Exploiting Efficient Densest Subgraph Discovering Methods for Big Data

Bo Wu and Haiying Shen

Agenda

- Introduction of Graph
- Densest subgraph problem
- Bottleneck of current algorithm

Introduction of Graph

- A graph is a structure that comprises a set of ***vertices*** and ***edges***
  - ***Vertices*** are the elementary units of graph, which can represent as

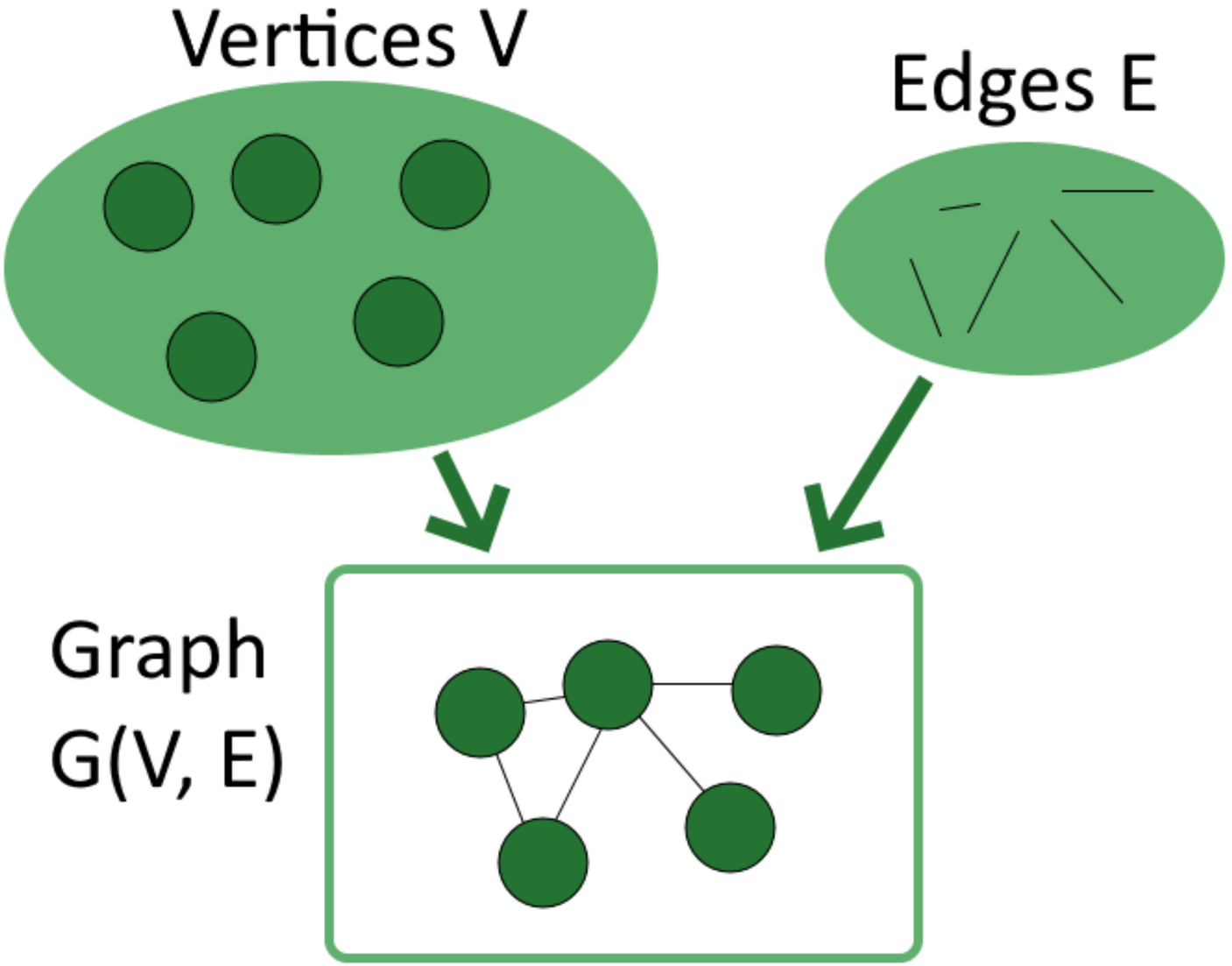
    $$V = \{v_1, v_2, \ldots v_n\}$$

  - ***Edges*** are connections between 2 ***vertices***, including vertex itself

    $$E = \{e_1, e_2, \ldots, e_n\}$$

- Therefore, we can define a graph G as the structure $G(V, E)$

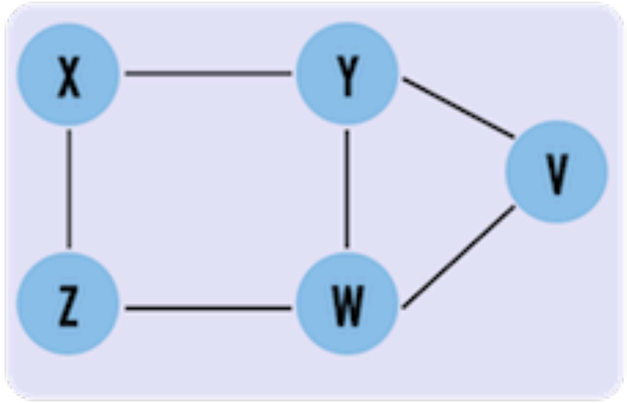
Introduction of Graph



Representation of a Graph

Undirected Graph

G_1

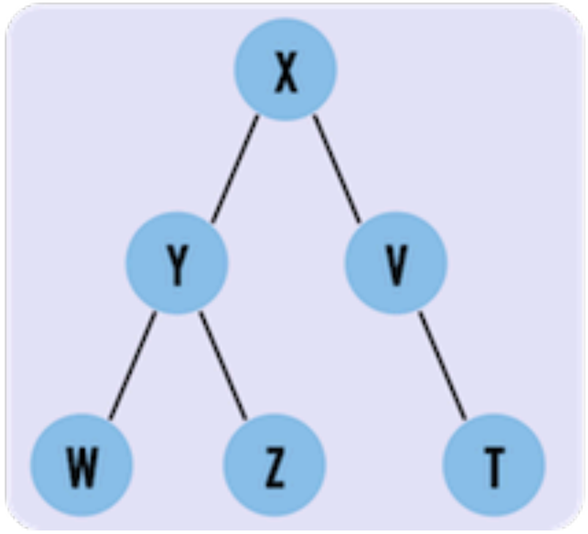


$V(G_1) : \{ X, Y, Z, W, V \}$

$E(G_1) : \{ (X, Y), (X, Z), (Y, W), (Y, V), (Z, W), (W, V) \}$

$|V| = 5 , |E| = 6$

G_2



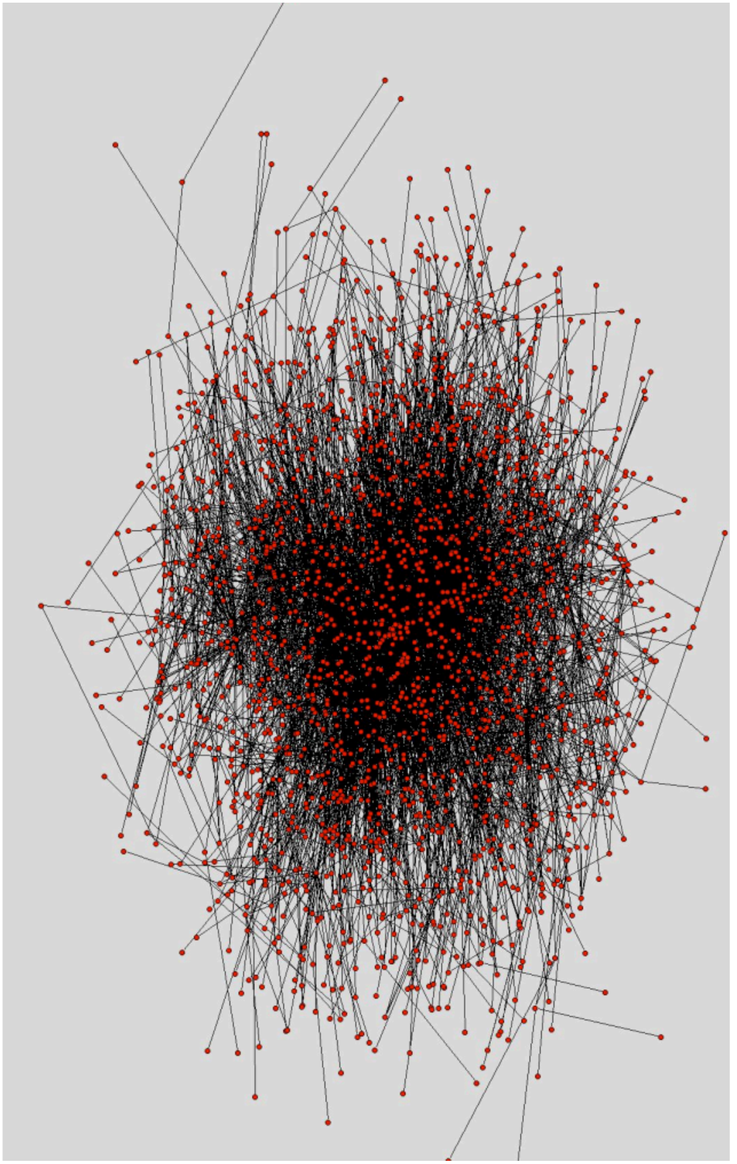
$V(G_2) : \{ X, Y, Z, W, V, T \}$

