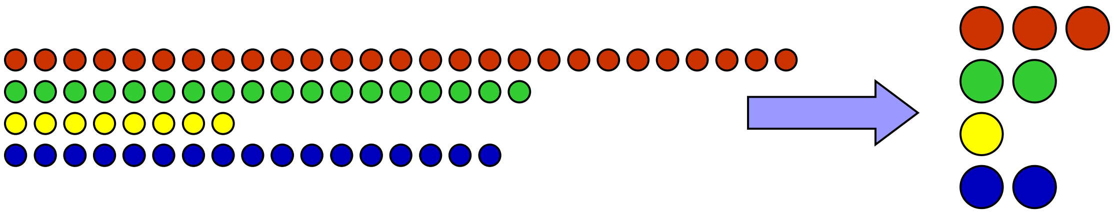
Applications of Sketching: Pathways to Impact



Graham Cormode

University of Warwick & Meta

G.Cormode@Warwick.ac.uk

Outline

- An overview and thumbnail history of sketching
- Research impacts of sketching
- Practical impacts of sketching
- Strategies and tactics for impact
- ~~Formal results, algorithms and proofs~~

Sketches

- Data summary algorithms ("sketches"):
  - Compact data structures that capture certain properties of data
  - May be updated incrementally (streaming) or merged together
- They answer queries with approximation guarantees
  - How many distinct items seen (count distinct, F_0 sketches)
  - Track frequency distributions (heavy hitters, quantiles)
  - Approximate Euclidean or other norms (dimensionality reduction)
  - Summarize more complex data types (graphs, matrices etc.)
- Sketch algorithms have been studied in CS for > 50 years
  - A brief history lesson now follows

Early History of Sketching

- A random sample gives a basic sketch (late 19th/early 20th C)
  - Reservoir sampling algorithms (1960s/70s: Fan et al., Waterman)
- Bloom filters for set summarization (1970)
- Morris counter for counting in O(log log n) bits (1977)
- Munro & Paterson median finding in few passes (1978)
- Flajolet & Martin distinct counting (1983)
- Johnson-Lindenstrauss lemma for dimensionality reduction (1984)

Initial results, with powerful, simple algorithms showing the first sublinear results for the constraints of the time

The Streaming Years

- AMS sketching for the frequency moments (1996)
- LSH (Indyk-Motwani, Broder) for approximate similarity (1998)
- MRL and GK algorithms for quantiles (1998, 2001)
- CCFC and CM sketches for frequency estimation (2002, 2003)
- Loglog and HLL sketches for count distinct (2003, 2007)
- Q-Digest for quantiles (2004)
- SpaceSaving for frequency estimation (2005)

Techniques motivated by ‘streaming’ computing models, with focus on the space complexity and time cost

From streaming to mergeable

- HLL++ from Google (2013-): optimizing accuracy and space for small distinct counts when running many counters in parallel
- Sparser Johnson-Lindenstrauss (2010): very sparse transformations preserve Euclidean norm
- Mergeable Summaries (2012): placing emphasis on merging summaries together, for quantiles and approximations
- KLL (2016): optimal approximate quantile sketch via sampling
- & Subspace embeddings (2010s), compressed sensing (2004-)
- … and too many more to list, right up to the present day

> Refinements and enhancements of previous techniques, targeting greater scalability (computation and memory cost) and/or theoretical optimality

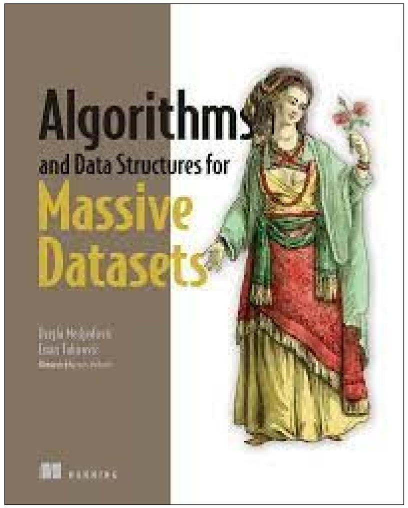
Sketches at PODS

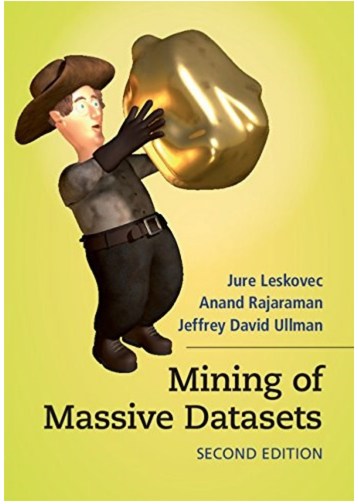
Many works on sketching have appeared at PODS, and several have been honoured with awards

