FlowCube

Constructing RFID FlowCubes for Multi-dimensional Analysis of Commodity Flows

Hector Gonzalez, Jiawei Han, Xiaolei Li

University of Illinois at Urbana-Champaign
Department of Computer Science
The Datatabase and Information Systems Laboratory

VLDB'06

Outline

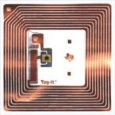
RFID Technology

What is it?

RFID is a technology that allows a reader to detect, from a distance, and without line of sight, a unique electronic product code (EPC) that is transmitted by a tag.

Tag

- Attached to items
- Stores item EPCs



Reader

- Periodical tag scans
- Records (EPC, time)

Why is it important?

- Real time tracking of individual items
    - Originating factory
    - Locations visited before arrival
    - Individuals in charge of quality control
- Improved operational efficiency
    - Reduced product scanning costs
    - Improved inventory management policies
    - More precise product recalls
- Current implementations
    - Pallet tracking at Walmart
    - Airline luggage management pilot at British airways
    - Container tracking initiative by the US Government

Motivating Example

Problem setup

- A large retailer with RFID tags placed at the item level, sells millions of items per day.
- We store the path traversed by each item:
  - laptop 1231 : (factory, 10 days) $\longrightarrow$ (warehouse, 2 days) $\longrightarrow$ (shelf, 5 days)

    printer 2453: (factory, 1 day) $\longrightarrow$ (backroom, 1 day) $\longrightarrow$ (shelf, 10 days)

Questions

- Summarize the flow patterns of electronic goods in Illinois and contrast it to those of California.
- Find products with correlations between time spent at quality control and returns.
- Identify conditions that increase total path duration for printers in the northeast.

Problem Statement

FlowCube construction problem

- Fact table: RFID Path data set.
- Dimensions: Item dimensions and path dimensions.
- Measure: Probabilistic workflow summarizing the flow patterns of the paths aggregated in the cell.

Why is this problem hard?

- The fact table is very large (terabytes or maybe petabytes).
- The number of cuboids is exponential in the number of dimensions.
- Computing the workflow for each cell is expensive.

Outline

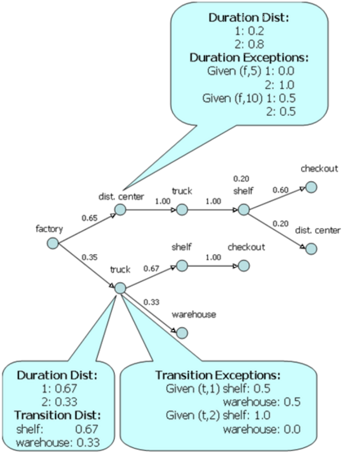
FlowGraph Definition

- Tree shaped workflow that summarizes the flow patterns for an item or group of items.
  - Nodes: Locations
  - Edges: Transitions
- Each node is annotated with:
  - Distribution of durations at the node
  - Distribution of transition probabilities
  - Exceptions to duration and transition probabilities
    - Minimum support: Frequent exceptions
    - Minimum deviation: Surprising exceptions
- Highly compressed and accurate representation

RFID Data

- Readers generate raw tuples of the form:
  (EPC, location, time)
- We can sort the tuples on EPC and generate paths of the form:
  $\langle \text{EPC}, (l_1, t_1), (l_2, t_2), ..., (l_k, t_k) \rangle$
  where l_i is the i-th location, and t_i is i-th duration.
- The paths can be augmented with item dimensions, e.g.:
  $$\langle \underbrace{Product, Manufacturer, Price}_{item\ dimensions}, \underbrace{(l_1, t_1), (l_2, t_2), ..., (l_k, t_k)}_{path\ stages} \rangle$$

