Report& Literature Review for Graph Sampling/ Pattern Mining

# 参考资料简述

## Understanding Graph Sampling Algorithms for Social Network Analysis

文章的目的：保持图的规模较小，同时捕获原始社会图的属性

分析了四种最先进的图采样算法：BFS, RW, MHRW, FS

RW在文章中并没有被介绍，因此前的文献中1.2, 1.3中已验证比MHRW和FS性能弱

1. BFS
   1. 随机选择起始点存储在处理队列
   2. 对处理队列中的选择点V的邻居进行抽样，将选择点V存储在已采样队列，点V的邻居存储在处理队列
   3. 重复以上步骤直到达到固定预算
      1. 预算：采样点的数目，规定采样的时间等固定值

BFS偏向于高度节点，在BFS中，节点的访问度越高，访问频率越高。BFS获得了比原来更高的局部聚类系数

1. MHRW：Metropolis-Hasting Random Walk

是一种根据节点的度概率分布获取随机节点样本的马尔科夫链蒙特卡罗(MCMC)算法。在MHRW中，基于概率分布设计了一个建议函数。通过随机接受或拒绝提议，提议函数改变转移概率，使样本收敛于概率分布

* 1. 选取非0度节点起始点v
     1. 记录v点的度为kv
  2. 从该点的所有相邻节点中任意选取点w
     1. 生成0~1的随机数p
     2. 若随机数，则选取该邻点，并将点w作为下一次循环起始点
     3. 否则重新执行步骤b
  3. 重复步骤直至达到预算

1. FS：Frontier Sampling
   1. 选出一组节点集S（随机，无关）
      1. 获取每个点的度
      2. 计算每个点的选取概率，公式：p=
      3. 根据每个点的选取概率选取点v
   2. 从v出边中均匀随机的选一条（v,w）将节点集合中的点v替换为点w
   3. 重复以上步骤直至达到预算

评估在使用大规模有向图上一些被广泛认可的图属性上的性能

1. Node Degree Distribution (NDD)
   1. 在有向图中节点出入度
   2. normalized mean square error (NMSE) 节点度的归一化均方误差

其中表示图中入/出度小于或等于k的节点的占比，是抽样图对的估计

表示采样图与原始图的度分布的差异

1. Clustering Coefficient (CC)

图中节点聚在一起的程度的度量

1. Relative Error (RE)

量化算法在不同图属性的采样性能，测试采样图和原始图的属性度量上的相对误差

结论：

BFS偏向于较高的节点度，获得较大的平均聚类系数。

MHRW和FS保持较好的度分布。聚类系数方面对数据集的依赖较大

紧密连接图中表现均很好，FS比MHRW收敛更快，更准确

## Walking in Facebook: A case study of unbiased sampling of OSNs

相比1.1：

1. 更加详细的阐述了MHRW的具体执行步骤
2. 阐述收敛判断，并在线评估样本质量
   1. 确定一个样本何时足够用于后续使用
   2. 何时停止采样是合理的

两种标准的收敛测试：

1. Evaluating the accuracy of sampling-based approaches to calculating posterior moments（**Geweke Diagnostic**）

检测单个马尔可夫链的收敛性

* 1. X是一个样本序列
  2. 考虑X的两个子序列，是序列的前10%，是序列的后50%。基于和，计算统计量
  3. 随着迭代次数的增加，Xa和Xb的距离越来越远，这限制了它们之间的相关性。
  4. 由于它们度量的是相同的度量值，所以它们在收敛时应该是均匀分布的，并且根据大数定律，z值变成均值为0，方差为1的正态分布。
  5. 当大多数值落在[−1,1]区间时，可以声明收敛

1. Inference from iterative simulation using multiple sequences（**Gelman-Rubin Diagnostic**）
   1. 将单个链的empirical distribution与所有序列的empirical distribution进行比较，如果两个序列相似，宣布为收敛的
   2. 输出一个值R，是所有链的均值和方差。随着时间推移，R趋于1。当小于1.02时被认为收敛

