CS 8803-MDS

Human-in-the-loop Data Analytics

Lecture 24

11/15/23

Today's class

M4: A Visualization-Oriented Time Series Data Aggregation

Authors: Faith, Yihan

Archaeologists: Yuxin, Daniel

Hacker: Tony

Practitioner: Pearl

M4: A Visualization-Oriented Time Series Data Aggregation

Paper Author: Yihan Ping, Faith Womack

Have you ever dealt with this problem?

121. Best Time to Buy and Sell Stock

Solved

Easy | Topics | Companies

You are given an array `prices` where `prices[i]` is the price of a given stock on the i^{th} day.

You want to maximize your profit by choosing a **single day** to buy one stock and choosing a **different day in the future** to sell that stock.

Return *the maximum profit you can achieve from this transaction*. If you cannot achieve any profit, return `0`.

Example 1:

```
Input: prices = [7,1,5,3,6,4]
Output: 5
Explanation: Buy on day 2 (price = 1) and sell on day 5 (price = 6), profit = 6-1 = 5.
Note that buying on day 2 and selling on day 1 is not allowed because you must buy before you sell.
```

Example 2:

```
Input: prices = [7,6,4,3,1]
Output: 0
Explanation: In this case, no transactions are done and the max profit = 0.
```

time	symbol	price
2023-02-06 15:05:26	PFE	44.19
2023-02-06 15:05:25	WMT	141.27
2023-02-06 15:05:24	KO	59.67
2023-02-06 15:05:24	TSLA	194.08
2023-02-06 15:05:24	SNAP	10.88

High-Volume Time Series Data

High-Volume Time Series Data

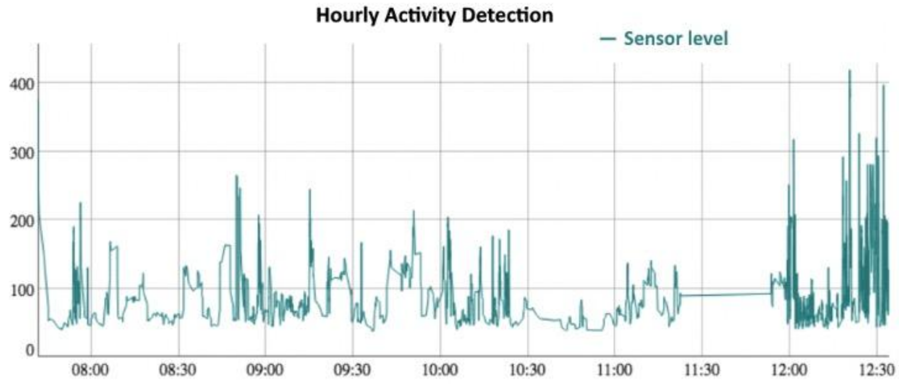
- Finance:

  Transaction Record

  Stock Market Data

- Manufacture:

  Sensor Data



Industry workflow

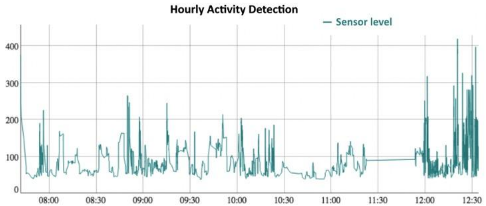
Raw high volume time series data

id	name	type	size
1	Rex	dog	big
2	Cleo	dog	small
3	Leroy	dog	medium
4	Sunny	cat	big
5	Snow Ball	cat	medium

id_pet	weight	age
1	32,3	7
2	4,35	13,6
3	24,9	4
5	8,75	5

name	food	danger_level
Cleo	diet	5
Sunny	wet	1
Rex	dry	4
Leroy	wet	3

A series of queries



Backend Visual Analysis

1.

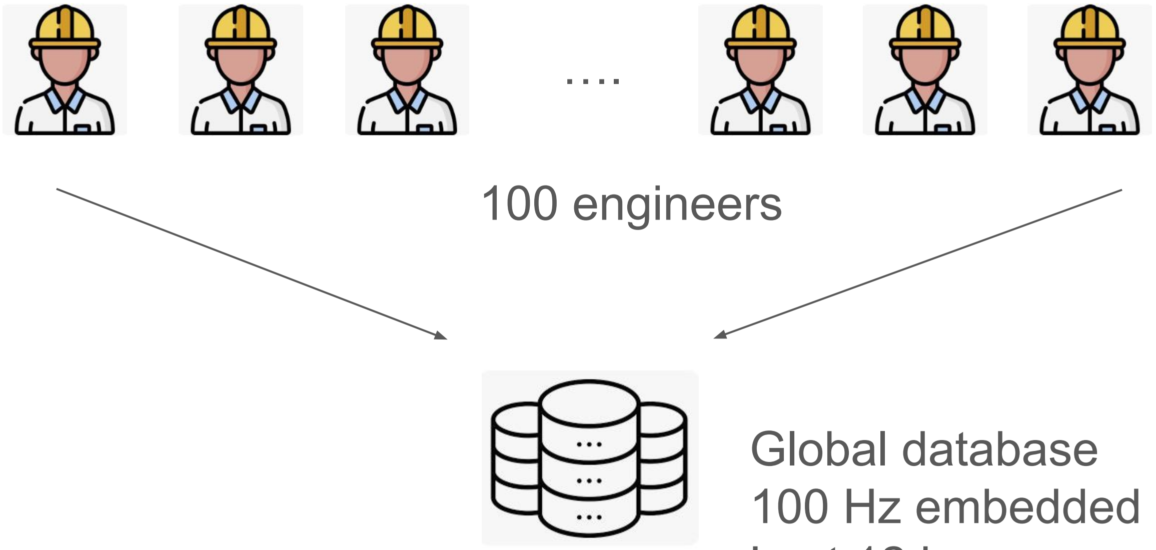
Question: How to balance time complexity and visualization accuracy

Current Visual Analytics Tools

Tableau, QlikView, SAP Lumira

- Didn’t consider the cardinality of the query result
- > Millions of rows
- > High Bandwidth consumption

Example: Equipment Sensor Data from Manufacturing Machine



Global database
100 Hz embedded sensor data
Last 12 hours

Example: Equipment Sensor Data from Manufacturing Machine

100 engineers

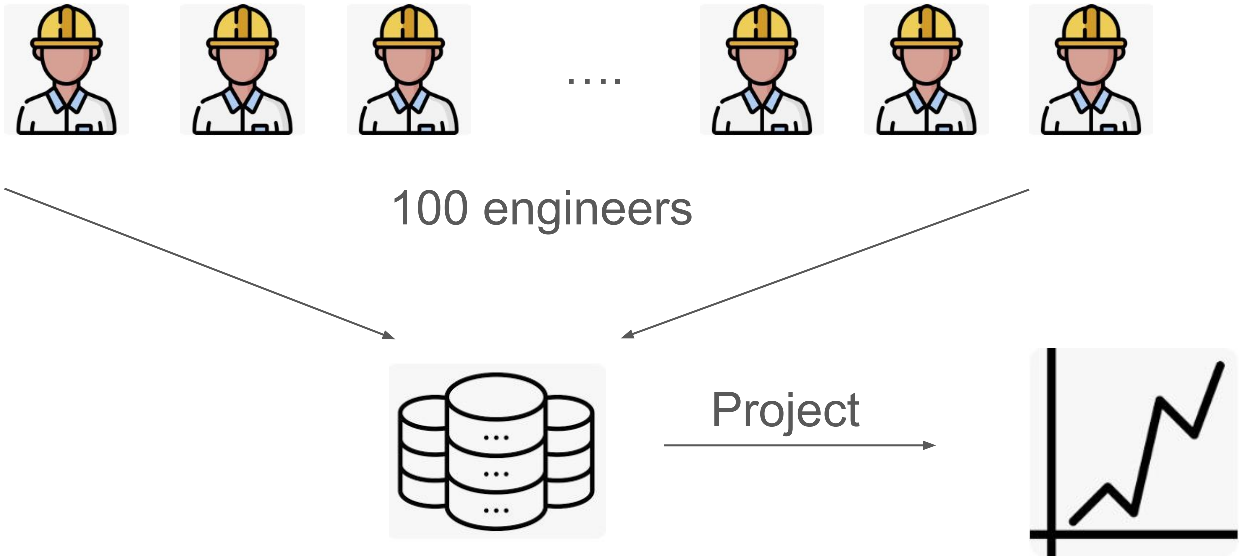
Non-aggregating Query

```
SELECT time,value FROM sensor WHERE time > NOW()-12*3600
```

$100_{\text{users}} * (12 * 3600)_{\text{seconds}} * 100_{\text{Hz}} = 432$ million

