Real-time Acquisition of Buyer Behaviour Data —
The Smart Shop Floor Scenario

Bo Yuan, Maria E Orlowska, Shazia Sadiq

School of Information Technology and Electrical Engineering
The University of Queensland, St Lucia QLD 4072
Brisbane, Australia

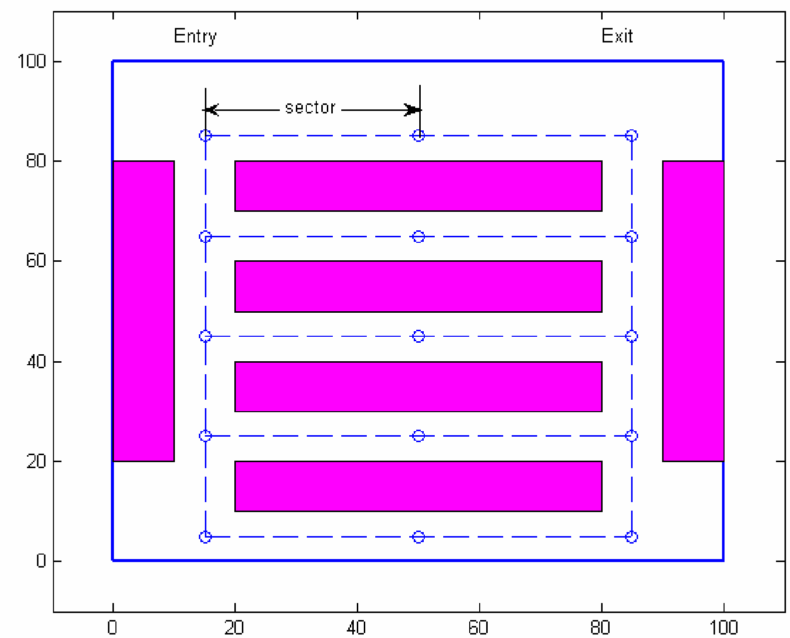
THE UNIVERSITY OF QUEENSLAND
AUSTRALIA

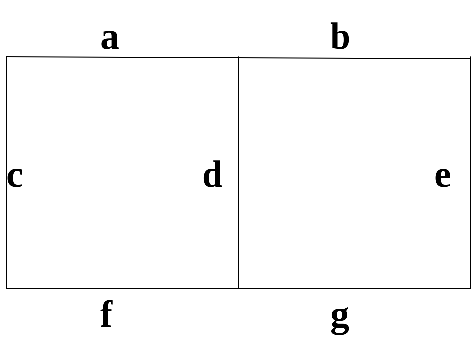
Infrastructure

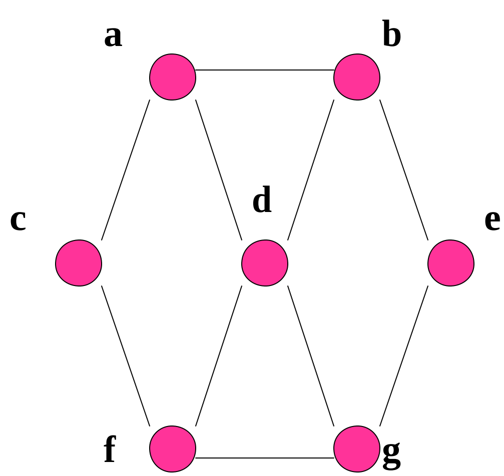
We are interested in characterizing the behaviour of buyers in terms of their navigation paths defined by shelf sectors passed by.

- Smart Trolley
  - Capture buyer profile.
  - Read product item tags.
  - Receive location data from sector indicators.
- Sector Indicators
  - Sense the existence of a trolley and send it the unique sector identification.
- Secure Wireless Network
  - Provide the communication between smart trolleys, sector indicators and back-end servers to collect real-time path and shopping data

High-Level Representation







Example Paths

a › b › e › g › d

f › g › d › f › c

b › d › g › f › d › b

Queries

Given the path records and shopping records, we can conduct the following queries:

- **Query 1**. To discover the most frequent path with a given length.
- **Query 2.** To find out the longest path with a minimum support.
- **Query 3**. To find out sectors where buyers visit frequently but seldom purchase any products.