$E(G_2) : \{ (X, Y), (X, V), (Y, W), (Y, Z), (V, T) \}$

$|V| = 6 , |E| = 5$

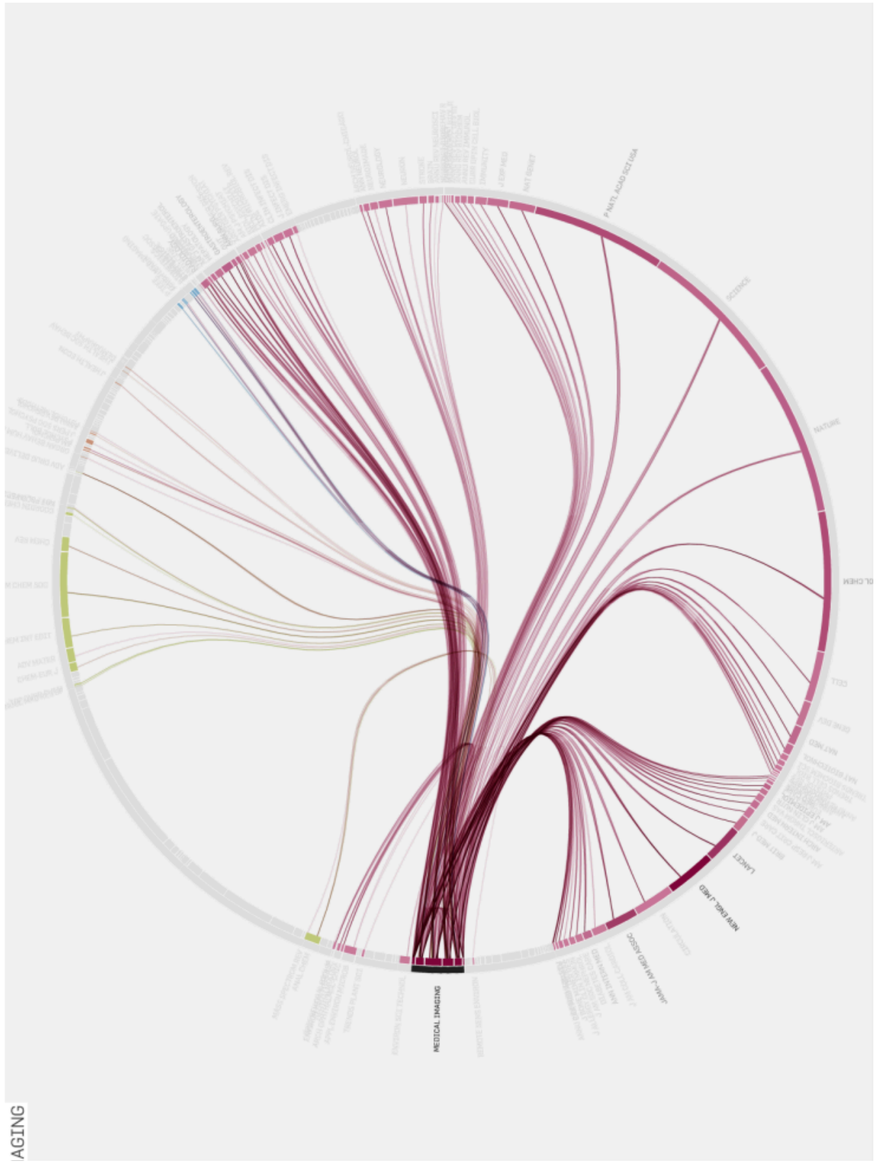
Types of Graphs

- links denote a social interaction
  - networks of acquaintances
  - collaboration networks
    - actor networks
    - co-authorship networks
    - director networks
  - phone-call networks
  - e-mail networks
  - IM networks
  - sexual networks



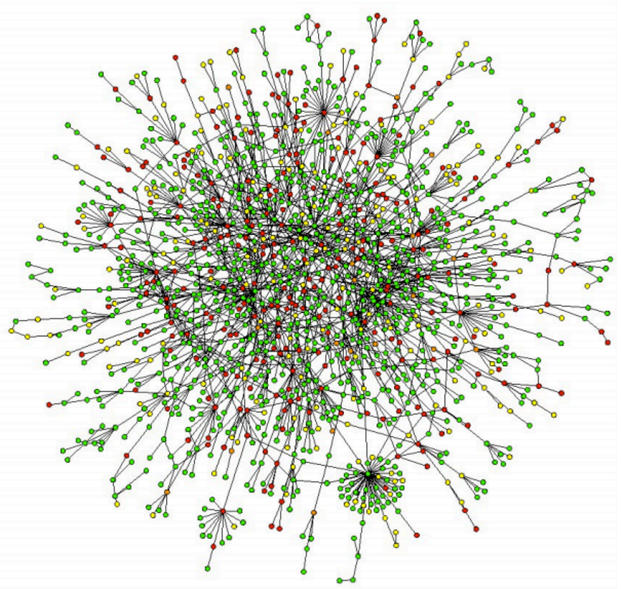
Types of Graphs

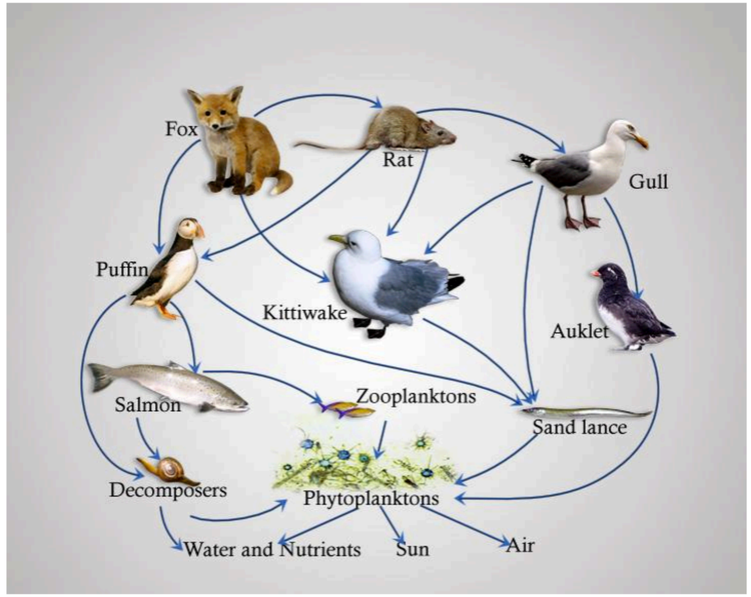
- nodes store information, links associate information
  - citation network (directed acyclic)
  - the web (directed)
  - peer-to-peer networks
  - word networks
  - networks of trust
  - software graphs
  - bluetooth networks
  - home page/blog networks



Types of Graphs

- biological systems represented as networks
  - protein-protein interaction networks
  - gene regulation networks
  - gene co-expression networks
  - metabolic pathways
  - the food web
  - neural networks