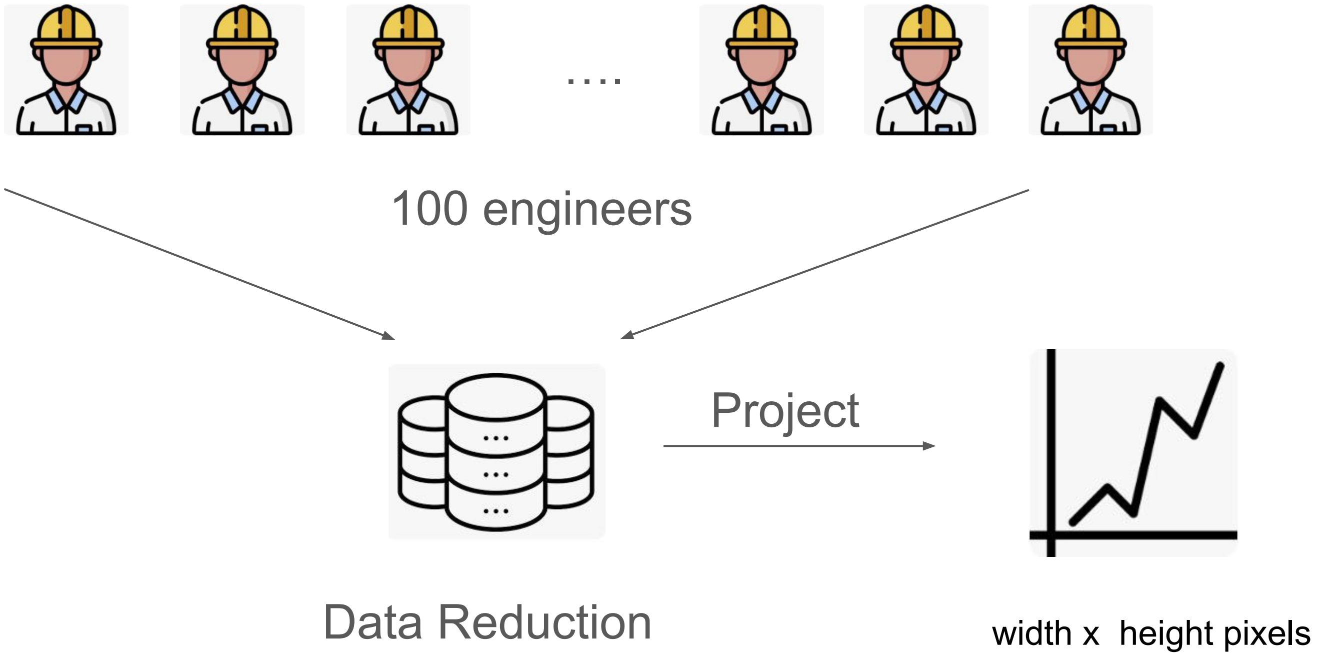
- Large amount of data
- Long waiting time

Example: Equipment Sensor Data from Manufacturing Machine



width x height pixels

Example: Equipment Sensor Data from Manufacturing Machine



Two Key Technical Ideas:

1. Query Rewrite

- apply an appropriate data reduction at the query level within the database
- Relying on relational operators and Parameter(Width and height)

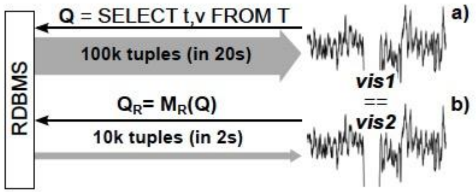


Figure 1: Time series visualization: a) based on a unbounded query without reduction; b) using visualization-oriented reduction at the query level.

Large amount of data (Not compute inside the database)

Long waiting time

2. Line Charts

- Select only necessary data points
- Each interval (pixel column) : selecting tuples with the MIN and MAX value, MIN and MAX timestamp

1. Query Rewriting

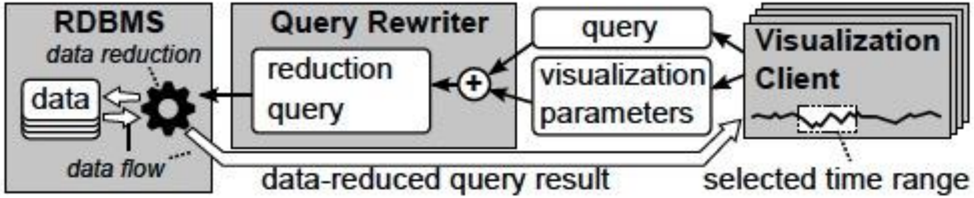


Figure 2: Visualization system with query rewriter.

Visualization Client: (1) Query Definition (2)Visualization Parameters

1. Query Rewriting

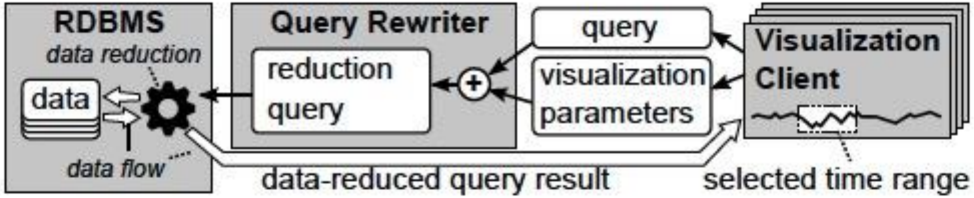


Figure 2: Visualization system with query rewriter.

(1) Query Definition:

SELECT time , value FROM series WHERE time > t1 AND time < t2

1. Query Rewriting

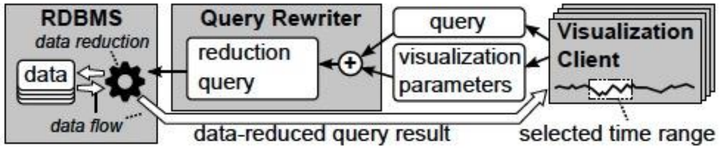
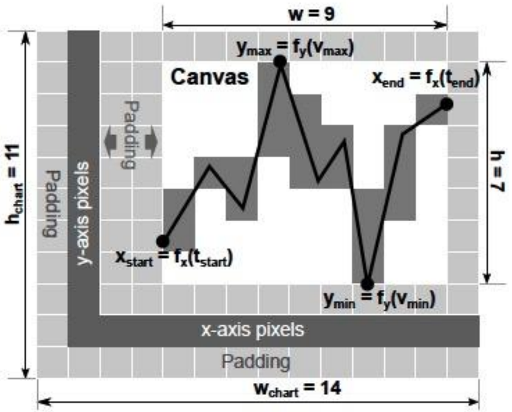


Figure 2: Visualization system with query rewriter.

(2) Visualization Parameters: visualization width and height



$$f_x(t) = w \cdot (t - t_{start})/(t_{end} - t_{start})$$
$$f_y(v) = h \cdot (v - v_{min})/(v_{max} - v_{min})$$

1. Query Rewriting

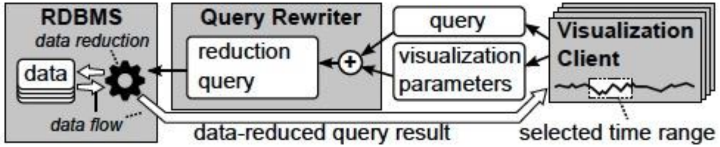


Figure 2: Visualization system with query rewriter.

Query Rewriter

```
WITH Q  AS (SELECT t,v FROM sensors WHERE               1) original query Q
            id = 1 AND t >= $t1 AND t <= $t2),
     QC AS (SELECT count(*) c FROM Q)                   2) cardinality query Q_C
SELECT * FROM Q WHERE (SELECT c FROM QC) <= 800         3a) use Q if low card.
UNION
SELECT * FROM (                                         reduction query Q_D:
     SELECT min(t),avg(v) FROM Q                        compute aggregates
     GROUP BY round(200*(t-$t1)/($t2-$t1))              for each pixel-column
) AS QD         WHERE (SELECT c FROM QC) >  800         3b) use Q_D if high card.
```

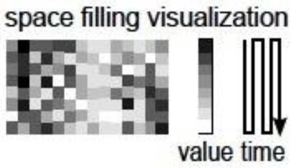
$$f_g(t) = round(w \cdot (t - t_{start})/(t_{end} - t_{start}))$$

Eg. w = 200
discrete group key between 0 and w = 200

2. Line Chart - High Volume Data

Too Much Space/Element!

Limited Potential for Data Reduction



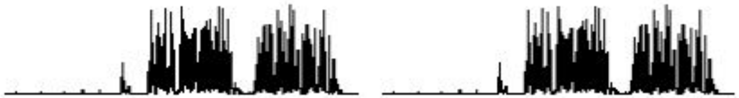
M4 Visualization-Oriented Data Aggregation Model

- Groups time series into equidistant time spans for each pixel column
- For each interval (pixel column) compute aggregated values according to

- **Composite Value preserving Aggregation**
    - Preserves shape of a time series by focusing on extrema (important for line visualizations because representing peaks and troughs is essential)
    - Necessary for line rasterization (lines visualized with a set wxh raster)
    - Superior in maintaining visualization quality and data efficiency.
    - Just using MinMax aggregation ignores the first and last tuples of each group, changing the shape of the time series

a) value-preserving M4 aggregation query

```
SELECT t,v FROM Q JOIN
(SELECT round($w*(t-$t1)/($t2-$t1)) as k, --define key
        min(v) as v_min, max(v) as v_max, --get min,max
        min(t) as t_min, max(t) as t_max  --get 1st,last
        FROM Q GROUP BY k) as QA          --group by k
ON k = round($w*(t-$t1)/($t2-$t1))        --join on k
   AND (v = v_min OR v = v_max OR         --&(min|max|
        t = t_min OR t = t_max)           --  1st|last)
```



b) resulting image == expected image

Figure 7: M4 query and resulting visualization.