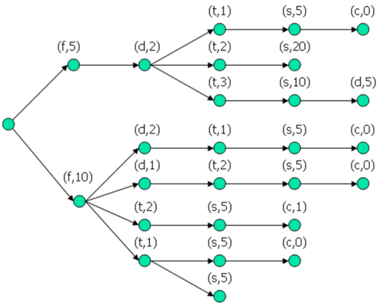
FlowGraph Example



Path Data

id	product	brand	path
1	tennis	nike	$(f, 10)(d, 2)(t, 1)(s, 5)(c, 0)$
2	tennis	nike	$(f, 5)(d, 2)(t, 1)(s, 10)(c, 0)$
3	sandals	nike	$(f, 10)(d, 1)(t, 2)(s, 5)(c, 0)$
4	shirt	nike	$(f, 10)(t, 1)(s, 5)(c, 0)$
5	jacket	nike	$(f, 10)(t, 2)(s, 5)(c, 1)$
6	jacket	nike	$(f, 10)(t, 1)(w, 5)$
7	tennis	adidas	$(f, 5)(d, 2)(t, 2)(s, 20)$
8	tennis	adidas	$(f, 5)(d, 2)(t, 3)(s, 10)(d, 5)$

Alternative FlowGraph Design



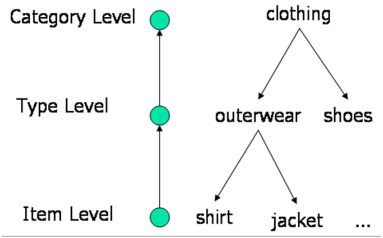
Duration dependent nodes

- Distinct node for every location and duration combination
- Significantly larger workflow
- Lots of redundancy if durations and transitions are independent of the path.

Outline

1. Motivation
   - RFID Technology
   - Problem Statement
2. FlowGraphs
   - Definition
   - Alternative Design
3. FlowCubes
   - Abstraction Lattice
   - FlowCube Design
   - Algorithm

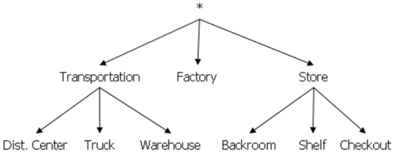
Item abstraction lattice

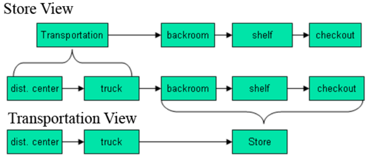


Item lattice

- Each item dimension has a concept hierarchy
- The set of concept hierarchies for all item dimensions forms an item lattice
- Item dimensions can be aggregated to any level in the item lattice

Path abstraction lattice





Path lattice

- the levels of the location and time dimensions of each path stage forms a path lattice
- Path stages can be aggregated to a given level in the path lattice.

Path views

- Each path can be aggregated at different abstraction levels
- We collapse path stages using the location lattice

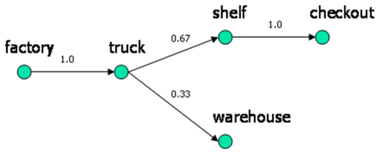
FlowCube Design

FlowCube

- Data cube computed on path data set
- Cuboids for interesting levels of the item and path lattices.
- Cells record a FlowGraph as measure.

Example cuboid

cell id	category	brand	path ids
1	shoes	nike	1,2,3
2	shoes	adidas	7,8
3	outerwear	nike	4,5,6



Which cells to compute?

Non-Redundant cells

- Cells which FlowGraph can not be inferred from available cells
- If the FlowGraph for milk 2% is the same as for milk then it is redundant
- non-redundant cells generate smaller cuboids and highlight important properties of flow patterns

Frequent cells

- Compute only cells that pass minimum support
- Well supported FlowGraphs are statistically significant
- Iceberg FlowCubes provide significant compression.

FlowCube construction - key ideas

- Compute the FlowGraph for each frequent cell
- Main cost: Determine frequent cells, and frequent path segments (used for exception computation).
- We can compute frequent path segments and cells simultaneously
- Transform the path database into a transaction database and do Apriori mining of frequent cells and frequent path segments
- Compute cells with minimum support, and frequent path segments simultaneously
  - Cross-pruning: Infrequent path segments at high level cells, can not be frequent at low level cells and infrequent cells can not contain frequent path segments
- In a single scan count frequent cells and frequent path segments at every abstraction level

