一、AG-Skyline

1. AG-Skyline定义

给定一个数据集P，具有n个点，所有点均属于一个维度为d的空间。令，是两个具有k个点的不同组，我们说 ag-domiante 如果对于所有的对，，并且对于至少一对，。AG-Skyline包含所有不被任何其他具有相同大小的组ag-dominate的组。

1. 对比g-dominate和ag-dominate。g-dominate只要求在两个组各自点集的某一个组合次序上具有一一dominate关系。而ag-dominate要求每个点对另一个组中的所有点均具有dominate关系，换句话说，ag-dominate要求在所有的组合次序上均具有一一dominate关系。显然，ag-dominate关系比g-dominate更严格，所以在相同的数据集和相同的k值上，AG-Skyline是G-Skyline的超集。

二、判断一个组是否属于AG-Skyline，为AG-Group

1. 对于一个大小为k的组：。我们判断其是否属于AG-Skyline，就是看是否找得到一个组，组里面任意一个点对于G里面的每一个点，均具有dominate或者等于关系，当然必须存在至少一个dominate关系。
2. 我们定义一个点的Unit group：Unit group包含和的所有父亲，记为。所以如果存在一个点对于G里面的一个点，具有dominate或者等于关系，显然：。而如果存在一个点对于G里面的任何一个点均具有dominate或者等于关系，对应的：。令，即。
3. 如果， ag-dominate G，则对于中的每一个点，，显然。
4. 判断一个组G是否属于AG-Skyline，即判断是否存在一个组，其ag-dominate G，。可以求。如果，证明找得到一个大小为k的组，组里面的点均来自，显然这个组ag-dominate G。
5. 对于，如果存在某个点，，一定属于AG-Skyline。因为，所以找不到一个大小为k的组，组里面的点均来自。
6. 对于，如果存在某个点，。那么只要存在，不是的孩子，那么一定属于AG-Skyline。因为，，并且只有是的孩子时，。
7. 对于，如果所有点，。我们可以通过求来判断其是否属于AG-Skyline。

三、计算数据集P的AG-Skyline。

1. 由于判断一个组是否属于AG-Skyline，我们需要利用一个点的父亲和孩子集合，所以，类似G-skyline的做法，我们首先计算P的所有层Skyline，然后生成DSG，这个DSG与G-skyline中的DSG的区别是每个DSGNode不仅包含该点的父亲和孩子集合parents, children，还包含该点的unit group: unit，unit就是自身加上parents构成的集合。
2. 预处理：（1）当某个点自身的unit的大小小于k时，可以直接构造包含该点的AG-Group。即在剩下的点中任选k-1个点，与该点就能构成AG-Group。（2）当某个点自身的unit的大小等于k时，也可以直接构造包含该点的AG-Group。即在剩下的点中选k-1个点，这k-1个点不能全是该点的孩子就能构成AG-Group。
3. 我们令Group中所有点的unit的交集为AG-Group的totalUnit，令AG-Group中所有点的children的交集为AG-Group的totalChildren。我们可以构造一棵Group枚举树。由Group初始大小为0，每次往里面添加一个点直到group大小为k。
4. 剪枝策略1: 每次计算新的group的totalUnit和totalChildren，如果totalUnit大小小于k，则包含此group的大小为k的group一定为AG-Group。如果totalUnit大小等于k，则包含此group的大小为k的group剩余的点只要不全属于totalChildren集合，就一定是AG-Group。这两种情况可以直接输出AG-Group集合，结束往group增加新元素。
5. 剪枝策略2:每个group都有一个tail set，我们每次都向group中添加tail set中的一个candidate point。仅当candidate point属于group的totalUnit中某个点的孩子集合时，新的group的totalUnit才有可能不为空，当candidate point不满足这种条件时，新的group的totalUnit必为空，可以直接利用新group构造AG-Group输出。所以我们将遍历tail set删减为仅检查满足这种条件的candidate。
6. 算法：

input: DSG and group size k

output: AG-Skyline(k) groups

initialize the Group (0) at root node as an empty set and its tail set as all points from DSG after preprocessing

for i = 1 to k:

for each Group (i-1) G do

for each point in Group’s totalUnit do

add ’s children to Children Set

for each point in Tail Set do

if is not in Children Set then

construct AG-Group concluding G and

delete

for each remaining point in Tail Set do

add to G to form a Group (i)

if the size of new candidate group’s totalUnit < k do

construct AG-Group concluding new candidate group

delete

if the size of new candidate group’s totalUnit = k do

construct AG-Group concluding new candidate group and the rest points don’t all belong to children set of new candidate group

delete