Hashed Cubes: Simple, Low Memory, Real-Time Visual Exploration of Big Data

Cícero A.L Pahins, Sean A. Stephens, Carlos Scheidegger, João L. D. Comba, *IEEE Transactions on Visualization and Computer Graphics* 23.1 (2017): 671-680.

What Data?

- HashedCube used to store datasets with spatial, categorical, and temporal attributes

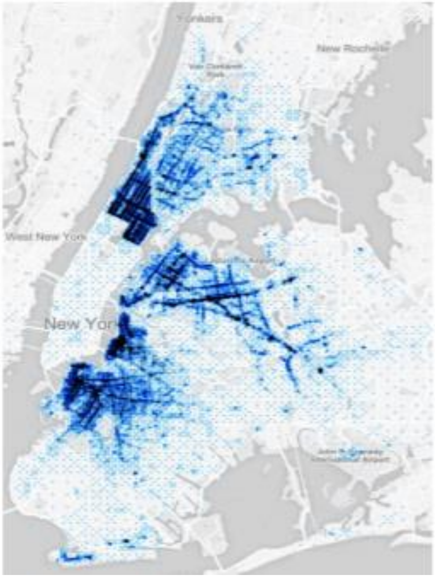
4.5 million BrightKite check-ins from April 2008 to Oct 2010

- Spatial dimension:	Geographical location
- Categorical dimension:	Day and hour
- Temporal dimension:	Time

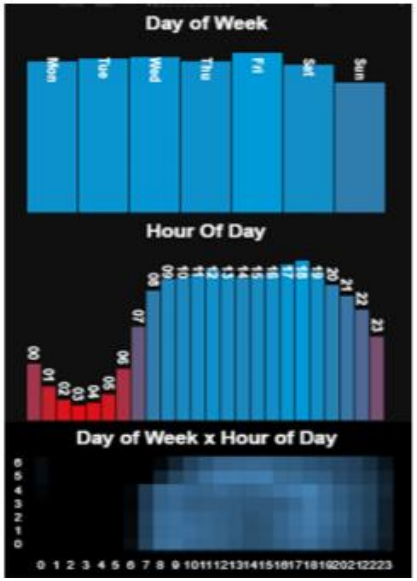
What Tasks?

- Aggregate items in the dataset to answer questions such as:
  - How many people checked in on Brightkite in Europe on a Friday?
  - What does the trend in the number of global Brightkite check-ins look like in a year?

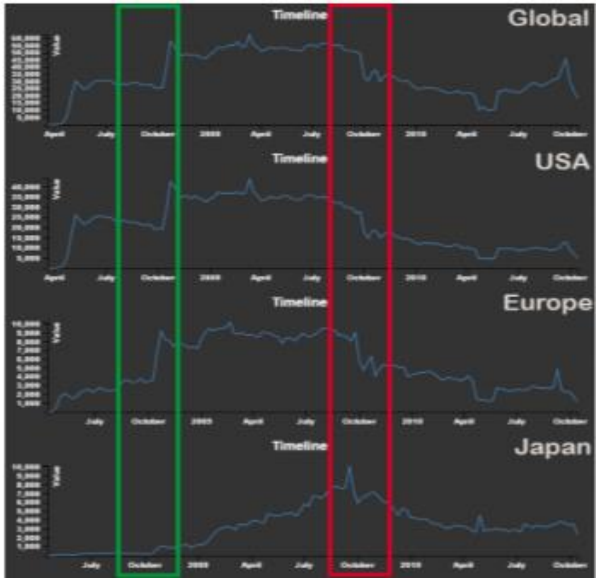
What Visual Encodings?



Heat map



Histogram



Line graph

Video: https://vimeo.com/161051233

[Fig 1. Pahins, Cícero A. L, et al. "Hashedcubes: Simple, low memory, real-time visual exploration of big data." *IEEE Transactions on Visualization and Computer Graphics* 23.1 (2017): 671-680.]

