Frequent subgraph mining

Philippe Fournier-Viger
http://www.philippe-Fournier-viger.com

Source code and datasets available in the SPMF library
Frequent subgraph mining

- A graph (图表) is a set of vertices (顶点) and edges (边)
- e.g.

This graph has four vertices (in yellow color). Each vertices has a label (10 or 11) that may not be unique.

This graph has five edges (black lines) Each edge has a label (20, 21, 22, 23) that may not be unique.
Types of graphs

**connected graph**: by following the edges, it is possible to go from any vertex to any other vertices

**disconnected graph**: a graph that is not a connected graph

- **A connected graph**: in this graph, it is possible to go from any vertex to any other vertices by following the edges.
- **A disconnected graph**: this graph is disconnected because Vertex A cannot be reached from the other vertices by following the edges.

**Example**: a graph where it is possible to go from any city to any other cities by following the roads.
Types of graphs

**Undirected graph (无向图):** Edges are bidirectional

**Directed graph (有向图):** Edges are unidirectional

A real-life example:
- Graphs where vertices are cities and edges are roads
- Some roads are "one-way" while others are bidirectional
Analyzing graphs

- Many data mining tasks on graphs:
  - detecting communities, predicting friendship links, detecting influence between users, etc.
  - what is our goal?

- **Frequent subgraph mining:**
  - discover *interesting subgraph(s)* appearing often in a set of graphs (a graph database)
Frequent subgraph mining

Input:
• a graph database (a set of graphs)
• a minimum support threshold (minsup).

Example:

A graph database

\[(\text{minsup} = 3)\]
Output:
all subgraphs appearing in at least $\text{minsup}$ graphs.

A graph database

Graph 1

Graph 2

Graph 3

Frequent subgraph 1:

Frequent subgraph 2:

Frequent subgraph 3:

$\text{minsup} = 3$
Output:
all subgraphs appearing in at least \textit{\textit{minsup}} graphs.

\textbf{A graph database}

\begin{itemize}
  \item Graph 1
  \begin{itemize}
    \item Node 10
    \item Node 11
    \item Edge 20
    \item Edge 23
    \item Edge 22
  \end{itemize}
  \item Graph 2
  \begin{itemize}
    \item Node 10
    \item Node 11
    \item Edge 20
  \end{itemize}
  \item Graph 3
  \begin{itemize}
    \item Node 10
    \item Node 11
    \item Node 21
    \item Node 23
    \item Node 22
    \item Edge 20
  \end{itemize}
\end{itemize}

\textit{\textit{minsup}} = 3

\begin{itemize}
  \item Frequent subgraph 1:
    \begin{itemize}
      \item Node 10
      \item Node 11
    \end{itemize}
  \item Frequent subgraph 2:
    \begin{itemize}
      \item Node 11
    \end{itemize}
  \item Frequent subgraph 3:
    \begin{itemize}
      \item Node 10
      \item Node 11
      \item Edge 20
    \end{itemize}
\end{itemize}

This subgraph has a support of 3
Frequent subgraph mining with a single graph

- A variation of the previous problem.
- We want to find **frequent subgraphs** in a **single large graph**.
- The **support of a subgraph** is the number of times that it appears in the single input graph.

A single graph:

```
10 --- 21 --- 10
  |      |
  v      v
23 --- 21 --- 10
```

10 --- 20 --- 11
  |      |
  v      v
20 --- 20 --- 11
```
Frequent subgraph mining with a single graph

A single graph

\[ \text{minsup} = 2 \]
Frequent subgraph mining with a single graph

A single graph

Frequent subgraph 1

Frequent subgraph 2

Frequent subgraph 3

Frequent subgraph 4

Frequent subgraph 5

This subgraph has a support of 2

$\text{minsup} = 2$
Algorithms for subgraph mining

- Several algorithms:
  - FFSM, GSPAN, Gaston, etc.

- The same algorithm can usually be applied on a single graph or multiple graphs.

- Other variations:
  - finding frequent paths
  - finding frequent trees
  - finding closed/maximal subgraphs…
  - …
Performance comparison

Authors of data mining papers often do not compare their algorithms with the best ones published until now.

Frequent subgraph mining (before 2014)

Legend: arrow $X \rightarrow Y$ from an algorithm $X$ to an algorithm $Y$ indicates that $X$ was shown to be a better algorithm than $Y$ in terms of execution time by the authors of $X$ in an experiment.