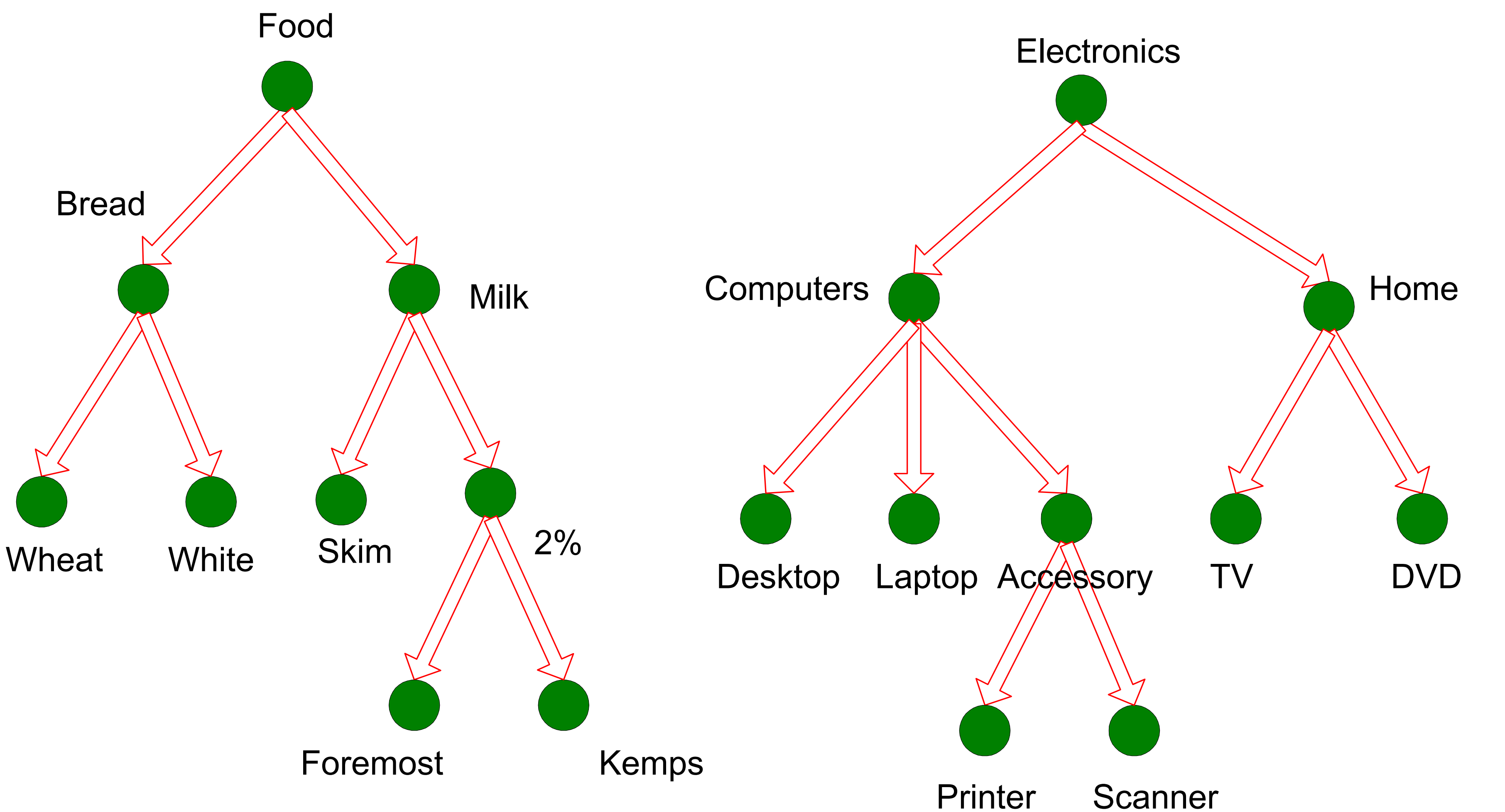
Concept Hierarchies



Multi-level Association Rules

- Why should we incorporate concept hierarchy?
  - Rules at lower levels may not have enough support to appear in any frequent itemsets

  - Rules at lower levels of the hierarchy are overly specific
    - e.g., skim milk → white bread, 2% milk → wheat bread, skim milk → wheat bread, etc.

    are indicative of association between milk and bread

  - Rules at higher level of hierarchy may be too generic
    - e.g., food->electronics

Multi-level Association Rules

- How do support and confidence vary as we traverse the concept hierarchy?
  - If X is the parent item for both X1 and X2, then $\sigma(X) \leq \sigma(X1) + \sigma(X2)$
  - If $\sigma(X1 \cup Y1) \geq$ minsup,
    and X is parent of X1, Y is parent of Y1
    then $\sigma(X \cup Y1) \geq$ minsup
    $\sigma(X1 \cup Y) \geq$ minsup
    $\sigma(X \cup Y) \geq$ minsup
  - If conf$(X1 \Rightarrow Y1) \geq$ minconf,
    then conf$(X1 \Rightarrow Y) \geq$ minconf

Multi-level Association Rules

- Approach 1:
  - Extend current association rule formulation by augmenting each transaction with higher level items

  Original Transaction: {skim milk, wheat bread}

  Augmented Transaction:
  {skim milk, wheat bread, milk, bread, food}

- Issues:
  - Items that reside at higher levels have much higher support counts
    - if support threshold is low, too many frequent patterns involving items from the higher levels
  - Increased dimensionality of the data
  - Produce redundant rules

Multi-level Association Rules

- Approach 2:
  - Generate frequent patterns at highest level first
  - Then, generate frequent patterns at the next highest level, and so on
- Issues:
  - I/O requirements will increase dramatically because we need to perform more passes over the data
  - May miss some potentially interesting cross-level association patterns