Transaction encoding

Concept hierarchy encoding

- Values for item dimensions encode their abstraction level, e.g., Jacket = 1112, outerwear = 111*, clothing = 11**, product = 1***
- Benefit: In a single scan values at all abstraction levels are counted

Path encoding

- Path stages encode their prefix, location level, and time level, e.g., given the path:

  (factory, 10) $\longrightarrow$ (dist, 2) $\longrightarrow$ (truck, 1) $\longrightarrow$ (shelf, 5) $\longrightarrow$ (checkout, 0)

  we can encode the third stage as

  (factory:dist,truck,1), (factory:Transportation,1), (factory:dist:truck,*)
- Benefit: In a single scan paths at at abstraction levels can be counted

Example transaction encoding

Path Database

id	product	brand	path
1	tennis	nike	$(f, 10)(d, 2)(t, 1)(s, 5)(c, 0)$
2	tennis	nike	$(f, 5)(d, 2)(t, 1)(s, 10)(c, 0)$
3	sandals	nike	$(f, 10)(d, 1)(t, 2)(s, 5)(c, 0)$
4	shirt	nike	$(f, 10)(t, 1)(s, 5)(c, 0)$
5	jacket	nike	$(f, 10)(t, 2)(s, 5)(c, 1)$
6	jacket	nike	$(f, 10)(t, 1)(w, 5)$
7	tennis	adidas	$(f, 5)(d, 2)(t, 2)(s, 20)$
8	tennis	adidas	$(f, 5)(d, 2)(t, 3)(s, 10)(d, 5)$

$\Downarrow$

transaction database

tid	items
1	{121, 211, (f,10),(fd,2),(fdt,1),(fdts,5),(fdtsc,0)}
2	{121,211,(f,5),(fd,2),(fdt,1),(fdts,10),(fdtsc,0)}
3	{122,211,(f,10),(fd,1),(fdt,2),(fdts,5),(fdtsc,0)}
4	{111,211,(f,10),(ft,1),(fts,5),(ftsc,0)}
5	{112,211,(f,10),(ft,2),(fts,5),(ftsc,1)}
6	{112,211,(f,10),(ft,1),(ftw,5)}
7	{121,221,(f,5),(fd,2),(fdt,2),(fdts,20)}
8	{121,221,(f,5),(fd,2),(fdt,3),(fdts,10),(fdtsd,5)}

Shared Algorithm

1. Compute transaction database, count frequent cells and frequent path segments of length 1 into L_1, pre-count high level patterns of length > 1 into P_1
2. For $k = 2$, L_{k-1} not empty, $k++$
3. Generate candidates C_k by joining L_{k-1}
4. Prune unpromising candidates
    - Based on P_k
    - Unrelated stages
    - Item and ancestor
5. Collect counts for C_k into L_k and compute P_k
6. Return $\bigcup_i L_i$

Shared Algorithm

1. Compute transaction database, count frequent cells and frequent path segments of length 1 into L_1, pre-count high level patterns of length > 1 into P_1
2. For $k = 2$, L_{k-1} not empty, $k++$
3. Generate candidates C_k by joining L_{k-1}
4. Prune unpromising candidates
   - Based on P_k
   - Unrelated stages
   - Item and ancestor
5. Collect counts for C_k into L_k and compute P_k
6. Return $\bigcup_i L_i$

Shared Algorithm

1. Compute transaction database, count frequent cells and frequent path segments of length 1 into L_1, pre-count high level patterns of length > 1 into P_1
2. For $k = 2$, L_{k-1} not empty, $k++$
3. Generate candidates C_k by joining L_{k-1}
4. Prune unpromising candidates
    - Based on P_k
    - Unrelated stages
    - Item and ancestor
5. Collect counts for C_k into L_k and compute P_k
6. Return $\bigcup_i L_i$

Shared Algorithm

1. Compute transaction database, count frequent cells and frequent path segments of length 1 into L_1, pre-count high level patterns of length > 1 into P_1
2. For $k = 2$, L_{k-1} not empty, $k++$
3. Generate candidates C_k by joining L_{k-1}
4. Prune unpromising candidates
   - Based on P_k
   - Unrelated stages
   - Item and ancestor
5. Collect counts for C_k into L_k and compute P_k
6. Return $\bigcup_i L_i$