M4 Aggregation Performance

- Complexity of M4 Aggregation:
  - The grouping and computation of aggregated values can be done in $O(n)$ time for n tuples
  - An equi-join is performed between the aggregated values and Q, matching n tuples in Q with $4 \cdot w$ aggregated tuples.
  - This join uses a hash-join method, which has a complexity of $O(n + 4 \cdot w)$.
  - The value of w (width) does not depend on n and is limited by the physical display resolutions (e.g., $w = 5120$ pixels for WHXGA displays).

Overall, the M4 aggregation process has a complexity of $O(n)$.

- M4 Upper Bound
  - Proof there's an upper bound of tuples necessary for an error-free visualization
  - Conclusion: no matter how big of a T(time series), selecting 4*w tuples (where w is width of raster)

Evaluation

- Conducted using real-world datasets, focusing on the visualization quality and query execution performance. The datasets include stock price data, soccer ball speed sensor data, and machine sensor data.
- Used SSIM to compare visualizations considering human perception
  - Normalized distance measure between visualizations 1 and 2.

$$DSSIM(V_1, V_2) = \frac{1 - SSIM(V_1, V_2)}{2} \qquad (6)$$

- Evaluated quality of line visualization based on the reduced time series compared to the original line visualization

Evaluation - Query Time

Evaluation Queries

- **Baseline Query:** Selects all tuples for visualization.
- **PAA-Query:** Computes up to 4·w average tuples.
- **Two-Dimensional Rounding Query:** Selects up to w·h rounded tuples.
- **Stratified Random Sampling Query**: Selects 4·w random tuples.
- **Systematic Sampling Query:** Selects 4·w first tuples.
- **MinMax Query:** Selects two minimum and maximum tuples from 2·w groups.
- **M4 Query:** Selects all four extrema (minimum, maximum, first, and last values) from w groups.

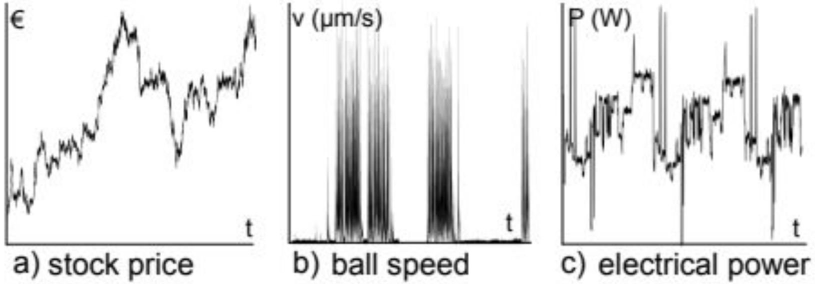


Figure 11: a) financial, b) soccer, c) machine data.

Evaluation - Query Time

M4 reduces the time the user has to wait for the data by one order of magnitude in all tested scenarios, and still provides the correct tuples for high quality line visualizations.

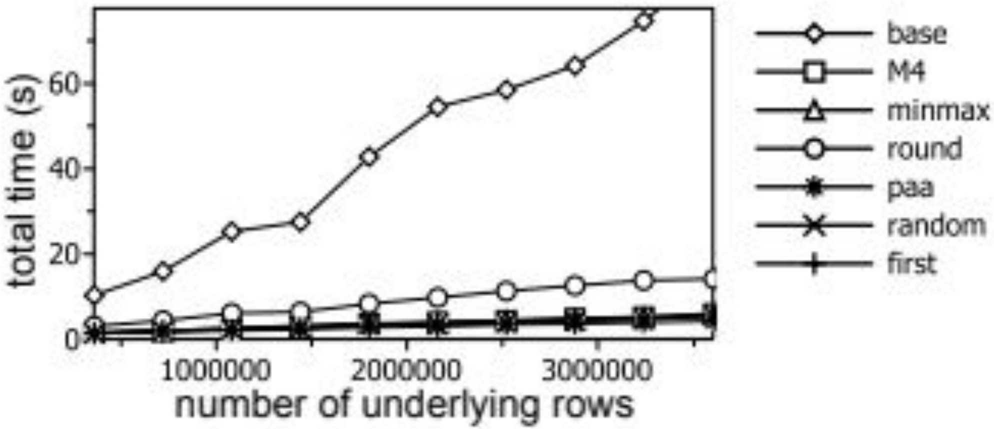


Figure 13: Performance with varying row count.

Evaluation - Data Efficiency

Resulting visualization quality (DSSIM) over the resulting number of tuples of each different groupings from {1, 2.5x*w*} of an applied data reduction technique.

- Average M4 visualization quality of DSSIM > 0.9 but usually below MinMax and line simplification techniques.
- at n_h = w, i.e., at any factor k of w (width of the visualization), M4 provides perfect (error-free) visualizations.

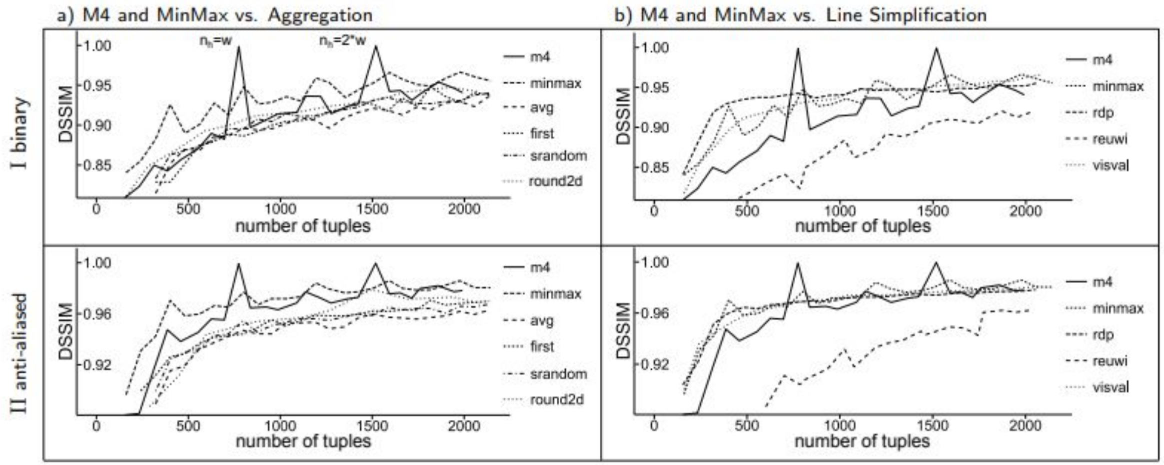


Figure 14: Data efficiency of evaluated techniques, showing DSSIM over data volume.

Overall

This is making the process of making charts computationally less intensive and faster (less data needs less bandwidth) without affecting the quality of produced visualizations.

MinMax Method:
- Creates long, incorrect connection lines, especially noticeable on the right of the main positive spikes in the chart.
- Introduces smaller errors due to the same issue.

RDP (Ramer-Douglas-Peucker Algorithm):
- Performs better in breaking up incorrect lines by detecting significant distances of unselected points.
- Applies averaging in low-variance areas of the time series.

PAA (Piecewise Aggregate Approximation):
- Leads to the most pixel errors, mainly due to averaging of vertical extremes.
- Results in over 100 false pixels.

Comparison of Pixel Errors:

MinMax: 30 false pixels.
RDP: 39 false pixels.
PAA: Over 100 false pixels.
M4 Method: 0

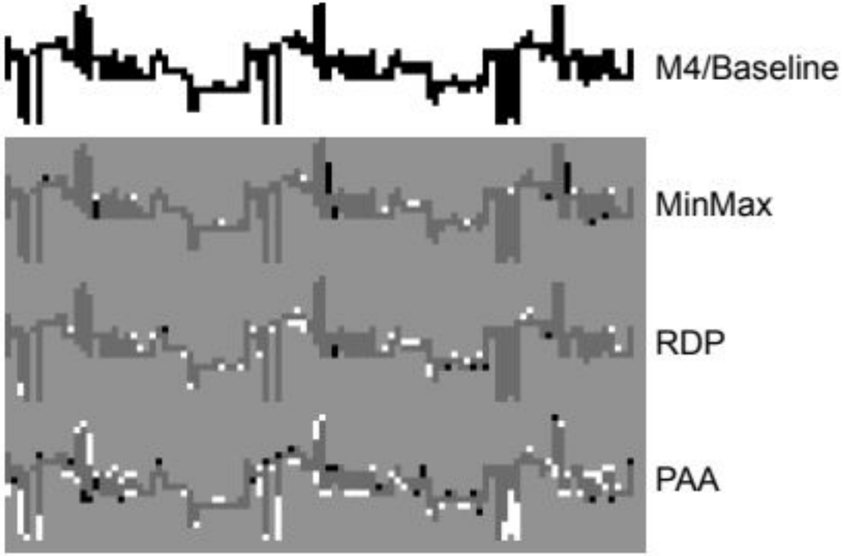


Figure 15: Projecting 40k tuples to 100x20 pixels.