## Estimating and Sampling Graphs with Multidimensional Random Walks

与1.2同为1.1的引用文献。在此项目中，认为1.1仅对应用实践有一定的参考意义。在此不做深入考虑，后面有需要时再深入细节

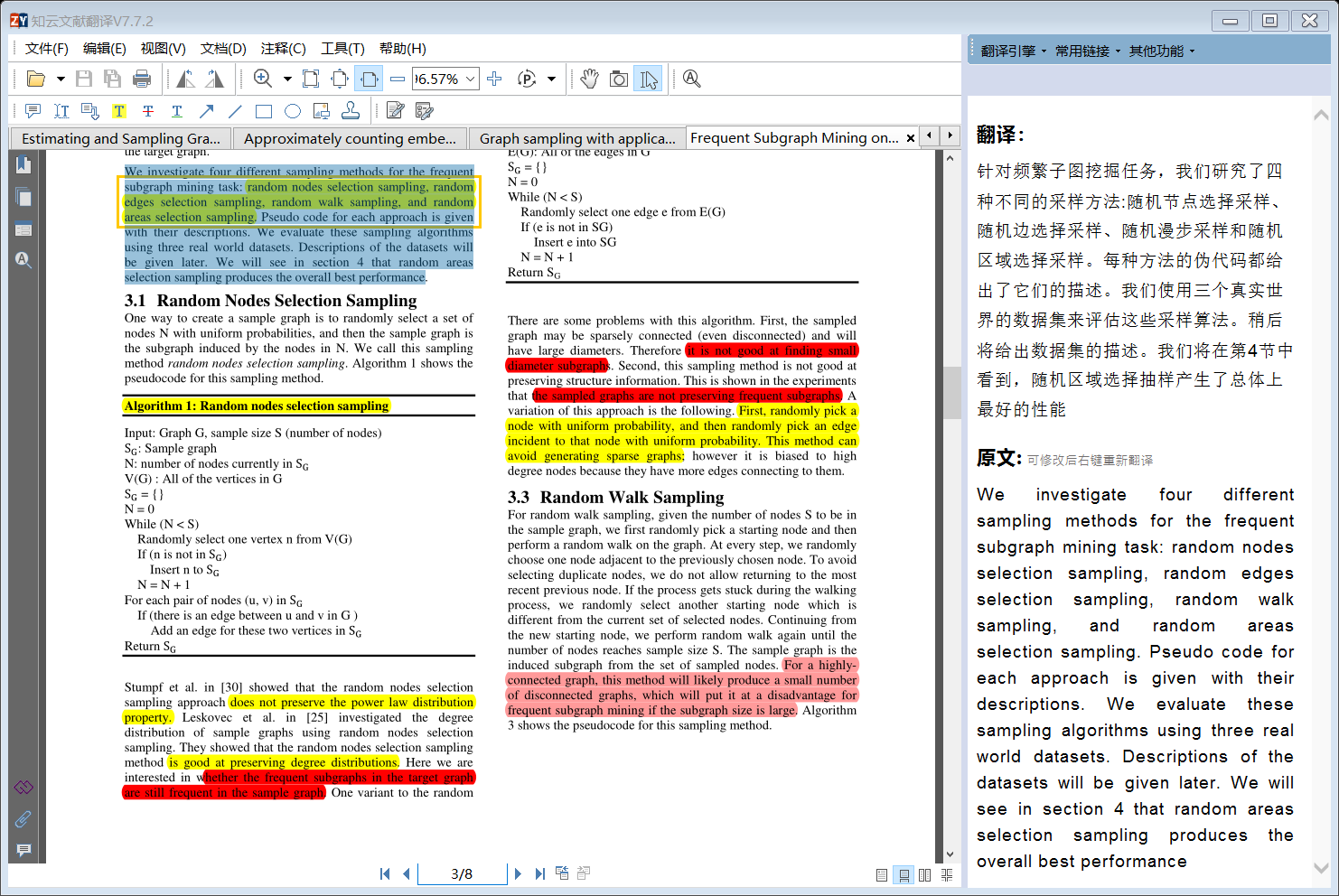
## Frequent Subgraph Mining on a Single Large Graph Using Sampling Techniques

