# Frequent Subgraph Discovery

2017201975 杨越千

# Introduction

- 频繁项集挖掘=>频繁子图挖掘
- 已知若干个图的集合D，希望找到出现频率超过σ|D|的子图

- FSG与AGM的主要区别：AGM每次加一个点来扩充频繁集合，FSG每次加一条边

# Back to Apriori

- 1. 已知size=k的频繁项集F_k，求F_(k+1);
- 2. 从F_k中枚举两个元素gi,gj, 要求|gi∩gj|=k-1, 然后将它们合并，放入候选集合C_k;
- 3. 对C_k中每一个候选者gi, 对其重新计数，若满足gi出现次数>=σ|D|则将其放入F_(k+1);
- 4. k+=1, 重复步骤2-4直到F_k= Φ.

# Key1: canonical labeling

- 怎样给F_k中的元素(子图)排序

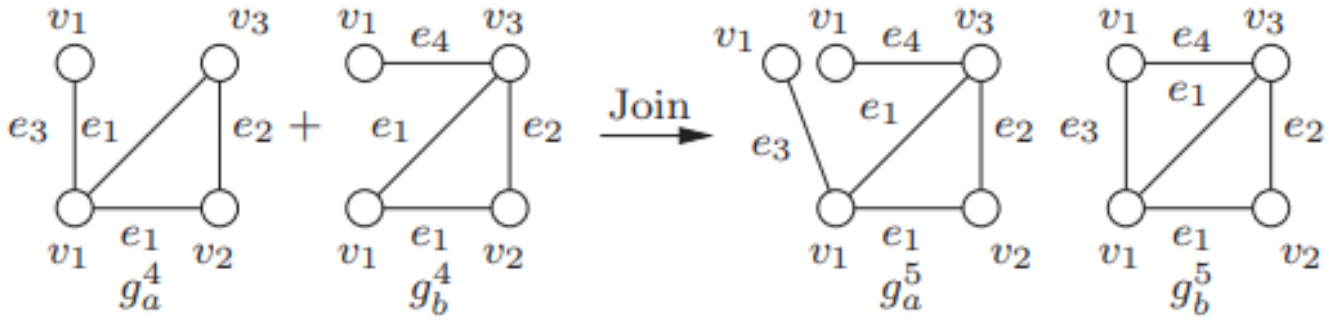
- 对每个子图生成一个”标识码”
- 将邻接矩阵的每一行拼起来拼成一个串作为排序的键（为保证唯一性，取所有可能的键中字典序最小的那个)

# Key2: candidate generation

- 频繁序列中，A和B很容易合并(abc + abd => abcd)
- 频繁子图中，A和B合并的结果可能有多个
- 为方便讨论，设core=A∩B，rA=A-core, rB=B-core

# Key2: candidate generation

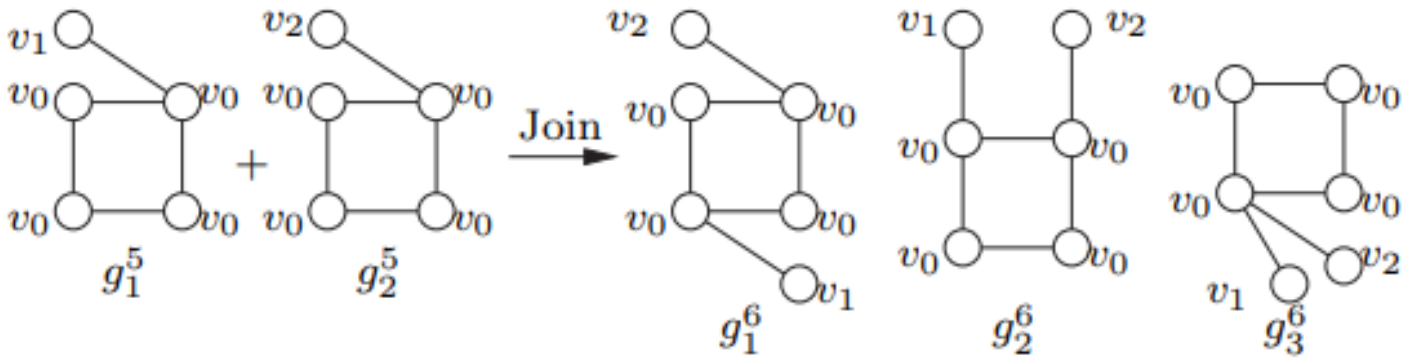
- (1) rA的label和rB的label一致，合并后存在保留两个点/一个点两种情况