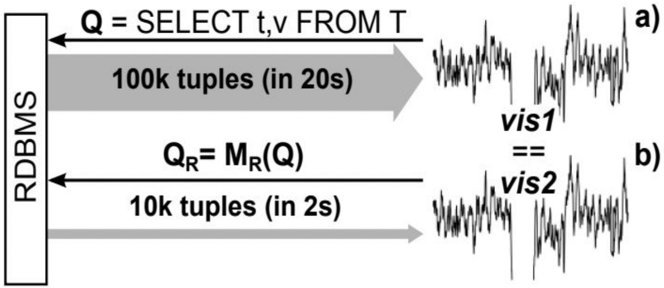
Questions

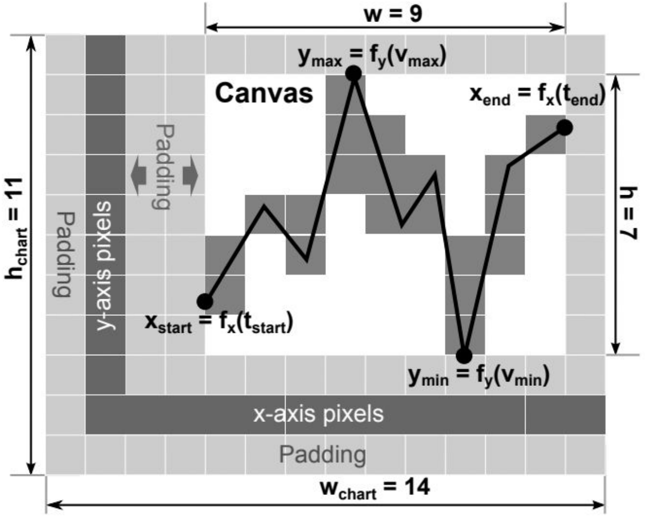
M4: A Visualization-Oriented Time Series Data Aggregation

Prior Work Archaeologist: Yuxin Cao

M4 Aggregation

- Groups time-series data into w equidistant time spans ⇔ pixel columns in visual graph
- For each group, compute
  - min(v), max(v)
  - min(t), max(t)
- Advantages
  - Error-free line visualization
  - Smaller result set => Low latency
  - Efficient aggregation: O(n)





A Simple Dimensionality Reduction Technique for Fast Similarity Search in Large Time Series Databases

Eamonn J. Keogh and Michael J. Pazzani

A Simple Dimensionality Reduction Technique for Fast Similarity Search in Large Time Series Databases

- Similarity search in time series databases
  - Useful for applications like clustering, classification, and data mining
- Time series databases are typically very large in data size
  - Dimensionality reduction on data
  - Indexing the data in the transformed space

Problem

- Similarity search for time series data
  - Whole Matching
  - Subsequence Matching
- Focus on nearest neighbor
  - Given a query X and a database consisting of K time series Y_{1,...,k}, find the most similar time series Y_i such that Euclidean distance between X and Y_i is minimized

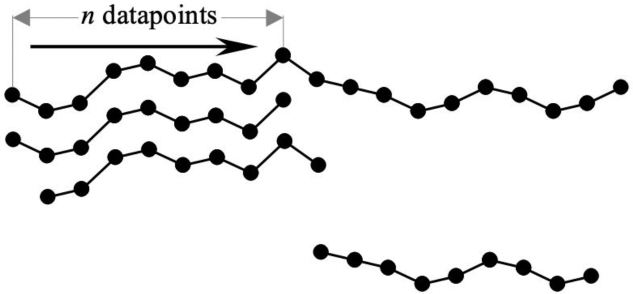
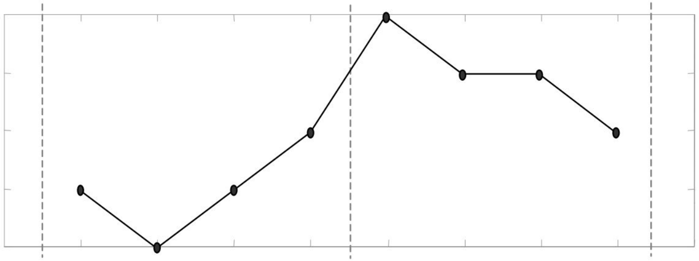


Figure 1: The subsequence matching problem can be converted into the whole matching problem by sliding a "window" of length n across the long sequence and making copies of the data falling within the windows

Approach: PCA Indexing



$X = (-1, -2, -1, 0, 2, 1, 1, 0)$

$n = |X| = 8$

$\overline{X} = (\text{mean}(-1,-2,-1,0), \text{mean}(2,1,1,0))$

$\overline{X} = (-1, 1)$ $\quad N = |\overline{X}| = 2$

- Dimensionality reduction
  - Time series data of length n is divided into N equi-sized frames (n > N)
  - Record the mean of each frame => mean value vector becomes reduced representation
- Building the index
  - Indexing all transformed sequences as points in an N-dimensional space
  - Each point contains a pointer to the corresponding original sequence on disk
- Searching the index
  - Search the indexing structure for NN of transformed query X*
  - Calculate true Euclidean distance between original sequence of NN and X*
- Handling queries of various length
  - Handle queries shorter or longer than the length for which the index was built

Strengths and Major Contributions

- Simple to understand and implement
- Allows more flexible queries
    - Handle many different distance measures (e.g. weighted Euclidean distance)
- Allows queries to have shorter or longer length than the index
- Complexity of index building: O(n)

Representing financial time series based on data point importance

Tak-chung Fu, Fu-lai Chung, Robert Luk, Chak-man Ng

Representing Financial Time Series Based on Data Point Importance

- Characteristics of time series
  - Large in data size
  - High dimensionality
  - Update continuously
- Financial time series is represented according to importance of data points
- Uses a binary tree data structure to represent the time series
  - Supports incremental updating
- Present the time series in different levels of details

Data Point Importance (PIPs)

- To what extent does each data point affect the **overall shape** of the time series?
- Evaluated with **Perceptually Important Points (PIPs)**

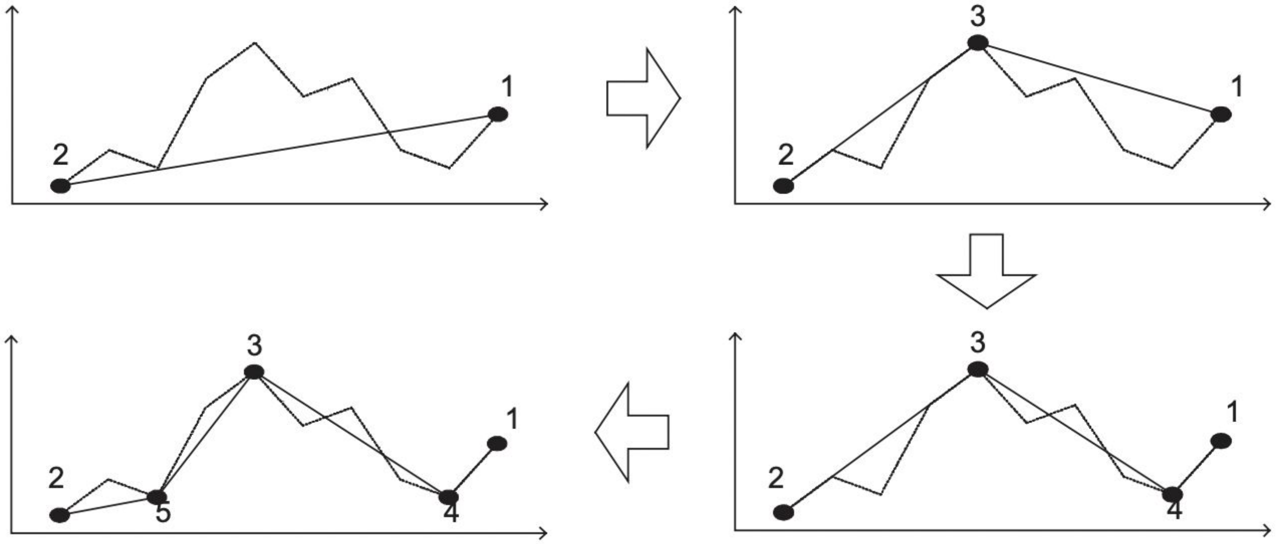


Fig. 6. Identification of the first 5 PIPs using PIP-VD.

Data Point Importance (PIPs)

- How to calculate the distance to the **two adjacent PIPs**?
- 3 evaluation methods:

Euclidean distance (PIP-ED)

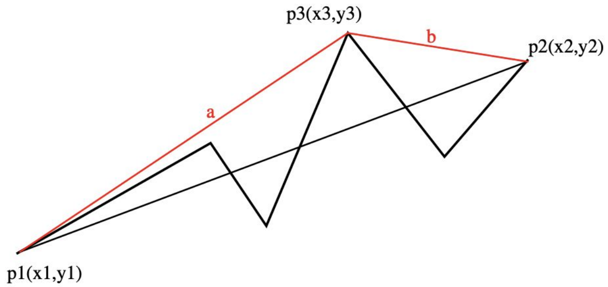


Fig. 3. Euclidean distance-based data point importance evaluation method (PIP-ED).

Perpendicular distance (PIP-PD)

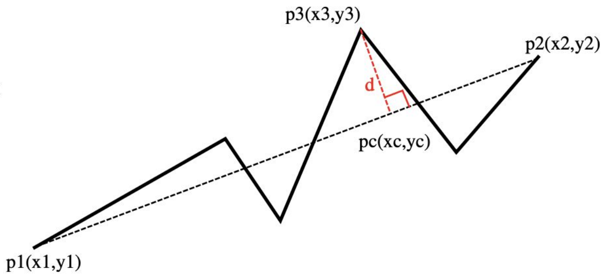


Fig. 4. Perpendicular distance-based data point importance evaluation method (PIP-PD).

