Introduction to Data Mining

Lecture #20: Analysis of Large Graphs-2

U Kang
Seoul National University

In This Lecture

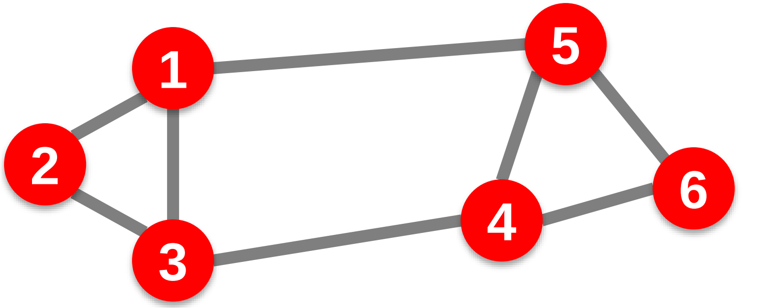
- Learn the min-cut problem in graphs, and its solutions
- Learn an example of spectral graph theory (how linear algebra and graph problems interact)
- Learn how to find small communities in graphs

Outline

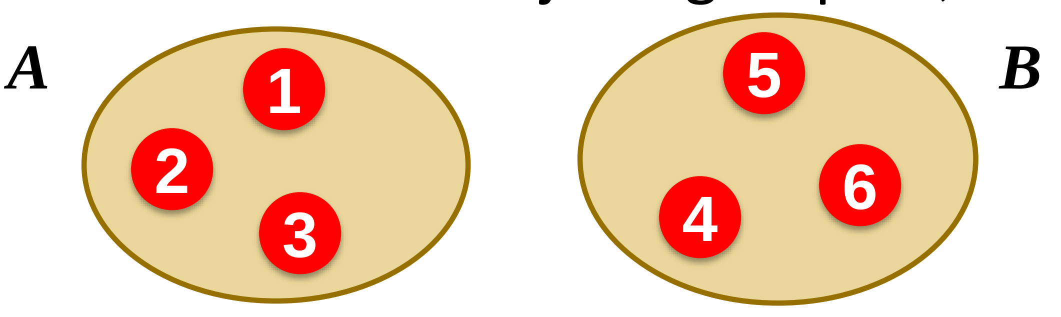
- ☐ **Spectral Clustering**
- ☐ Small Communities

Graph Partitioning

- **Undirected graph $G(V, E)$:**



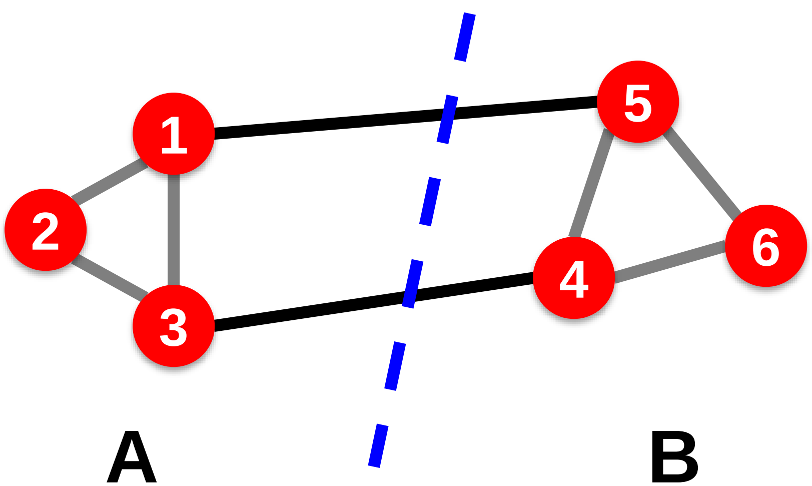
- **Bi-partitioning task:**
  - Divide vertices into two disjoint groups A, B



- **Questions:**
  - How can we define a "good" partition of G?
  - How can we efficiently identify such a partition?

Graph Partitioning

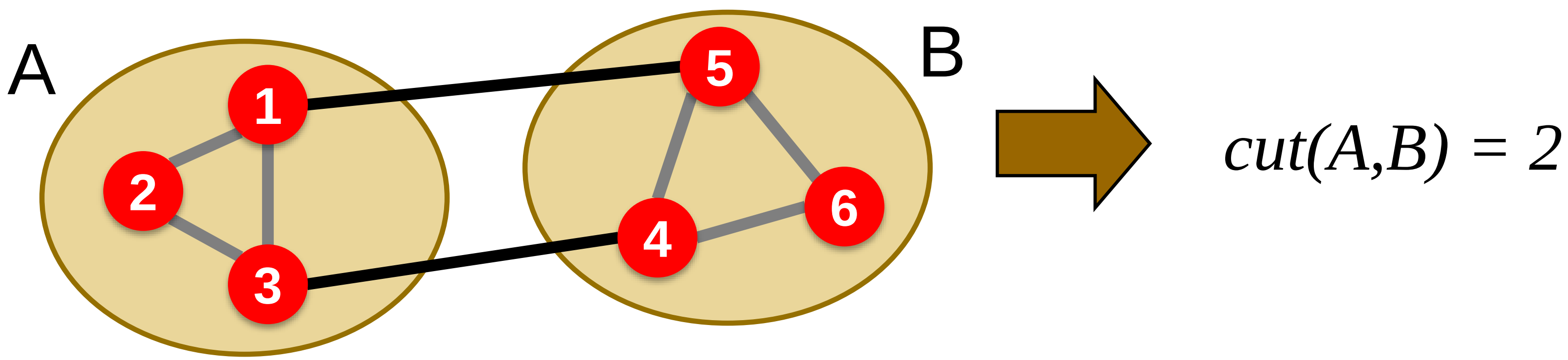
- **What makes a good partition?**
  - Maximize the number of within-group connections
  - Minimize the number of between-group connections



