The DESQ Framework for Declarative and Scalable Frequent Sequence Mining

Kaustubh Beedkar[1] Rainer Gemulla [2]
Alexander Renz-Wieland [1]

[1]Technische Universität Berlin

[2]Universität Mannheim

INFORMATIK '19, Kassel
September 24th, 2019

Presentation of work originally published in IEEE 16th Intl. Conf. on Data Mining, IEEE 35th Intl. Conf. on Data Engineering, and 2019 ACM Trans. on Database Syst.

Outline

1. Frequent Sequence Mining
2. Declarativity
3. Scalability
4. Summary

Outline

1. Frequent Sequence Mining
2. Declarativity
3. Scalability
4. Summary

Before and after

Anni wants to watch a movie. Anni loves LOTR1. But she does not want to see it. She had seen LOTR2 last week!

Movie streaming site

Let's look at some data

- Data from Netflix' online movie-streaming platform
  - 500k users, 18k movies, 100M ratings with timestamps
- 125k users rated both LOTR1 and LOTR2
- In which order?

105k users 20k users

- Order matters!
  - How to discover patterns in sequential data?

Frequent Sequence Mining

- Frequent sequence mining is a fundamental task in data mining
  - Data modeled as collection of **sequences of items or events**
  - Often items are arranged in a **hierarchy**
  - We seek frequent **sequential patterns**
- E.g., market-basket data
  - Sequence = purchases of a customer over time
  - Item = product (or set of products) + product hierarchy
  - Example pattern: *DSLR Camera* $\rightarrow$ *Tripod* $\rightarrow$ *Flash*
- E.g., natural-language text
  - Sequence = sentence or document
  - Item = word + syntactic/semantic hierarchy
  - Example pattern: *person* was born in *location*
- E.g., amino acid sequences
  - Sequence = protein
  - Item = amino acid
  - Example pattern: S L R

What constitutes a good pattern?

- Extensively studied
  - Interesting patterns should be new, surprising, understandable, actionable
  - No random patterns, common knowledge, redundancy
  - Details application-specific
- Many different variants, many algorithms
  - Constraints: length, positional/temporal, hierarchy, regex, ...
  - Scoring: **frequency**, utility, information gain, significance, ...
  - Pattern sets: all, top-k, maximality, closedness, MDL, ...
- Our research focuses on **unifying frequent sequence mining**
  - Study general properties instead of special cases
  - Avoid need for customized mining algorithms

DESQ

- DESQ = framework for declarative and scalable frequent sequence mining [TODS19, ICDM16, ICDE19]
  - Open source

- Key design goals are
  1. **Usefulness**
     - Can be tailored to application
     - Flexible constraints
  2. **Usability**
     - Describe pattern mining task in an intuitive, declarative way
     - Hide technical and implementation details
  3. **Efficiency**
     - Fast
     - Scalable
     - Competitive to specialized miners

