CSIT5210

Association Rule Mining

Prepared by Raymond Wong
Presented by Raymond Wong
raywong@cse

Introduction

Supermarket Application

Item

History or Transaction

Raymond

apple

coke

coffee

David

diaper

coke

…

Emily

milk

biscuit

…

Derek

coke

milk

…

We want to find some **associations** between items.

An interesting association:

Diaper and **Beer** are usually bought together.

Why? Is it strange?

diaper

beer

Introduction

Supermarket Application

An interesting association:

Diaper and **Beer** are usually bought together.

Why? Is it strange?

diaper

beer

Introduction

Supermarket Application

An interesting association:

Diaper and **Beer** are usually bought together.

Why? Is it strange?

diaper

beer

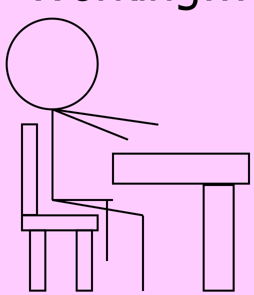
Reasons:

This pattern occurs frequently in the **early evening**.

Daytime

Office

Working…



Introduction

Supermarket Application

An interesting association:

Diaper and **Beer** are usually bought together.

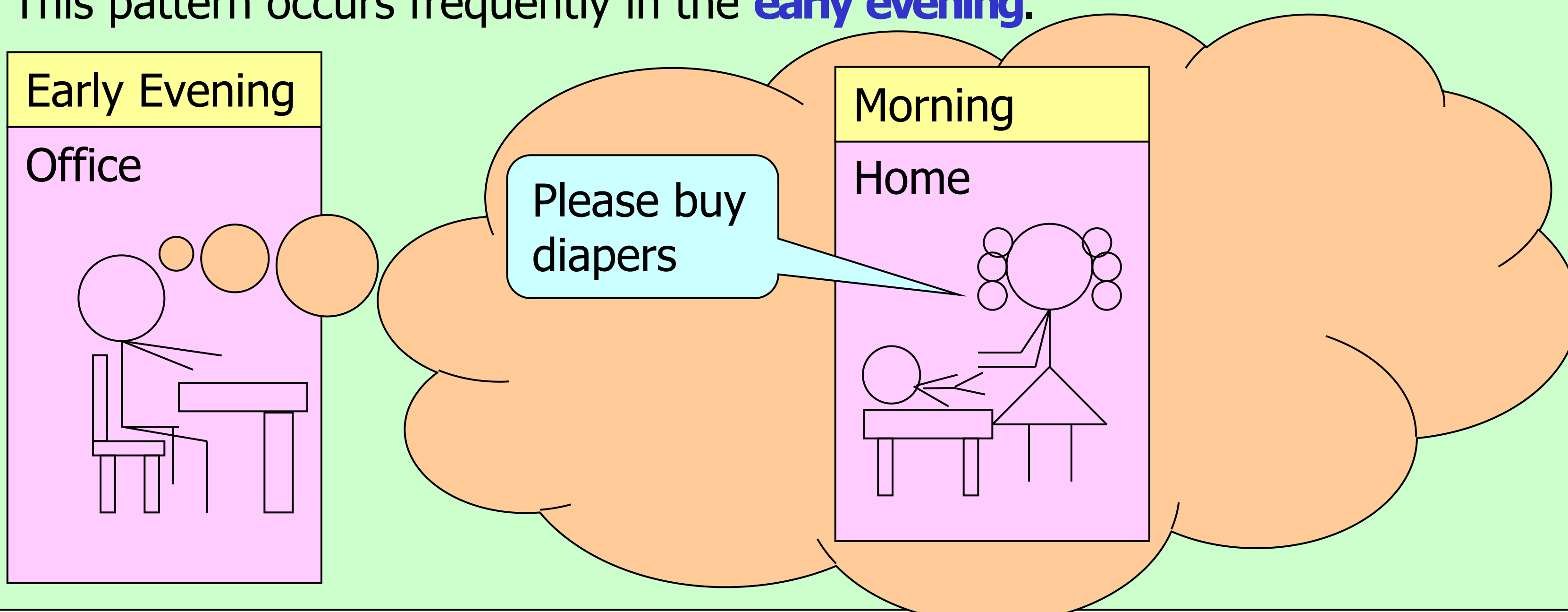
Why? Is it strange?

diaper

beer

Reasons:

This pattern occurs frequently in the **early evening**.