Graph Cuts

- **Express partitioning objectives as a function of the "edge cut" of the partition**
- **Cut:** Set of edges with only one vertex in a group:

$$cut(A,B) = \sum_{i \in A, j \in B} w_{ij}$$

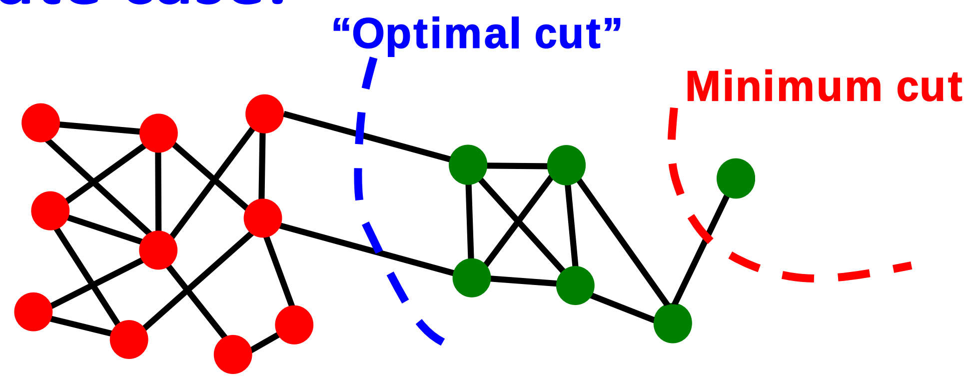


Graph Cut Criterion

- **Criterion: Minimum-cut**
  - Minimize weight of connections between groups

$$\arg\min_{A,B} cut(A,B)$$

- **Degenerate case:**



- **Problem:**
  - Only considers external cluster connections
  - Does not consider internal cluster connectivity

Graph Cut Criteria

- **Criterion: Normalized-cut** [Shi-Malik, ’97]
  - Connectivity between groups relative to the density of each group

$$ncut(A,B) = \frac{cut(A,B)}{vol(A)} + \frac{cut(A,B)}{vol(B)}$$

$\boldsymbol{vol(A)}$: total weight of the edges with at least one endpoint in $\boldsymbol{A}$: $\boldsymbol{vol(A) = \sum_{i \in A} k_i}$

- **Why use this criterion?**
  - Produces more balanced partitions
- **How do we efficiently find a good partition?**
  - **Problem:** Computing optimal cut is NP-hard

Spectral Graph Partitioning

- $\boldsymbol{A}$: adjacency matrix of undirected **G**
  - $\boldsymbol{A}_{ij}$ **=1** if $(\boldsymbol{i}, \boldsymbol{j})$ is an edge, else **0**
- $\boldsymbol{x}$: a vector in $\Re^n$ with components $(\boldsymbol{x_1}, \dots, \boldsymbol{x_n})$
  - Think of it as a label/value of each node of $\boldsymbol{G}$
- **What is the meaning of $A \cdot x$?**

$$\begin{bmatrix} a_{11} & \dots & a_{1n} \\ \vdots & & \vdots \\ a_{n1} & \dots & a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix}$$

$$y_i = \sum_{j=1}^{n} A_{ij} x_j = \sum_{(i,j) \in E} x_j$$

- **Entry y_i is a sum of labels x_j of neighbors of i**

What is the meaning of Ax?

$$\begin{bmatrix} a_{11} & \dots & a_{1n} \\ \vdots & & \vdots \\ a_{n1} & \dots & a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} = \lambda \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}$$

$$\boldsymbol{A} \cdot \boldsymbol{x} = \boldsymbol{\lambda} \cdot \boldsymbol{x}$$

- **j^{th} coordinate of $A \cdot x$:**
  - Sum of the x-values of neighbors of j
  - Make this a new value at node j
- **Spectral Graph Theory:**
  - Analyze the "spectrum" of matrix representing G
  - **Spectrum:** Eigenvectors x_i of a graph, ordered by the magnitude (strength) of their corresponding eigenvalues λ_i: $\Lambda = \{\lambda_1, \lambda_2, \dots, \lambda_n\}$

$$\lambda_1 \le \lambda_2 \le \dots \le \lambda_n$$

Example: d-regular graph

- Suppose all nodes in $\boldsymbol{G}$ have degree $\boldsymbol{d}$ and $\boldsymbol{G}$ is connected

definition of
d-regular graph

- **What are some eigenvalues/vectors of $\boldsymbol{G}$?**

$\boldsymbol{A} \cdot \boldsymbol{x} = \boldsymbol{\lambda} \cdot \boldsymbol{x}$ What is λ? What x?

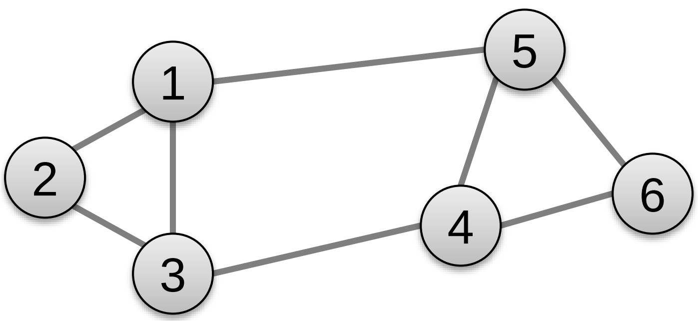
  - **Let's try:** $\boldsymbol{x} = (\mathbf{1}, \mathbf{1}, \ldots, \mathbf{1})$
  - **Then:** $\boldsymbol{A} \cdot \boldsymbol{x} = (\boldsymbol{d}, \boldsymbol{d}, \ldots, \boldsymbol{d}) = \boldsymbol{\lambda} \cdot \boldsymbol{x}$. **So:** $\boldsymbol{\lambda} = \boldsymbol{d}$
  - **We found eigenpair of $\boldsymbol{G}$:** $\boldsymbol{x} = (\mathbf{1}, \mathbf{1}, \ldots, \mathbf{1})$, $\boldsymbol{\lambda} = \boldsymbol{d}$

Remember the meaning of $\boldsymbol{y} = \boldsymbol{A} \cdot \boldsymbol{x}$:

$$y_j = \sum_{i=1}^{n} A_{ij} x_i = \sum_{(j,i) \in E} x_i$$

Matrix Representations

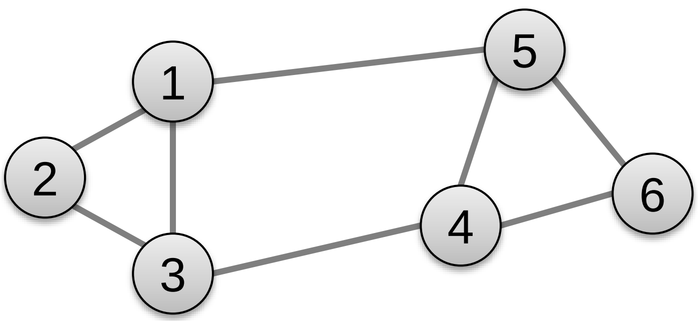
- **Adjacency matrix (A):**
  - $n \times n$ matrix
  - $A=[a_{ij}]$, $a_{ij}=1$ if edge between node i and j