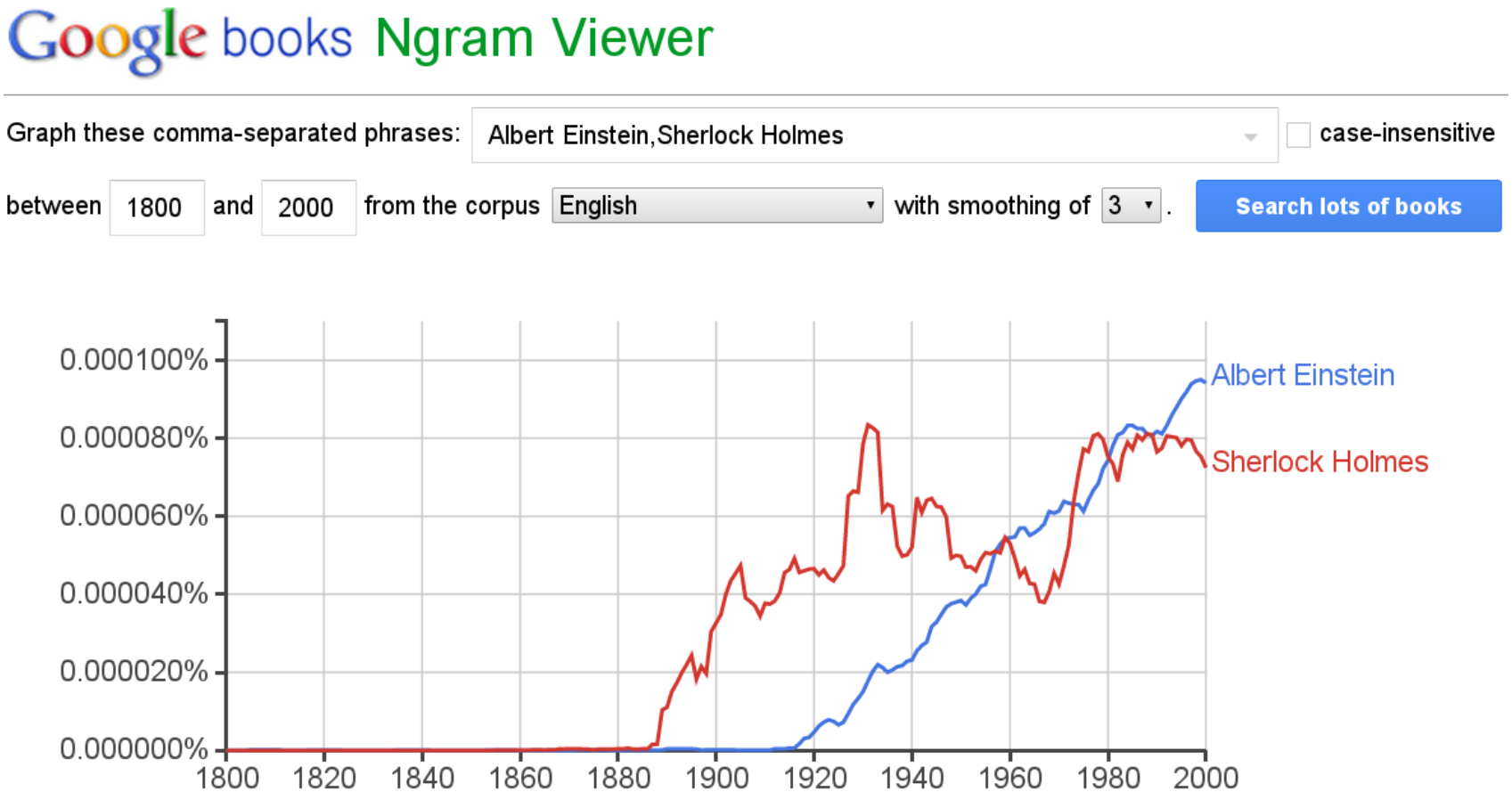
Outline

1. Frequent Sequence Mining
2. Declarativity
3. Scalability
4. Summary

Special case: *n*-gram mining

An ***n*-gram** is a sequence of n consecutive words

- Extensively used in text mining and natural-language processing
- Web-scale *n*-gram models published by Google and Microsoft



Special case: *n*-gram mining

An ***n*-gram** is a sequence of n consecutive words

- Extensively used in text mining and natural-language processing
- Web-scale *n*-gram models published by Google and Microsoft

Going declarative

- If we simply mined all frequent n-grams, we may
  1. Produce many uninteresting patterns (low frequency threshold)
  2. Miss out on interesting patterns (high frequency threshold)
- DESQ allows data analysts to focus on what they consider relevant
  - Supports all traditional constraints (length, gap, hierarchy, ...)
  - Supports customized constraints that go beyond traditional constraints
- Based on a declarative **pattern expression language**
  - Describe relevant patterns, let DESQ take care of mining them
  - Syntax like regular expression
  - Adds capture groups and hierarchies

Some examples for text mining

1. **Noun modified by adjective or noun**
   Ex: *big country (110), green tea (337), research scientist (473)*
   PE: `([ADJ|NOUN] NOUN)`