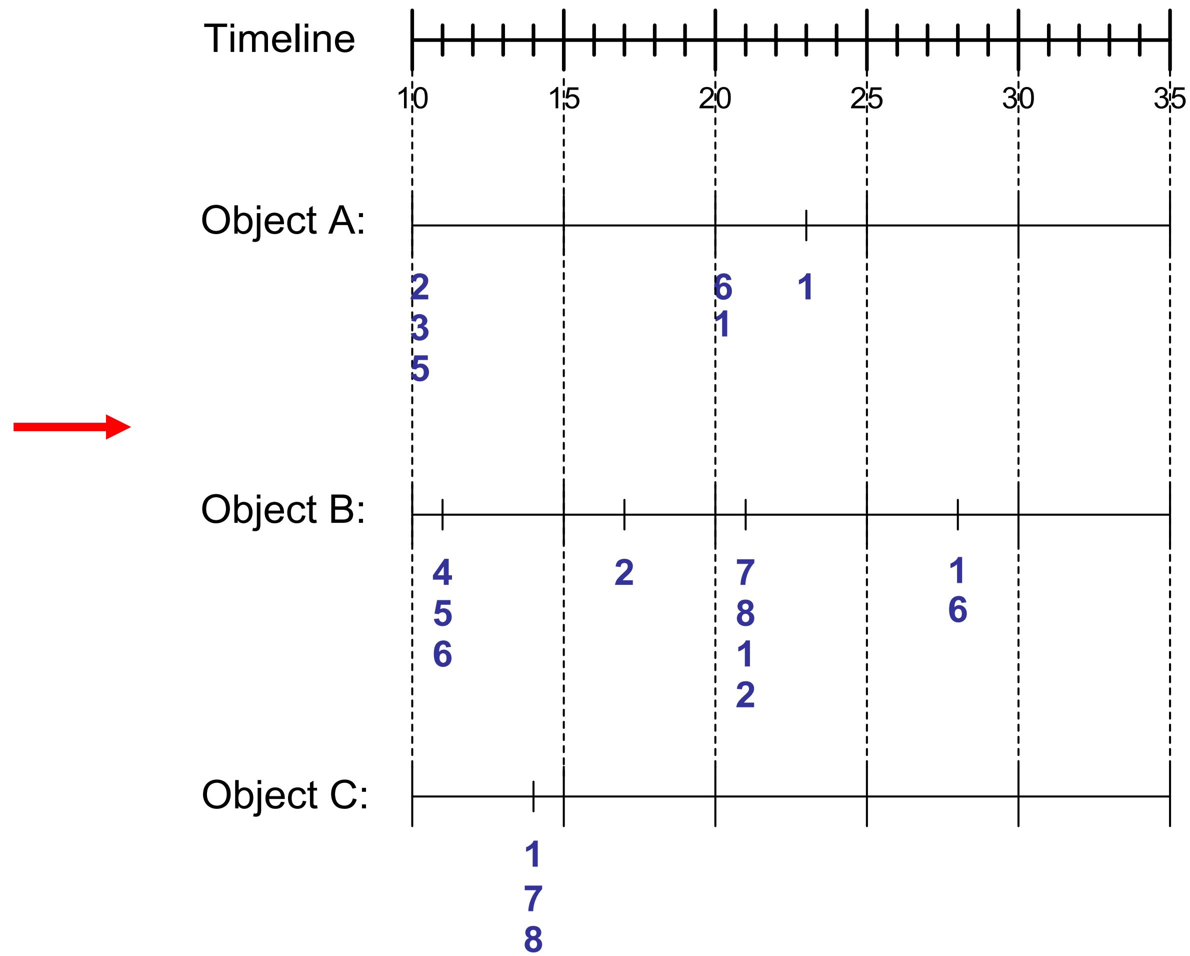
Data Mining
Association Analysis: Advanced Concepts

Sequential Patterns

Sequence Data

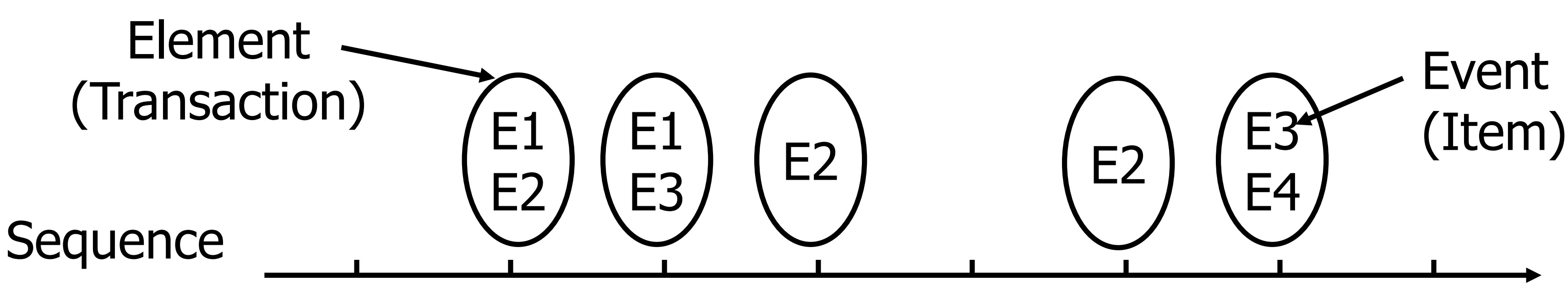
Sequence Database:

Object	Timestamp	Events
A	10	2, 3, 5
A	20	6, 1
A	23	1
B	11	4, 5, 6
B	17	2
B	21	7, 8, 1, 2
B	28	1, 6
C	14	1, 8, 7