	1	2	3	4	5	6
1	0	1	1	0	1	0
2	1	0	1	0	0	0
3	1	1	0	1	0	0
4	0	0	1	0	1	1
5	1	0	0	1	0	1
6	0	0	0	1	1	0

Matrix Representations

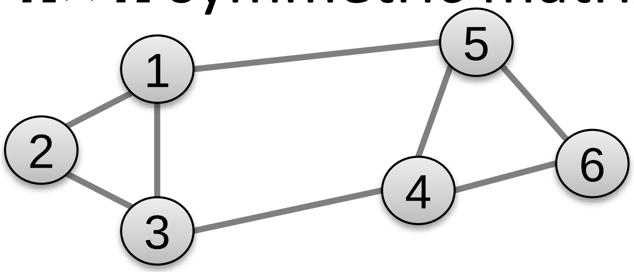
- **Degree matrix (D):**
  - $n \times n$ diagonal matrix
  - $D=[d_{ii}]$, d_{ii} = degree of node i



	1	2	3	4	5	6
1	3	0	0	0	0	0
2	0	2	0	0	0	0
3	0	0	3	0	0	0
4	0	0	0	3	0	0
5	0	0	0	0	3	0
6	0	0	0	0	0	2

Matrix Representations

- **Laplacian matrix (L):**
  - $n \times n$ symmetric matrix



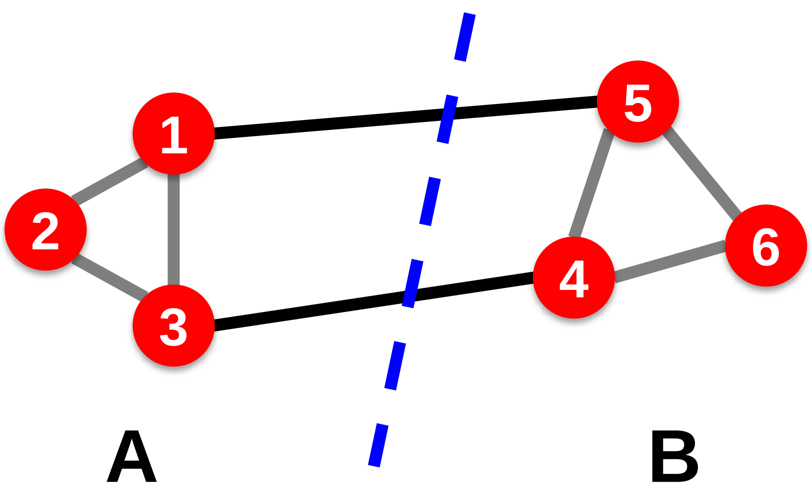
	1	2	3	4	5	6
1	3	-1	-1	0	-1	0
2	-1	2	-1	0	0	0
3	-1	-1	3	-1	0	0
4	0	0	-1	3	-1	-1
5	-1	0	0	-1	3	-1
6	0	0	0	-1	-1	2

$$\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{A}$$

- **What is trivial eigenpair?**
  - $\boldsymbol{x} = (\mathbf{1}, \dots, \mathbf{1})$ then $\boldsymbol{L} \cdot \boldsymbol{x} = \mathbf{0}$ and so $\boldsymbol{\lambda} = \boldsymbol{\lambda}_{\mathbf{1}} = \mathbf{0}$
- **Important properties of L:**
  - **Eigenvalues** are non-negative real numbers
  - **Eigenvectors** are real and orthogonal

Our Goal

- Recall: our goal is to find the minimum cut (or normalized cut)



New Formulation of Min Cut [Fiedler'73]

- Express partition (A,B) as a vector

$$y_i = \begin{cases} +1 & if\ i \in A \\ -1 & if\ i \in B \end{cases}$$

- Problem 1: Find a non-trivial vector *y* that **minimizes** **f(y)**:

$$\arg\min_{y \in [-1,+1]^n} f(y) = \sum_{(i,j) \in E} (y_i - y_j)^2$$

- Problem 1 is equivalent to finding the min-cut (A,B)
  - Why?

New Formulation of Min Cut [Fiedler'73]

- **Connection of min-cut problem and Laplacian L**

$$\arg\min_{y \in [-1,+1]^n} f(y) = \sum_{(i,j)\in E} (y_i - y_j)^2 = y^T L y$$

- **Why?**
  - $\mathrm{y}^{\mathrm{T}} \mathrm{L}\, \mathrm{y} = \sum_{i,j=1}^{n} L_{ij}\, y_i y_j = \sum_{i,j=1}^{n} \left(D_{ij} - A_{ij}\right) y_i y_j$
  - $= \sum_i D_{ii} y_i^2 - \sum_{(i,j)\in E} 2 y_i y_j$
  - $= \sum_{(i,j)\in E} (y_i^2 + y_j^2 - 2 y_i y_j) = \boldsymbol{\sum_{(i,j)\in E} \left(y_i - y_j\right)^2}$

New Formulation of Min Cut [Fiedler'73]

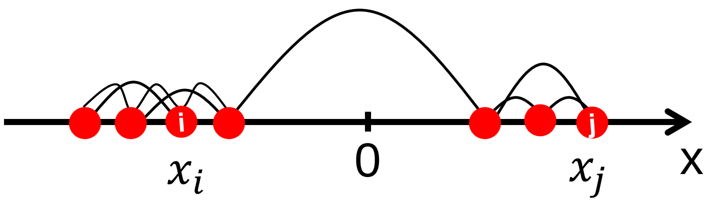
- Until now: finding the min-cut is equiv. to solving the following problem

$$\arg\min_{y\in[-1,+1]^n} f(y) = \sum_{(i,j)\in E} (y_i - y_j)^2 = y^T L y$$

- But, the problem is NP-hard
- So, we relax the constraint to make it viable
  - y_i can be any number
- Surprisingly, the solution of the problem is tightly connected with the eigenvector of L
  - Detail: next slide