Shared Algorithm

1. Compute transaction database, count frequent cells and frequent path segments of length 1 into L_1, pre-count high level patterns of length > 1 into P_1
2. For $k = 2$, L_{k-1} not empty, $k++$
3. Generate candidates C_k by joining L_{k-1}
4. Prune unpromising candidates
   - Based on P_k
   - Unrelated stages
   - Item and ancestor
5. Collect counts for C_k into L_k and compute P_k
6. Return $\bigcup_i L_i$

Alternative - Cubing based algorithm

Cubing algorithm

Using a bottom up algorithm, construct an Iceberg data cube on the item dimensions. Run a frequent pattern mining algorithm on each cell of the cube, and build the FlowGraphs.

Issues

- FlowGraphs are holistic measures difficult to compute bottom up
- cross pruning opportunities between path and item lattices are lost, e.g., infrequent path segments at high level cells are repeatedly counted on every cell
- Large cost of storing lists of transaction identifiers during cuboid phase

Experimental Setup

Data synthesis

- Synthetic path generator, emulates large retailer
- Path dimensions have 3 levels each
- Location, and duration dimensions 2 levels each
- Process: generate item dimensions, generate path, assign durations

Algorithms

- Shared: simultaneous counting + pruning
- BUC: cubing + Apriori
- Basic: Shared without pruning

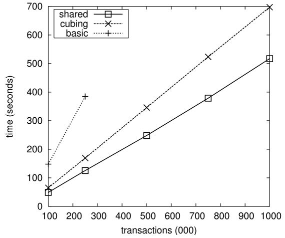
Experiments

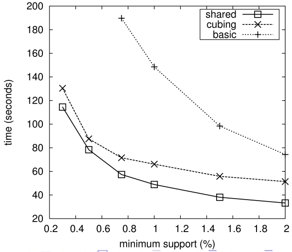
Path database size

- Construction time vs db size
- min sup = 0.01, item dimensions = 5
- Shared scales well, cubing slows with dense cube

Minimum support

- Construction time vs support
- Paths = 100,000, item dims = 5
- Shared better, basic improves when few candidates

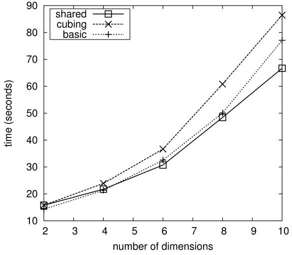




Experiments

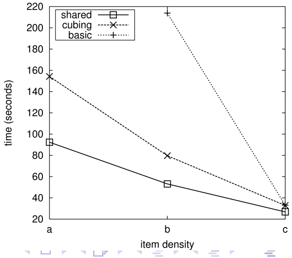
Number of dimensions

- Construction time vs item dimensions
- min sup = 0.01, paths = 100,000
- spare cube ⇒ similar performance



Item density

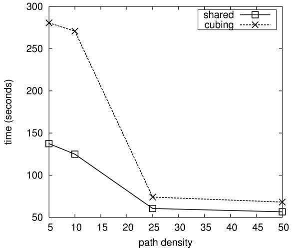
- Construction time vs Item dimension density
- Paths = 100,000, item dims = 5, a dense, c sparse
- Shared much better in dense cubes



Experiments

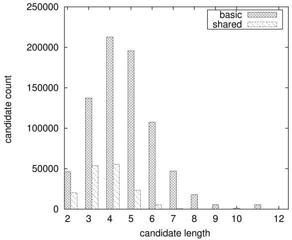
Path density

- Construction time vs distinct paths
- min sup = 0.01, paths = 100,000, item dims = 5
- dense paths ⇒ shared shines



Pruning power

- Candidates to evaluate, with and without pruning
- Pruning techniques provide dramatic advantage



Conclusions

- FlowGraph: Succinct summary of general flow patterns and exceptions.
- FlowCube: Data cube on paths with FlowGraphs for measure. OLAP over flow patterns.
- Algorithm: Shared computation of frequent cells, and frequent path segments.