KDD 2012 BEIJING
THE 18TH ACM SIGKDD CONFERENCE ON KNOWLEDGE DISCOVERY AND DATA MINING
Beijing, China August 12-16, 2012

The Long and the Short of It

Summarizing Event Sequences with Serial Episodes

Nikolaj Tatti and **Jilles Vreeken**
{nikolaj.tatti, jilles.vreeken}@ua.ac.be, University of Antwerp, Belgium

The Problem

An ideal outcome of pattern mining are small sets of informative patterns, containing no redundancy or noise, identifying the key structure of the data.

Standard frequent pattern mining does not achieve this goal, as it is does not penalize redundancy

We pursue the ideal for sequential data and serial episodes, taking a *pattern set* mining approach: instead of ranking patterns individually, we aim for the best *set of patterns*

The Big Question

How do we identify the optimal set of patterns?

- we need a score rewarding models that identify real structure, while punishing redundancy
- no off-the shelf score fits our setting, but we can make our intuitions concrete by MDL

The Minimum Description Length (MDL) principle
given a dataset D and a collection of models $\mathcal{H}$ the best model H is the model that minimizes

$$L(H) + L(D|H)$$

MDL for Event Sequences

By MDL we define the optimal set of patterns as the set that describes the data most succinct

To use MDL we need to define a *lossless* encoding for our models, and for data given a model.

As models we use *code tables*—dictionaries of patterns and associated codes

Encoding Event Sequences

We encode the data D using two *code streams:* the pattern-stream C_p and the gap-stream C_g

Data D: a b d c a d b a a b c

Encoding 1: using only singletons

C_p a b d c a d b a a b c

CT_1: a a, b b, c c, d d

Encoding 2: using patterns

C_p p d a q b p

C_g □ ■ □ ■ □ □ □

gap, gap

alignment: a b d c a d b a a b c; p, q, p

CT_2: a a, b b, c c, d d, abc p ■ □, da q ■ □ (gaps, non-gaps)

How to Cover your String?

Given a pattern set, there are many ways to cover a sequence database. We are after the **optimum.**

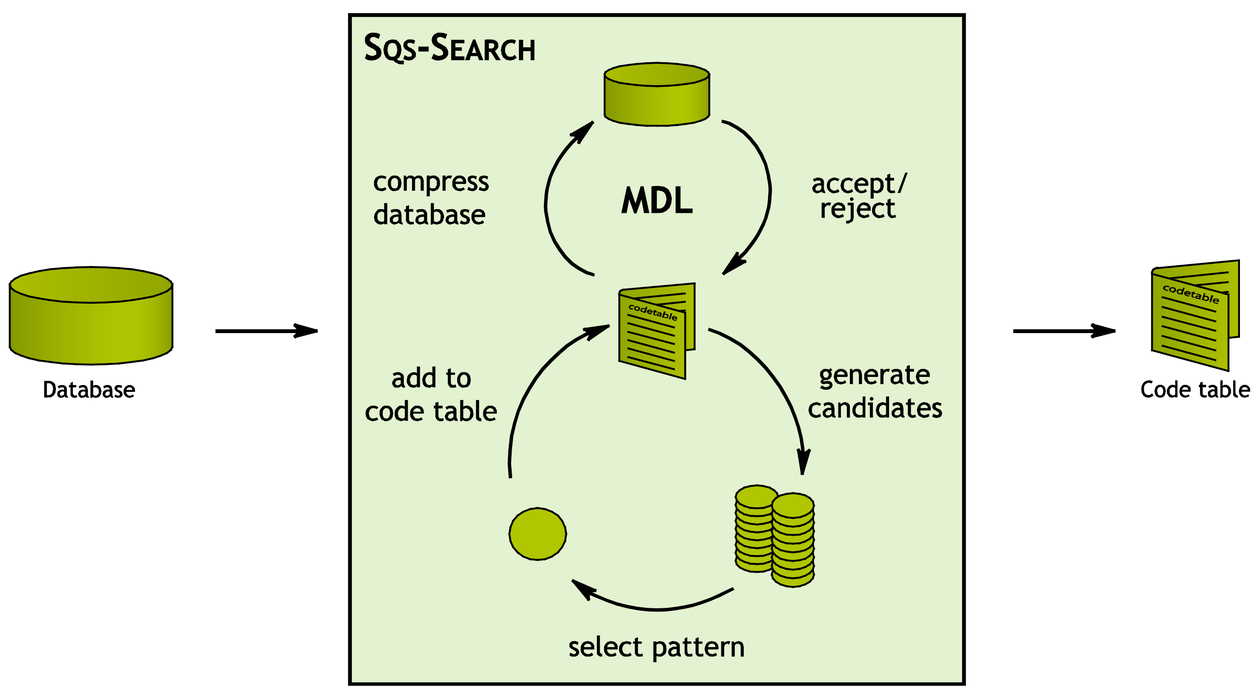
- if we fix the cover, we can compute code lengths
- if we fix the code lengths, we can obtain the optimal cover by dynamic programming