Vertical distance (PIP-VD)

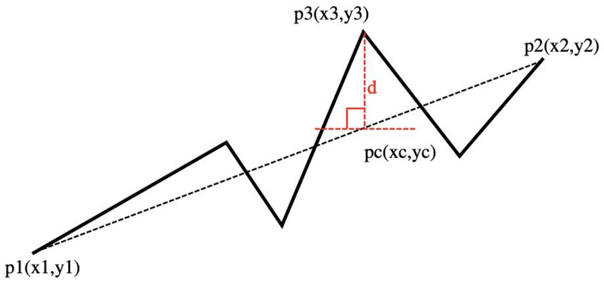


Fig. 5. Vertical distance-based data point importance evaluation method (PIP-VD).

Specialized Binary Tree (SB-Tree)

- SB-Tree structure
  - Hierarchy of data points (PIPs)
  - **Node**: x- and y-coordinates of PIP
  - **Path**: distance to child node (next PIP)

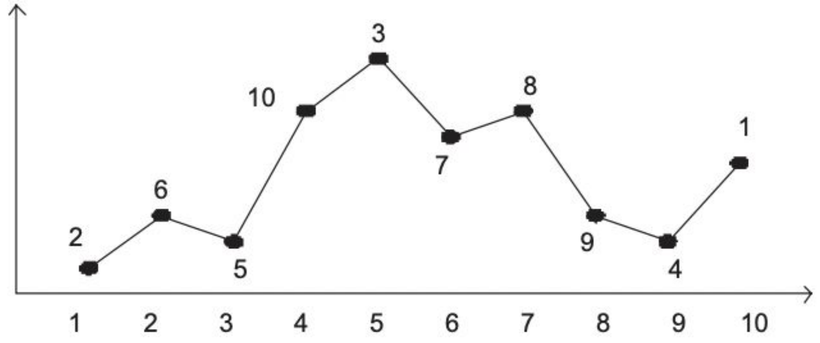


Fig. 7. The importance of the PIPs after the identification process using PIP-VD.

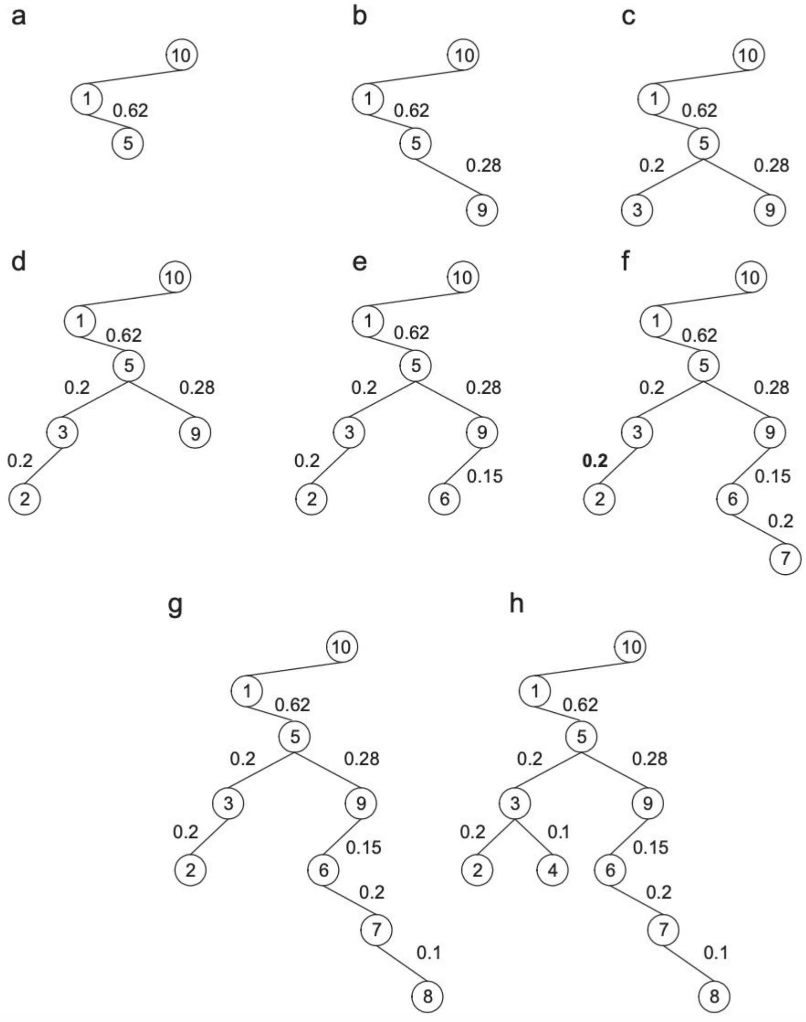


Fig. 9. SB-Tree building process.

SB-Tree: Incremental Updating

- New time series data is available frequently and continuously
- Efficient point-by-point updating mechanism for SB-Tree is necessary
    - Appending a new data point to the rightmost of the time series can have different effects on the tree structure

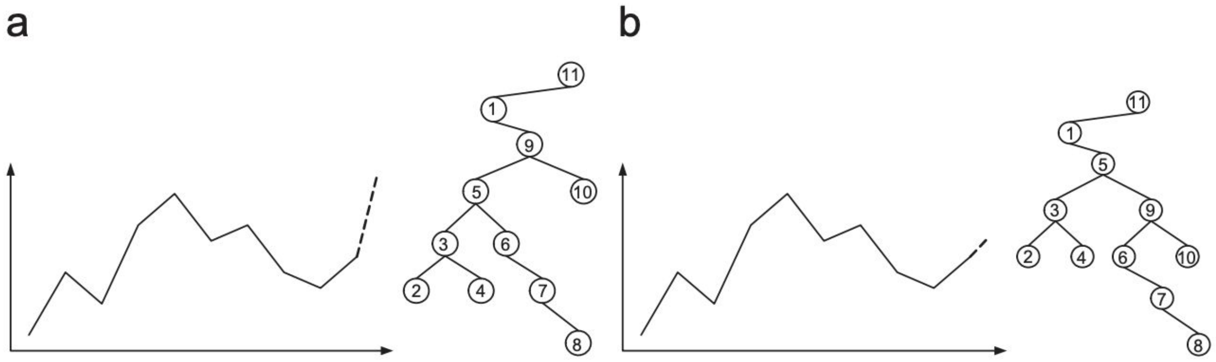


Fig. 10. Different behaviors after adding a new data point.

SB-Tree: Dimensionality Reduction

- Reduce the size of the tree to minimize space consumption

1. Tree Pruning Approach

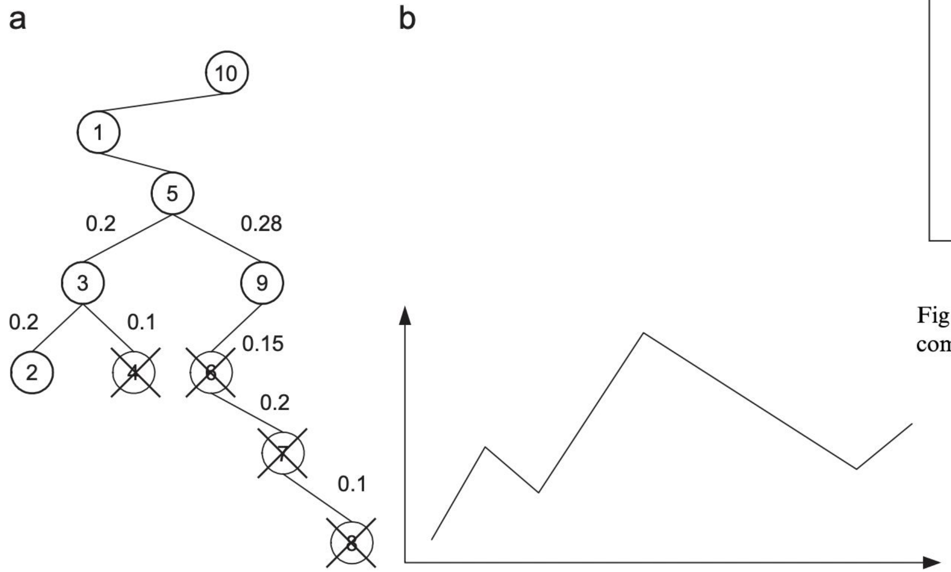


Fig. 19. Dimensionality reduction by tree pruning approach: ($\lambda = 0.2$) (a) the accessing result of the pruned SB-Tree and (b) the result after dimensionality reduction.