Examples of Sequence Data

Sequence Database	Sequence	Element (Transaction)	Event (Item)
Customer	Purchase history of a given customer	A set of items bought by a customer at time t	Books, diary products, CDs, etc
Web Data	Browsing activity of a particular Web visitor	A collection of files viewed by a Web visitor after a single mouse click	Home page, index page, contact info, etc
Event data	History of events generated by a given sensor	Events triggered by a sensor at time t	Types of alarms generated by sensors
Genome sequences	DNA sequence of a particular species	An element of the DNA sequence	Bases A,T,G,C



Formal Definition of a Sequence

- A sequence is an ordered list of elements (transactions)

$$s = < e_1 \, e_2 \, e_3 \, \ldots >$$

  - Each element contains a collection of events (items)

$$e_i = \{i_1, i_2, \ldots, i_k\}$$

  - Each element is attributed to a specific time or location

- Length of a sequence, $|s|$, is given by the number of elements of the sequence

- A k-sequence is a sequence that contains k events (items)

Examples of Sequence

- Web sequence:

  < {Homepage} {Electronics} {Digital Cameras} {Canon Digital Camera} {Shopping Cart} {Order Confirmation} {Return to Shopping} >

- Sequence of books checked out at a library:

  <{Fellowship of the Ring} {The Two Towers} {Return of the King}>

Sequence Data vs. Market-basket Data

Sequence Database:

Customer	Date	Items bought
A	10	2, 3, 5
A	20	1,6
A	23	1
B	11	4, 5, 6
B	17	2
B	21	1,2,7,8
B	28	1, 6
C	14	1,7,8

Market- basket Data

Events
2, 3, 5
1,6
1
4,5,6
2
1,2,7,8
1,6
1,7,8

Sequence Data vs. Market-basket Data

Sequence Database:

Customer	Date	Items bought
A	10	2, 3, 5
A	20	1,6
A	23	1
B	11	4, 5, 6
B	17	2
B	21	1,2,7,8
B	28	1, 6
C	14	1,7,8

Market- basket Data

Events
2, 3, 5
1,6
1
4,5,6
2
1,2,7,8
1,6
1,7,8

Formal Definition of a Subsequence

- A sequence $<a_1\ a_2 \ldots a_n>$ is contained in another sequence $<b_1\ b_2 \ldots b_m>$ $(m \geq n)$ if there exist integers $i_1 < i_2 < \ldots < i_n$ such that $a_1 \subseteq b_{i1}$, $a_2 \subseteq b_{i2}$, …, $a_n \subseteq b_{in}$

- Illustrative Example:

s:	b_1	b_2	b_3	b_4	b_5
t:		a_1	a_2		a_3

t is a subsequence of s if $a_1 \subseteq b_2$, $a_2 \subseteq b_3$, $a_3 \subseteq b_5$.

Data sequence	Subsequence	Contain?
< {2,4} {3,5,6} {8} >	< {2} {8} >	Yes
< {1,2} {3,4} >	< {1} {2} >	No
< {2,4} {2,4} {2,5} >	< {2} {4} >	Yes
<{2,4} {2,5}, {4,5}>	< {2} {4} {5} >	No
<{2,4} {2,5}, {4,5}>	< {2} {5} {5} >	Yes
<{2,4} {2,5}, {4,5}>	< {2, 4, 5} >	No

Sequential Pattern Mining: Definition

- The support of a subsequence w is defined as the fraction of data sequences that contain w

- A *sequential pattern* is a frequent subsequence (i.e., a subsequence whose support is ≥ *minsup*)

- Given:
  - a database of sequences
  - a user-specified minimum support threshold, *minsup*
- Task:
  - Find all subsequences with support ≥ *minsup*

Sequential Pattern Mining: Challenge

- Given a sequence: <{a b} {c d e} {f} {g h i}>
  - Examples of subsequences:

    <{a} {c d} {f} {g} >, < {c d e} >, < {b} {g} >, etc.

- How many k-subsequences can be extracted from a given n-sequence?

<{a b} {c d e} {f} {g h i}> n = 9

k=4: Y _ _ Y Y _ _ _ Y

<{a} {d e} {i}>

Answer :

$$\binom{n}{k} = \binom{9}{4} = 126$$

Sequential Pattern Mining: Example

Object	Timestamp	Events
A	1	1,2,4
A	2	2,3
A	3	5
B	1	1,2
B	2	2,3,4
C	1	1, 2
C	2	2,3,4
C	3	2,4,5
D	1	2
D	2	3, 4
D	3	4, 5
E	1	1, 3
E	2	2, 4, 5

Minsup = 50%

Examples of Frequent Subsequences:

< {1,2} >	s=60%
< {2,3} >	s=60%
< {2,4}>	s=80%
< {3} {5}>	s=80%
< {1} {2} >	s=80%
< {2} {2} >	s=60%
< {1} {2,3} >	s=60%
< {2} {2,3} >	s=60%
< {1,2} {2,3} >	s=60%

Extracting Sequential Patterns

- Given n events: $i_1, i_2, i_3, \ldots, i_n$
- Candidate 1-subsequences:

  $\langle\{i_1\}\rangle, \langle\{i_2\}\rangle, \langle\{i_3\}\rangle, \ldots, \langle\{i_n\}\rangle$

- Candidate 2-subsequences:

  $\langle\{i_1, i_2\}\rangle, \langle\{i_1, i_3\}\rangle, \ldots, \langle\{i_1\}\ \{i_1\}\rangle, \langle\{i_1\}\ \{i_2\}\rangle, \ldots, \langle\{i_{n-1}\}\ \{i_n\}\rangle$