Query 1: Finding the most popular path

- The length of the path is defined as the number of sectors involved.
- This is different from traditional sequence mining problems.
  - No support threshold given.
  - Sectors must be directly connected to each other.
- The number of all possible paths could be huge given a shop floor.
  - Cannot afford enumerating all candidates in advance.
- However, a path record with M sectors supports at most M-N+1 unique candidate paths of length N.

  {a b c d e f g h} › {a b c d e}, {b c d e f}, {c d e f g}, {d e f g h}

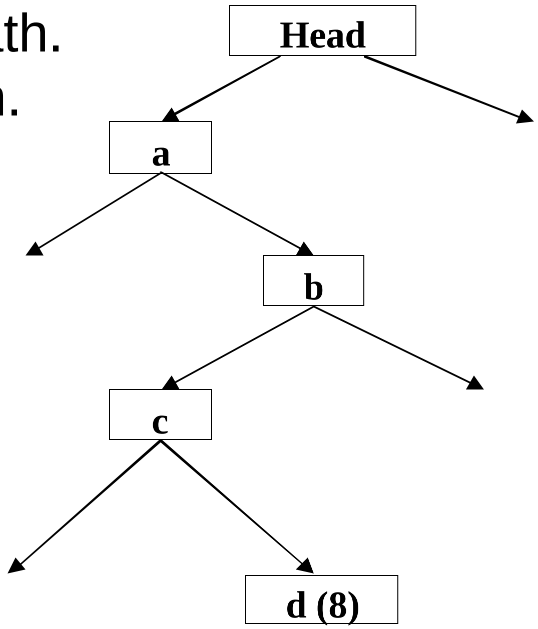
- A dataset with K records supports at most (M-N+1)·K unique paths.
- A single scan of the dataset is sufficient for solving this query.

Query 1: Algorithm Details

- Step 1: Select a new path record from the dataset. If all records have been processed, go to Step 5.
- Step 2: Find out all N-sector candidate paths supported by it.
- Step 3: Assign a new *id* to each candidate path generated in Step 2 that has never been met before.
- Step 4: Increase the counter of each unique candidate path. Go to Step 1.
- Step 5: Return the *id* of the counter with the maximum value.

Query 1: Data Structure

- A data structure for the candidate-*id* mapping.
  - *ids* are integers sequentially starting from 1.
  - Assign a new *id* to each new candidate path.
  - Return the *id* of an existing candidate path.
  - Each operation needs N steps.



- An array of counters
  - Store the frequency of each candidate path.
  - Updated for candidate paths found for the first time in a record.
  - Do not count multiple occurrences in the same record.

Query 1: Examples

- Suppose the first record is {a b c d e b c d}, N=3

No.	Candidate	ID	Array of Counters
1	{a b c}	1	[1]
2	{b c d}	2	[1, 1]
3	{c d e}	3	[1, 1, 1]
4	{d e b}	4	[1, 1, 1, 1]
5	{e b c}	5	[1, 1, 1, 1, 1]
6	{b c d}	2	[1, 1, 1, 1, 1]

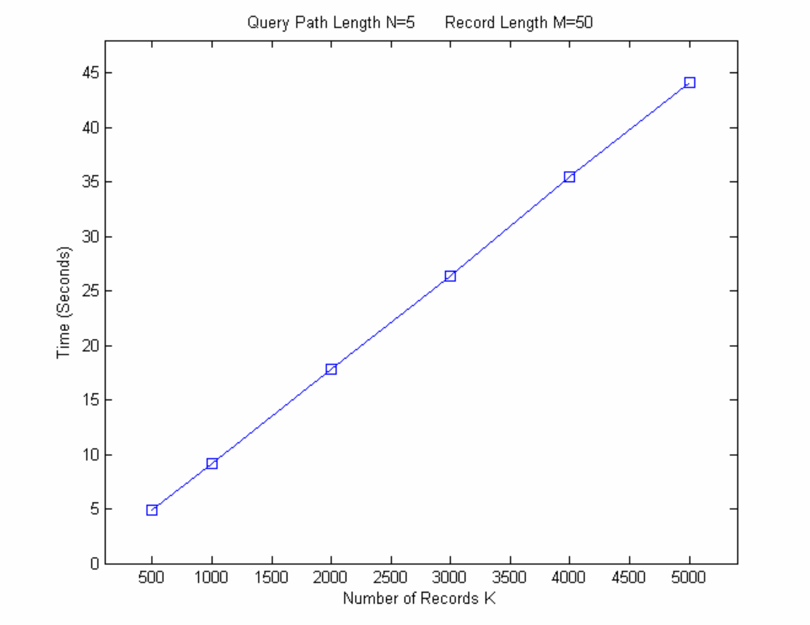
Duplicates (rows 2 and 6)