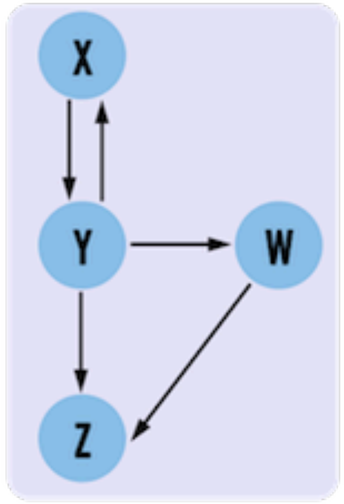


Introduction of Subgraph

- A subgraph $G^{'}$ is a structure that comprises partial set of ***vertices*** and ***edges*** of the original graph G
  - $V(G^{'}) \subseteq V(G)$
  - $E(G^{'}) \subseteq E(G)$
- Therefore, we can define a $G^{'}$ as the subgraph of G

Introduction of Subgraph

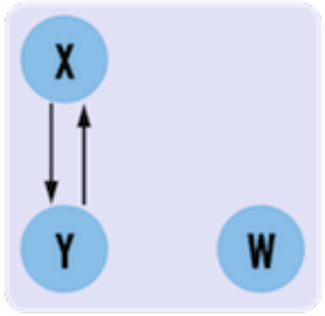
Graph G



$V(G) : \{ X, Y, Z, W \}$
$E(G) : \{ X \rightarrow Y, Y \rightarrow X, Y \rightarrow Z, Y \rightarrow W, W \rightarrow Z \}$
$|V| = 4 , |E| = 5$

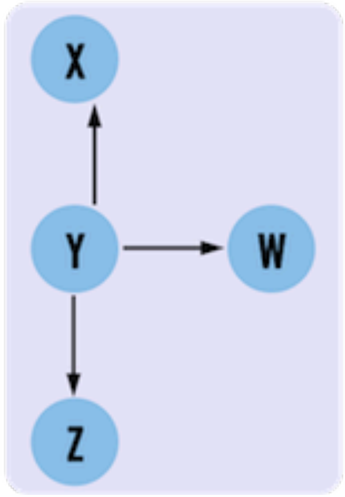
Subgraph of G

G_1



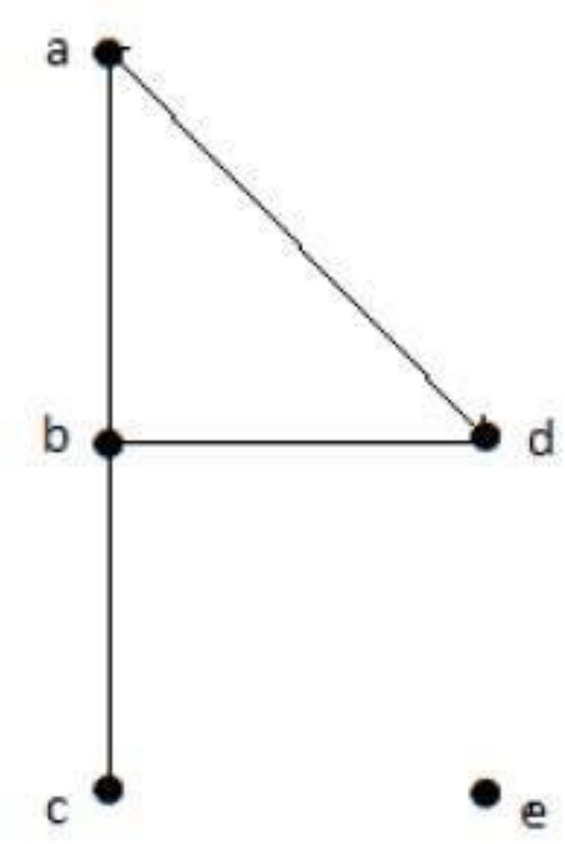
$V(G_1) : \{ X, Y, W \}$
$E(G_1) : \{ X \rightarrow Y, Y \rightarrow X \}$

G_2



$V(G_2) : \{ X, Y, Z, W \}$
$E(G_2) : \{ Y \rightarrow X, Y \rightarrow Z, Y \rightarrow W \}$

Introduction of degree of vertex



- deg(a) = 2, as there are 2 edges meeting at vertex 'a'
- deg(b) = 3, as there are 3 edges meeting at vertex 'b'
- deg(c) = 1, as there is 1 edge formed at vertex 'c'

  So 'c' is a **pendent vertex**
- deg(d) = 2, as there are 2 edges meeting at vertex ‘d'
- deg(e) = 0, as there are 0 edges formed at vertex 'e'

  So 'e' is an **isolated vertex**

Introduction of Density of graph

- Density of graph indicates how “full” a graph is, which estimates the comparison between the number of ***edges*** and the ***maximal*** number of ***edges***

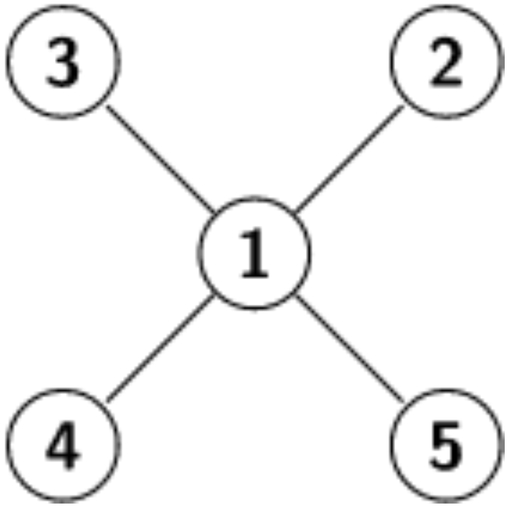
- We can define Density D as

$$D = \frac{Number\ of\ edges\ in\ G}{Maximal\ number\ of\ edges\ in\ G} = \frac{|E|}{\binom{|V|}{2}} = \frac{2|E|}{|V|(|V|-1)}$$

- The maximal density is 1, and the minimal density is 0

- Density can help us to judge whether graph G is dense or not

Introduction of Density of graph



- $G(V, E)$ where $V = \{1,2,3,4,5\}$ and $E = \{(1,2), (1,3), (1,4), (1,5)\}$
- $|V| = 5, |E| = 4$
- $Max(V) = \begin{pmatrix} |V| \\ 2 \end{pmatrix} = \frac{5 \times (5-1)}{2} = 10$
- $D(V, E) = \frac{|E|}{Max(V)} = \frac{4}{10} = 0.4$
- Because $D(V, E) < 0.5$, we can conclude that $G(V, E)$ is a sparse graph

Introduction of Dense subgraph

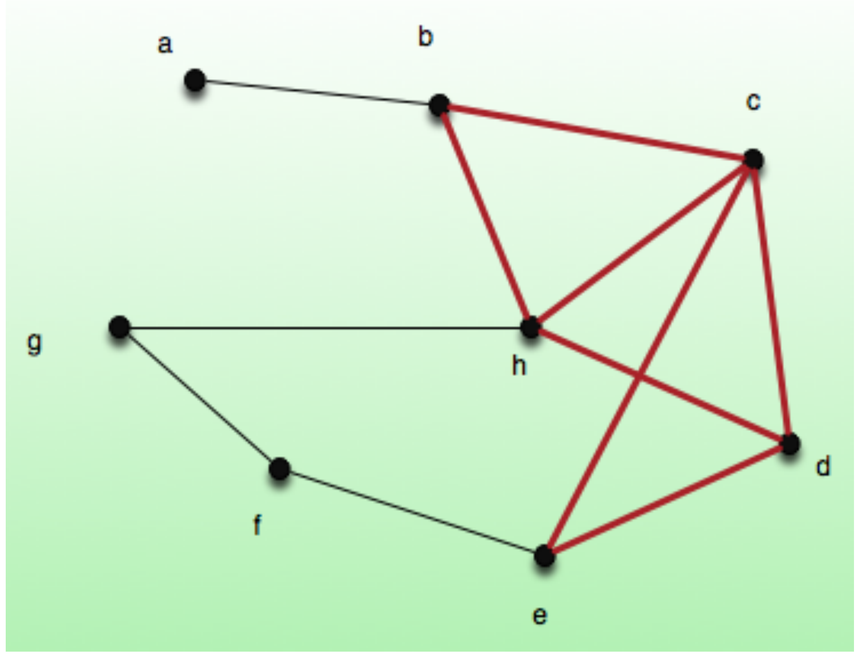
- Dense subgraph estimates the comparison between the number of ***edges of the subgraph*** and the number of ***vertices of the subgraph***

- Let $G(V, E)$ be a graph and $G^{'}(V_{G^{'}}, E_{G^{'}})$ be a subgraph of $G(V, E)$

