Review on some graph mining papers

June 7, 2015

1 Graph-based knowledge discovery: compression and frequency

1. Two primary types of graph-based data miners, frequent subgraph and compression-based miners. With the frequent subgraph miners, the most interesting substructure is the largest one (or ones) that meet the minimum support. The compression-based graph miners discover those subgraphs that maximize the amount of compression that a particular substructure provides a graph.
2. GASTON is to return all frequent substructures in a graph using a depth-first search on the input graphs.
3. SUBDUE aims to return the substructures that compress the graph the best.

2 A quantitative comparison of the subgraph ners MoFa, gSpan, FFSM, and Gaston

1. SUBDUE, AGM, FSG generate refinements in a breadth first way.
2. MoFa, gSpan, FFSM, Gaston are depth-first search methods.
3. DFS approaches need less memory to store appearance lists because the number of lists that have to be stored in memory is proportional to the depth of the lattice (i.e., the size of the biggest graph), whereas it is proportionl to its width (i.e., the maximal number of subgraphs in one level) in BFS.
4. On MoFa, gSpan, FFSM, and Gaston
   - All these four algorithms are DFS-based methods.
   - MoFa (molecule fragment miner, 2002): It still generates many isomorphic fragments and then use standard isomorphism testing to prune duplicates.

- gSpan (graph-based substructure pattern, 2002) uses a canonical representation for graphs, called dfs-code. A dfs-traversal of a graph defines an order in which the edges are visited. Refinement generation is restricted by gSpan in two ways: first, fragments can only be extended at noes that lie on the rightmost path of the dfs-tree. Secondly, fragment generation is guided by occurrence in the appearance lists. These two pruning rules cannot fully prevent isomorphic fragment generation, gSpan computes the canonical (lexicographically smallest) dfs-code for each refinement by means of a series of permutation.
- FFSM (Fast frequent subgraph mining, 2003) represents graphs as triangle matrices.
- Gaston (Graph/sequence/tree extraction, 2004).
- Regarding to the runtime, MoFa is the slowest. gSpan is faster then FFSM on large datasets. Gaston is the fastest.
- Gaston does not produce duplicates for non-cyclic graphs. FFSM and MoFa tend to perform rather weak in terms of extenion methos and pruning rules.
- gSpan uses the least memory as it does not use embedding lists. MoFa needs less memory than FFSM, even though MoFa stores both edges and nodes while FFSM only stores nodes. Gaston need the most memory because embedding lists for a new fragment are built based on the embedding lists of the parent.
- All these algorithms scale linearly with the database size, but with different factors.
- FFSM and Gaston cannot work with directed graphs without major changes. MoFa can work with directed graphs. gSpan can also be used for directed graphs with minor changes.
-