Rayleigh Theorem

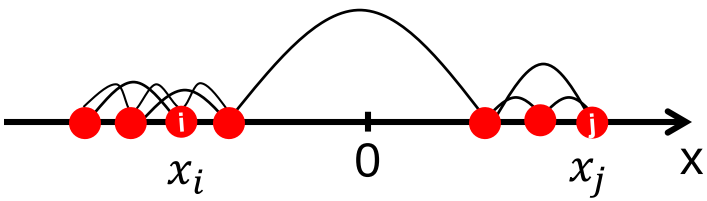
$$\min_{y \in \Re^n} f(y) = \sum_{(i,j) \in E} (y_i - y_j)^2 = y^T L y$$



- $\boldsymbol{\lambda_2 = \min_y f(y)}$: The minimum value of $\boldsymbol{f(y)}$ is given by the 2nd smallest eigenvalue λ_2 of the Laplacian matrix $\boldsymbol{L}$
- $\mathbf{x} = \arg\min_y f(y)$: The optimal solution for $\boldsymbol{y}$ is given by the corresponding eigenvector $\boldsymbol{x}$, referred as the **Fiedler vector**

Rayleigh Theorem

$$\min_{y \in \Re^n} f(y) = \sum_{(i,j) \in E} (y_i - y_j)^2 = y^T L y$$



- In fact, the minimum solution is given by y = **1** vector (the smallest eigenvector w/ eigenvalue 0); however, this does not say anything about the partition
- Thus, we find the next best solution which is the Fiedler vector
- Let the Fiedler vector = $(\alpha_1, \ldots \alpha_n)$. Then, $\sum \alpha_i^2 = 1$ (unit length), $\sum \alpha_i = 0$ (orthogonal to the first eigenvector).

So far...

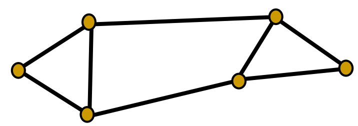
- **How to define a "good" partition of a graph?**
  - Minimize a given graph cut criterion
- **How to efficiently identify such a partition?**
  - Approximate using information provided by the eigenvalues and eigenvectors of Laplacian
- **Spectral Clustering**

Spectral Clustering Algorithms

- **Three basic stages:**
  - **1) Pre-processing**
    - Construct a matrix representation of the graph
  - **2) Decomposition**
    - Compute eigenvalues and eigenvectors of Laplacian
    - Map each point to a lower-dimensional representation based on one or more eigenvectors
  - **3) Grouping**
    - Assign points to two or more clusters, based on the new representation

Spectral Partitioning Algorithm

- **1) Pre-processing:**
  - Build Laplacian matrix $\boldsymbol{L}$ of the graph



	1	2	3	4	5	6
1	3	-1	-1	0	-1	0
2	-1	2	-1	0	0	0
3	-1	-1	3	-1	0	0
4	0	0	-1	3	-1	-1
5	-1	0	0	-1	3	-1
6	0	0	0	-1	-1	2

- **2) Decomposition:**
  - Find eigenvalues λ and eigenvectors $\boldsymbol{x}$ of the matrix $\boldsymbol{L}$

λ=

0.0
1.0
3.0
3.0
4.0
5.0

X =

0.4	0.3	-0.5	-0.2	-0.4	-0.5
0.4	0.6	0.4	-0.4	0.4	0.0
0.4	0.3	0.1	0.6	-0.4	0.5
0.4	-0.3	0.1	0.6	0.4	-0.5
0.4	-0.3	-0.5	-0.2	0.4	0.5
0.4	-0.6	0.4	-0.4	-0.4	0.0

  - Map vertices to corresponding components of $\boldsymbol{x}$

1	0.3
2	0.6
3	0.3
4	-0.3
5	-0.3
6	-0.6

Spectral Partitioning

- **3) Grouping:**
  - Sort components of reduced 1-dimensional vector
  - Identify clusters by splitting the sorted vector in two
- **How to choose a splitting point?**
  - Naïve approaches:
    - Split at **0** or median value
  - More expensive approaches:
    - Attempt to minimize normalized cut in 1-dimension (sweep over ordering of nodes induced by the eigenvector)

1	0.3
2	0.6
3	0.3
4	-0.3
5	-0.3
6	-0.6

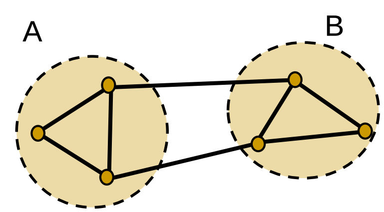
Split at 0:

Cluster A: Positive points

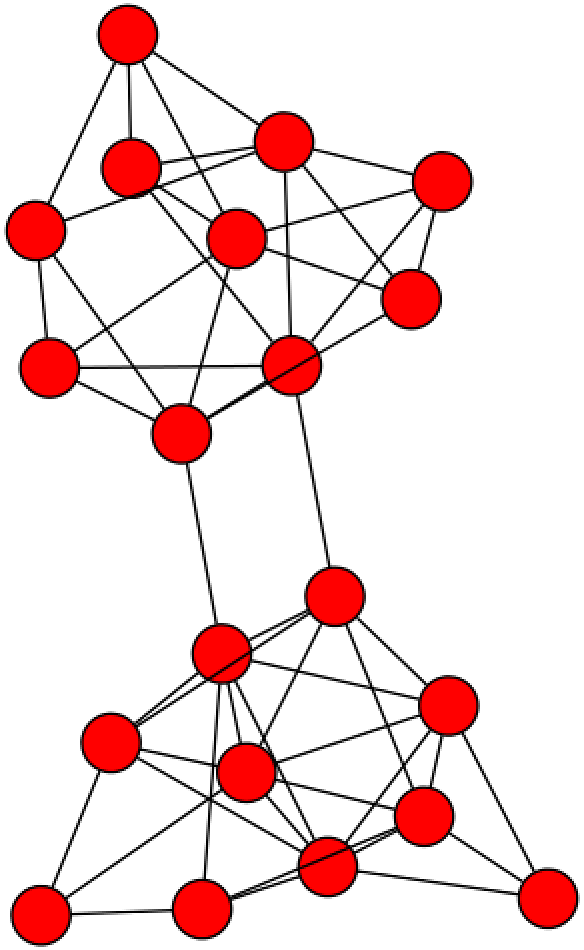
Cluster B: Negative points

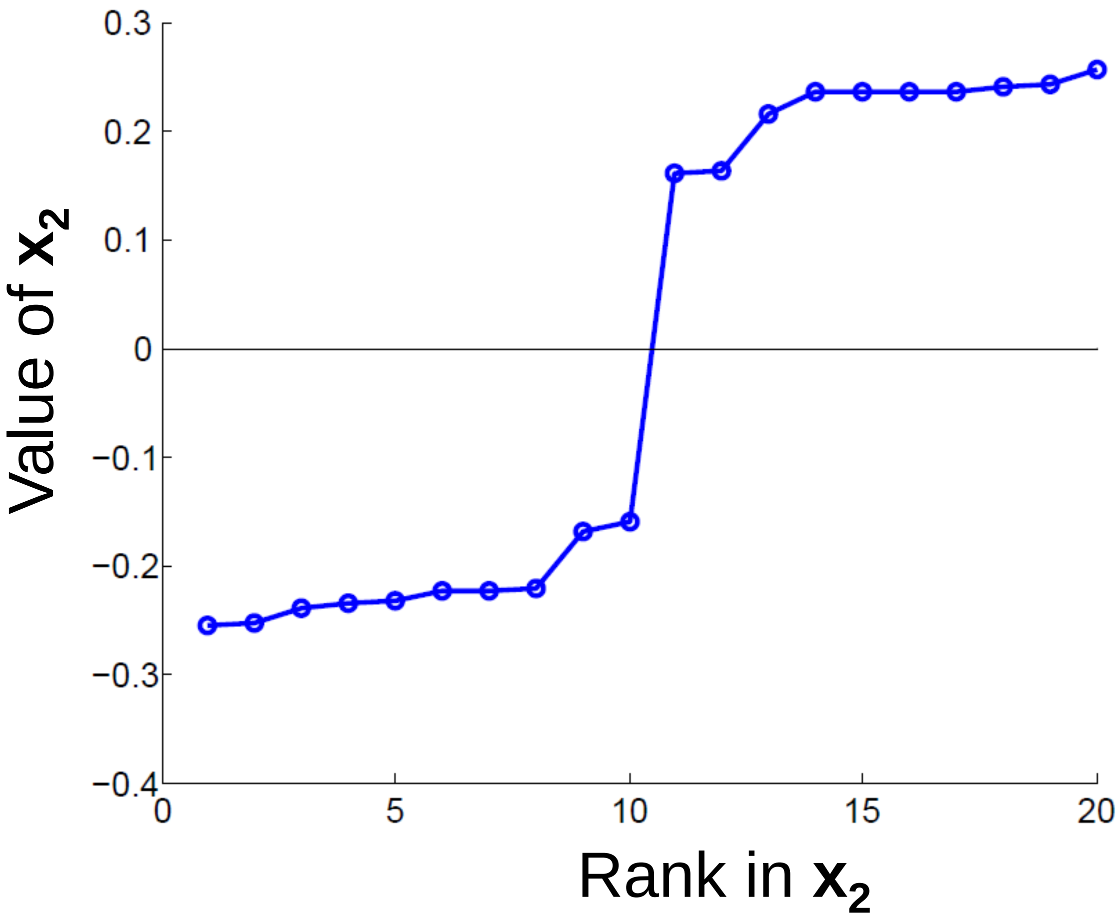
1	0.3
2	0.6
3	0.3

4	-0.3
5	-0.3
6	-0.6

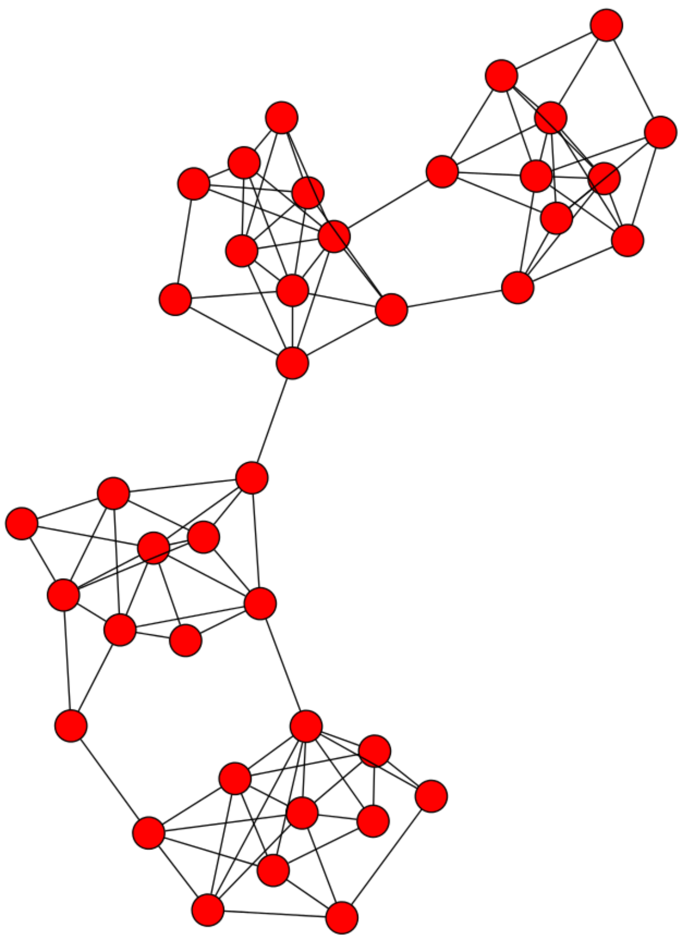


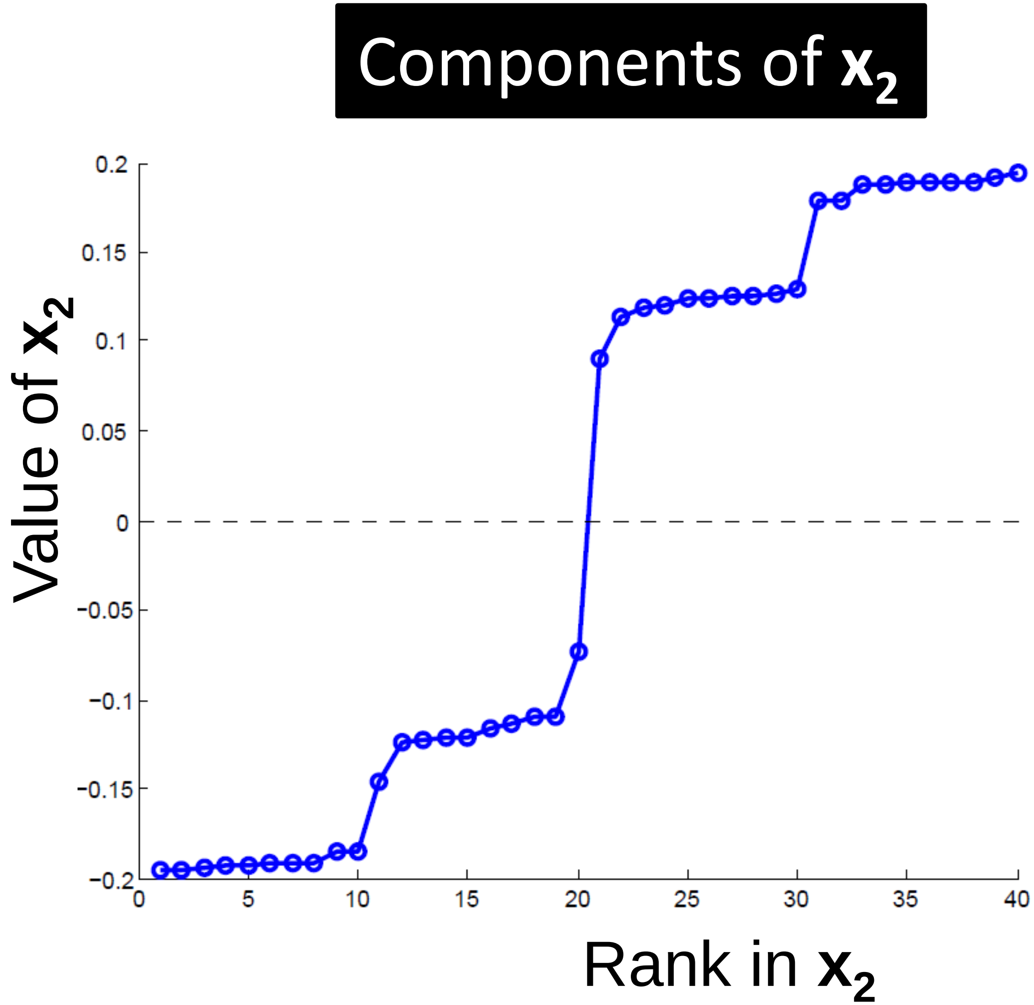
Example: Spectral Partitioning



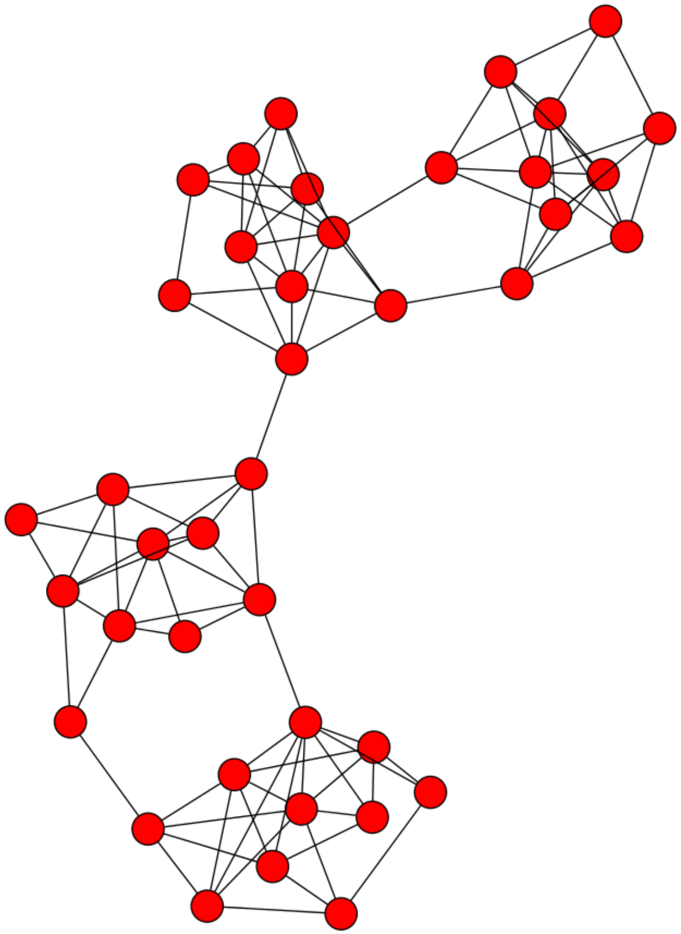


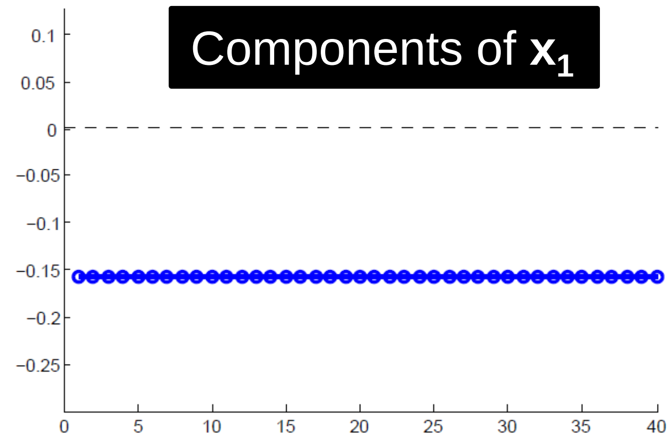
Example: Spectral Partitioning

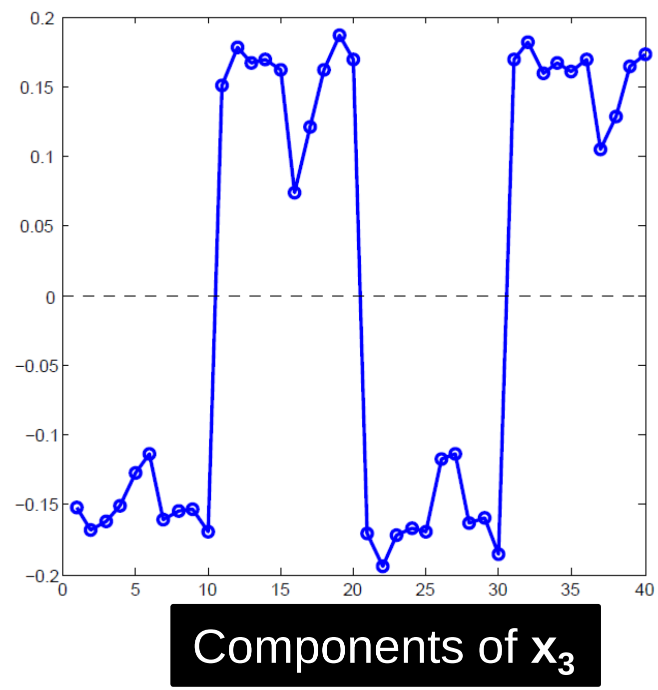




Example: Spectral partitioning







k-Way Spectral Clustering

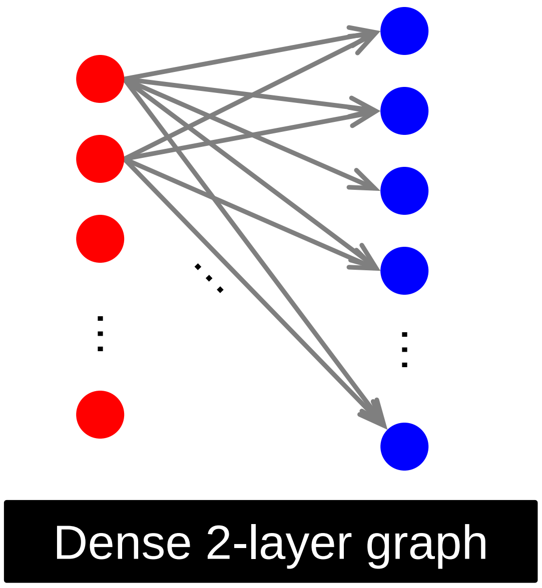
- **How do we partition a graph into *k* clusters?**
- **Two basic approaches:**
  - **Recursive bi-partitioning** [Hagen et al., '92]
    - Recursively apply bi-partitioning algorithm in a hierarchical divisive manner