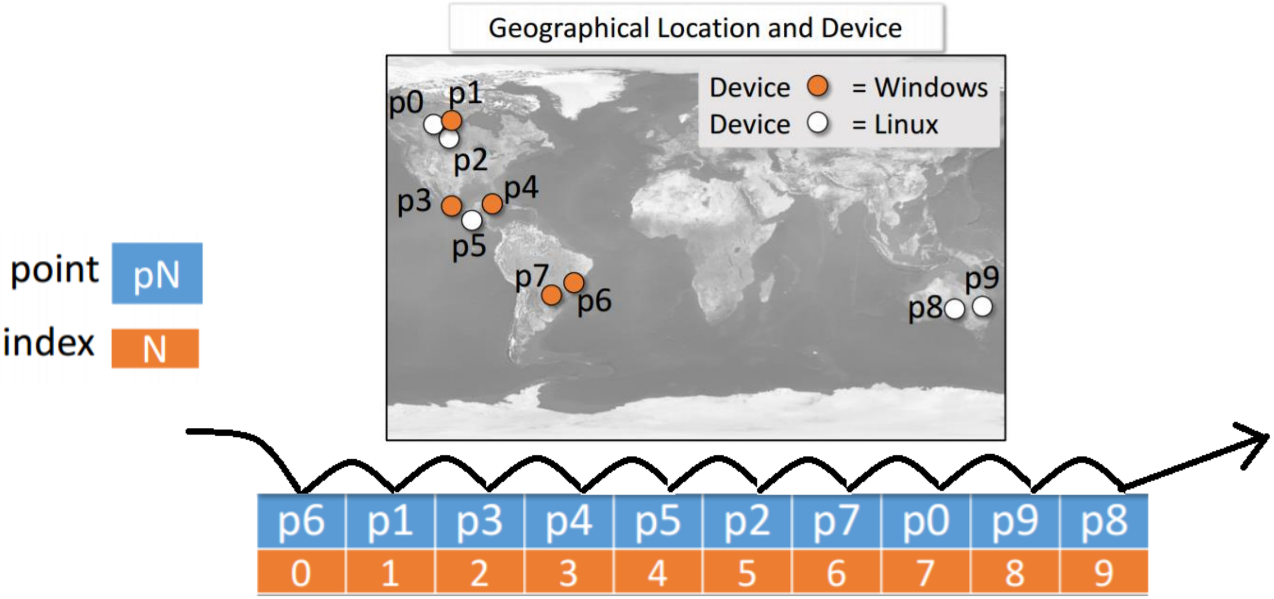
Storing Data in Memory

- Laying out datasets thoughtfully in memory means faster query times with large visualizations
- Think of data in computer memory like books in a library – the neater the better!

VS

The Array: a Naïve Approach

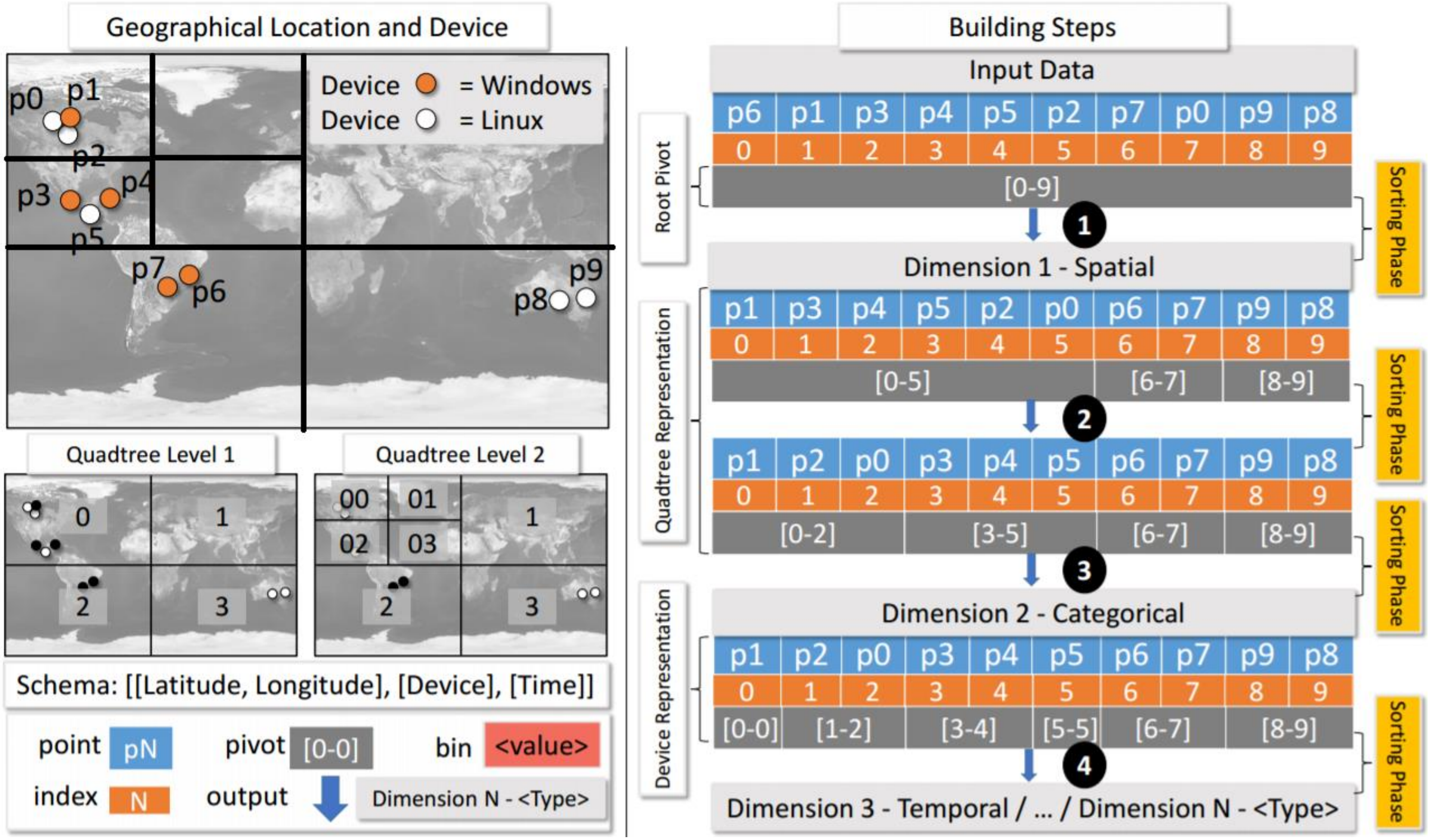
- Let’s say we wish to store the dataset below in an array
  - Traversing each index in the array is tedious!