Introduction

- Applications of Association Rule Mining
  - Supermarket
  - Web Mining
  - Medical analysis
  - Bioinformatics
  - Network analysis
    (e.g., Denial-of-service (DoS))
  - Programming Pattern Finding

Outline

- Association Rule Mining
  - Problem Definition
  - NP-hardness
- Algorithm Apriori
  - Properties
  - Algorithm

Association Rule Mining

TID	A	B	C	D	E
t1	1	0	0	1	0
t2	1	1	0	1	1
t3	0	1	1	0	0
t4	1	1	1	1	1
t5	0	1	1	0	1

A, D

A, B, D, E

B, C

A, B, C, D, E

B, C, E

Single Items (or simply items): A B C D E

Itemsets: {B, C} {A, B, C} {B, C, D} {A}

2-itemset 3-itemset 3-itemset 1-itemset

Association Rule Mining

Large itemsets:
itemsets with support >= a threshold (e.g., 3)

Frequent itemsets

e.g., {A}, {B}, {B, C}
but NOT {A, B, C}

TID	A	B	C	D	E
t1	1	0	0	1	0
t2	1	1	0	1	1
t3	0	1	1	0	0
t4	1	1	1	1	1
t5	0	1	1	0	1

Support = 3

Support = 4

Single Items (or simply items): A B C D E

Itemsets: {B, C} {A, B, C} {B, C, D} {A}

1-frequent itemset of size 3

Support = 3

Support = 1

3-frequent itemset of size 2

Association Rule Mining

TID	A	B	C	D	E
t1	1	0	0	1	0
t2	1	1	0	1	1
t3	0	1	1	0	0
t4	1	1	1	1	1
t5	0	1	1	0	1

Support = 2

Association rules: {B, C} → E

Association Rule Mining

TID	A	B	C	D	E
t1	1	0	0	1	0
t2	1	1	0	1	1
t3	0	1	1	0	0
t4	1	1	1	1	1
t5	0	1	1	0	1

Support = 2

Confidence = 2/3 = 66.7%

Association rules: {B, C} → E

Association Rule Mining

TID	A	B	C	D	E
t1	1	0	0	1	0
t2	1	1	0	1	1
t3	0	1	1	0	0
t4	1	1	1	1	1
t5	0	1	1	0	1

Association rules:

{B, C} → E

Support = 2

Confidence = 2/3 = 66.7%

B → C

Support = 3

Association Rule Mining

TID	A	B	C	D	E
t1	1	0	0	1	0
t2	1	1	0	1	1
t3	0	1	1	0	0
t4	1	1	1	1	1
t5	0	1	1	0	1

Association rules:

{B, C} → E

Support = 2

Confidence = 2/3 = 66.7%

B → C

Support = 3

Confidence= 3/4 = 75%

Association Rule Mining

Association rules with
1. Support >= a threshold (e.g., 3)
2. Confidence >= another threshold (e.g., 50%)

TID	A	B	C	D	E
t1	1	0	0	1	0
t2	1	1	0	1	1
t3	0	1	1	0	0
t4	1	1	1	1	1
t5	0	1	1	0	1

Problem:
We want to find some "interesting" association rules

{B, C} → E

Support = 2

Confidence = 2/3 = 66.7%

B → C

Support = 3

Confidence = 3/4 = 75%

…

How can we find all "interesting" association rules?

