# Graph Virus Cluster

首先GVC的类似于label propagation，只不过前者从一个点向外传播，后者从邻近的点向内传播；

其次，GVC并不把图聚类看作是一个图划分任务，而是看作，以N个点为centroid，并计算每个点到centroid的相关度；

怎么计算相关度呢?

仿照virus的传播，首先从Graph的点列表中选择一个未被染感的点（0表示未被感染过）作为centroid，并让virus依次从centroid向外传播，virus初始值为1；

传播过程：

1. 首先设定传播次数（不选择用threshold的原因是因为，一方面threshold不好人工设定，另一方面传播次数更方便对计算资源上进行控制），Edge衰减系数（virus每次经过edge都会衰减一次，相当于当前值乘上衰减系数）；
2. 每次都选择可以让virus衰减最小的路径进行传播，经过edge时，会发生衰减，new_virus = virus*decaf*edge_weight（edge_weight需要被归一化）；
3. 每经过一个node，node会被virus感染，感染值为衰减后的virus值；

传播后，每个node的virus值，作为node和centroid的相关度，同时当前centroid会记录传播经过的node列表；

每次传播后，继续从Graph的点列表剩余的点中，选择未被感染的点作为centroid，继续一次新的传播；

不断重复，直到Graph的点列表没有未感染的点；

当然也可以继续选择最健康的点（virus值最低），进行传播，直到完成设定的centroid数值；

对于新插入的node，因为不存在未感染节点，因此可以通过node和所连接的neighbors进行virus传播，然后获得新node的感染状态（都被哪些virus感染了，virus值多少）；

在完成所有的centroid之后，我们会得到一个centroid列表，每个centroid都对应一个node列表，包含每个node和centroid的相似度；

通过centroid列表，我们可以得到每个node属于的centroid；

修改：

本来让virus衰减最小的路径进行传播是为了减少计算量，但因为计算virus衰减最小路径存在一定的开销，而且由于图是很稀疏的，因此更换成random walk；

传播过程修改：

1. 设定每次的random walk的传播次数；
2. 设定random walk的次数，根据计算资源自行设置；

3. 每次walk，都会增加经过node的virus值；

为什么选择GVC模型?

GVC计算量小，可以加大计算量而提升精度；

GVC可以适应图的增长；

GVC可以将所有的对象进行计算，不管是用户，用户词，用户词的关联关键词，还是以后的UGC，只要可以被词所连接，就可以被计算；

GVC的source选择很灵活，可以是用户，可以是关键词，可以是UGC，而且选择条件可以由其他数据所决定，比如用户活跃度，关键词的词频，UGC的热度；

GVC可以结合图数据库一起开发，同时满足存储，计算，查询；

GVC的结果可以提供给其他数据挖掘任务，同时了解整个数据的分布情况，适合进行推荐；

GVC模型有效性:

GVC基于蒙特卡洛思想，因为每一次传播都是从用户，到用户词，到关键词，因此对于两个用户词的相关关键词越多，被访问的概率就越大；

比如下面这个图，假设我们以Paul为source，那么Paulwalk到Aaron的概率要高于Alice，到Alice的概率要高于Roger，因此随机访问时，只要增大数量，就可以接近真实的概率分布；