p6	p1	p3	p4	p5	p2	p7	p0	p9	p8
0	1	2	3	4	5	6	7	8	9

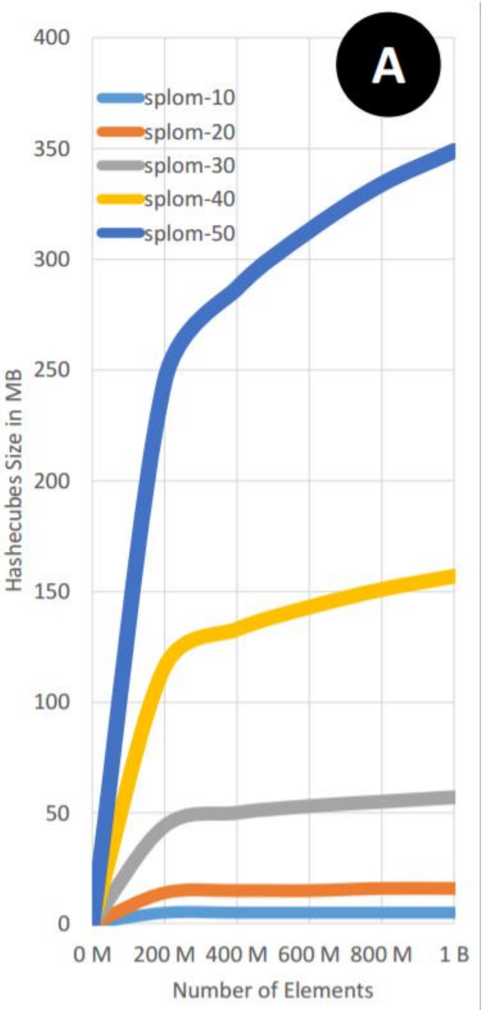
[Fig 2. Pahins, Cícero A. L, et al. "Hashedcubes: Simple, low memory, real-time visual exploration of big data." *IEEE Transactions on Visualization and Computer Graphics* 23.1 (2017): 671-680.]

Building Hashedcubes



[Fig 2. Pahins, Cícero A. L, et al. "Hashedcubes: Simple, low memory, real-time visual exploration of big data." *IEEE Transactions on Visualization and Computer Graphics* 23.1 (2017): 671-680.]

Hashedcube Memory Usage

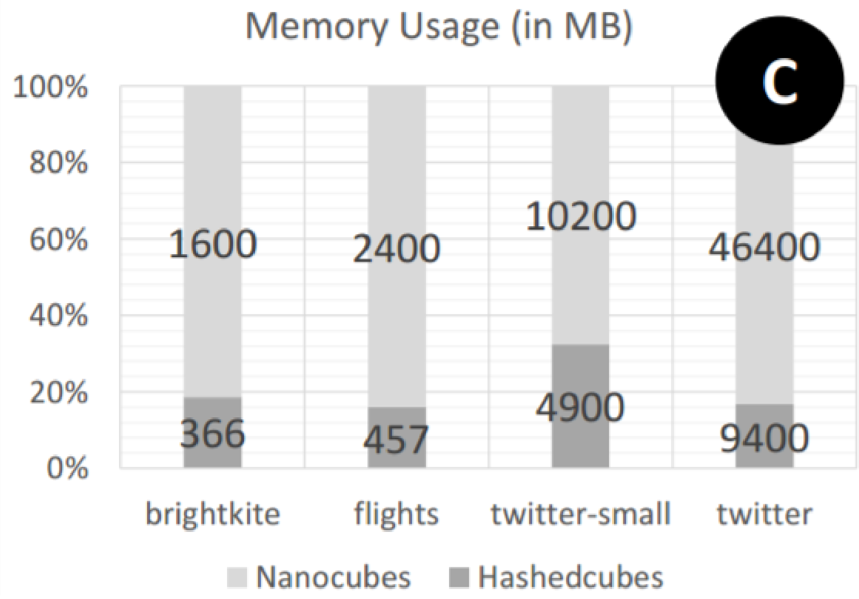


- Memory usage of hashedcubes directly proportional to number of pivots
  - Key saturation reduces memory footprint