    - Disadvantages: Inefficient, unstable
  - **Cluster multiple eigenvectors** [Shi-Malik, '00]
    - Build a reduced space from multiple eigenvectors
    - Commonly used in recent papers
    - A preferable approach…

Outline

- ☑ Spectral Clustering
- ☐ **Small Communities**

Small Communities in Web

- **What is the signature of a community / discussion in a Web graph?**

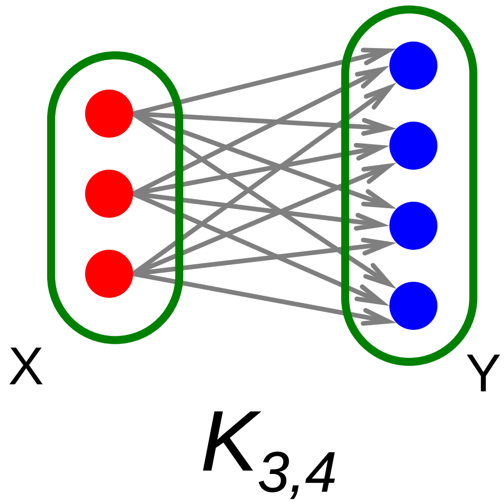


Use this to define "topics":
What the same people on the left talk about on the right
Remember HITS!

Intuition: Many people all talking about the same things

Searching for Small Communities

- **A more well-defined problem:**
  Enumerate complete bipartite subgraphs $K_{s,t}$