(a) By vertex labeling

# Key2: candidate generation

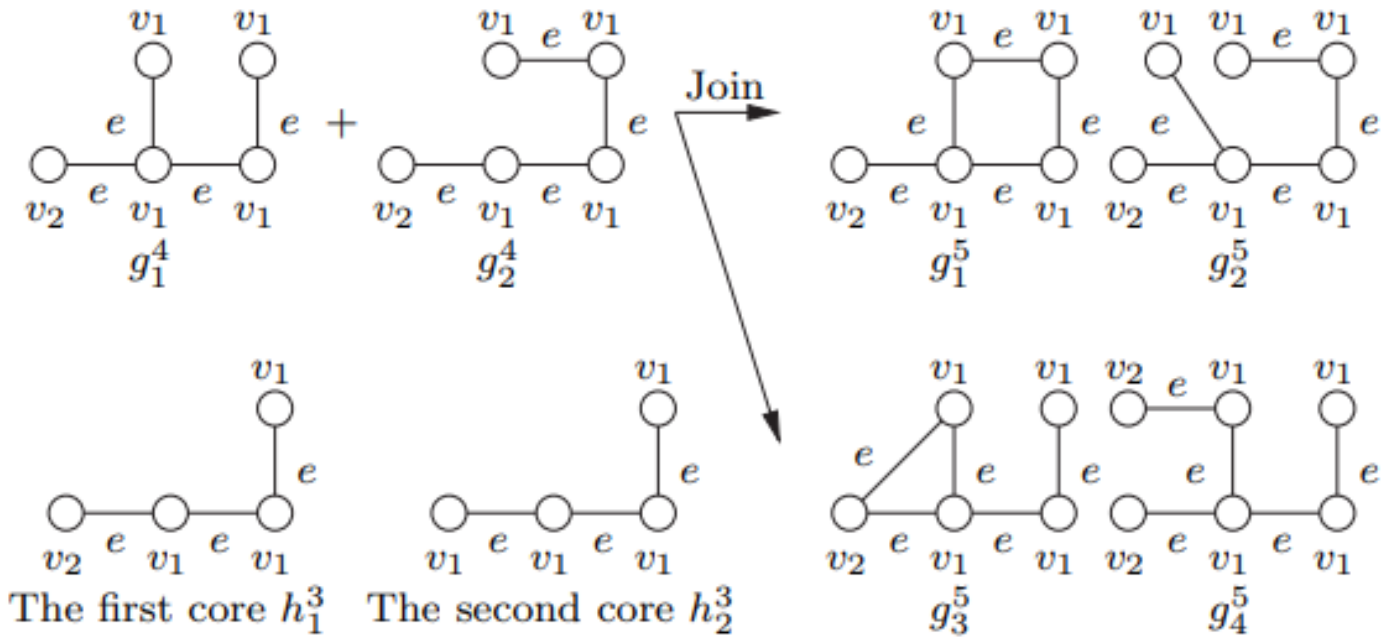
- （2）core可能是自同构的(具体如下)




(b) By multiple automorphisms of a single core

# Key2: candidate generation

- (3) core可能有多个




(c) By multiple cores

# Key3: isomorphism

- 怎样判断两个子图是“相等”(同构)的?
- G1={e1<v0,v1>,e2<v0,v2>}, G2={e2<v2,v0>,e1<v0,v1>}

- 从G1中随机地选一个点,在G2中找是否有结构对应的点(映射),不断地选只到G1所有点选完为止.

- 类似地，寻找G1是否在G2中出现也可以这样做

# Back to Apriori

- 1. 已知size=k的频繁项集F_k，求F_(k+1);
- 2. 从F_k中枚举两个元素gi,gj, 要求|gi∩gj|=k-1, 然后将它们合并，放入候选集合C_k;
- 3. 对C_k中每一个候选者gi, 对其重新计数，若满足gi出现次数>=σ|D|则将其放入F_(k+1);
- 4. k+=1, 重复步骤2-4直到F_k= Φ.

# reference


- 论文:Frequent Subgraph Discovery, Michihiro Kuramochi and George Karypis, 2001
- Web:http://www.cs.unc.edu/Research/datamining/data/kuramochi01frequent.pdf