2. Error Threshold Approach

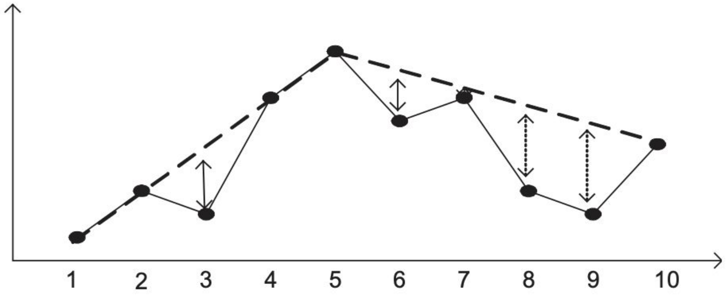
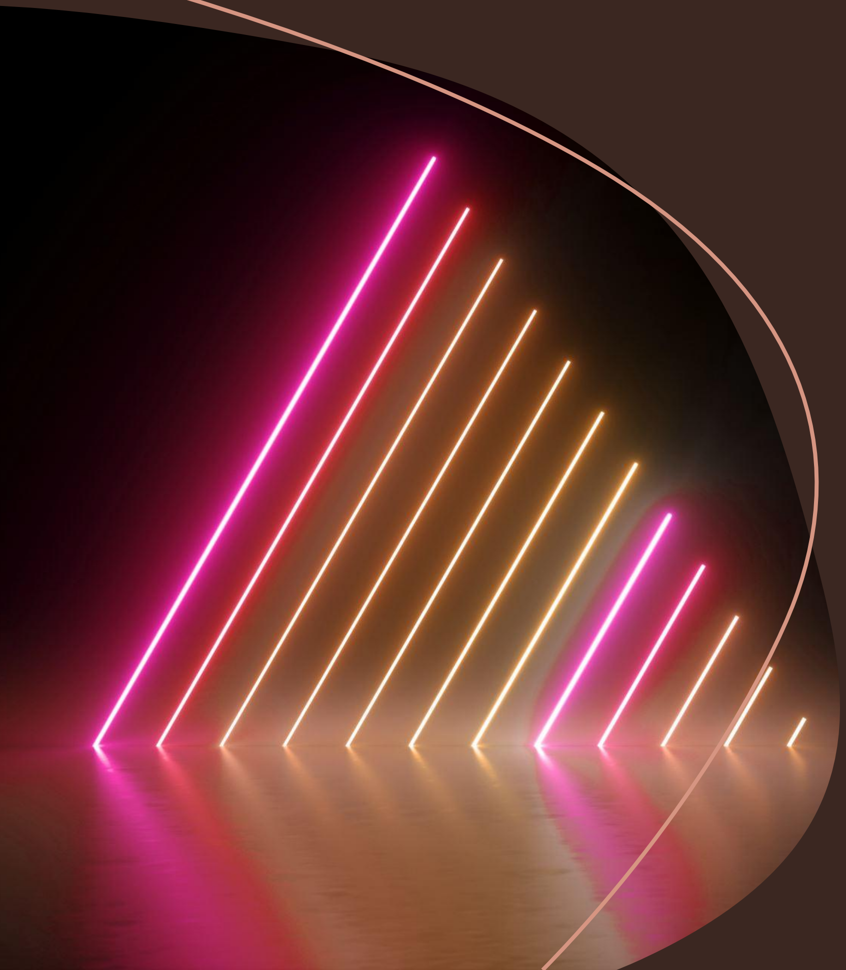


Fig. 20. Error in representing a time series when only 3 PIPs are used compared with the original time series.

Thank you!



M4: A Visualization-Oriented Time Series Data Aggregation

Archeologist 2: Daniel Lyczak

Load-n-Go: Fast Approximate Join Visualizations That Improve Over Time

2017

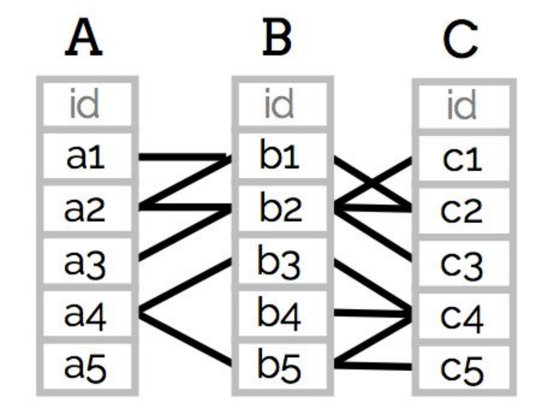
Authors:
Marianne Procopio
Carlos Scheidegger
Eugene Wu
Remco Chang

Bottle Necking [The Issue]

- Responsive visualizations require interactive speeds
- Doesn’t scale -> increased data --> exponential wait
- Joins intensify this problem as they are costly
- Precomputing helps query response but requires long wait times
- Enter Approximate Query Processing

Approximate Query Processing

- Sample records while executing the query and provide results with a confidence interval
- Ripple Join randomly samples from each table, but its CI convergence can require [too] many records
- Wander Join uses graphs, walking edge to find a valid join from record A to table B
- Not good for visualizations by drawing samples independent of WHERE and GROUP BY <-> High Selectivity <-> Wander Join Struggles

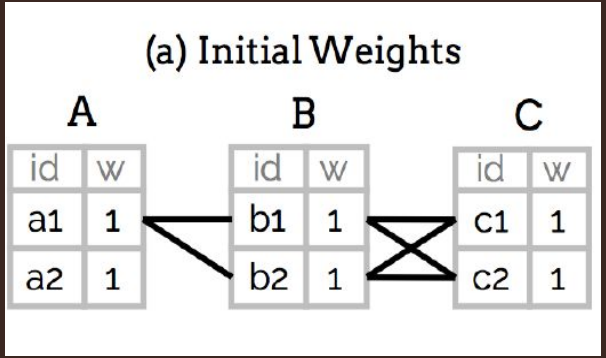


Load-n-Go

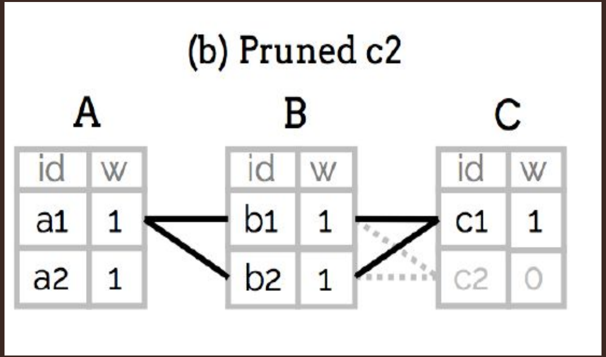
- Builds on Wander Join by considering query filters and skewness (disproportionate groups in GROUP BY)
- Prioritizes samples most likely to satisfy filters for 6x faster results
- Faster results as measured by number of samples needed to reach convergence
- Accomplished by injecting importance sampling to Wander Join, weighting samples by importance to query, moving away from uniform sampling of the entire table used in Wander Join

Importance Sampling

- In filter queries, non-matching records assigned weight 0. Reduces sample failure and number of samples needed for convergence
- Uniformly sample from all groups in GROUP BY to converge at the same rate regardless of skewness <-> Sample by group instead of full records (obscure group members harder to find)
- Considers the relative size of the group, the number of groups, and weighting the group record accordingly



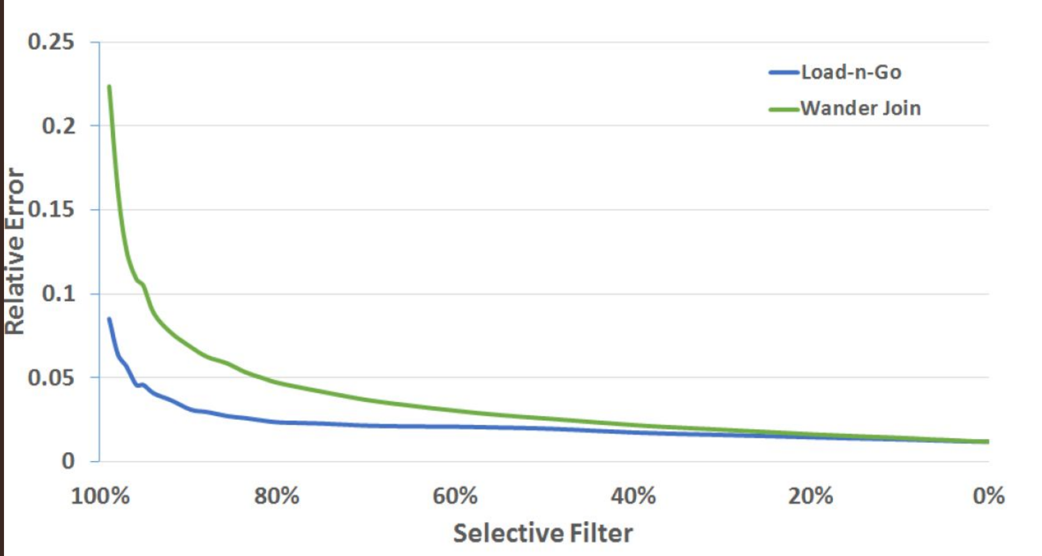
Initialized with uniform weight



Recursively pruned (WJ samples with replacement)

Performance

- Wander Join considers it a failed walk as the sample doesn't meet the criteria of the filtering clause
- Load-n-Go required 25% to 50% fewer samples in skewed groups