文章目的：频繁子图挖掘，关注点在对图像pattern挖掘和数量估计不影响的情况下对图进行抽样。选取小部分的图以估计整张图中感兴趣的pattern的量。

1. 利用抽样方法研究在单个大型图上进行频繁子图挖掘。
2. 在单个大型图上比较频繁子图挖掘的不同采样方法
3. 在不同数据集上进行实验

四种采样方法

随机节点选择、随机边选择、随机漫步、随机区域

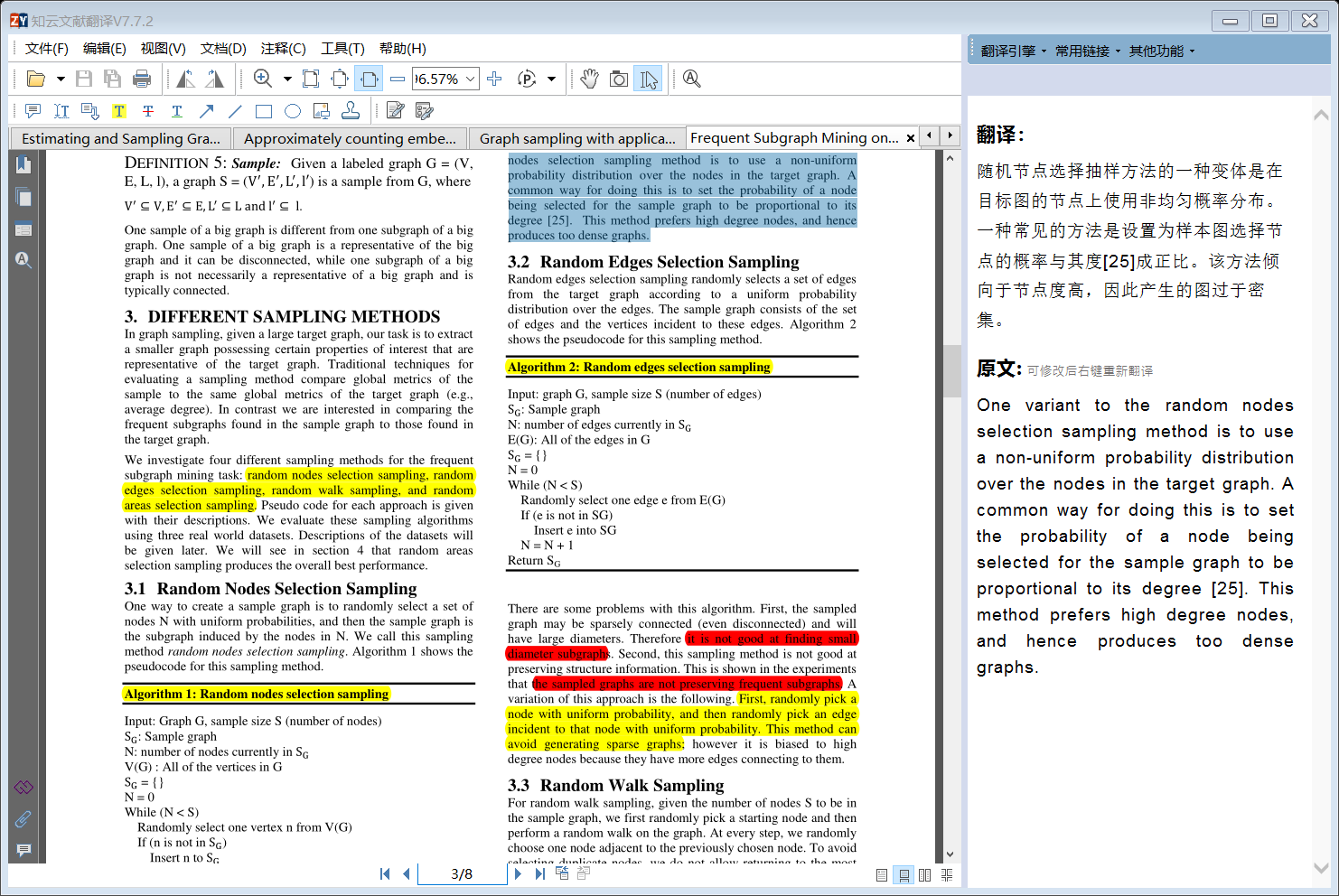
随机点选择：

1. 设抽样点数量上限N
2. 每次从图中任意选取不重复的点，直至达到上限
3. 对于所有的选取点之间，若相互之间有边连接，将边加入到图中

结果：该方法不保留幂律分布性质，能很好地保持度分布。

而目标是图中的频繁子图在样本图中是否仍然频繁。

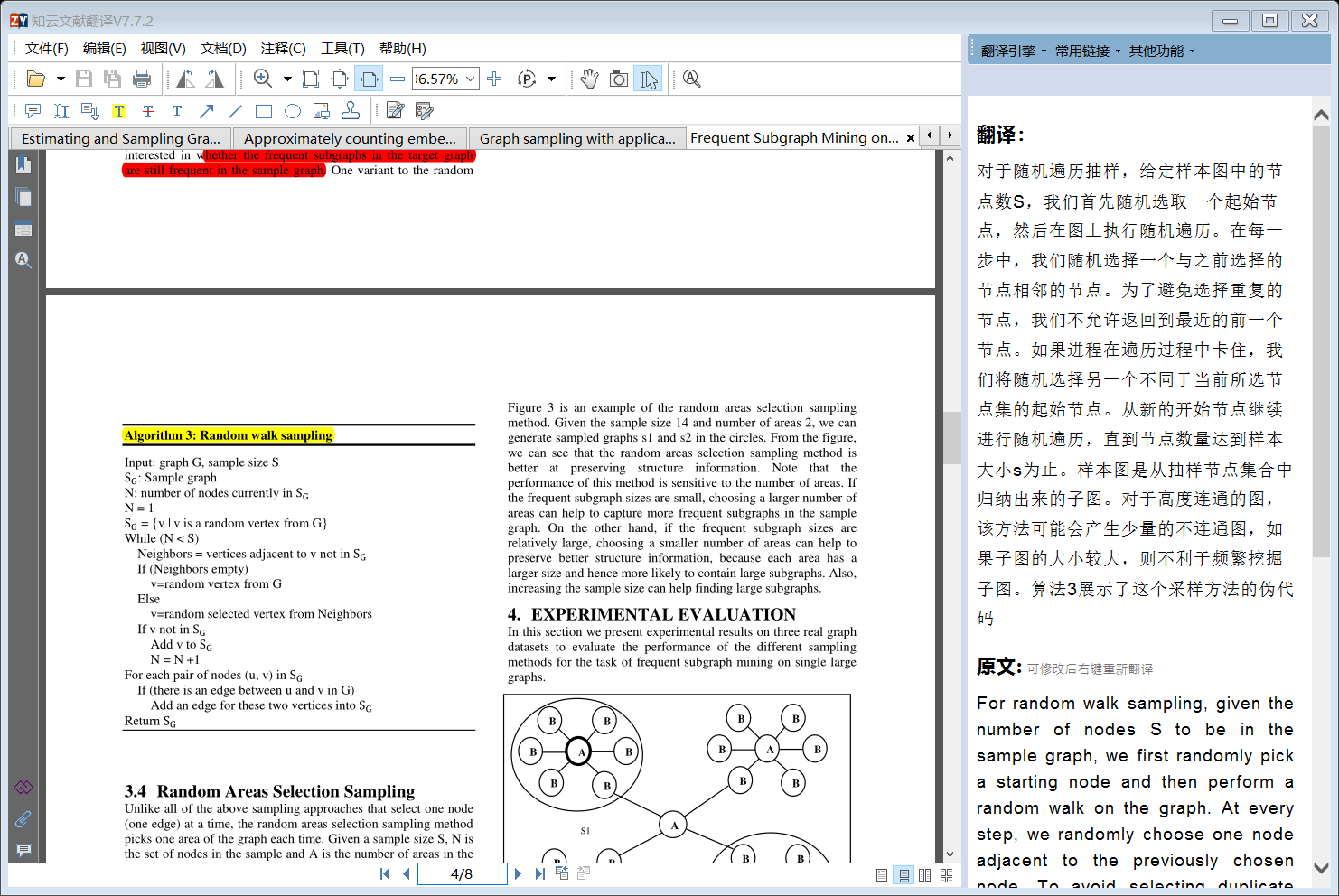