No new *id* given

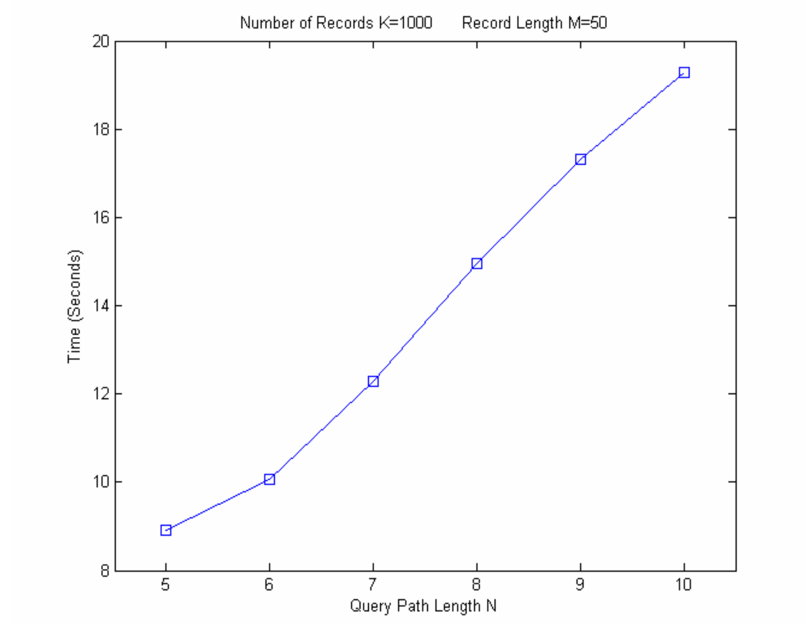
No new counter created

Query 1: Time Complexity

- There are totally (M-N+1)·K candidate paths to be processed.
- For each candidate path, we need to find out its *id* or create one, which requires N steps.
- The time complexity is *O*((M-N+1)·N·K) *O*(M·N·K) for M>>N.



Scalability with the number of records



Scalability with the length of candidate paths

Query 2: Finding the longest path

- A minimum support threshold is given.
- Find the path with as many sectors as possible, subject to the threshold.
- The basic procedure is to repeatedly apply Query 1 with N=1,2,... until no candidate paths are frequent.
- This is similar to the sequence mining problem.
- The key is how to reduce the number of candidate paths.
  - Any path containing an infrequent sub-path is deemed to be infrequent.
- Use the information in previous queries to prune the search space.
  - Still no need to enumerate candidates in advance.
  - Update the dataset instead.