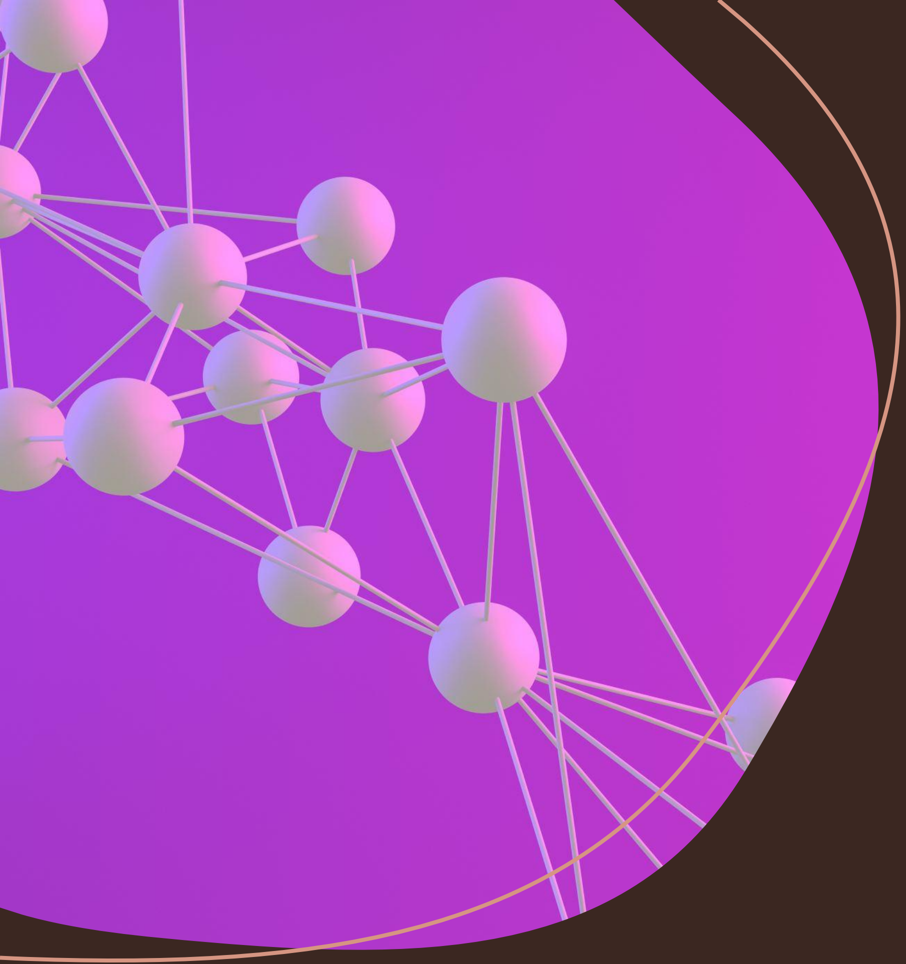


Selective Wander Join: Fast Progressive Visualizations for Data Joins

2019

Authors:
Marianne Procopio
Carlos Scheidegger
Eugene Wu
Remco Chang

Builds on Previous Work



Progressive Visual Analytics

Quintessential Human in the loop approach, users interact with automation through direction of information and entity of interest

Users can prioritize data attributes / groups while being provided an updated view to reduce [perceived] latency

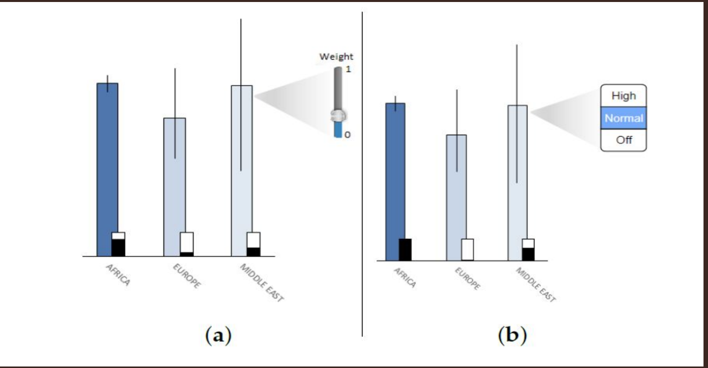
Selective Wander Join

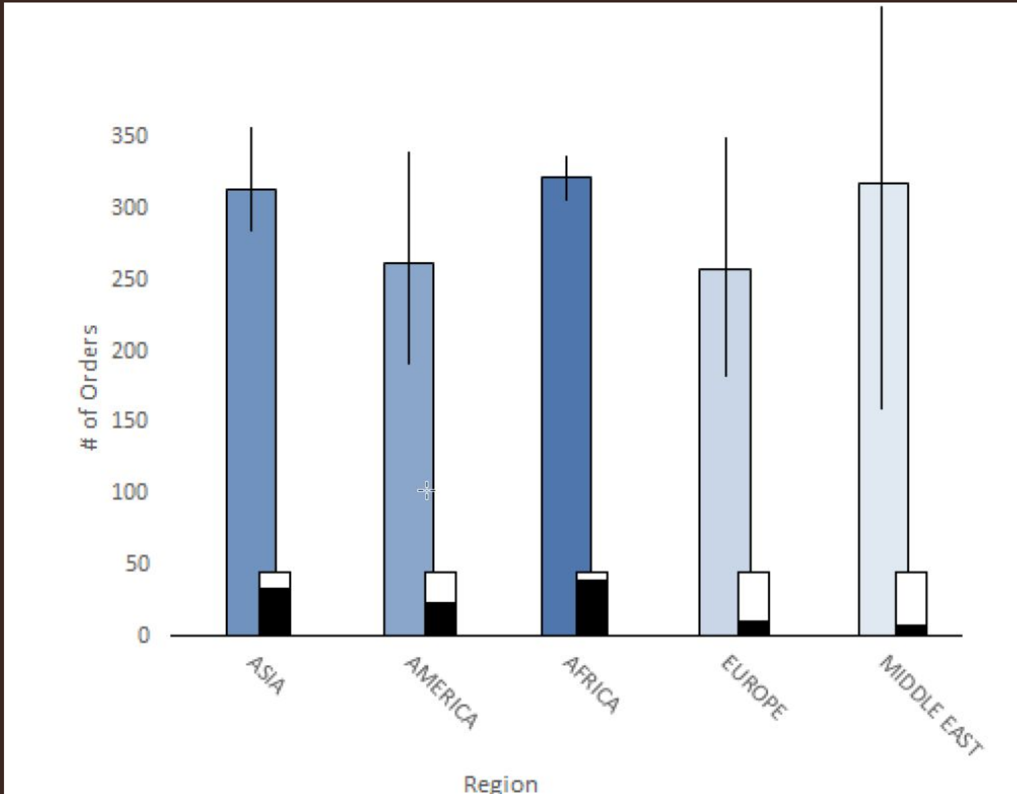
- Allow users to prioritize or cancel query executions leading to transparency of aggregation process and prioritizing interests
- Users benefit from seeing intermediate results along with incremental improvements in accuracy
- Users are empowered to actively explore data instead of waiting
- Users can allocate computation resources to JOINs to align relevance or interest

Novelty of Selective Wander Join

- Wander JOIN is a "black box" agnostic to front-end viz and doesn't consider user interaction
- Wander Join is iterative, leverage this to show users results and allow drilling down
- Load-n-Go treated everything independently and reassign weights.
  - For example, GROUP BY:
    - Wander Join -> view table in totality (uniform)
    - Load-n-Go -> non-uniform, ranking by groups
    - Selective Wander Join -> User can prioritize groups

Selective Wander Join UI



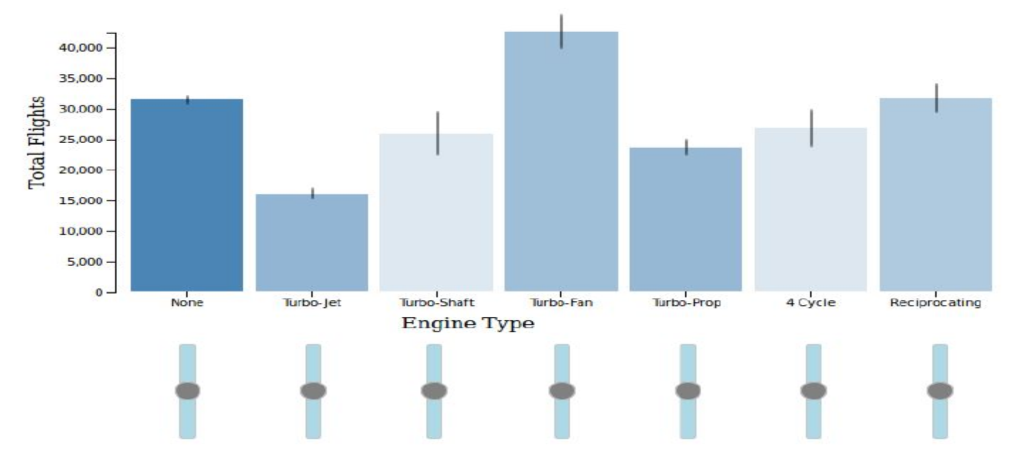


Example

```
SELECT count(*)
FROM plane_data, flights
WHERE plane_data.tail_num = flights.tail_num
GROUP BY plane_data.engine_type
```

- Wander Join 997,000 records to reach 0.05 relative error
- SWJ needed 11,000 records to reach 0.01, 0.04 and 0.1 respectively

- Turbo-fan: 68%
- Turbo-jet: 27%
- Turbo-Prop: 4%
- Remaining 4: <1%



Thank you

M4: A Visualization-Oriented Time Series Data Aggregation

Industry Practitioner

Industry Practitioner : Trading Firm

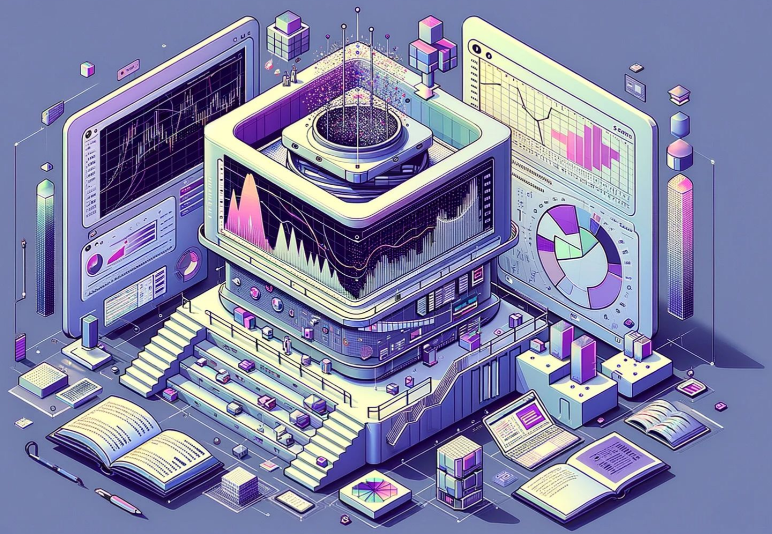
- High Volume Time Series Data is important for our firm because:
  - Market Analysis and Decision Making: Allows traders to analyze market movements in real-time, enabling quick decision-making based on the latest information.
  - Trading: Trading algorithms rely on high-frequency data to execute trades automatically based on predefined criteria. High-volume time series data is essential for the proper functioning of algorithmic trading strategies.
  - Risk Management: High-volume time series data aids in assessing market volatility, allowing trading firms to adjust risk management strategies accordingly.
  - News: Rapid analysis of high-volume time series data allows trading firms to quickly respond to market-moving news and events, minimizing the impact on their portfolios.