- We can define Density $D_{G^{'}} = \frac{Number\ of\ edges\ in\ G^{'}}{Number\ of\ vertices\ in\ G^{'}} = \frac{|E_{G^{'}}|}{|V_{G^{'}}|}$

- Unlike density of graph, Dense subgraph indicates ***the ratio between edges and vertices***

Introduction of Dense subgraph



- $G(V, E)$ where $V = \{a, b, c, d, e, f, g, h\}$ and

  $E = \{(a, b), (b, c), (b, h), (c, d), (c, e), (c, h), (d, e), (d, h), (e, f), (f, g), (g, h)\}$

- $|V| = 8, |E| = 11$

- $D(G) = \frac{|E|}{|V|} = \frac{11}{8} = 1.375$

- $G'(V_{G'}, E_{G'})$ where $V_{G'} = \{b, c, d, e, h\}$ and

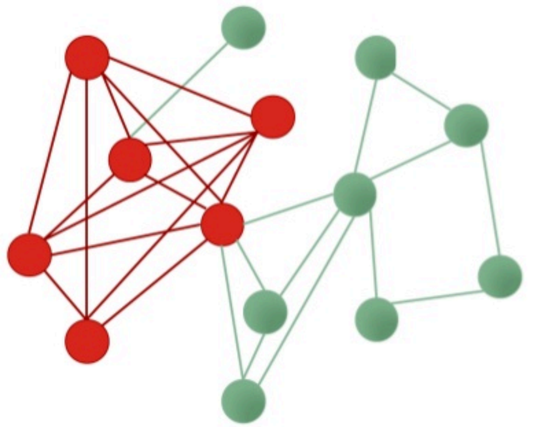
  $E = \{(b, c), (b, h), (c, d), (c, e), (c, h), (d, e), (d, h)\}$

- $|V_{G'}| = 5, |E_{G'}| = 7$

- $D(G') = \frac{|E_{G'}|}{|V_{G'}|} = \frac{7}{5} = 1.4$

Densest subgraph problem (DSP)

- Densest subgraph problem is that of finding a ***subgraph of maximum density***
- Densest subgraph discovery is a key graph mining primitive
- Densest subgraph has lots of real application such as
  - Spam detection in the Web graph
  - Correlation mining
  - Bioinformatics
  - Mining Twitter data



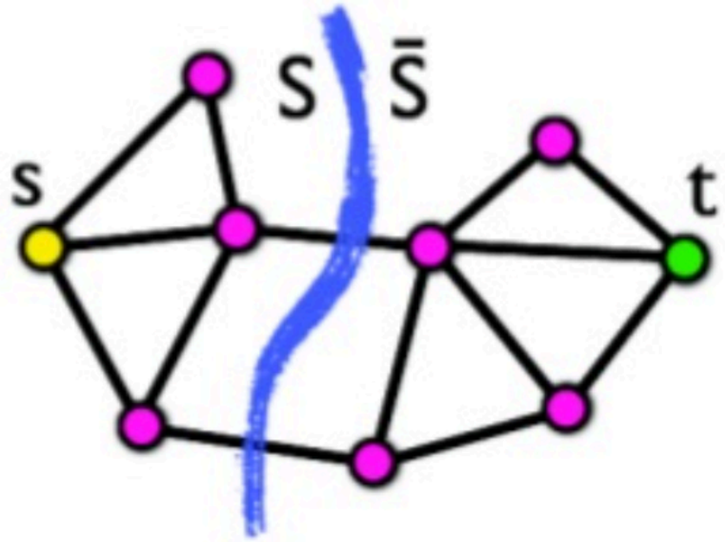
Application of Densest subgraph

- Youtube social network and ground-truth communities
  - Youtube is a video-sharing web site that includes a social network
  - In the Youtube social network, users form friendship each other and users can create groups which other users can join
  - We consider such user-defined groups as ground-truth communities

Goldberg's algorithm of DSP

- Requires: min-cut problem

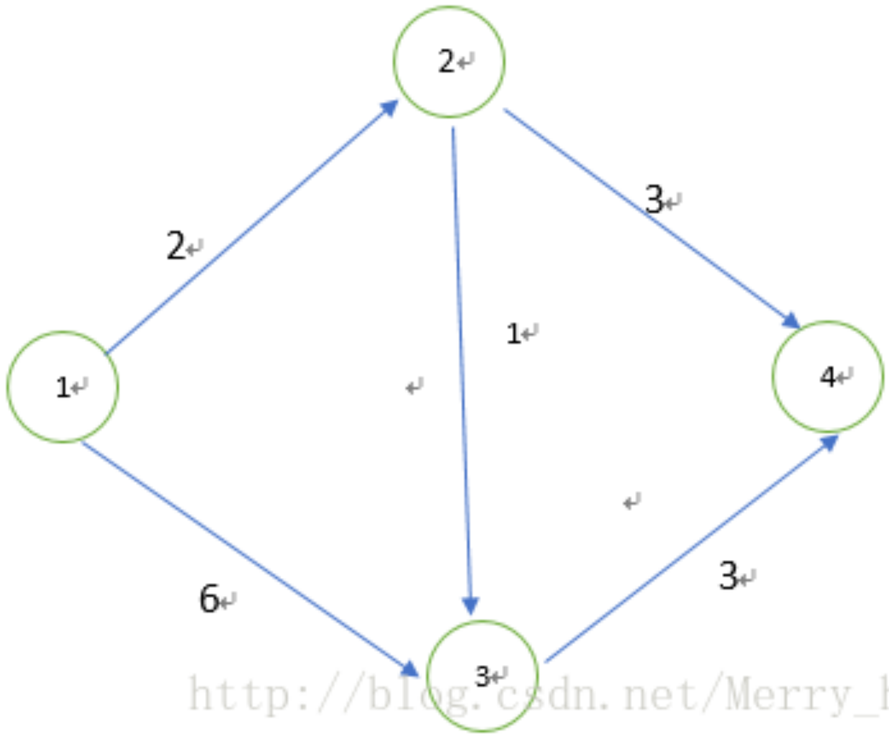
min-cut problem

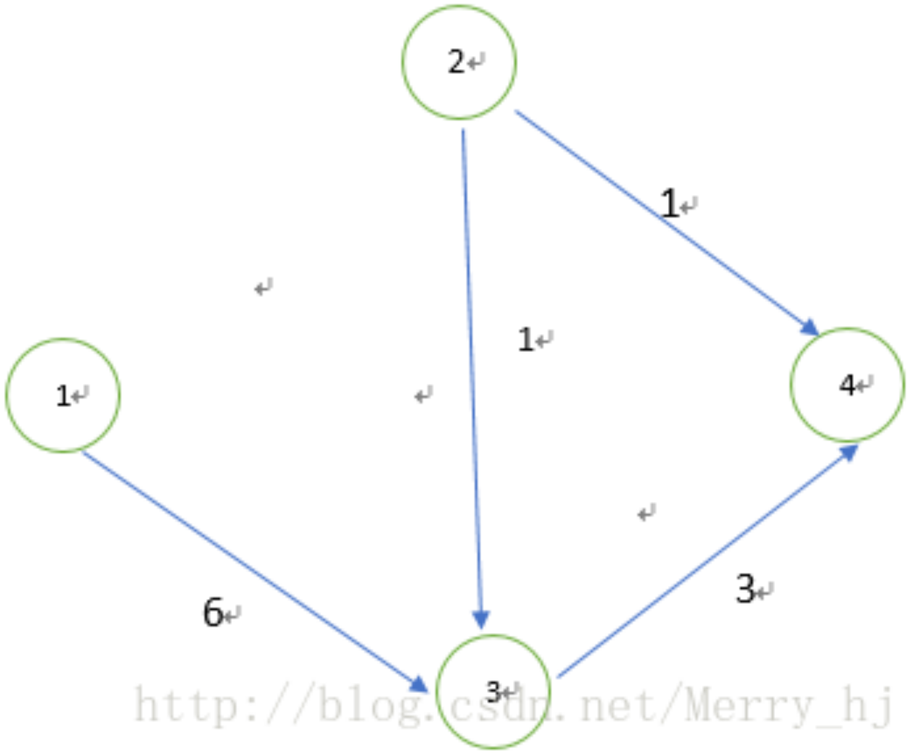


- source $s \in V$, destination $t \in V$
- find $S \subseteq V$, s.t.,
- $s \in S$ and $t \in \bar{S}$, and
- minimize $e(S, \bar{S})$
- polynomially-time solvable
- equivalent to max-flow problem