2. **Relational phrase between entities**
   Ex: *lives in (847), is being advised by (15), has coached (10)*
   PE: `ENTITY (VERB+ NOUN+? PREP?) ENTITY`

3. **Typed relational phrases**
   Ex: *ORG headed by ENTITY (275), PERS born in LOC (481)*
   PE: `(ENTITY↑ VERB+ NOUN+? PREP? ENTITY↑)`

4. **Google *n*-gram viewer data**
   Ex: *a good day, a ADJ day, DET ADJ NOUN, have a good day*
   PE: `(.↑) (.↑)? (.↑)? | (.....?)`

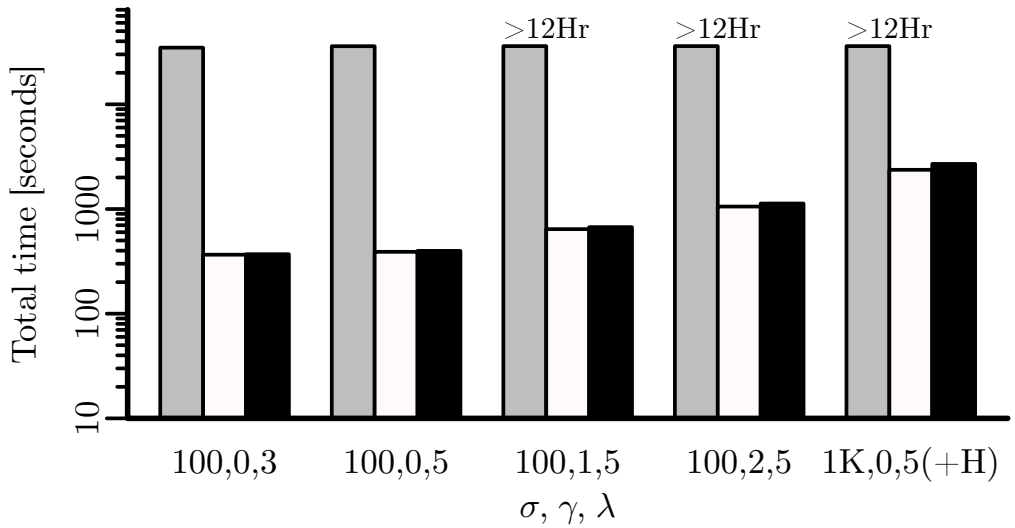
Pattern mining

- Under the hood, DESQ translates pattern expressions to finite state transducers (FST)
  - FST outputs all patterns that occur in a given input sequence

- Multiple sequential mining algorithms
  - Naive approach ("WordCount")
  - DesqCount ("WordCount" with frequency pruning)
  - DesqDfs (depth-first search)