We alternate these steps until **convergence**

Mining Code Tables

We propose two algorithms for mining code tables

- SQS-CND filters a list of ordered candidate patterns
- SQS-SEARCH mines code tables directly from data, iteratively considering joins of current patterns



Starting with the basic singleton-event code table, SQS-SEARCH iteratively finds codes that frequent co-occur, and considers their join as a candidate

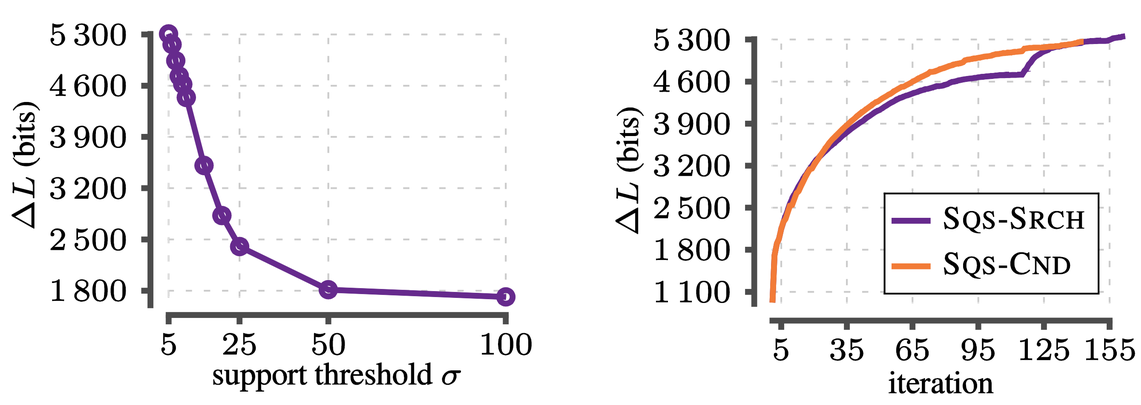
Experiments

For ease of interpretation we evaluate on text data

JMLR
supp. vector machine
machine learning
state [of the] art
data set
Bayesian network

MOBY DICK
sperm whale
Moby Dick
seven hundr.
seventy seventh
Hast seen White Whale

PRES. ADDRESSES
unit[ed] state[s]
economi public expenditur
take oath
equal right
exercis power



Presidential Addresses dataset. Gain in bits over the singleton-only code table, higher is better. (left) for SQS-CND per min-sup (right) per iteration

Conclusions

Mining highly informative patterns is an important aspect of exploratory data mining

- we score sets of serial episodes **by MDL**, and formalize how to **optimize a cover** of the data
- SQS efficiently mines good code tables, by filtering pre-mined candidates, or **directly from data**
- experiments show the discovered code tables are **small**, **non-redundant** and **characteristic** for the data
- **future work** includes
  - **overlapping, nesting** and **interleaving** patterns

Universiteit Antwerpen
FWO VLAANDEREN