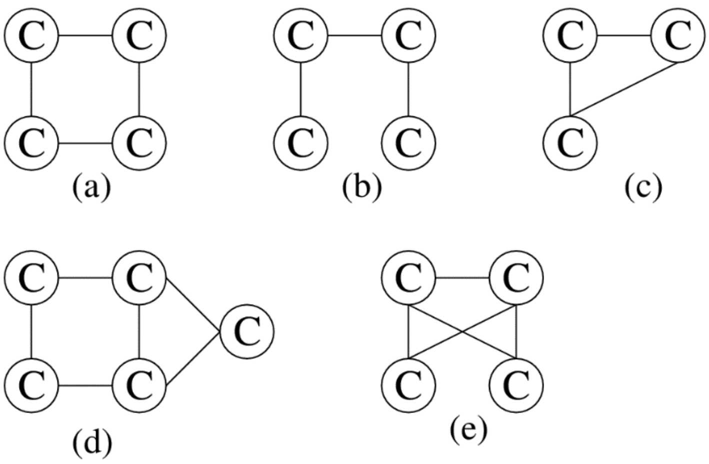
# Assignment 3

**The deadline of assignment 3 is:**

**Fri 25 May, 5:00 pm**

## Question 1 (5 marks)

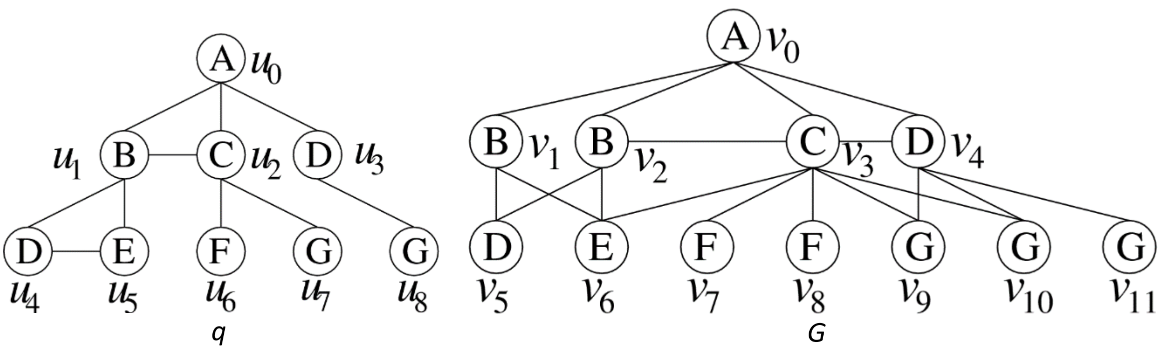
Given a graph database *D* containing following graphs:



1) Suppose *minFreq* = 3, draw at least 4 frequent patterns/fragments in the graph database *D*. A graph/pattern *g* is frequent if its occurrence frequency is no less than *minFreq*. (5 marks)

## Question 2 (10 marks)

Given the following query *q* and data graph *G*.




1) Please draw a Neighborhood Equivalence Class tree (NEC tree) of query *q*. (5 marks)

The Neighborhood Equivalence Class(NEC) of a query vertex $u$ is a set of query vertices, which are equivalent to $u$. The equivalence is defined as follows:

Let $\cong$ be an equivalence relation over all query vertices in $q$ such that,

$u_i\,(\in V(q)) \cong u_j\,(\in V(q))$ if for every embedding $m$ that contains $(u_i, v_x)$ and $(u_j, v_y)$ $(v_x, v_y \in V(g))$, there exists an embedding $m'$ such that $m' = m - \{(u_i, v_x), (u_j, v_y)\} \cup \{(u_i, v_y), (u_j, v_x)\}$.

Please read the following paper for more detail:

**Han, W. S., Lee, J., & Lee, J. H. (2013, June). Turbo iso: towards ultrafast and robust subgraph isomorphism search in large graph databases. In Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data (pp. 337-348). ACM.**

2) Please decompose the vertex set of query $q$ according to Core-Forest-Leaf decomposition. That is, decompose the vertex set of $q$ into three sets including the core-set, the forest-set and the leaf-set. (5 marks)

Given a query $q$, the Core-Forest-Leaf decomposition consists of core-forest decomposition and forest-leaf decomposition.

