与其前驱相连，向第三种节点游走则为深搜的过程。为了控制广搜与深搜，因此设计了参数p和q，

所以deepwalk中的随机游走过程，就是一个p=1、q=1的node2vec。

二、HARP

1、基本都把考虑的重点放在了网络的局部结构关系（如：一阶相似性和二阶相似性），而忽略了距离较长的全局关系。

2、它们都是通过随机梯度下降法对一个非凸的目标函数进行优化，这样如果初始化不好就很容易陷入局部最优。

作者提出的HARP算法希望通过递归地粗粒化方式，将原网络图的节点和边通过合并划分成一系列分层的结构更小的网络图，然后再利用现有的算法进行不断的特征提取，从而实现最终的network embedding特征提取。

边合并和中心点合并，分别维护的是一阶相似度和二阶相似度。

边合并：先选择 E'，使得 E' 中的两条边不会连接于相同的点，然后对于 E' 中的每个节点对 (u_i,v_i) ,合并为单一节点 w_i ，并把之间的边去掉，这样，图的规模就能够大大的减小了。

中心点合并：在真实世界中的网络，某一个中心节点往往同时连接着非常多的节点，如果按照边合并的策略，则这种情况对图的粗粒度化作用非常不明显，如上图2(b)，所以，作者同时又采用了中心点合并的方法来进行粗粒度化操作，而它也能保持图结构中的二阶相似度。中心点合并的方法也很简单，就是把共同以中心点为邻居的节点两两进行合并，如上图2(c)