  - Where $K_{s,t}$: s nodes on the "left" where each links to the same t other nodes on the "right"



$|X| = s = 3$
$|Y| = t = 4$

Fully connected

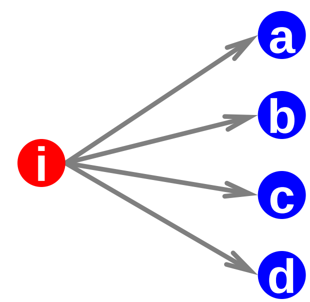
Frequent Itemset Enumeration

- **Market basket analysis.** Setting:
  - **Market:** Universe U of n items
  - **Baskets:** m subsets of U: $S_1, S_2, \ldots, S_m \subseteq U$ (S_i is a set of items one person bought)
  - **Support:** minimum number of occurrence f
- **Goal:**
  - **Find all subsets T appearing in at least f sets S_i** (items in T were bought together at least f times)
- **What's the connection between the itemsets and complete bipartite graphs?**

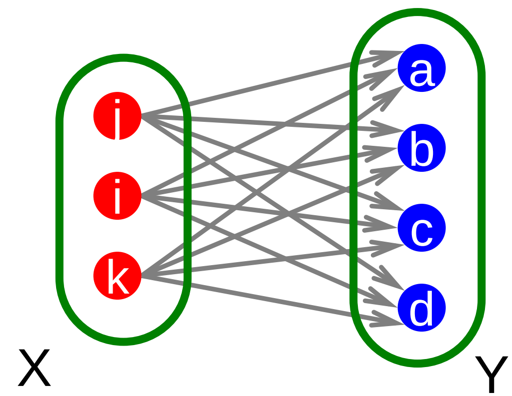
From Itemsets to Bipartite $K_{s,t}$

Frequent itemsets = complete bipartite graphs!