- Candidate 3-subsequences:

  $\langle\{i_1, i_2, i_3\}\rangle, \langle\{i_1, i_2, i_4\}\rangle, \ldots, \langle\{i_1, i_2\}\ \{i_1\}\rangle, \langle\{i_1, i_2\}\ \{i_2\}\rangle, \ldots,$

  $\langle\{i_1\}\ \{i_1, i_2\}\rangle, \langle\{i_1\}\ \{i_1, i_3\}\rangle, \ldots, \langle\{i_1\}\ \{i_1\}\ \{i_1\}\rangle, \langle\{i_1\}\ \{i_1\}\ \{i_2\}\rangle, \ldots$

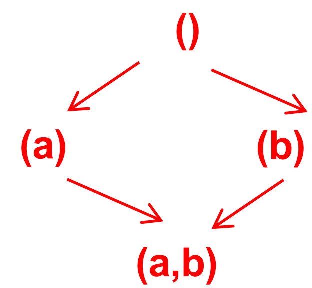
1. An event can appear more than once in a sequence

2. Order matters in sequences

Extracting Sequential Patterns: Simple example

- Given 2 events: a, b
- Candidate 1-subsequences:

  <{a}>, <{b}>.



Item-set patterns

- Candidate 2-subsequences:

  <{a} {a}>, <{a} {b}>, <{b} {a}>, <{b} {b}>, <{a, b}>.

- Candidate 3-subsequences:

  <{a} {a} {a}>, <{a} {a} {b}>, <{a} {b} {a}>, <{a} {b} {b}>,
  <{b} {b} {b}>, <{b} {b} {a}>, <{b} {a} {b}>, <{b} {a} {a}>
  <{a, b} {a}>, <{a, b} {b}>, <{a} {a, b}>, <{b} {a, b}>

Generalized Sequential Pattern (GSP)

- **Step 1**:
  - Make the first pass over the sequence database D to yield all the 1-element frequent sequences
- **Step 2**:

  Repeat until no new frequent sequences are found

  - **Candidate Generation**:
    - Merge pairs of frequent subsequences found in the (k-1)*th* pass to generate candidate sequences that contain k items
  - **Candidate Pruning**:
    - Prune candidate *k*-sequences that contain infrequent *(k-1)*-subsequences
  - **Support Counting**:
    - Make a new pass over the sequence database D to find the support for these candidate sequences
  - **Candidate Elimination**:
    - Eliminate candidate *k*-sequences whose actual support is less than *minsup*

Candidate Generation

- Base case (k=2):
  - Merging two frequent 1-sequences
- General case (k>2):
  - A frequent *(k-1)*-sequence w_1 is merged with another frequent *(k-1)*-sequence w_2 to produce a candidate *k*-sequence *if the subsequence obtained by removing the first event in w_1 is the same as the subsequence obtained by removing the last event in w_2*
    - The resulting candidate after merging is given by the sequence w_1 extended with the last event of w_2.
      - If the last two events in w_2 belong to the same element, then the last event in w_2 becomes part of the last element in w_1
      - Otherwise, the last event in w_2 becomes a separate element appended to the end of w_1

Candidate Generation Examples

- Merging the sequences
  w_1=<{1} {2 3} {4}> and w_2 =<{2 3} {4 5}>
  will produce the candidate sequence < {1} {2 3} {4 5}> because the last two events in w_2 (4 and 5) belong to the same element

- Merging the sequences
  w_1=<{1} {2 3} {4}> and w_2 =<{2 3} {4} {5}>
  will produce the candidate sequence < {1} {2 3} {4} {5}> because the last two events in w_2 (4 and 5) do not belong to the same element

- We do not have to merge the sequences
  w_1 =<{1} {2 6} {4}> and w_2 =<{1} {2} {4 5}>
  to produce the candidate < {1} {2 6} {4 5}> because if the latter is a viable candidate, then it can be obtained by merging w_1 with
  < {1} {2 6} {5}>

Candidate Generation: Examples (ctd)

- Can <{a},{b},{c}> merge with <{b},{c},{f}> ?
- Can <{a},{b},{c}> merge with <{b,c},{f}>?
- Can <{a},{b},{c}> merge with <{b},{c,f}>?
- Can <{a,b},{c}> merge with <{b},{c,f}> ?
- Can <{a,b,c}> merge with <{b,c,f}>?
- Can <{b}{a}{b}> merge with <{a}{b}{a}> ?

Candidate Generation: Examples (ctd)

- <{a},{b},{c}> can be merged with <{b},{c},{f}> to produce <{a},{b},{c},{f}>
- <{a},{b},{c}> cannot be merged with <{b,c},{f}>
- <{a},{b},{c}> can be merged with <{b},{c,f}> to produce <{a},{b},{c,f}>
- <{a,b},{c}> can be merged with <{b},{c,f}> to produce <{a,b},{c,f}>
- <{a,b,c}> can be merged with <{b,c,f}> to produce <{a,b,c,f}>
- <{b}{a}{b}> can be merged with <{a}{b}{a}> to produce <{b},{a},{b},{a}>

GSP Example

Frequent 3-sequences

< {1} {2} {3} >
< {1} {2 5} >
< {1} {5} {3} >
< {2} {3} {4} >
< {2 5} {3} >
< {3} {4} {5} >
< {5} {3 4} >

Candidate Generation

< {1} {2} {3} {4} >
< {1} {2 5} {3} >
< {1} {5} {3 4} >
< {2} {3} {4} {5} >
< {2 5} {3 4} >

GSP Example

Frequent 3-sequences

< {1} {2} {3} >
< {1} {2 5} >
< {1} {5} {3} >
< {2} {3} {4} >
< {2 5} {3} >
< {3} {4} {5} >
< {5} {3 4} >