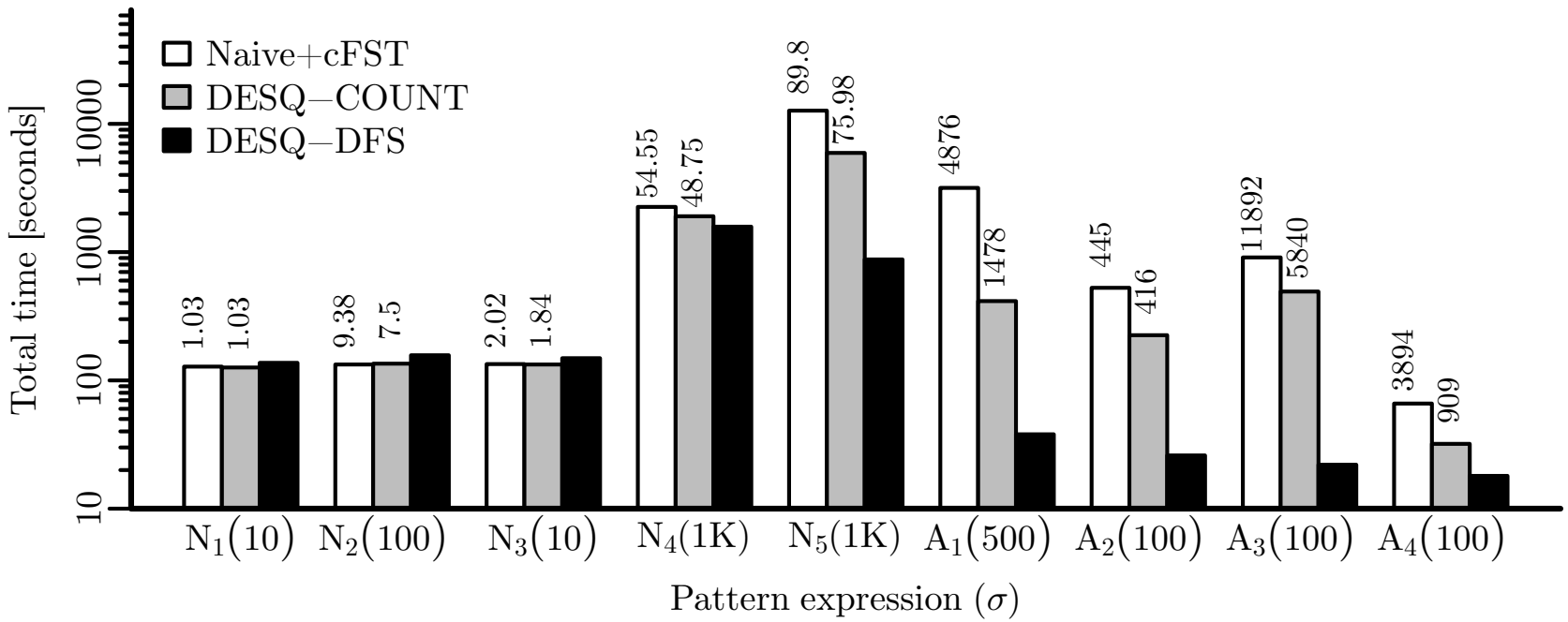
Performance comparison (traditional constraints)

Left: cSPADE, center: prefix-growth, right: DesqDfs



DESQ is competitive to state-of-the-art miners for traditional constraints.

Performance comparison (new constraints)



DesqDfs is method of choice and can be orders of magnitude faster than Naive or DesqCount.

Outline

1. Frequent Sequence Mining
2. Declarativity
3. Scalability
4. Summary

Distributed mining

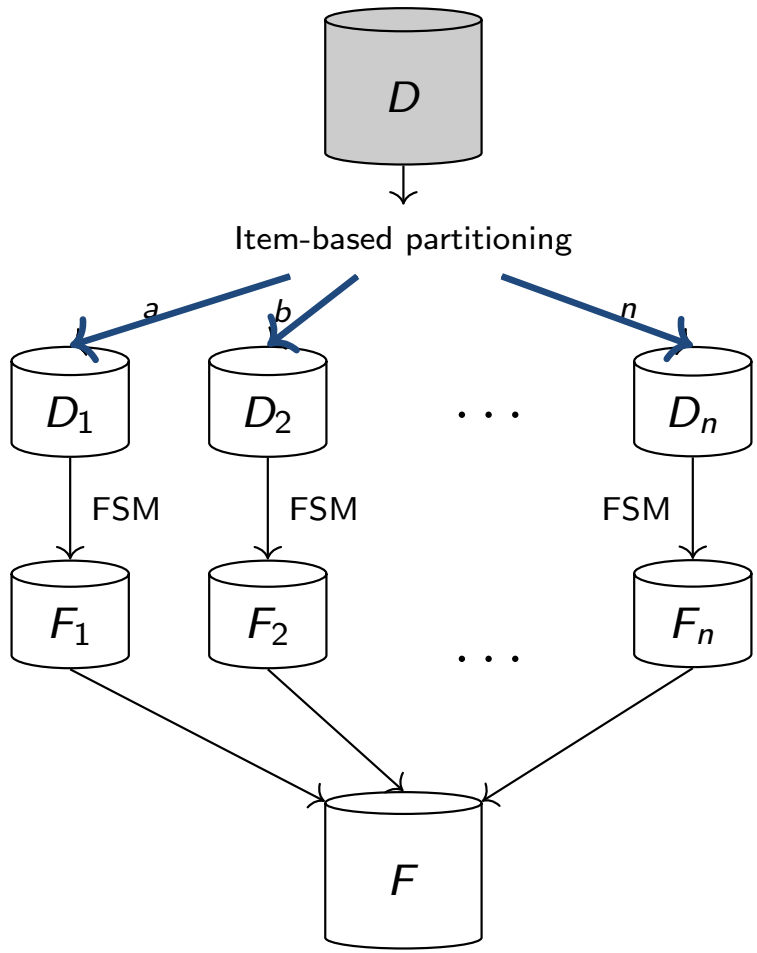
- Based on bulk synchronous parallel model