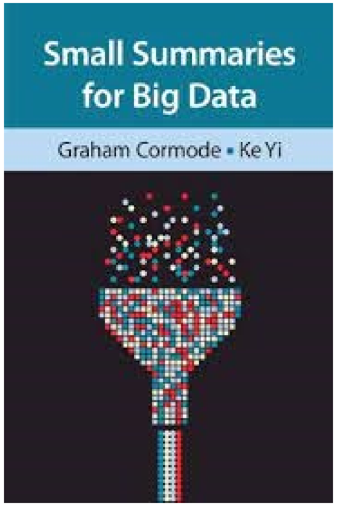
- An Optimal Algorithm for the Distinct Elements Problem, Kane, Nelson, Woodruff (2010 best paper)
- Tight bounds for Lp samplers, Jowhari, Saglam, Tardos, (2011, Test-of-time award in 2021)
- Mergeable summaries, Agarwal, C, Huang, Phillips, Wei, Yi, (2012, Test-of-time award in 2022)
- A Framework for Adversarially Robust Streaming Algorithms, Ben-Eliezer, Jayaram, Woodruff, Yogev (2020 best paper award)
- Relative Error Streaming Quantiles, C, Karnin, Liberty, Thaler, Vesely (2021 best paper award)
- Better DP Approximate Histograms and Heavy Hitters Using the Misra-Gries Sketch, Lebeda, Tetek (2023 distinguished paper)

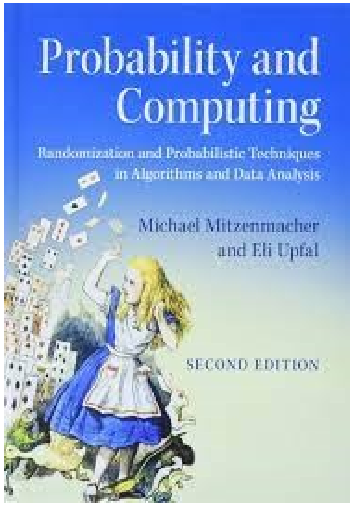
Further reading…

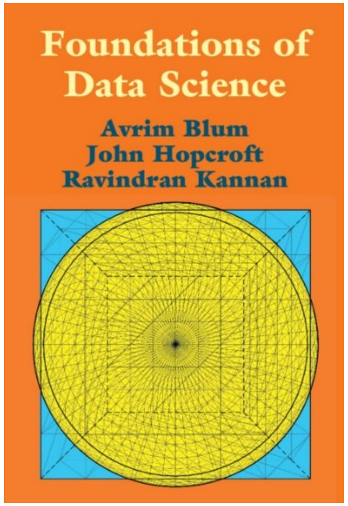
- Sketch algorithms are now presented in several textbooks











- But how have they been used in practice – and why?

Shifting Motivations for Sketching (1)

Memory constrained systems (1970s – 1980s)

- Morris counting, Bloom filters assumed memory was tiny
- Munro-Paterson: taking multiple passes over tapes
- Ratio of data size : storage capability shifted, so the need diminished

"Massive Data" (late 1990s – early 2000s)

- Network/ISP data motivated the streaming paradigm for data analysis
- Systems: Gigascope (AT&T), CMon (Sprint), Aurora/Borealis (academic)…

Multimedia search (2000s onwards)

- Locality Sensitivity hashing was a key component of fast search
- Still relevant: vector embeddings etc. still benefit from LSH

Shifting Motivations for Sketching (2)

Online advertising (2010s)

- Track distinct impressions across many clients
- Still some concern about approximate counting
- Need for sketching reduced: warehouses could count exactly!

“Big Data” (2010s onwards) – analytics

- Twitter counting embedded tweets (CM sketch)
- Quantile algorithms for tracking distributions (t-digest, KLL)
- Built into tools including Splunk, Presto, Salesforce, …
- Apache Data Sketches library supports Spark, Java
- Mostly invisible to outsiders

Shifting Motivations for Sketching (3)

Private data analysis (late 2010s-)

- Google's RAPPOR: Bloom filters + randomized response
- Apple DP deployment: Count/Count Min sketches + RR
- Federated analytics ≈ sketches + (differential) privacy

Communication efficient ML (mid 2010s-)

- Send a sketch of model updates e.g., SketchSGD
- Other tools: kernels, epsilon-approximations
- Still emerging – unclear if this will be mainstream

Lessons learned from applying sketches

You don't have to launch a startup based on your paper

- A company needs a business plan, not a scientific idea
- No business has emerged based solely on sketching (yet)

The language of CS is (open source) code

- Bad prototype code is better than no code
- A reference implementation shows the feasibility & basic ideas
- Code lives on github: hundreds of sketch implementations there

Put research ideas into (undergrad) teaching

- Find ways to incorporate research ideas into core topics
- E.g., sketches fit well into algorithms or database classes
- Students forced to study something may eventually deploy it!

Lessons learned from applying sketches

Write accessible notes where people read them	• Writing non-academic texts to reach working coders • Medium/substack/arxiv may be better than articles/books
Give talks and present tutorials online/meetups	• Need to reach beyond academia to software engineers • Make material freely available (YouTube, social media)
Work directly with companies	• Ideally, fully embedding yourself with a company • “Real world” problems are both simpler and more complex than research problems • Helps identify where research effort is really needed

Closing remarks

- The biggest contributor towards impact can be time
  - With encouragement, good ideas will find applications eventually
  - **Corollary**: many ideas sink without trace (too early or too late)

- Sketches are a good example of theory to applications
  - Many sketching ideas had compelling real motivations
  - This led to some deep theory and clever algorithms
  - The ultimate applications were not the original motivations!
  - Motivations and applications can change a lot over time