Candidate Generation

< {1} {2} {3} {4} >
< {1} {2 5} {3} >
< {1} {5} {3 4} >
< {2} {3} {4} {5} >
< {2 5} {3 4} >

Candidate Pruning

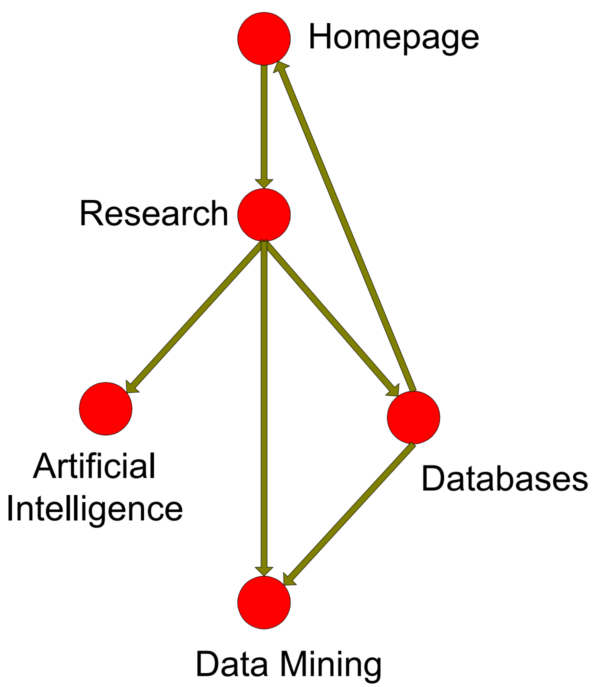
< {1} {2 5} {3} >

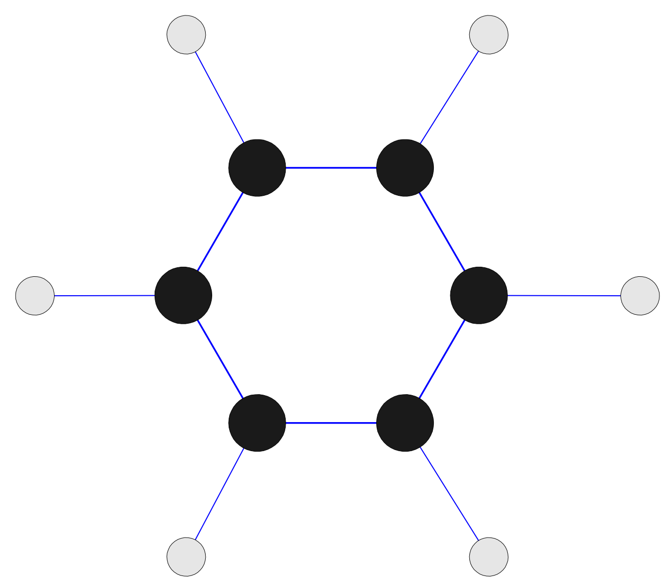
Data Mining
Association Analysis: Advanced Concepts

Subgraph Mining

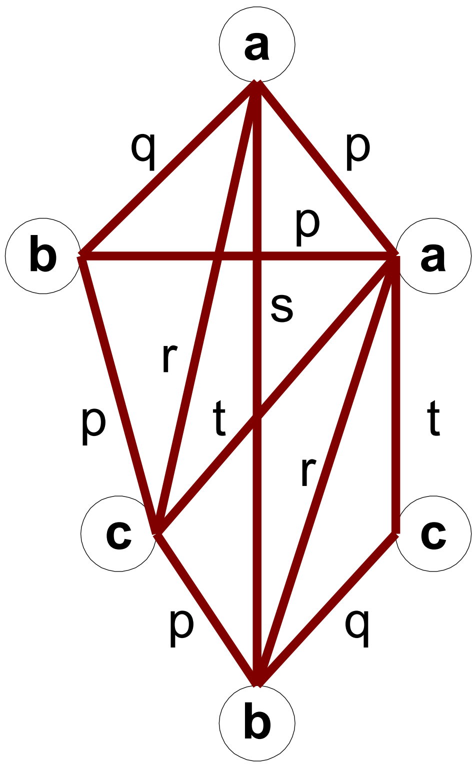
Frequent Subgraph Mining

- Extend association rule mining to finding frequent subgraphs
- Useful for Web Mining, computational chemistry, bioinformatics, spatial data sets, etc