[Fig 7a. Pahins, Cícero A. L, et al. "Hashedcubes: Simple, low memory, real-time visual exploration of big data." *IEEE Transactions on Visualization and Computer Graphics* 23.1 (2017): 671-680.]

Hashedcube Memory Usage

- Hashedcubes required less memory than Nanocubes
  - Up to 5.2 times less in the best case



[Fig 7c. Pahins, Cícero A. L, et al. "Hashedcubes: Simple, low memory, real-time visual exploration of big data." *IEEE Transactions on Visualization and Computer Graphics* 23.1 (2017): 671-680.]

Hashedcube Query Times

- Only one in 50 queries took more than 40 ms
  - Most time consuming queries required large number of aggregates of many small pivots

- Hashedcube query times **worse** than state-of-the-art
  - Nanocube worst case value around 12 ms
  - imMens had 20 ms query time on average

Critique

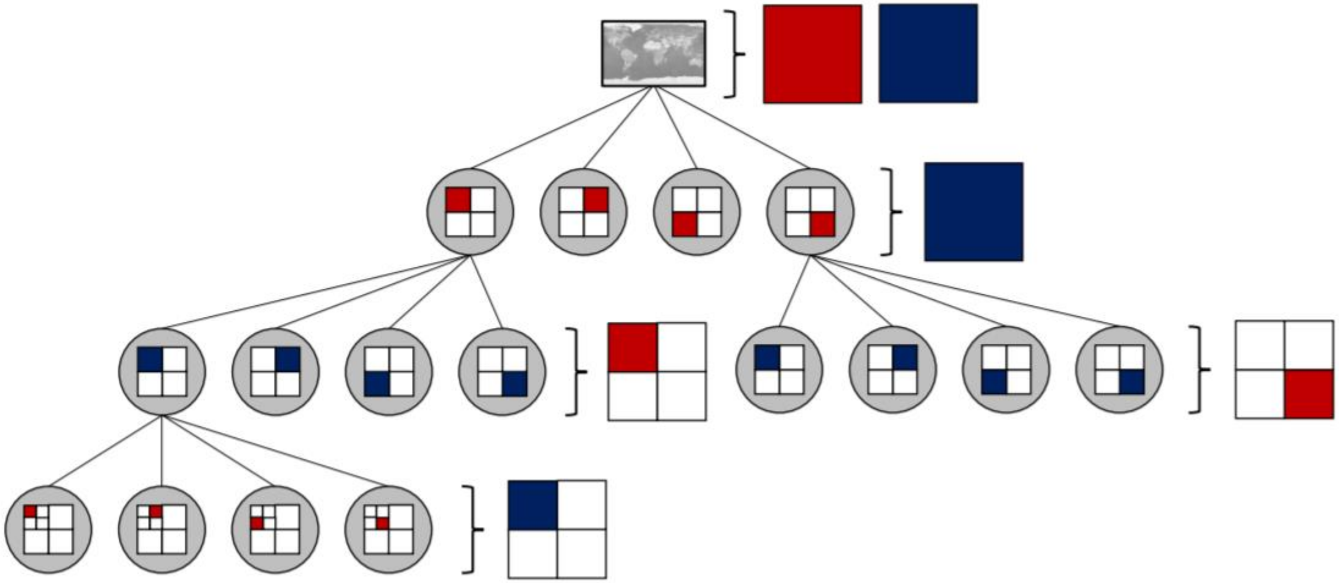
- Strengths:
  - Code available online!
  - Most query times are tolerable
  - Occupies less computer memory than the state-of-the-art

- Weaknesses:
  - Query times longer than the state-of-the-art
  - Need to tune the algorithm for generating Hashedcubes (Ex: Pick dimension sort order)

Thank you!

Multiple Spatial Dimensions

- Hashcubes support multiple spatial dimensions by using interweaved quadtrees



[Fig 4. Pahins, Cícero A. L, et al. "Hashedcubes: Simple, low memory, real-time visual exploration of big data." *IEEE Transactions on Visualization and Computer Graphics* 23.1 (2017): 671-680.]