Key idea

- Partition data into smaller overlapping partitions using **item-based partitioning**
  - One partition for each frequent item
- Mine each partition locally
- Combine results

Key question

- What to communicate to partitions?
  - Inputs
  - Candidates



Communicate inputs

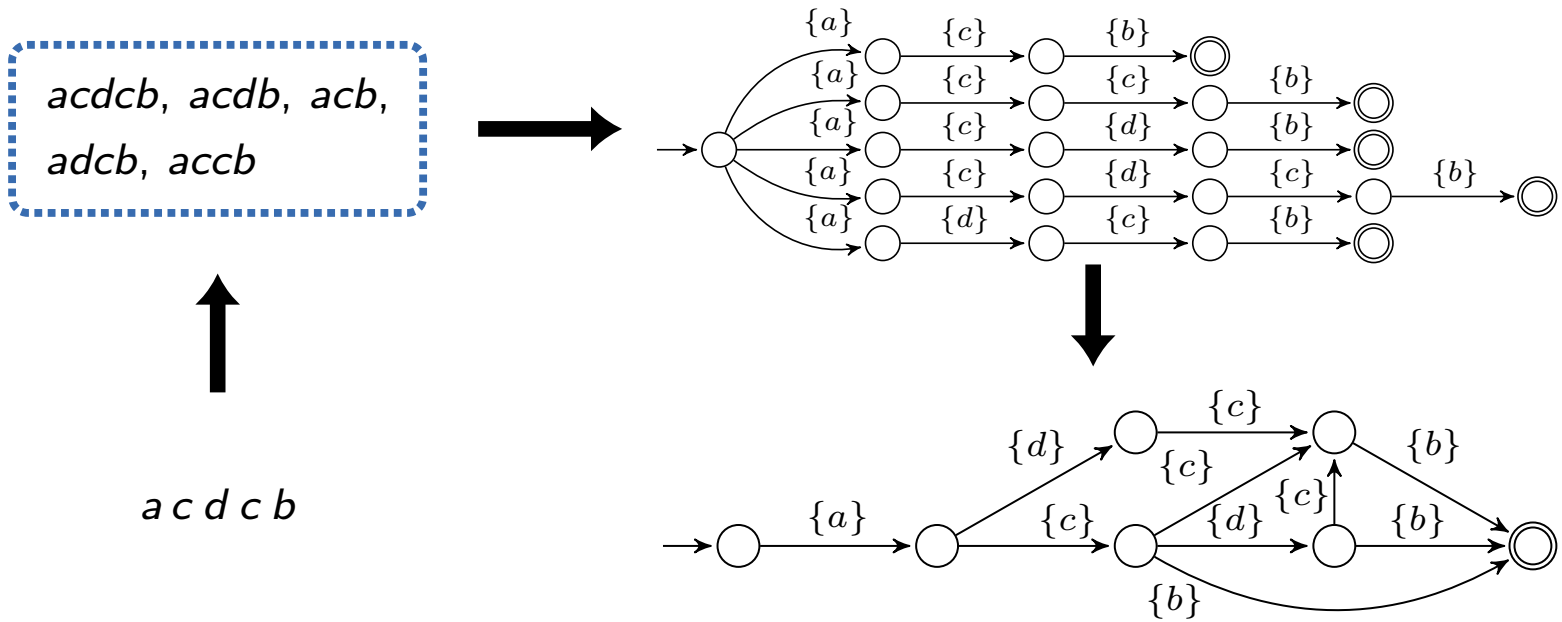
- Naïve approach: send each input sequence to all partitions for which it is "relevant"