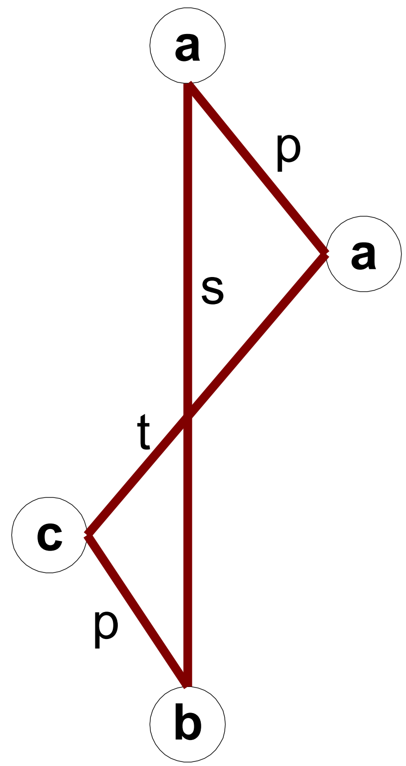




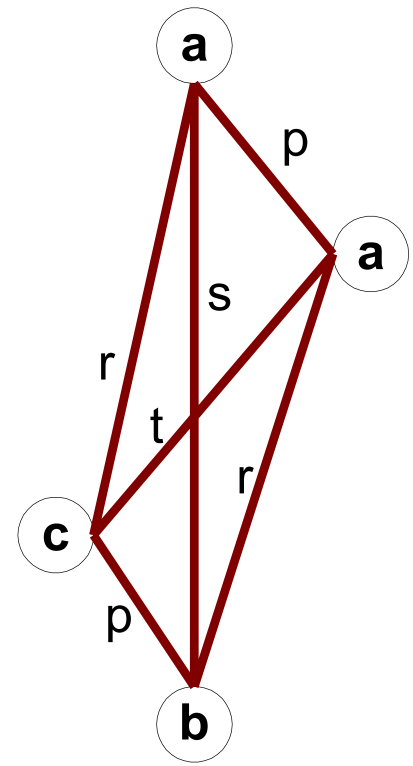
Graph Definitions



(a) Labeled Graph



(b) Subgraph

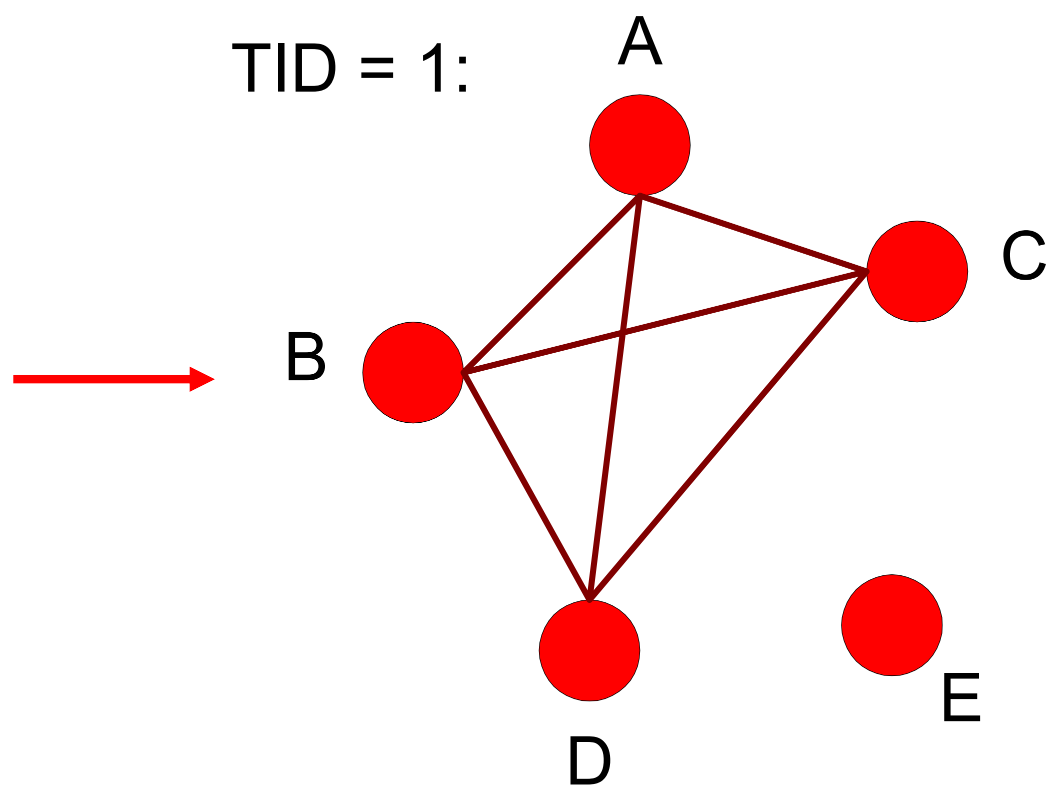


(c) Induced Subgraph

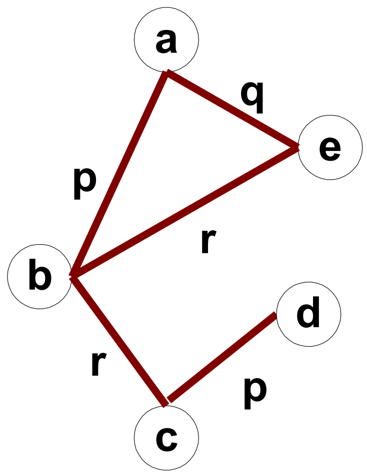
Representing Transactions as Graphs

- Each transaction is a clique of items

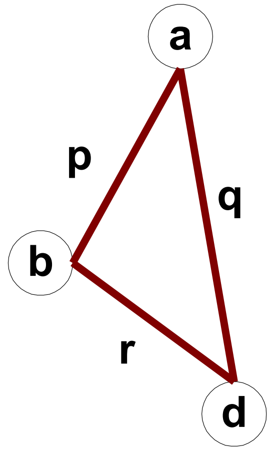
Transaction Id	Items
1	{A,B,C,D}
2	{A,B,E}
3	{B,C}
4	{A,B,D,E}
5	{B,C,D}