**Core-Forest Decomposition**

Edges of $q$ can be categorized into two categories regarding a spanning tree $q_T$ of $q$: edges in $q_T$ are called tree edges while edges of $q$ that are not in $q_T$ are called non-tree edges regarding $q_T$.

Our core-forest decomposition is to compute a small dense subgraph containing all non-tree edges regarding any spanning tree, which is defined as follows. Given a query $q$, the core-forest decomposition of $q$ is to compute the minimal connected subgraph $g$ of $q$ that contains all non-tree edges of $q$ regarding any spanning tree of $q$; g is called the core-structure of $q$. The subgraph of $q$ consisting of all other edges not in the core-structure called the forest-structure of $q$, denoted $T$. We call the vertex set of the core-structure as the core-set $\boldsymbol{V_C}$ and the forest-structure of $q$ doesn't contain any vertices in $\boldsymbol{V_C}$.

**Forest-Leaf Decomposition**

Given the forest-structure $T$, rooting each tree in forest-structure at its connection vertex with core-structure. The set $\boldsymbol{V_I}$ is called the leaf set which contains all the degree-one vertices in the trees of forest-structure. The set of vertices not in $\boldsymbol{V_C} \cup \boldsymbol{V_I}$ is called the forest set $\boldsymbol{V_T}$.

Let $V(q)$ denotes the vertex set of $q$, $V(q) = \boldsymbol{V_C} \cup \boldsymbol{V_T} \cup \boldsymbol{V_I}$ and $\boldsymbol{V_C} \cap \boldsymbol{V_T} = \boldsymbol{V_C} \cap \boldsymbol{V_I} = \boldsymbol{V_T} \cap \boldsymbol{V_I} = \emptyset$.

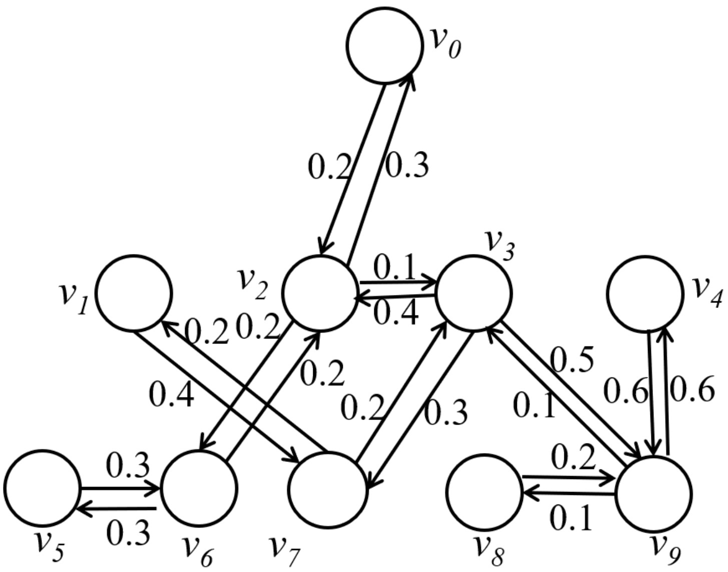
Please read the following paper for more detail:

**Bi, F., Chang, L., Lin, X., Qin, L., & Zhang, W. (2016, June). Efficient subgraph matching by postponing cartesian products. In Proceedings of the 2016 International Conference on Management of Data (pp. 1199-1214). ACM.**

Considering Figure 4 in the above paper, we can decompose the vertex set of $q$ into:
The core set: $u_0$, $u_1$, $u_2$
The forest set: $u_3$, $u_4$, $u_5$, $u_6$
The leaf set: $u_7$, $u_8$, $u_9$, $u_{10}$

# Question 3 (5 marks)

Given a social influence graph $G_1$ as following:




1) Choose one activated seed $s$ from $v_0$ ~ $v_9$ which can generate the largest influence spreads (i.e., let $w(s) = 1$, maximize $\sum_{i=0}^{9} w(v_i)$). (5 marks)

Initially, all the vertices are inactivated. We define $w(u)$ as the probability of a vertex $u$ which can be activated. In graph $G_1$, $p(u, v)$ on a directed link from $u$ to $v$ is the probability that $v$ is activated by $u$ after $u$ is activated (e.g., ~~$p(v_0, v_1) = 0.3$~~ $p(v_0, v_2)=0.2$). For example, $w(v_0) = 1$, $w(v_2) = 0.2$, and $w(v_3) = 0.2 * 0.1 = 0.02$ if we choose $v_0$ as the activated seed.