一种变体：在目标图的节点上使用非均匀概率分布（个人联想：1.1中类似方法）

随机边选择：

1. 设抽样边的数量上限N
2. 随机从图中选择一条不重复的边，直到达到上限
3. 得到抽样子图

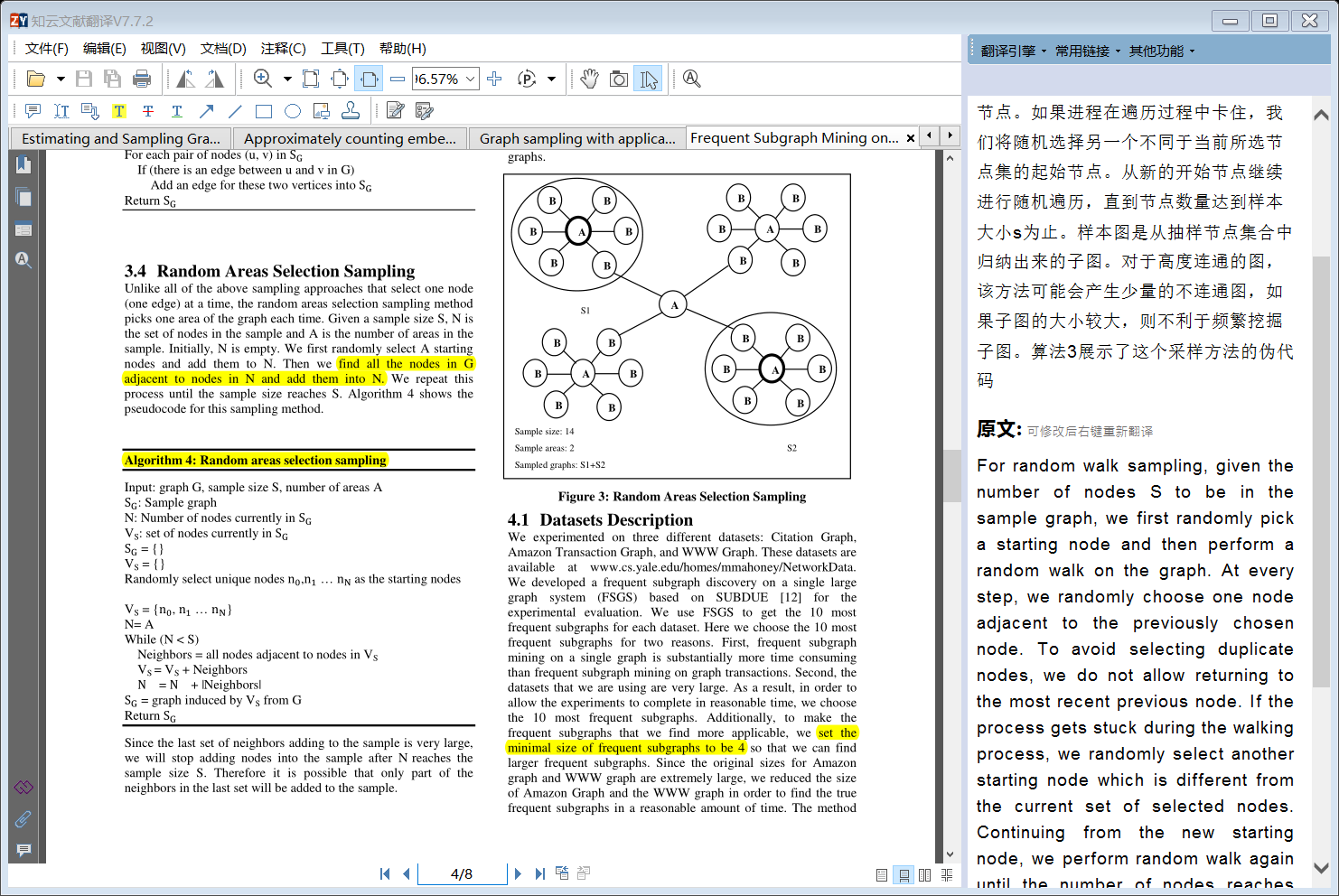
结果：采样图可能是稀疏连接的。且采样有较大的直径（稀疏）不擅长寻找小直径的子图。采样方法不能很好地保存结构信息

一种变体：首先等概率的随机选取节点，如何再等概率选取该节点的一条边。避免生成稀疏图。

随机漫步

1. 设抽样点数量上限N
2. 随机从图中选取起始点
3. 选取非重复的该点的邻点的其中一个，若没有邻点，重新随机选择新的起始点
4. 重复以上步骤至达到N个点
5. 对于所有的选取点之间，若相互之间有边连接，将边加入到图中