Goldberg’s algorithm of DSP

- Example: From Vertex 1 to 4

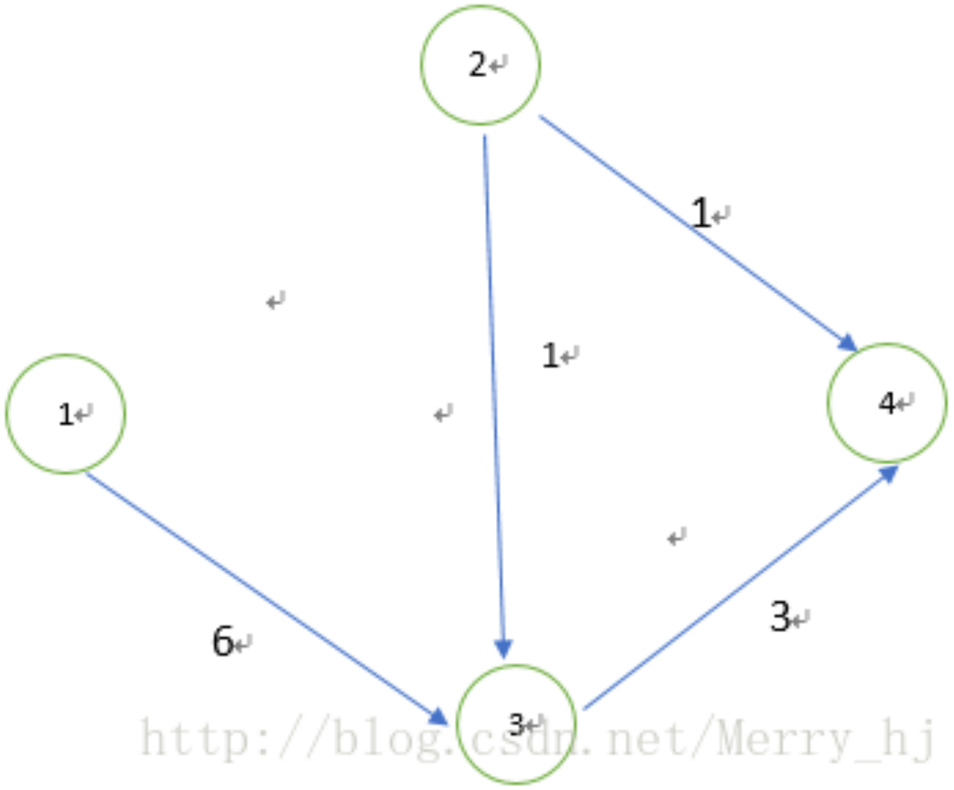


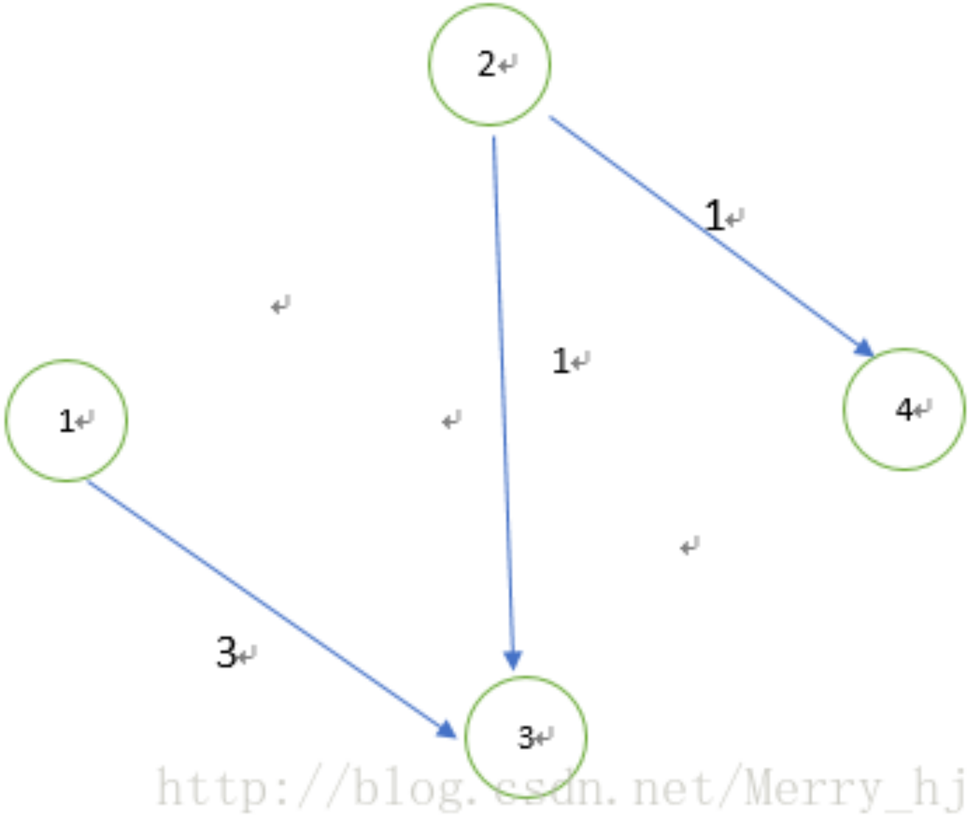


- (1,2,4) min(2,3)=2 and remove (1,2)