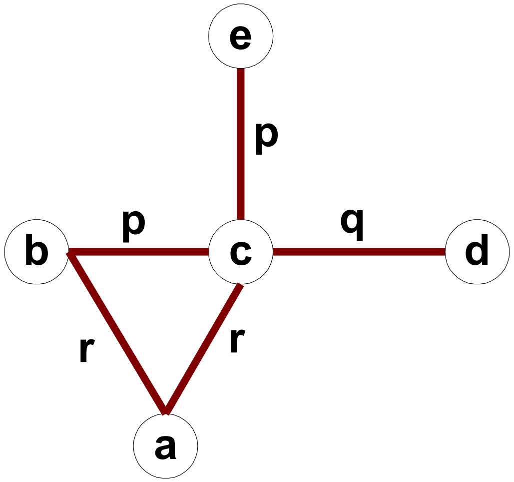
Representing Graphs as Transactions



G1

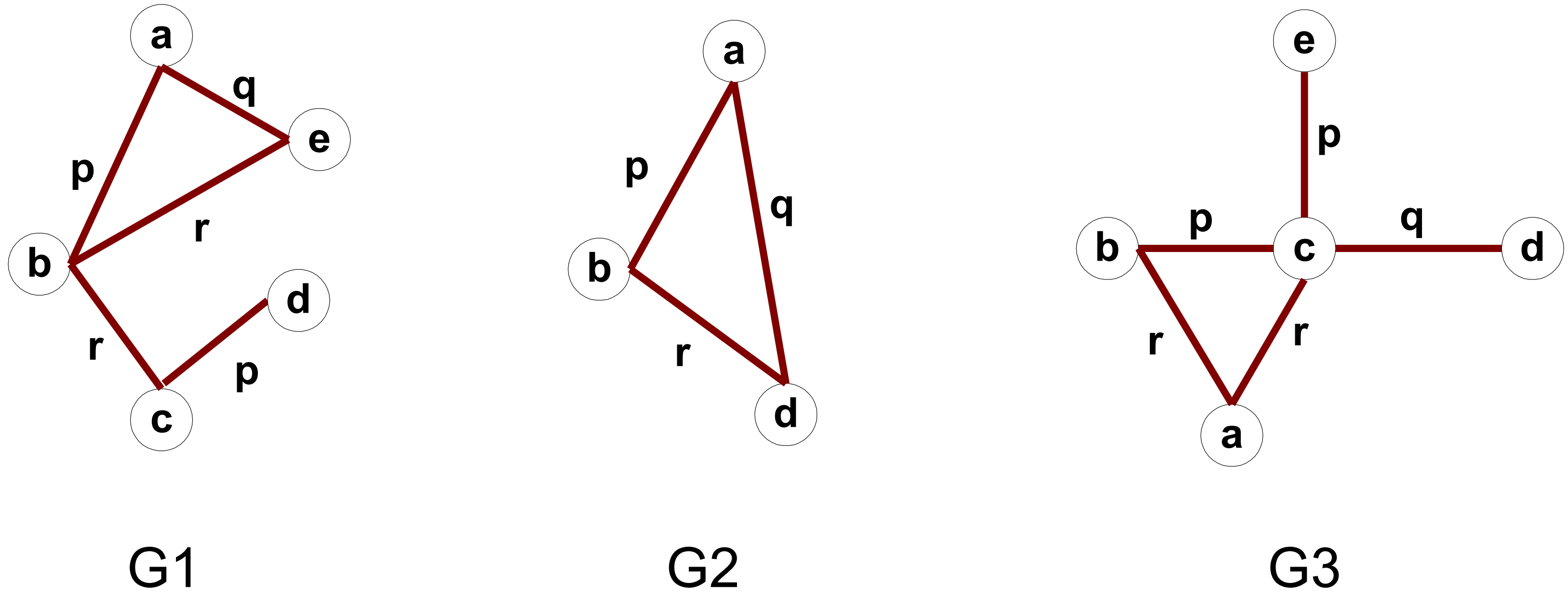


G2



G3

Representing Graphs as Transactions



	(a,b,p)	(a,b,q)	(a,b,r)	(b,c,p)	(b,c,q)	(b,c,r)	…	(d,e,r)
G1	1	0	0	0	0	1	…	0
G2	1	0	0	0	0	0	…	0
G3	0	0	1	1	0	0	…	0
G3	…	…	…	…	…	…	…	…

Frequent subgraph mining

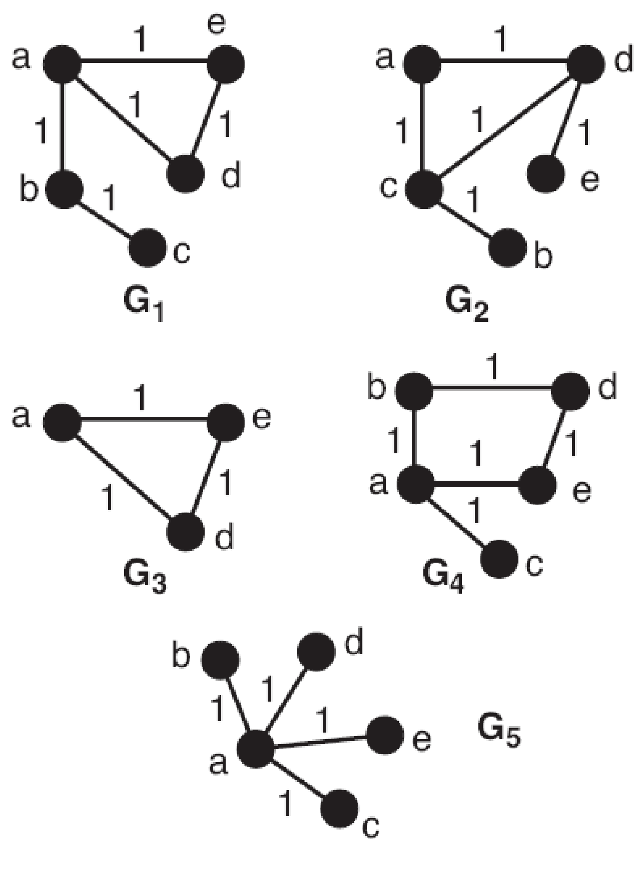
- Input:
  - A set of graphs
  - A support threshold, $minsup$
- Output:
  - Find all connected subgraphs such that s(g)$\geq minsup$

Challenges

- Node may contain duplicate labels
- Support and confidence
  - How to define them?
- Additional constraints imposed by pattern structure
  - Support and confidence are not the only constraints
  - Assumption: frequent subgraphs must be connected
- Apriori-like approach:
  - Use frequent k-subgraphs to generate frequent (k+1) subgraphs
    - What is k?