结果：对于高度连通的图，该方法可能会产生少量的不连通图。如果子图的大小较大，不利于频繁挖掘子图。

随机区域选择

1. 设抽样点数量上限N
2. 随机选取A个起始点
3. 对于每个起始点，寻找其所有的邻点并加入到集合
4. 重复3直至达到N个点

实验：对每个数据集选择10个最频繁的子图，子图最小大小设为4

对数据集运行FSGS(引用自Substructure Discovery in the Subdue System)得到真实频繁子图

使用不同抽样方法，得到抽样图。每种抽样图运行N次（default：15）使用FSGS进行子图发掘

比较15组中的10个频繁子图每个与真实的误差，误差相加取平均得到精确度。

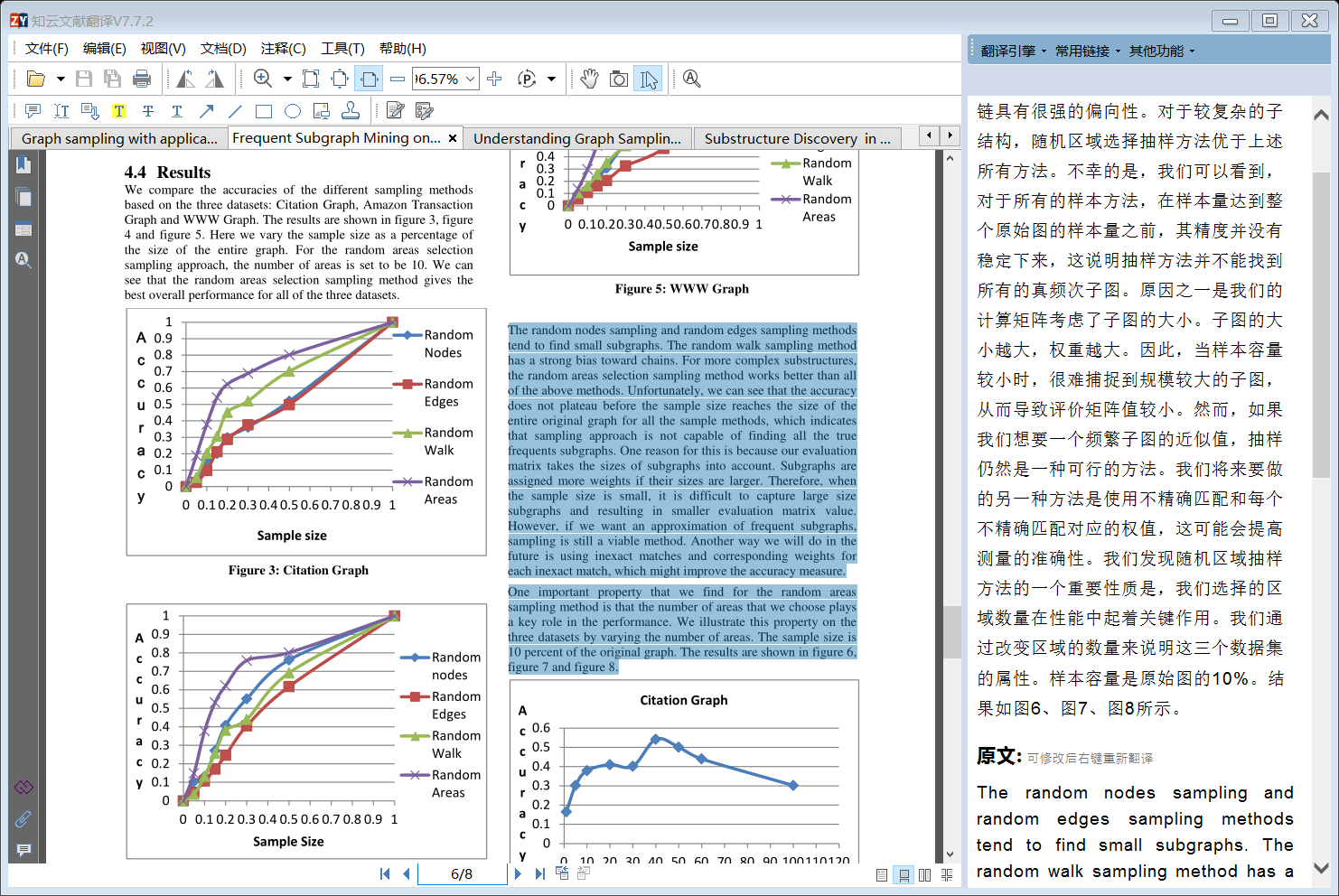
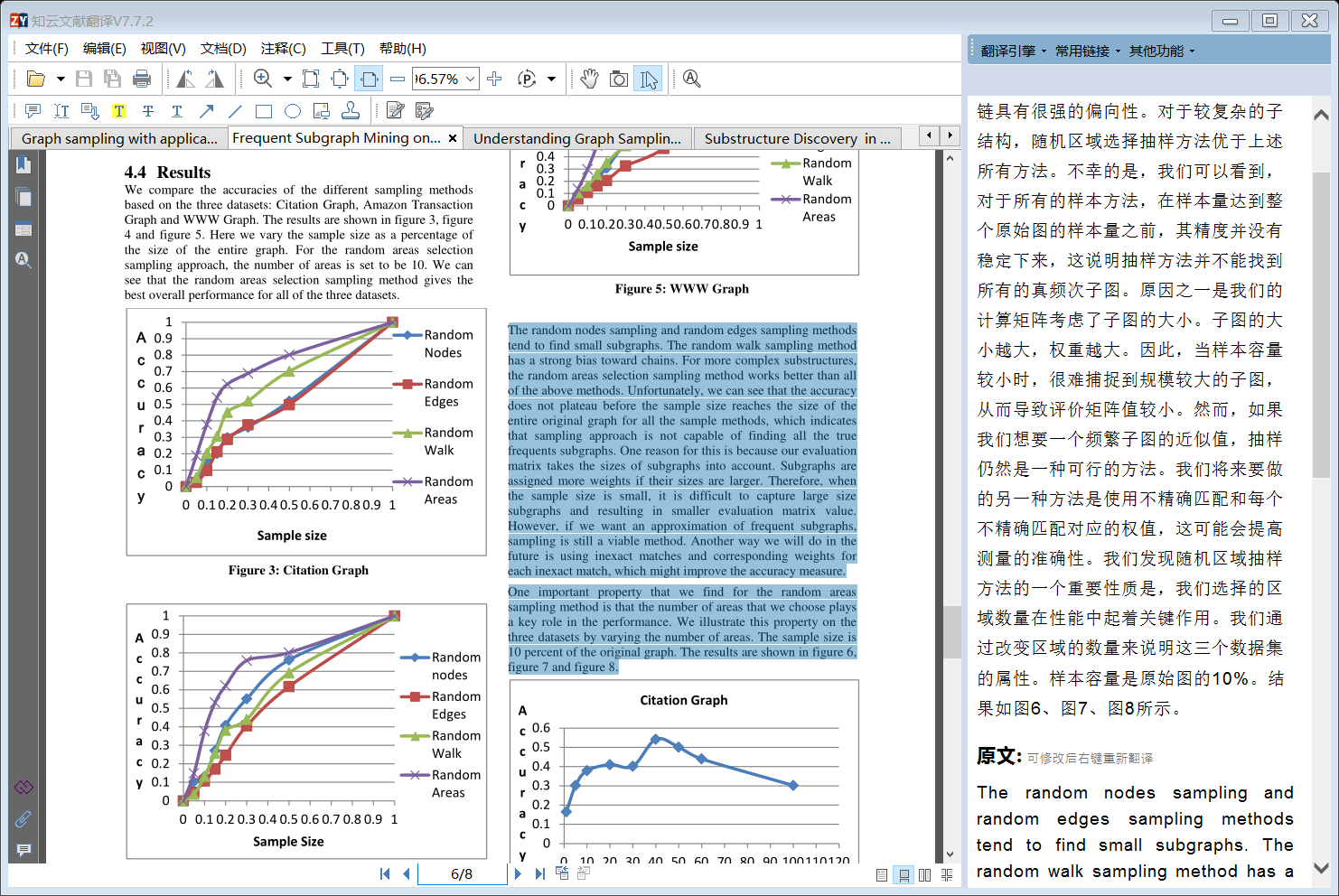
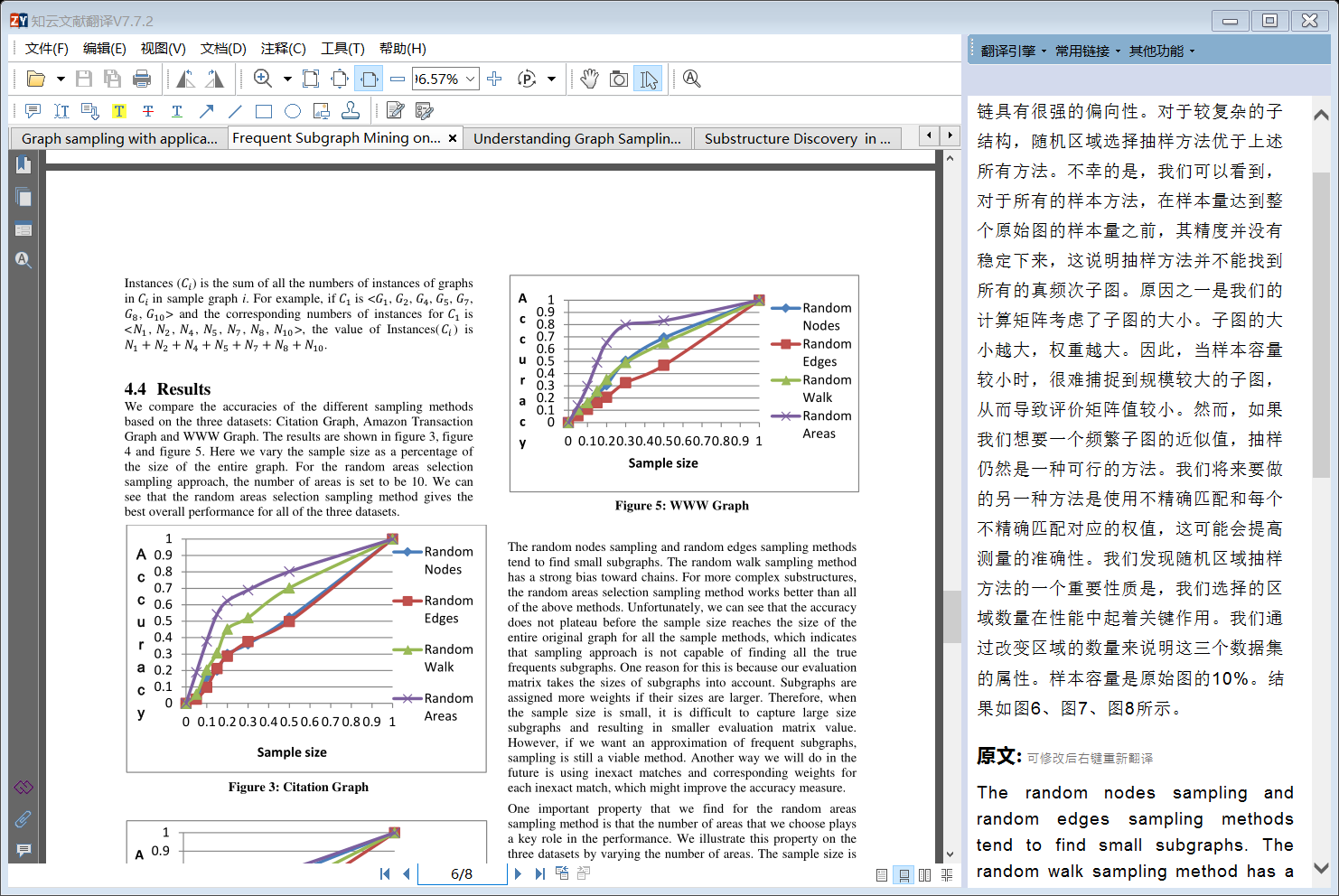
结论：随机区域选择抽样方法的总体性能最好