Chart representing stock value fluctuations

Use of M4 in Trading Firms

- As RDBMS systems cannot be only be used for visualizations for high volume time series data, we aim to use M4 in conjunction with it.
- Because M4 claims to be error free, the visualizations created by it will be helpful for our data analysts to see trends in stock markets.
- It can also be used to set alerts for individual stocks when a certain threshold is reached due to its accuracy.
- M4 also performs data reduction using aggregation which is highly used in trading firms to gauge the approximate values or total sum of investments by clients which would be very efficient in terms of time.

Thank you



M4: A Visualization-Oriented Time Series Data Aggregation

Demo by Tony Kim

Query Rewriter

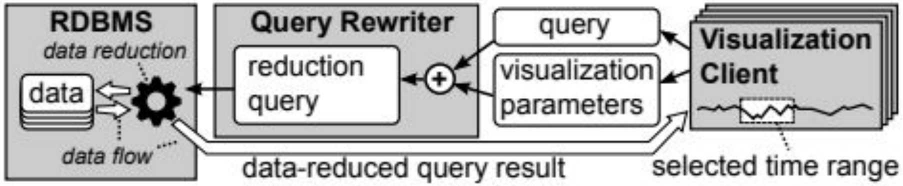
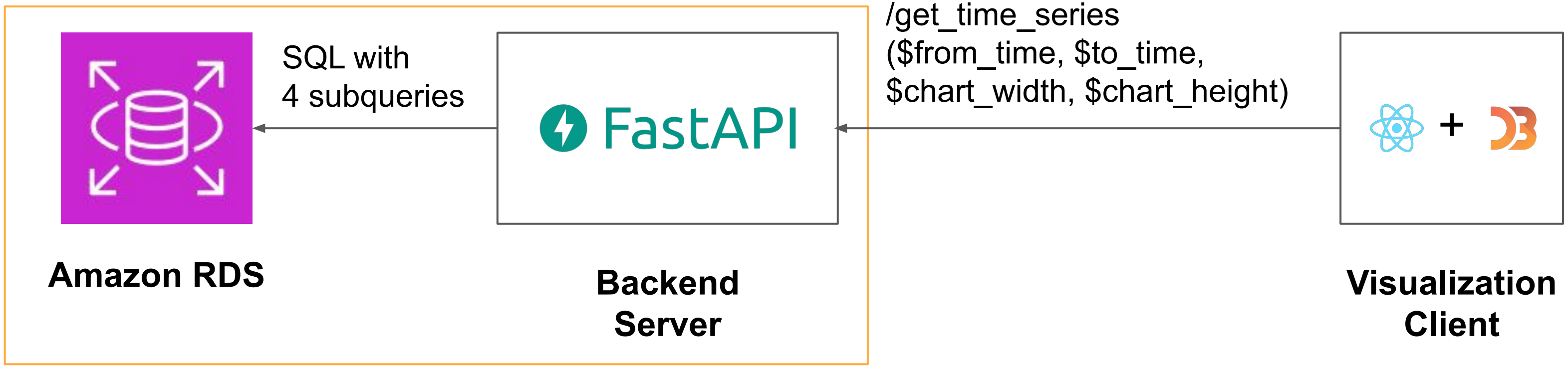


Figure 2: Visualization system with query rewriter.

Paper's Diagram

Implementation



Dataset

- Trades of SPDR S&P 500 ETF trust for the past 1 week (3.06 million tuples; 234 MB) accessed via Wharton Research Data Services.

Link to Visualization Client

http://3.144.70.123:4000/

M4 Limitations

- Only work for line charts
- Limited support for interaction: M4 needs to execute a new query for each user action on the visualization (e.g., panning and zooming)
- => How to fix it?

Last year's researcher presentation

Suppose we have precomputed a power-of two hierarchy of aggregates.

- For example, say a pixel column corresponds to time interval 0-98(s).
- We have two precomputed aggregates on 0-64 and 64-96 that covers most of this time interval.
- It remains to compute the aggregates online for the data in 96-98 only, and we can use only sampling to further speed up.

Last year's researcher presentation

- When the user zooms in (say gradually), the previous and new pixel columns still have much overlaps.
- If stateful computation is allowed, we can store the aggregates on previous pixel columns, and use AQP++ to efficiently compute the difference.

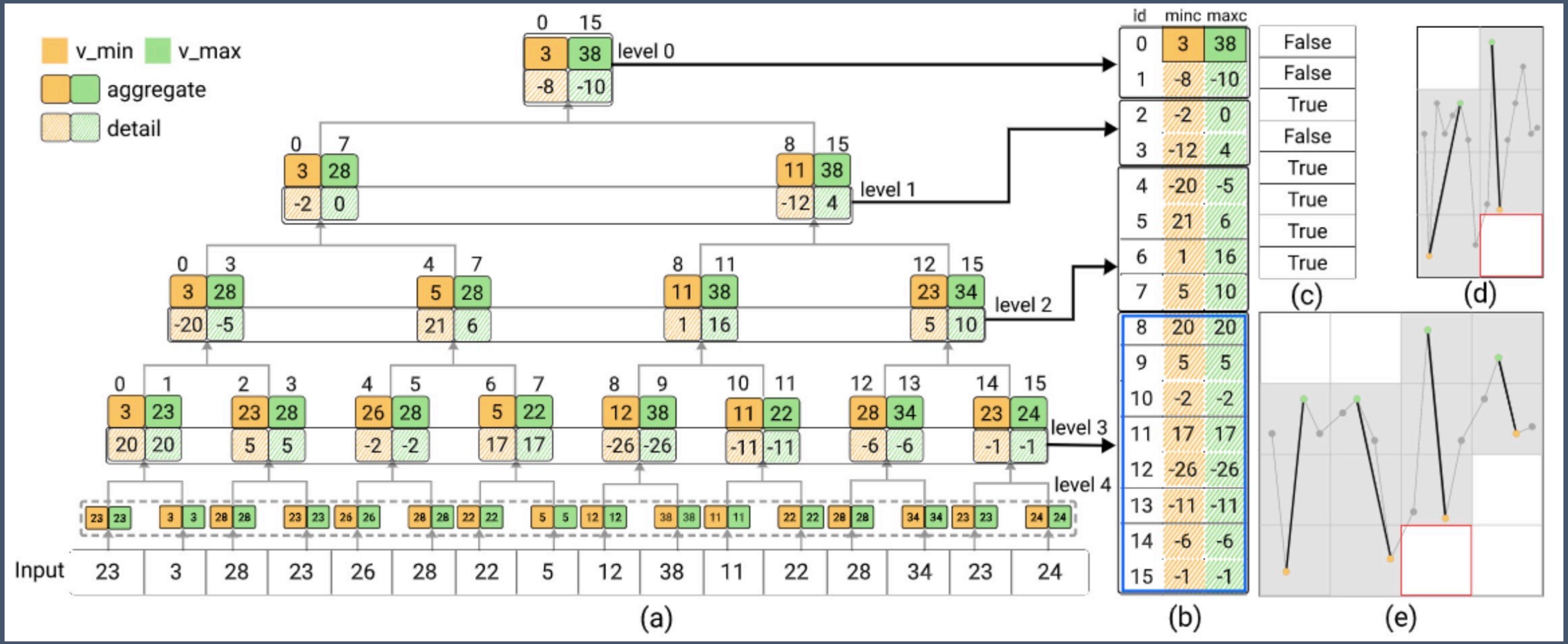
M4 + Interactivity

OM3: An Ordered Multi-level Min-Max Representation for Interactive Progressive Visualization of Time Series [SIGMOD'23]

- log(n) levels and maintains (min, max) value at each time interval
- ~75% of dataset size
- Extended to support time series of arbitrary length
- Incremental processing and progressive visualization

M4 + Interactivity

OM3: An Ordered Multi-level Min-Max Representation for Interactive Progressive Visualization of Time Series [SIGMOD'23]



Discussion: AQP and visualization

How are these ideas similar/different between AQP and viz?

- Definition of error/accuracy
- Presentation of error
- Language (viz=SQL?)
- Use of precomputation techniques
- Use of sampling techniques

Next class

MacroBase: Prioritizing Attention in Fast Data

- Authors: Tony, Amey
- Archaeologists: Xiaoyue
- Practitioner: Siddhant