Goldberg’s algorithm of DSP

- Min-cut is 2+3=5, Max-flow is 2+3=5



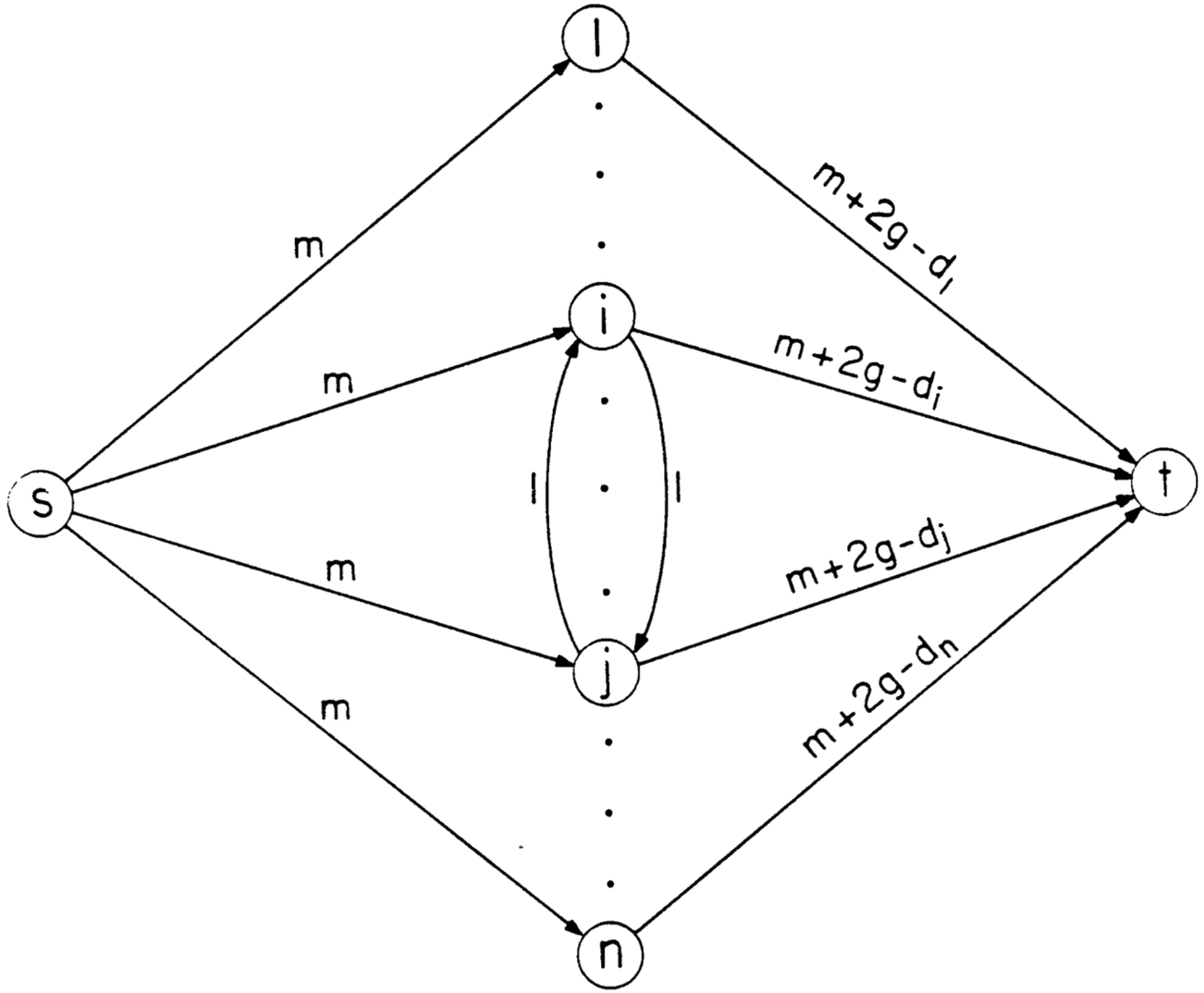


- (1,3,4) min(6,3)=3 and remove (3,4), no path 1 to 4, algorithm end

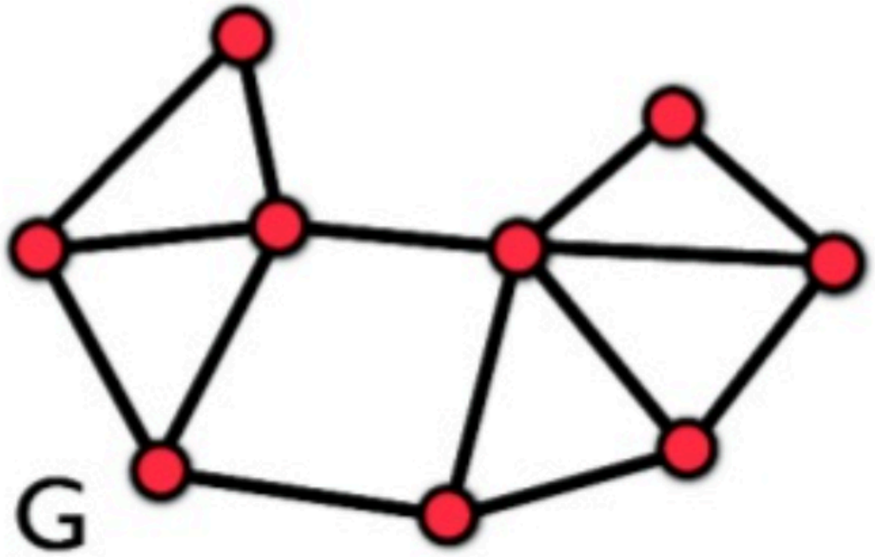
Goldberg's algorithm of DSP

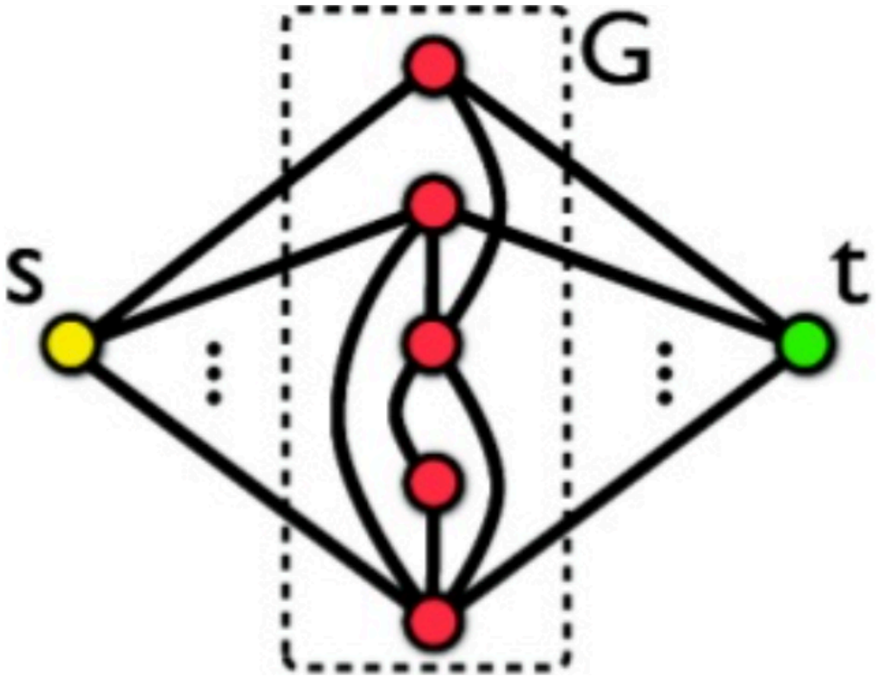
- Let d_i be the degree of vertex i of G. Given a "guess" g, we convert G into network $N = (V_N, E_N)$ as follow
- Add source s and sink t to the set of vertices of G_i, replace each undirected edge of G by two directed edges of capacity 1 each
- Connect source to each node i of G by an edge of capacity of m
- Connect sink to each node i of G by an edge of capacity of $m + 2g - d_i$

Goldberg's algorithm of DSP



Goldberg’s algorithm of DSP





Goldberg's algorithm of DSP

- A partition of V_N into two sets, S and T, such that $s \in S$ and $t \in T$, determines and $s - t$ cut
- Let $V_1 = S - \{s\}$, and $V_2 = T - \{t\}$. If $|V_1| = 0$, then the capacity of the cut $c(S,T) = m|V|$; otherwise, the capacity of the cut is given by

$$c(S,T) = \sum_{i \in S, j \in T} c_{ij} = \sum_{j \in V_2} c_{sj} + \sum_{i \in V_1} c_{it} + \sum_{i \in V_1, j \in V_2} c_{ij}$$

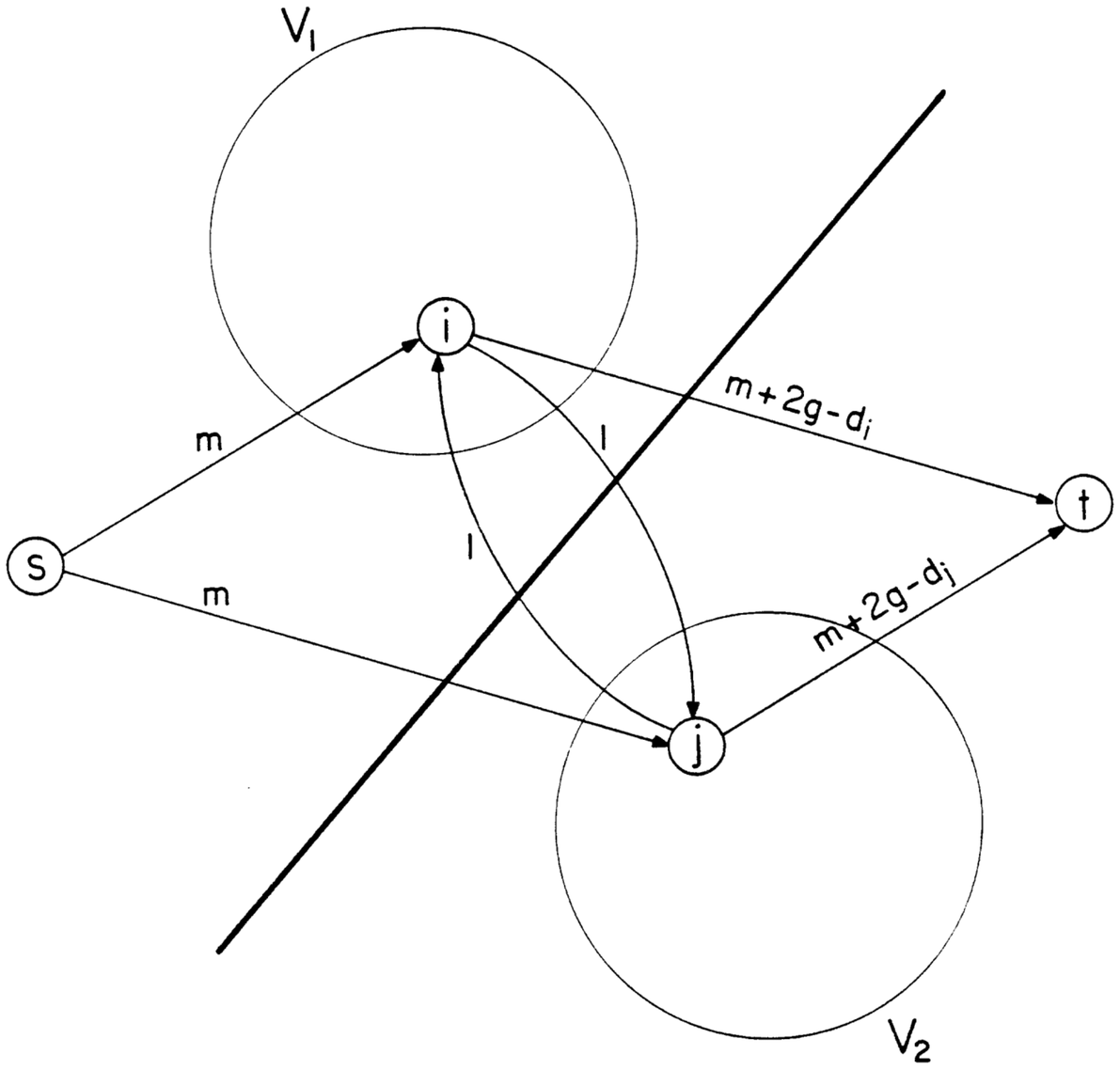
$$= m|V_2| + (m|V_1| + 2g|V_1| - \sum_{i \in V_1} d_i) + \sum_{i \in V_1, j \in V_2} c_{ij}$$