随机节点与随机边抽样倾向于找小的子图

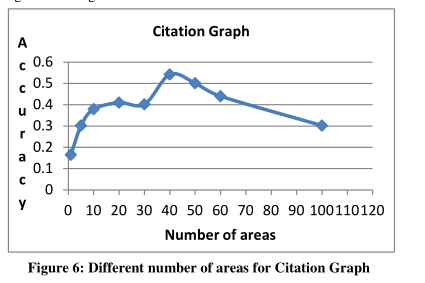
随机漫步抽样对链有很强的偏向

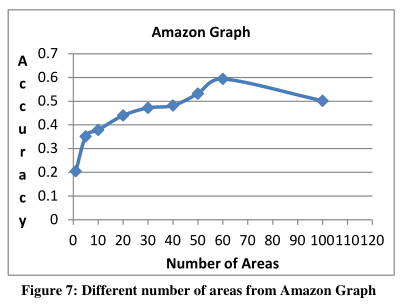
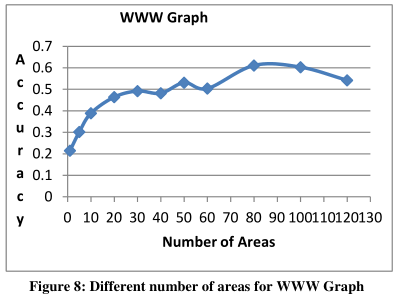
而抽样的结果同时也依赖于具体的图

X:采样占总图的数量，Y：pattern预估的精准度

线：不同的采样方法

上图为不同采样方法中，采样点数量对于最终结果的影响。

实验还讨论了在固定采样为原图10%时，采样选择区域的数量对结果的影响，结论是影响还挺明显



结果：对于每个图来说，区域选择对于精度的最优解都不一样，对于本项目的图来说需要进一步做实验来验证。

文章中further work：开发一个个更精确的框架来推导“随机区域选择“方法的样本大小边界。以及自动决定区域数量的随机区域抽样方法。

## Graph sampling with applications to estimating the number of pattern embeddings and the parameters of a statistical relational model