- More efficient: send only relevant parts of input sequence
  - Example: only fantasy movies relevant for mining task

  - Can reduce communication up to 100x

Communicate candidates

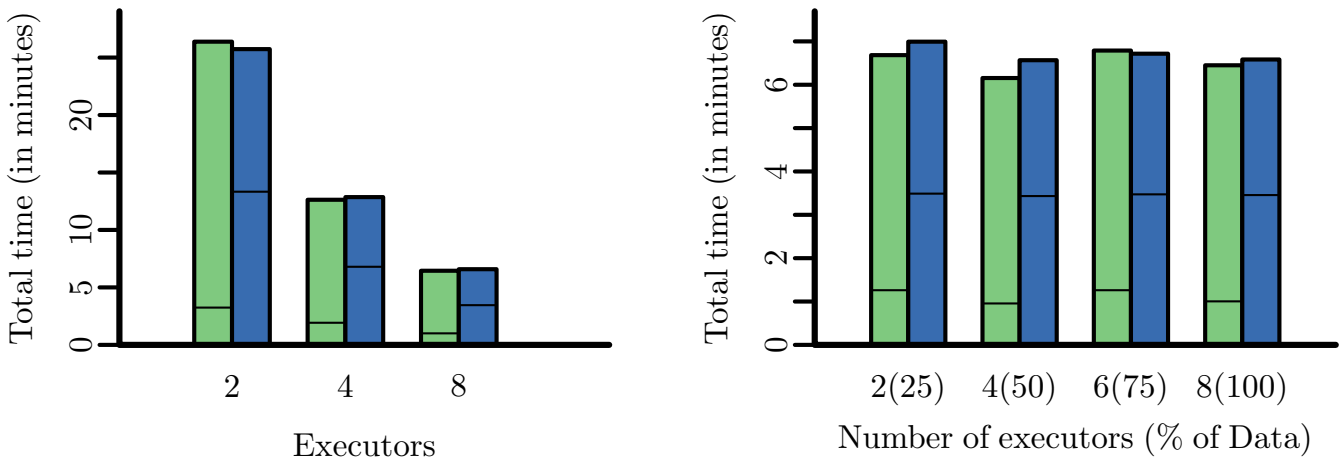
- Naïve approach: send each candidate subsequence to its corresponding partition
- More efficient: compress candidates
  - Shared structure
  - Non-deterministic finite automata (NFA)



  - Can reduce communication by up to 100x

Performance comparison

- Both approaches scale nearly linearly with number of input sequences. green: send inputs, blue: send candidates



(a) Strong scalability (b) Weak scalability

- Up to 50x faster than naïve approaches
- Sending candidates is up to 5x faster for selective constraints
- 1-4x generalization overhead over specialized approaches

Outline

1. Frequent Sequence Mining
2. Declarativity
3. Scalability
4. Summary

Summary

DESQ: framework for declarative and scalable frequent sequence mining

- Find patterns in sequential data
- Declarative language to specify interest
- Item-based partitioning to scale to large datasets
- Open source: https://github.com/rgemulla/desq

[ICDM16] Beedkar, K.; Gemulla, R.: DESQ: Frequent Sequence Mining with Subsequence Constraints. In: ICDM, 2016.

[TODS19] Beedkar, K.; Gemulla, R.; Martens, W.: A Unied Framework for Frequent Sequence Mining with Subsequence Constraints. ACM Trans. Database Syst., 2019.

[ICDE19] Renz-Wieland, A.; Bertsch, M.; Gemulla, R.: Scalable Frequent Sequence Mining With Flexible Subsequence Constraints. In: ICDE, 2019.

Thank you!