Step 1: to find all "large" itemsets
(i.e., itemsets with support >= 3)
(e.g., itemset {B, C} has support = 3)

Step 2: to find all "interesting" rules after Step 1
- from all "large" itemsets
find the association rule with confidence >= 50%

Outline

- Association Rule Mining
  - Problem Definition
  - NP-hardness
- Algorithm Apriori
  - Properties
  - Algorithm

NP-Completeness

Problem: to find all "large" itemsets
(i.e., itemsets with support >= 3)

Problem: to find all "large" J-itemsets for each positive integer J
(i.e., J-itemsets with support >= 3)

Step 1: to find all "large" itemsets
(i.e., itemsets with support >= 3)
(e.g., itemset {B, C} has support = 3)
Step 2: to find all "interesting" rules after Step 1
- from all "large" itemsets
find the association rule with confidence
>= 50%

NP-Completeness

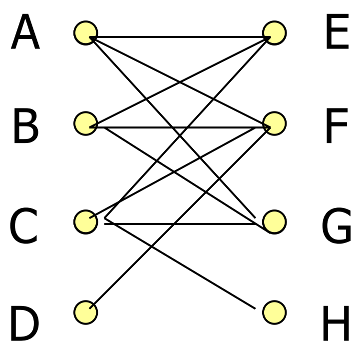
- Finding Large J-itemsets
  - INSTANCE: Given a database of transaction records
  - QUESTION: Is there an f-frequent itemset of size J?

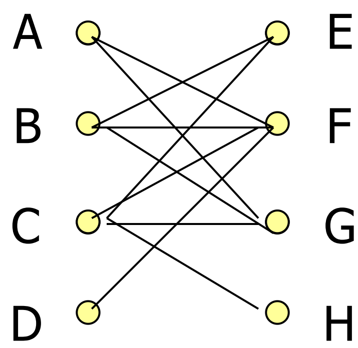
Egg	Rice	Oil	Juice
1	1	1	0
0	1	1	0
0	1	1	1
0	1	1	0

NP-Completeness

NP-complete problem

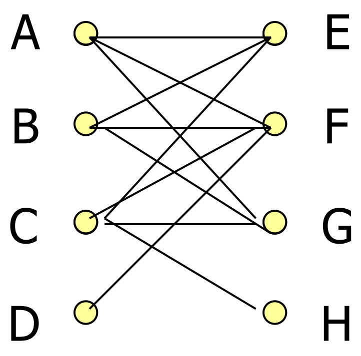
- Balanced Complete Bipartite Subgraph
  - INSTANCE: Bipartite graph $G = (V, E)$, positive integer $K \le |V|$
  - QUESTION: Are there two disjoint subsets $V_1, V_2 \subseteq V$ such that $|V_1| = |V_2| = K$ and such that, for each $u \in V_1$ and each $v \in V_2$, $\{u, v\} \in E$?





NP-Completeness

- We can transform the graph problem into itemset problem.
  - For each vertex in V_1, create a transaction
  - For each vertex in V_2, create an item
  - For each edge (u, v), create a purchase of item v in transaction u
  - f ←K
  - J ←K
- Is there a K-frequent itemset of size K?



	A	B	C	D
E	1	1	1	0
F	1	1	1	1
G	1	1	1	0
H	0	0	1	0

NP-Completeness

- It is easy to verify that solving the problem Finding Large K-itemsets is equal to solving problem Balanced Complete Bipartite Subgraph
- Finding Large K-itemsets is NP-hard.

Methods to prove that a problem P is NP-hard

- **Step 1:** Find an existing NP-complete problem (e.g., complete bipartite graph)
- **Step 2:** Transform this NP-complete problem to P (in polynomial-time)
- **Step 3:** Show that solving the "transformed" problem is equal to solving "original" NP-complete problem

Outline

- Association Rule Mining
  - Problem Definition
  - NP-hardness
- Algorithm Apriori
  - Properties
  - Algorithm

Apriori

Suppose we want to find all "large" itemsets (e.g., itemsets with support >= 3)

TID	A	B	C	D	E
t1	1	0	0	1	0
t2	1	1	0	1	1
t3	0	1	1	0	0
t4	1	1	1	1	1
t5	0	1	1	0	1

{B, C} is large

Support of {B, C} = 3

Is {B} large?

Is {C} large?

Property 1: If an itemset S is large, then any proper subset of S must be large.

Apriori

Suppose we want to find all “large” itemsets (e.g., itemsets with support >= 3)

TID	A	B	C	D	E
t1	1	0	0	1	0
t2	1	1	0	1	1
t3	0	1	1	0	0
t4	1	1	1	1	1
t5	0	1	1	0	1

{B, C, E} is NOT large

Support of {B, C, E} = 2

Is {A, B, C, E} large?