Challenges…

- Support:
  - number of graphs that contain a particular subgraph

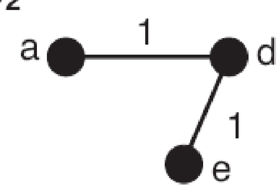


Graph Data Set

Subgraph g_1

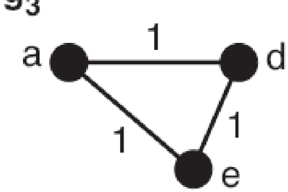
support = 80%

Subgraph g_2



support = 60%

Subgraph g_3



support = 40%

$$s(g) = \frac{|\{G_i | g \subseteq_S G_i,\ G_i \in \mathcal{G}\}|}{|\mathcal{G}|}$$

subgraph vs. itemset mining

- Frequent itemset mining:

Search space: 2^d

- Frequent subgraph mining

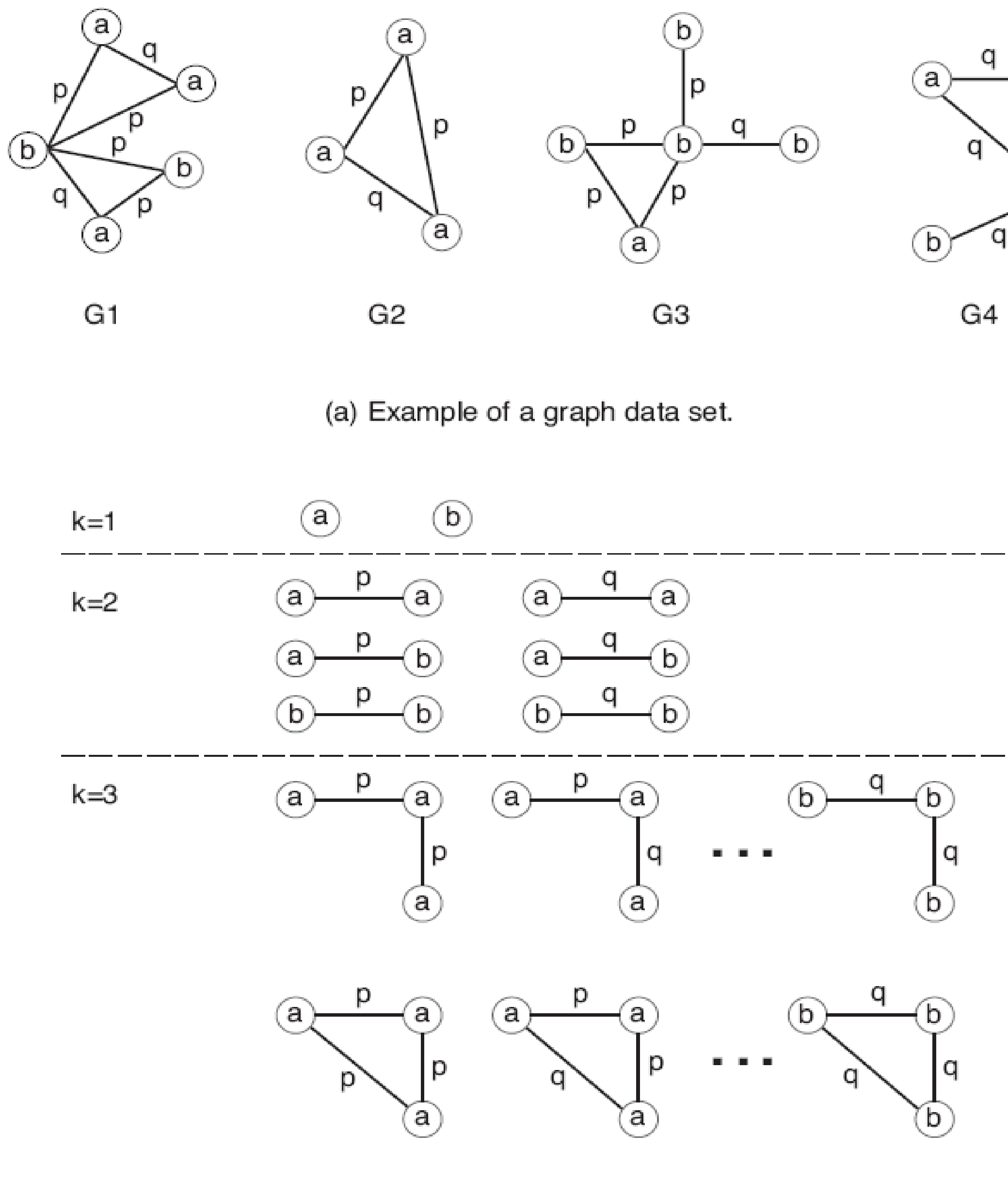
Search space:

$$\sum_{i}^{d} \binom{d}{i} * 2^{i(i-1)/2}$$

Table 7.8. A comparison between number of itemsets and subgraphs for different dimensionality, d.

Number of entities, d	1	2	3	4	5	6	7	8
Number of itemsets	2	4	8	16	32	64	128	256
Number of subgraphs	2	5	18	113	1,450	40,069	2,350,602	28,619,2513

Frequent subgraph mining



(a) Example of a graph data set.

(b) List of connected subgraphs.

- A vertex label can appear more than once

- The same pair of vertex labels can have multiple choices of edge labels