$$= m|V| + |V_1|2(g - (\frac{\sum_{i \in V_1} d_i - \sum_{i \in V_1, j \in V_2} 1}{2|V_1|}))$$

Goldberg's algorithm of DSP

- We can simplified $D_1 = \dfrac{\sum_{i \in V_1} d_i - \sum_{i \in V_1, j \in V_2} 1}{2 \mid V_1 \mid}$
- Then $c(S_s, T_s) = m \mid V \mid + 2 \mid V_1 \mid (g - D_1)$
- However, how to determine whether g is too big or too small is a problem
- Therefore, we have theorem 1
- Theorem 1: Assume that S and T give a minimum capacity cut. If $\mid V_1 \mid \neq 0$, then $g \leq D$; if $\mid V_1 \mid = 0$ $(i.e. S = \{s\})$, then $g \geq D$

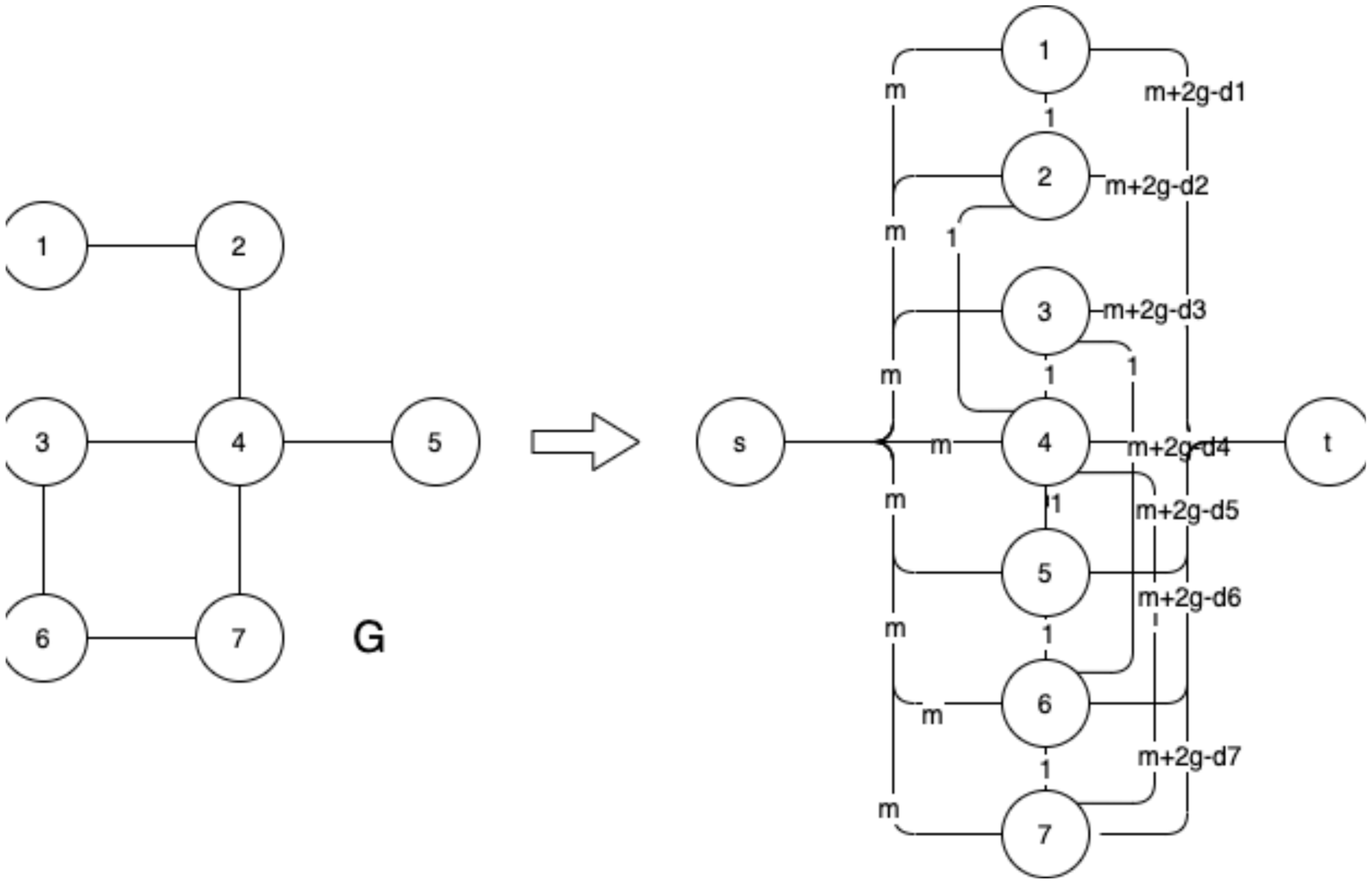
Goldberg's algorithm of DSP



Goldberg's algorithm of DSP

- The maximum desity, D, can take on only a finite set of values
- Recall that D is the density of subgraph G, so$D \in \{\frac{m^{'}}{n^{'}} | 0 \leq m^{'} \leq m, 1 \leq n^{'} \leq n\}$
- It follows that D lies between 0 and m; furthermore, the smallest distance between the two different possible values of D is $\frac{1}{n(n-1)}$
- To see this $|\Delta| = \frac{m_1}{n_1} - \frac{m_2}{n_2} = \frac{m_1 n_2 - m_2 n_1}{n_1 n_2}$
- If $n_1 = n_2$, then $|\Delta| \geq \frac{1}{n}$, otherwise $n_1 n_2 \leq n(n-1)$, so $|\Delta| \geq \frac{1}{n(n-1)}$
- Theorem 2: If $G^{'}$ is a subgraph of G with Density $D^{'}$, and no subgraph of G has a density greater than or equal to $D^{'} + \frac{1}{n(n-1)}$, then $G^{'}$ is a densest subgraph

Goldberg’s algorithm of DSP



Goldberg's algorithm of DSP

$l \leftarrow 0$; $u \leftarrow m$; $V_1 \leftarrow empty$;

while $u - l \geq \frac{1}{n(n-1)}$ **do**

begin

$g \leftarrow \frac{u+l}{2}$;

Construct $N = (V_N, E_N)$;

Find min-cut (S, T);

if $S = \{s\}$ **then** $u \leftarrow g$

else

begin

$l \leftarrow g$;

$V_1 \leftarrow S - \{s\}$;

end;

end;

return (*subgraph of* G *induced by* V_1)

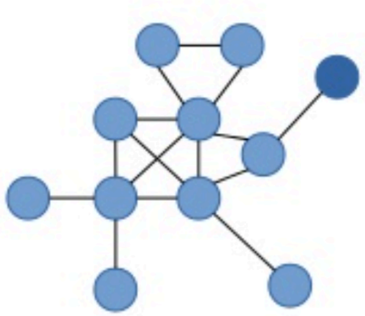
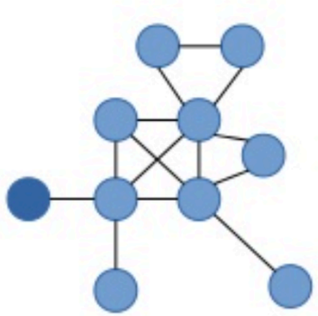
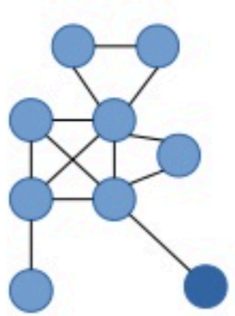
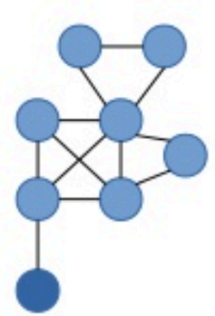
Theorem 2, the subgraph returned is a maximum density subgraph.