Is {B, C, D, E} large?

Property 2: If an itemset S is NOT large, then any proper superset of S must NOT be large.

Apriori

Property 1: If an itemset S is large, then any proper subset of S must be large.

Property 2: If an itemset S is NOT large, then any proper superset of S must NOT be large.

Outline

- Association Rule Mining
  - Problem Definition
  - NP-hardness
- Algorithm Apriori
  - Properties
  - Algorithm

Apriori

TID	A	B	C	D	E
t1	1	0	0	1	0
t2	1	1	0	1	1
t3	0	1	1	0	0
t4	1	1	1	1	1
t5	0	1	1	0	1

Item	Count
A	3
B	
C	
D	
E	

Apriori

Suppose we want to find all "large" itemsets (e.g., itemsets with support >= 3)

TID	A	B	C	D	E
t1	1	0	0	1	0
t2	1	1	0	1	1
t3	0	1	1	0	0
t4	1	1	1	1	1
t5	0	1	1	0	1

Item	Count
A	3
B	4
C	3
D	3
E	3

Thus, {A}, {B}, {C}, {D} and {E} are "large" itemsets of size 1 (or, "large" 1-itemsets).

We set L_1 = {{A}, {B}, {C}, {D}, {E}}

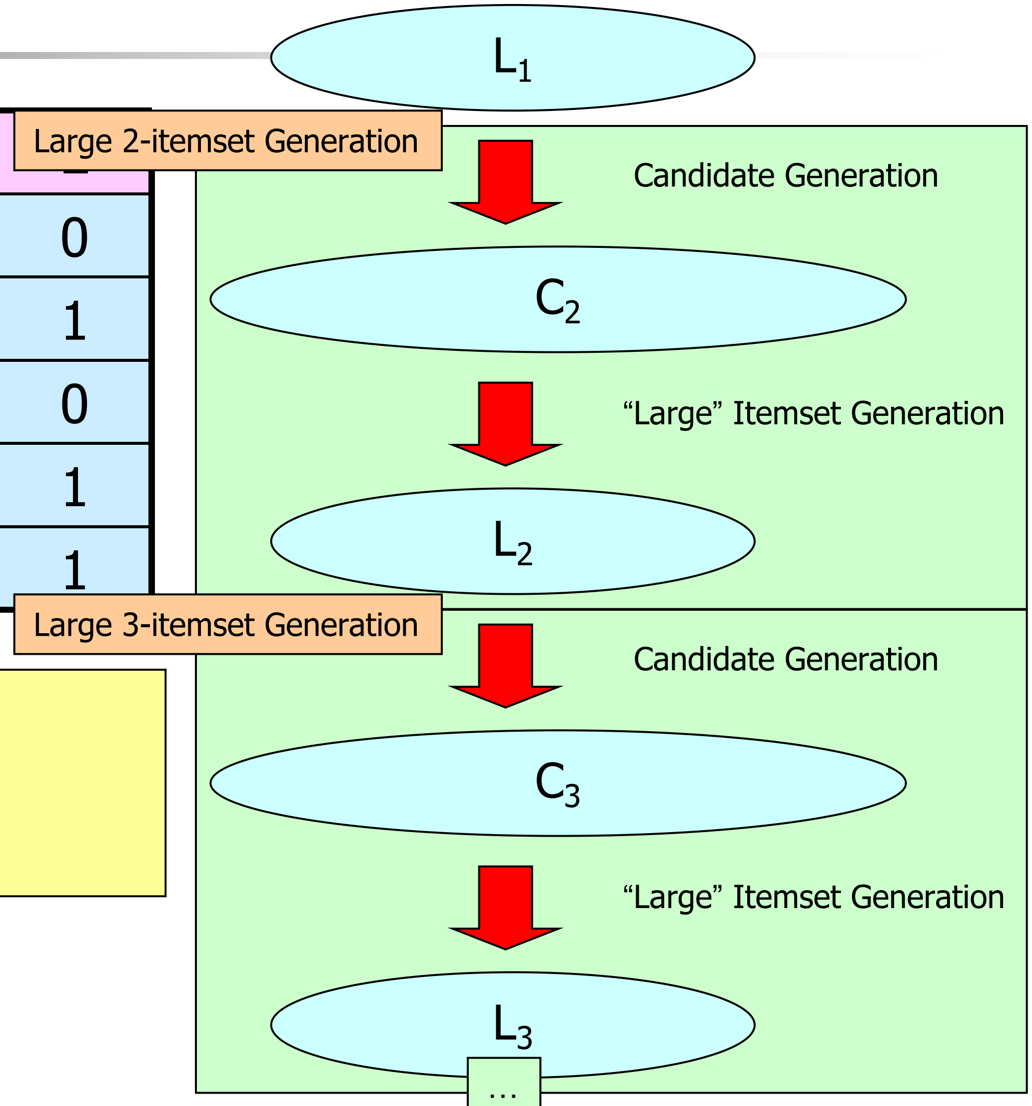
Apriori

Suppose we want to find all "large" itemsets (e.g., itemsets with support >= 3)

TID	A	B	C	D	E
t1	1	0	0	1	0
t2	1	1	0	1	1
t3	0	1	1	0	0
t4	1	1	1	1	1
t5	0	1	1	0	1

Thus, {A}, {B}, {C}, {D} and {E} are "large" itemsets of size 1 (or, "large" 1-itemsets).