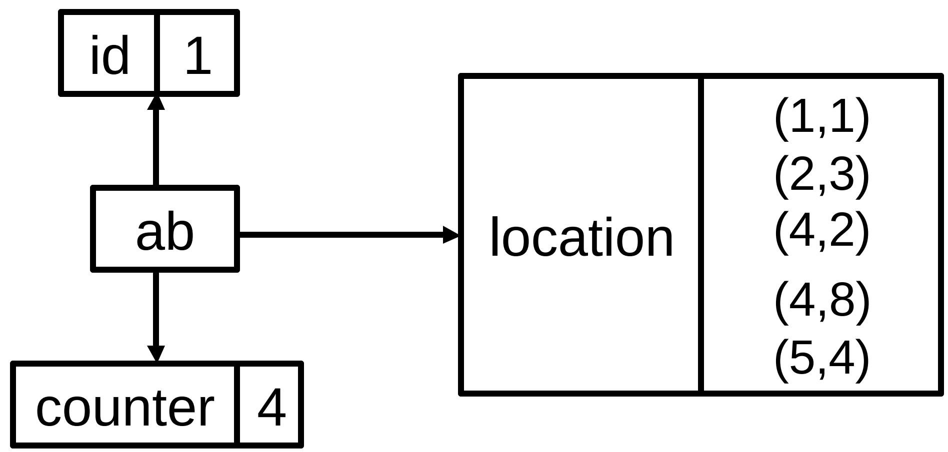
Query 2: Updating the dataset

- For each record with M sectors, maintain an array of M flags.
  - Initially all flags are set to zero.
- Once a candidate path of length N is found to be infrequent, the flag corresponding to its first sector in the record is set to N.
  - Multiple flag arrays need to be updated if it is contained in multiple records.
- This indicates that all candidate paths containing the sub-path of length N starting from this sector are deemed to be infrequent.
  - Check this condition before processing each candidate path generated.
  - The actual number of candidate paths that need to be processed may be much less than $(M-N+1)\cdot K$.
- There is a need to store the location information of each candidate path.
  - Which record does it belong to?
  - The offset within the record (i.e., position of its first sector in that record).

A dataset with five records

1	a b x x x x x x x
2	x x a b x x x
3	x x x x x x x x x x
4	x a b x x x x a b
5	x x x a b

Data structure



Three arrays used to store the location of ‘ab’

Id Array	1	...	1	...	1	...	1	...	1
Record Array	1	...	2	...	4	...	4	...	5
Offset Array	1	...	3	...	2	...	8	...	4

Mask Definition

Value 0: sector S is frequent

Value i (i=1,2,3, ...): all paths of length equal or greater than i starting from sector S is infrequent.

Since i=x also implies i=x+1, x+2, x+3, ..., the mask of a sector will not be updated to larger values once it receives a non-zero value.

Record	a	b	c	d	e	f	g	h
Initial Mask	0	0	0	0	0	0	0	0

↓

‘d’ is found to be infrequent after the first pass with N=1

↓

0	0	0	1	0	0	0	0

Candidates generated in the second pass with N=2

Candidates	Mask Values	Comments
ab	00	
bc	00	
cd	01	Infrequent
de	10	Infrequent
ef	00	
fg	00	
gh	00	

Sector ‘c’ is given mask 2 because ‘cd’ is infrequent.

0	0	2	1	0	0	0	0

The mask value of sector ‘d’ is unchanged because it is already non-zero.

Suppose ‘fg’ is found to be infrequent at the end of the second pass with N=2

a	b	c	d	e	f	g	h
0	0	2	1	0	2	0	0

Candidates generated in the third pass with N=3

Candidates	Mask Values	Comments
abc	002	
bcd	021	Infrequent
cde	210	Infrequent
def	102	Infrequent
efg	020	Infrequent
fgh	200	Infrequent

Elimination rule: the j^{th} ($j \epsilon [1,N]$) mask value i belongs to interval 0 i N-j+1. This means that there is one infrequent sub-path fully contained in the candidate.

Query 3: Finding sectors below a given purchase level

- The purchase level of a sector is defined as the ratio between
  - The number of records in which at least a product in this sector is purchased.
  - The number of records in which this sector is visited.
- Some sectors may be visited frequently but produce little profit.
- Assume that any product is only available from a certain sector.
- Transform the shopping/transaction records into sector records.
- Apply Query 1 with N=1 on the sector records
  - Frequency of each sector with purchasing activity.
- Apply Query 1 with N=1 on the path records
  - Frequency of each sector being visited
- Calculate the ratios.

More Queries …

- Classical data mining problems
  - Which sectors are likely to be visited during the same shopping trip?
  - The popular sequences of sectors

- Real-time path planning
  - The trolley knows your shopping list
  - Minimize the shopping time

- Time stamp information
  - How long did a customer spend on each sector?
  - When did a customer purchase a certain product?

- Queries can be conducted only on sectors with purchasing activities.
  - Understand the purchasing behaviour instead of the navigation behaviour.

Conclusions

- Presented data acquisition method may generate source material for marketing studies,
- The general solution is applicable to different domains,
- More interesting mining exercises can be design when combining generated data with other data sources.