- **How?**
  - View each node ***i*** as a set S_i of nodes ***i*** points to
  - $K_{s,t}$ = a set ***Y*** of size ***t*** that occurs in ***s*** sets S_i
  - Looking for $K_{s,t}$ → finding frequent sets of size ***t*** with support **s**



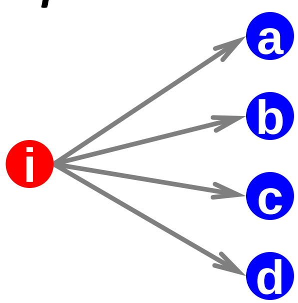
S_i={a,b,c,d}



s … minimum support (|X|=s)
t … itemset size (|Y|=t)

From Itemsets to Bipartite $K_{s,t}$

View each node ***i*** as a set S_i of nodes ***i*** points to

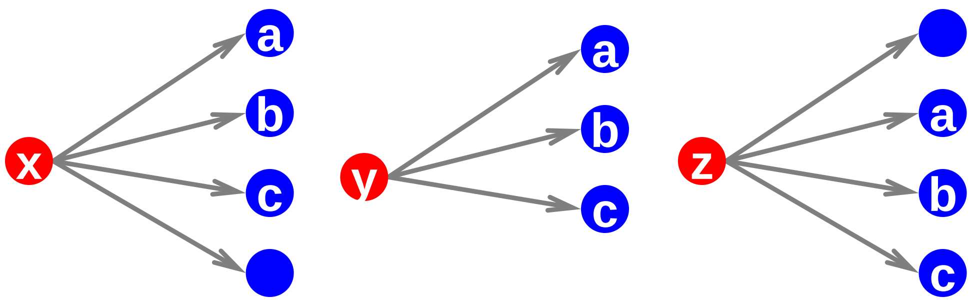


S_i={a,b,c,d}

Find frequent itemsets:
- **s** … minimum support
- **t** … itemset size

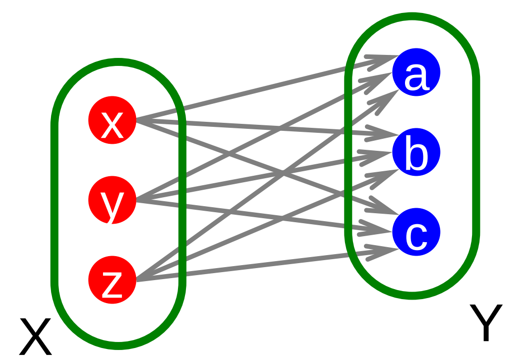
Say we find a **frequent itemset *Y={a,b,c}*** of supp ***s***

So, there are ***s*** nodes that link to all of **{a,b,c}**:

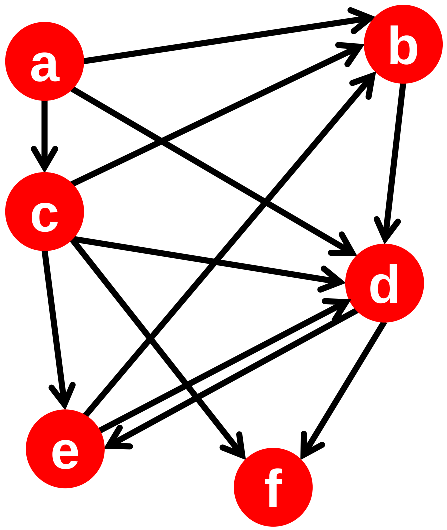


We found $K_{s,t}$!

$K_{s,t}$ = a set ***Y*** of size ***t*** that occurs in ***s*** sets S_i



Example (1)



Itemsets:

a = {b,c,d}

b = {d}

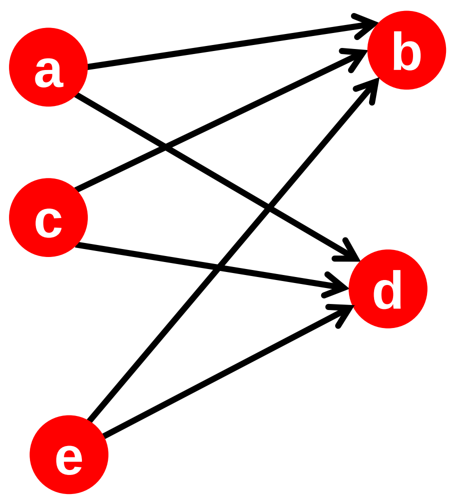
c = {b,d,e,f}

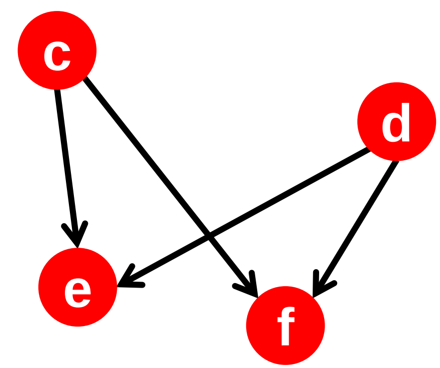
d = {e,f}

e = {b,d}

f = {}

- **Minimum support s=2**
  - **{b,d}**: support 3
  - **{e,f}**: support 2
- **And we just found 2 bipartite subgraphs:**





Example (2)

- **Example of a community from a web graph**

A community of Australian fire brigades

Nodes on the right	Nodes on the left
NSW Rural Fire Service Internet Site	New South Wales Fir...ial Australian Links
NSW Fire Brigades	Feuerwehrlinks Australien
Sutherland Rural Fire Service	FireNet Information Network
CFA: County Fire Authority	The Cherrybrook Rur...re Brigade Home Page
"The National Cente...ted Children's Ho...	New South Wales Fir...ial Australian Links
CRAFTI Internet Connexions-INFO	Fire Departments, F... Information Network
Welcome to Blackwoo... Fire Safety Serv...	The Australian Firefighter Page
The World Famous Guestbook Server	Kristiansand brannv...dens brannvesener...
Wilberforce County Fire Brigade	Australian Fire Services Links
NEW SOUTH WALES FIR...ES 377 STATION	The 911 F,P,M., Fir...mp; Canada A Section
Woronora Bushfire Brigade	Feuerwehrlinks Australien
Mongarlowe Bush Fire – Home Page	Sanctuary Point Rural Fire Brigade
Golden Square Fire Brigade	Fire Trails "l...ghters around the...
FIREBREAK Home Page	FireSafe – Fire and Safety Directory
Guises Creek Volunt...fficial Home Page...	Kristiansand Firede...departments of th...

[Kumar, Raghavan, Rajagopalan, Tomkins: Trawling the Web for emerging cyber-communities 1999]

Questions?