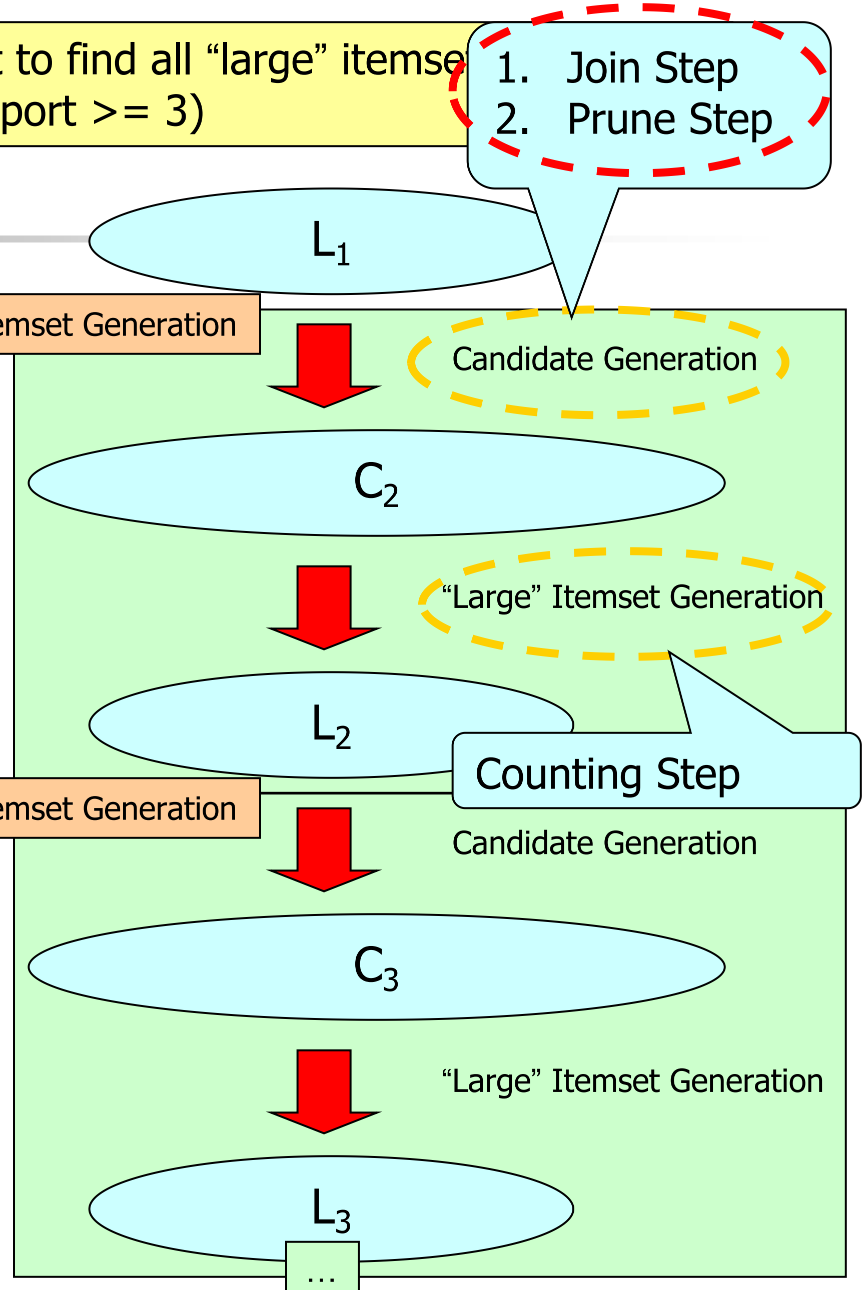
We set L_1 = {{A}, {B}, {C}, {D}, {E}}



Apriori

Suppose we want to find all “large” itemse
itemsets with support >= 3)

TID	A	B	C	D	E
t1	1	0	0	1	0
t2	1	1	0	1	1
t3	0	1	1	0	0
t4	1	1	1	1	1
t5	0	1	1	0	1



Thus, {A}, {B}, {C}, {D} and {E} are “large” itemsets of size 1 (or, “large” 1-itemsets).

We set L_1 = {{A}, {B}, {C}, {D}, {E}}

Candidate Generation

- Join Step
- Prune Step

Property 1: If an itemset S is large, then any proper subset of S must be large.

Property 2: If an itemset S is NOT large, then any proper superset of S must NOT be large.

Join Step

TID	A	B	C	D	E
t1	1	0	0	1	0
t2	1	1	0	1	1
t3	0	1	1	0	0
t4	1	1	1	1	1
t5	0	1	1	0	1

Suppose we know that itemset {B, C} and itemset {B, E} are large (i.e., L_2).

It is possible that itemset {B, C, E} is also large (i.e., C_3).

Join Step

- Join Step
  - Input: L_{k-1}, a set of all large (k-1)-itemsets
  - Output: C_k, a set of candidates k-itemsets
  - Algorithm:
    - insert into C_k
      select $p.item_1$, $p.item_2$, …, $p.item_{k-1}$, $q.item_{k-1}$
      from L_{k-1} p, L_{k-1} q
      where $p.item_1 = q.item_1$,
      $p.item_2 = q.item_2$,
      …
      $p.item_{k-2} = q.item_{k-2}$,
      $p.item_{k-1} < q.item_{k-1}$

Property 1: If an itemset S is large, then any proper subset of S must be large.

Property 2: If an itemset S is NOT large, then any proper superset of S must NOT be large.

Prune Step

TID	A	B	C	D	E
t1	1	0	0	1	0
t2	1	1	0	1	1
t3	0	1	1	0	0
t4	1	1	1	1	1
t5	0	1	1	0	1

Suppose we know that itemset {B, C} and itemset {B, E} are large (i.e., L_2).

It is possible that itemset {B, C, E} is also large (i.e., C_3).

Suppose we know that {C, E} is not large.
We can prune {B, C, E} in C_3.