文章目的：针对计算一个pattern在数据库中出现的次数的问题，提出一种用于估计较大pattern出现的统计信息的抽样方法

文章包含：抽样方法，实验评估，算法验证，准确性与效率权衡的估计、

引文：计算pattern在图中出现次数是NPC问题，早期工作考虑了所有可能的pattern，但只关注与小的pattern。

基于Fürer和Kasiviswanathan(2014)提出的理论采样思想，提出新的算法

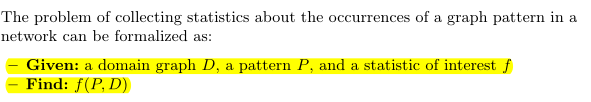
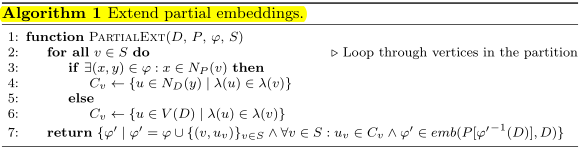
该文章与1.4区别：关注更大的图中的子图的频率，考虑size相对较大的子图。Work相近：Mining large networks with subgraph counting.（看不完了，后续再补充）

文章与我们要求相近的地方：允许使用通配符。即，节点的label在这种特殊情况下，不关心他是哪个值。

开源了所使用的pattern和代码：<https://dtai.cs.kuleuven.be/software/gs-srl>

（打不开）

文章思路

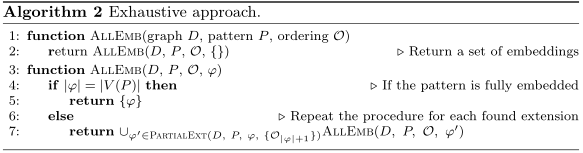
1. 图分解
   1. Ordered bipartite decompositions
      1. 本论文主要关注的分解方式，同时也是其主要引文1.6的图分解
      2. 考虑到LDBC图的复杂连接，暂不考虑OBD的图分解方式
      3. 文章后续涉及OBD的部分不再呈现（看不完）
   2. Arbitrary decompositions
      1. 在图无法进行OBD图分解的时候
2. 计算图中pattern的统计数量
   1. 基本过程是找到pattern的所有extension，和我们此前的思路高度相似。
      1. 算法1

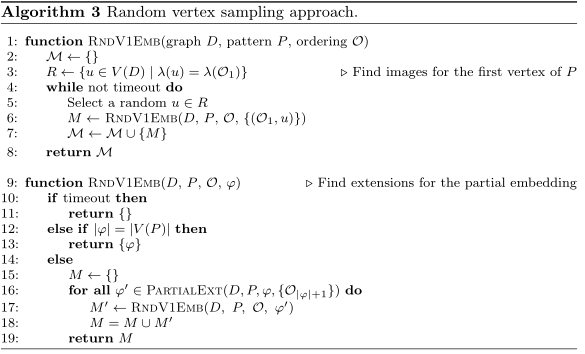
D: domain graph; P: pattern graph; : patrial embedding; S: set of vertices

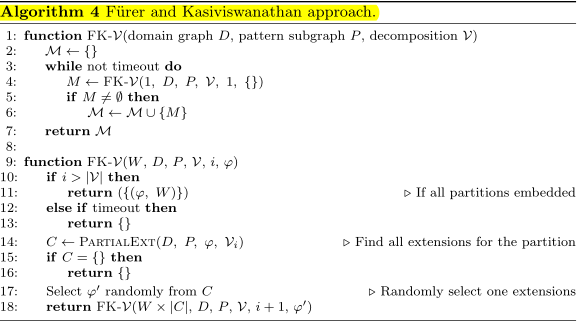
遍历所有S中的点，试图找到pattern v，并存在Cv中

构建所有可能的embedding集合，以便每个embedding都是和S组成

* + 1. 算法2 Exhaustive approach，该方法并非文章所推荐，结论上性能也不如推荐算法，因此略微带过



* + 1. 算法3 Random vertex sampling，该方法并非文章所推荐，结论上性能也不如推荐算法，因此略微带过
    2. 算法4，The F¨ urer-Kasiviswanathan algorithm

该算法为本文最常引用算法，也是本文的核心

该算法最初是用于无label无向图，将图划分成OBD，将总体问题分解为估计OBD的每个分区的embedding数量问题。对于每个分区，通过匹配分区中的顶点来找到当前部分的所有可能拓展。随机的对拓展中的一个进行采样，并将对pattern数量的估计×当前分区中可能发现的扩展的数量。

但当不是OBD时，不再保证算法的收敛速度

算法分为两部分

1. 循环控制
2. 单一尝试

在不超过时间限制的情况下，控制循环随机采样，并替换搜索空间中可能lead to 的pattern

函数FK-V返回一组(，W). 是发现的pattern（embedding）W是尝试找到该pattern的逆概率。W的期望值为的总数

FK-V识别算法1中的返回C，使用C的大小作为估计pattern的值

然后进行递归，更新参数,W

1. 计算样本图的相关统计量、估计量和准确度
   1. 统计量
      1. Count：D中pattern 的数量
      2. Mean and variance:
         1. Range(P,D)🡪R. 函数t将pattern映射到一个整数，该整数为他在D中的数量。
         2. 平均数：，方差：
         3. 混淆矩阵(Confusion matrix)
            1. pattern的某些顶点可以用一个集合标记，比较这些顶点映射到带有特定label的点的频率
   2. 估计量
      1. 结论：使用F¨urer-Kasiviswanathan对于非OBD的分解运行，估计量仍是无偏的（可推导，论文中由于篇幅不足而略去细节）
      2. 随着样本量的增加，误差收敛的更慢，且收敛接近于0
2. 实验与结果
   1. 实验目标解决的问题（以下算法大多指F¨urer-Kasiviswanathan算法）
      1. 抽样算法在多大程度上近似真实的pattern 统计信息（由穷举法得出）
         1. 取抽样5小时的前10%和后30%进行统计
         2. FK-AD在估计的pattern数量上的相对误差小于10%
      2. OBD与任意分解如何影响算法性能