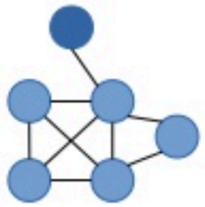
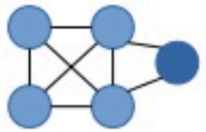
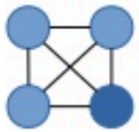
Greedy algorithm of DSP

input: undirected graph $G = (V, E)$
output: S, a dense subgraph of G

1 set $G_n \leftarrow G$
2 for $k \leftarrow n$ downto 1
2.1 let v be the smallest degree vertex in G_k
2.2 $G_{k-1} \leftarrow G_k \setminus \{v\}$
3 output the densest subgraph among $G_n, G_{n-1}, \ldots, G_1$

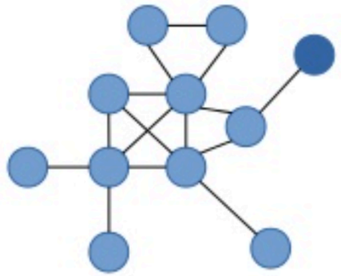
Greedy algorithm of DSP

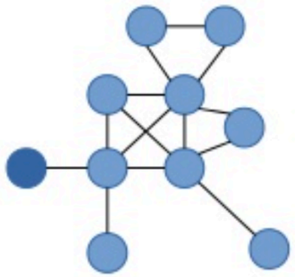





Done!

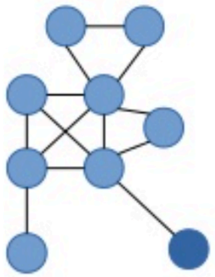
Greedy algorithm of DSP



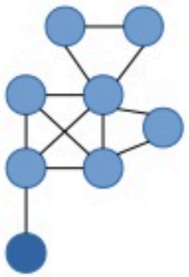
13/11=1.18



14/10=1.40

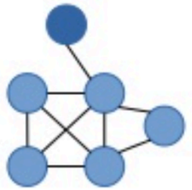


13/9=1.44



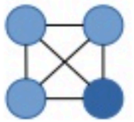
12/8=1.50

11/7=1.57



9/6=1.50

8/5=1.60



6/4=1.50

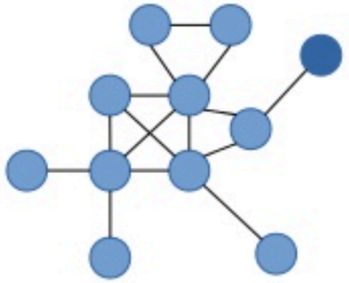
3/3=1.00

1/2=0.50

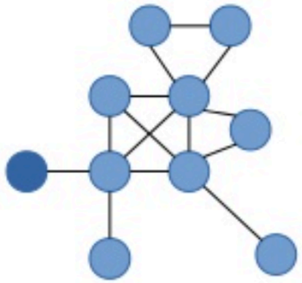
0/1=0.00

Density computed as |E|/|V|

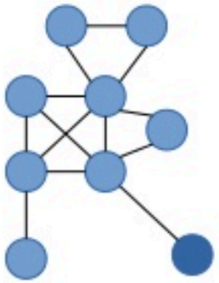
Greedy algorithm of DSP



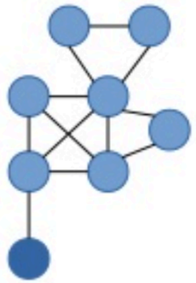
13/11=1.18



14/10=1.40

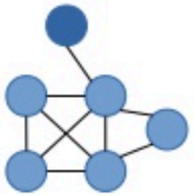


13/9=1.44

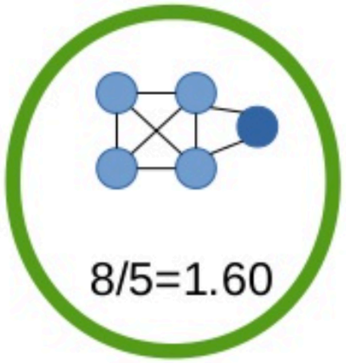


12/8=1.50

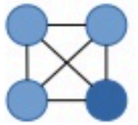
11/7=1.57



9/6=1.50



8/5=1.60



6/4=1.50

3/3=1.00

1/2=0.50

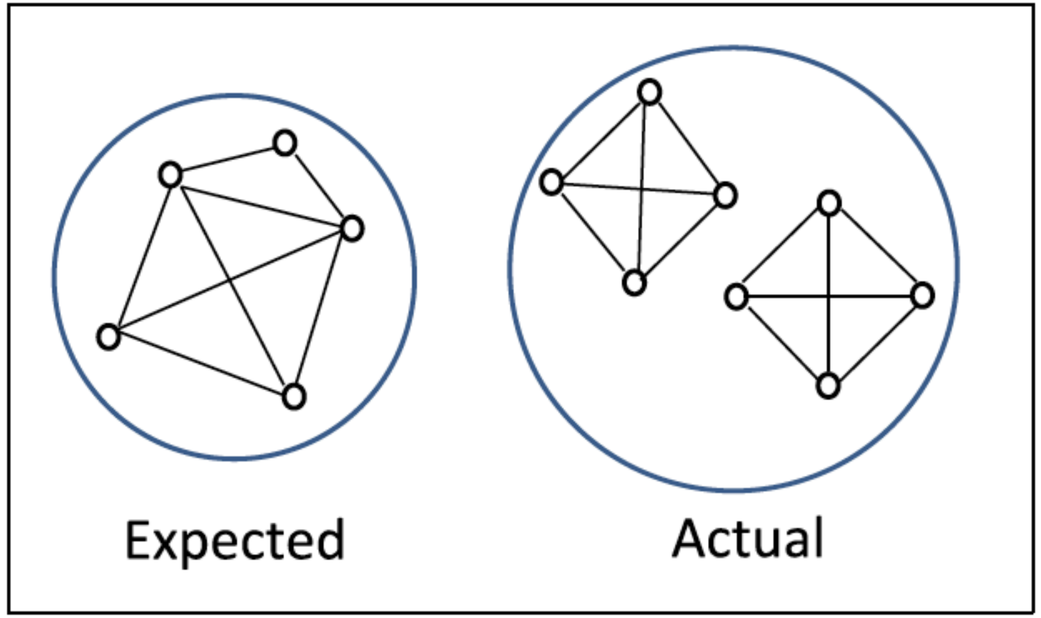
0/1=0.00

Done!

Bottleneck of current algorithm

- The previous algorithms ignore the connectivity of the returned densest subgraph

- Returned subgraph may consist of several isolated connected components that maximize its density



Bottleneck of current algorithm

- There are lacking of efficient algorithms for massive graphs, especially considering datasets become increasingly larger in this era of Big Data

- Another problem is that all the exact algorithms, are in-memory algorithms which are not suitable for big data

- The applicability of current algorithms to different kinds of graphs has not been discussed

- For example, some of the natural graphs have community structures and some others do not have community structures

Bottleneck of current algorithm

- To solve the problems, authors design a heuristic densest subgraph discovery algorithm that is more advantageous than the previous algorithms in that it is more time and space efficient and its outcomes are more precise

- Authors design an exact algorithm for big data for discovering the densest connected subgraph for undirected massive graphs in a MapReduce framework

Solver

- Authors' algorithm has two phases
  - In the first phase, it carefully reduces the dataset in a MapReduce framework without loss of any nodes existing in the exact densest subgraph
  - In the second phase, the reduced dataset can be handled in-memory in one computer by an exact densest subgraph discovery algorithm
- Therefore, we can find the exact densest subgraph for big data by taking advantage of both MapReduce and in-memory computing on one computer