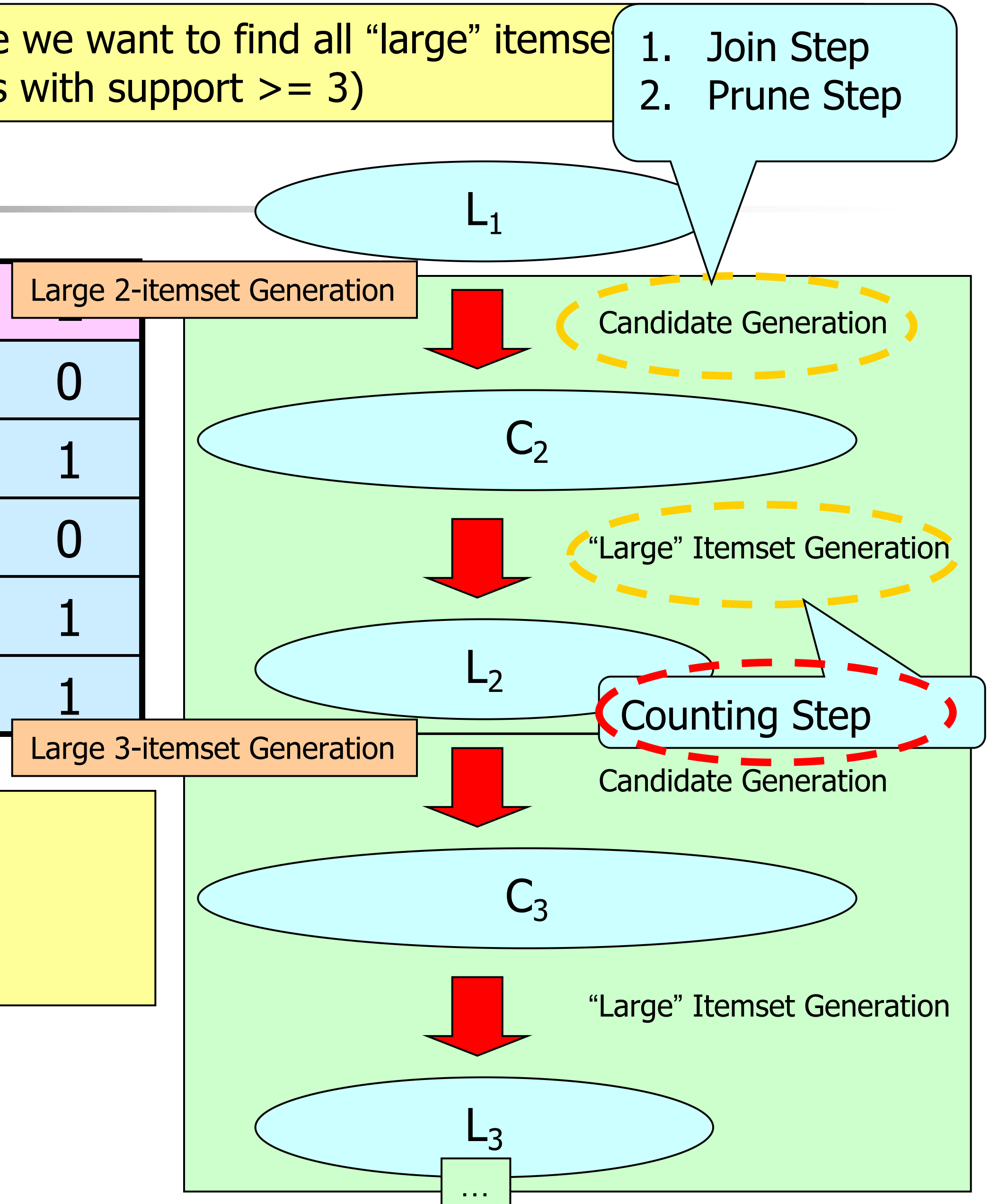
Prune Step

- Prune Step
  - forall itemsets $c \in C_k$ (from Join Step) do
    - for all (k-1)-subsets s of c do
      - if (s not in L_{k-1}) then
        - delete c from C_k

Apriori

Suppose we want to find all “large” itemse
itemsets with support >= 3)

TID	A	B	C	D	E
t1	1	0	0	1	0
t2	1	1	0	1	1
t3	0	1	1	0	0
t4	1	1	1	1	1
t5	0	1	1	0	1



Thus, {A}, {B}, {C}, {D} and {E} are “large” itemsets of size 1 (or, “large” 1-itemsets).

We set L_1 = {{A}, {B}, {C}, {D}, {E}}

Counting Step

- After the candidate generation (i.e., Join Step and Prune Step), we are given a set of **candidate** itemsets
- We need to **verify** whether these candidate itemsets are large or not
- We have to scan the database to obtain the count of each itemset in the candidate set.
- Algorithm
  - For each itemset c in C_k,
    - obtain the count of c (from the database)
    - If the count of c is smaller than a given threshold,
      - remove it from C_k
  - The remaining itemsets in C_k correspond to L_k