结论：

* + - 1. FK-OBD FK-AD非常快的收敛到稳定， 在Random时间10%左右
      2. pattern size增加精度略降
      3. 与random差距随pattern size增加而增加
    1. 确定任意分解的方法如何影响算法的性能
       1. FK-ODB和FK-AD性能非常相似
    2. 算法在随机和非随机图上表现如何
       1. FK-OBD和FK-AD即使在理论分析不成立的非随机图上也能表现良好
    3. 使用算法计算近似计数会怎么影响统计关系中的weight learning和inference
       1. 没看懂
    4. 在较长pattern中算法表现
       1. 相比穷举FK-ODB, FK-AD都只使用了穷举所需的一小部分时间就得到了相对稳定的估计
  1. 实验使用的不同算法
     1. Exhaustive: 算法2
     2. Random: 算法3
     3. FACT 通过使用表示数据库的图的进进出出度的汇总统计信息来计算满意的实例数
     4. FK-OBD: 算法4，使用深度优先搜索为pattern找到OBD。它只适用于生成的具有OBD的pattern
     5. FK-AD: 对pattern进行任意分解的算法4。要构造任意的分解，我们只需将OBD平展，这意味着每个分区只包含一个节点
  2. 为实际数据生成pattern graph
     1. 类似我们的extension：给定一个大小为L的pattern集合CL，遍历每个pattern在CL中的所有有效扩展，生成一个大小为L+1的pattern集合CL+1。
     2. 目标pattern：既不是太频繁，也不是太少见
     3. 从CL+1中随机取样pattern，并在较大的数据集上运行FK-OBD或FK-AD 1小时，在较小的数据集上运行5分钟。
     4. 如果pattern有OBD，则使用FK-OBD，如果没有，则使用FK-AD。继续采样，直到每个图案尺寸找到100个符合以下标准的图案

其中，M：图中节点数量，：运行一小时后的平均pattern估计量，：运行一小时后的估计量标准差

* 1. Pattern Evaluation

搁置，暂时没全看懂

## Approximately Counting Embeddings into Random Graphs

本文章为F¨urer-Kasiviswanathan算法，暂时没有去看

# 各文献对本项目参考意义

由于1.1为1.2 1.3的归纳总结，1.5为1.6的进一步优化，本节仅分3个板块讨论

## 1.1~1.3文献

### Pros：

文章给出了非常详细的抽样方法，可以直接更改至sampling中进行实现

在1.2 1.3中也提供了针对预算，收敛的具体解释，具有可执行性

用于量化的指标简单清晰，结论明显

### Cons：

涉及到的目标图为社交网络图，点的label与复杂度与我们目标LDBC不一定符合。在涉及多个不同label时，是否能保持在不同label的情况下维持良好的分布

量化的指标仅仅关注了度的分布和聚类的系数，忽略了对pattern的保持程度。即：在保持度分布和聚类相近的情况下，抽样产生的偏差有很大可能会导致unexpected pattern的产生或expected pattern的消失。

## 1.4文献

### Pros：

关注到了采样时对pattern数量的影响

清晰的讨论了影响采样精准度的影响因素

对于pattern挖掘和pattern计数都有参考意义

### Further

仅对频繁的pattern进行计数，未提及对所有pattern的计数。对少的仅给出一个default概率

Pattern size设置了为4以上，

文章中已提出的further work。自动生成采样的区域数量。

## 1.5~1.6文献

### Pros:

给出图分解思路，OBD, AD。是针对pattern mining进行的图分解策略。相比1.4中的方法，更加关注了graph size大的图，这一点很符合我们要处理的图规模。同时也关注了pattern size更大的图，这一点也很符合。

关注了算法需要的时间，提出了非常多的统计指标进行实验验证。应用至我们的课题时可以按需采纳。

关注了通配符#，这一点同时也涉及到我们具体图和模糊图转化中的pattern计数。（具体细节我没有看了，时间真的不够T\_T）

有清楚的采样收敛方式

源码和数据集都有给，可以复现

### Cons：

文章太TM长了，37页。看的我吐血了。

统计量太多，关于Q5, Q6还有好几页的理论推导和统计指标没看，实在是没时间看明白了

他引用的文献1.6也很长25页。真看不下去

# 基于分布式的考虑

以上几个都可以进行分布式的计算。

1.1中的MHRW在其引用文献1.2中已有提到分布式的多链同时采样

1.4中的分区域随机采样本身就非常符合分布式的逻辑思维。在决定采样区域后直接分别在各主机上进行多个区域采样即可。最后主机间的通信只需要传递最后的topN个频繁子图，收集起来计算accuracy即可。

1.5中的OBD和AD也同样地。选择多个起始点进行mining，加快采样速率的同时避免了单点抽样遇到的局限性。但具体地分解需要进一步讨论

# 文献内容的应用与后续工作

1 需要进一步研读当前的论文，仍存在一些统计上面的推导和统计量意义不全明白。

2 确定对于各文献的参考意义，将文献内容结合地来看，根据所需选择统计指标，统计方法

3 选择文章进行一定程度的复现，并归纳出针对我们工作需要进一步的改变的地方

4 探讨关于通配符等等，其他的思路参考意义

5 将论文方法带入到项目中，结合现在已有框架做个demo进行实验测试

6 针对实验结果进行改善

# 3.31会后计划Graph Sampling

|  |  |  |
| --- | --- | --- |
|  |  |  |
| 关于1.5中论文深入探索 |  |  |
| 1.4伪代码 |  |  |
| 1.4时间性能分析 |  |  |
| 1.4Rust实现 |  |  |
| 1.4实验复现（准确度，时间） |  |  |
|  |  |  |
|  |  |  |

# Question

为了实现随机点取样（无论是有规定label还是无规定label），在gremlin中有几种表达形式：

1 sample

g.V().values(‘###’).sample(N)

g.V().outE().sample(N)

g.V().hasLabel(‘###’).sample(N).by(‘%%%’)

2 coin()

g.V().coin(0.5)

logical中只找到outE inV的操作，那么对于inE呢？是不是只是测试用例里面没说

pegasus.run.blala

flatmap +function 那些

g.getV().flatmap(random)