A New Support Measure for Items in Streams *

Toon Calders
Eindhoven University of Technology

Nele Dexters   Bart Goethals
University of Antwerp

Abstract

Mining streams is a challenging problem, because the data can only be looked at once, and only small summaries of the data can be stored. We present a new frequency measure for items in streams that does not rely on a fixed window length or a time-decaying factor Based on the properties of the measure, an algorithm to compute it is shown Experimental evaluation supports the claim that the new measure can be computed from a summary with very small memory requirements, that can be maintained and updated efficiently In this extended abstract, the main points of the presentation are discussed

1 Motivation

Mining frequent items over streams received recently a lot of attention. It presents interesting new challenges over traditional mining in static databases. It is assumed that the stream can only be scanned once, and hence if an item is passed, it can not be revisited, unless it is stored in main memory. Storing large parts of the stream, however, is not possible because the amount of data passing by is typically huge

Different models have already been proposed in literature. The main characteristic is: how must the frequency of an item be measured? There are different types of models. (1) the sliding window model, (2) the time-fading model, or (3) the landmark model In the sliding window model [1, 3, 6, 8, 10], only the most recent events are used to determine the frequency of an item. In order to avoid having to count the supports on this window all over again in every time point, the algorithm in fact updates the frequency of the items based on the deletion of some transactions and the insertion of other In the time-fading model, the past is still considered important, but not as important as the present. This is modelled by gradually fading away the past [9] That is, there is, e g., a

*The presentation is based on material presented in the ECML/PKDD'06 workshop International Workshop on Knowledge Discovery from Data Streams (IWKDDS) [2] fixed, from the landmark designating the start of the system up till the current time [7, 8, 11]. The analysis of the stream is performed for only the part of the stream between the landmark and the current time instance.

Obviously, the landmark model is not very well suited to find current trends. The sliding window and the time-decaying models are more suitable; the sliding window method focusses solely on the present, while the time-decaying model still takes the past into account, although the effect fades away. For both the sliding window and the fading window approach, it is hard to determine the right parameter settings. Especially for the sliding window method, if the window length is set too high, interesting phenomena might get smoothed out. For example, suppose that the occurrence of an item a is cyclic; every month, in the beginning of the month, the frequency of a increases. If the length of the sliding window, however, is set to 1 month, this phenomenon will never be captured. In many applications it is not possible to fix a window length or a decay factor that is most appropriate for every item at every timepoint in an evolving stream.

2 Maximal Frequency Measure

Therefore, we propose a new frequency measure. We assume that on every timestamp, a new itemset arrives in the stream. We denote a stream as a sequence of itemsets; e.g., $\langle ab\ bc\ abc\rangle$ denotes the stream where at timestamp 1, the itemset ab arrives, at timestamp 2, bc, and at timestamp 3, abc. The *maximal frequency* measure for an itemset I in a stream S of length t, denoted $mfreq(a, S)$, is defined as the maximum of the frequency of the itemset over the time intervals $[s, t]$, with s any timepoint before t.

Example *We focus on target item a.*

$$\begin{aligned}
mfreq(a, \langle a\ \ b\ \ a\ \ a\ \ a\ \ b\rangle) &= \max\left(\frac{0}{1},\frac{1}{2},\frac{2}{3},\frac{3}{4},\frac{3}{5},\frac{4}{6}\right) = \frac{3}{4}\ . \\
mfreq(a, \langle b\ \ c\ \ d\ \ a\ \ b\ \ c\ \ d\ \ a\rangle) &= \max\left(\frac{1}{1},\cdots\right) = 1\ . \\
mfreq(a, \langle x\ \ a\ \ a\ \ x\ \ a\ \ a\ \ x\rangle) &= \max\left(\frac{0}{1},\frac{1}{2},\frac{2}{3},\frac{2}{4},\frac{3}{5},\frac{4}{6},\frac{4}{7}\right) = \frac{2}{3}\ .
\end{aligned}$$

3 Algorithm

For this frequency measure, we present an incremental algorithm that maintains a small summary of relevant information of the history of the stream that allows to produce the current frequency of a specific item in the stream immediately at any time. That is, when a new itemset arrives, the summary is updated, and when at a certain point in time, the current frequency of an item is required, the result can be obtained instantly from the summary. The structure of the summary is based on some critical observations about the windows with the maximal frequency. In short, many points in the stream can never become the starting point of a maximal window, no matter what the continuation of the stream will be.

Example In the following stream, the only positions that can ever become the starting point of a maximal interval for the singleton itemset a are indicated by vertical bars.

$$\langle |a\ a\ a\ b\ b\ b\ a\ b\ b\ a\ b\ a\ b\ a\ b\ a\ b\ b\ b\ b\ |a\ a\ b\ a\ b\ b\ |a\rangle$$

The summary will thus consist of some statistics about the few points in the stream that are still candidate starting points of a maximal window. These important points in the stream will be called the *borders*. More details can be found in [2].

4 Experimental Evaluation

Critical for the usefulness of the technique are the memory requirements of the summary that needs to be maintained in memory. We show experimentally that, even though in worst case the summary depends on the length of the stream, for realistic data distributions its size is extremely small. Obviously, this property is highly desirable as it allows for an efficient and effective computation of our new measure. Also note that our approach allows exact information as compared to many approximations considered in other works. In Figure 1, for some distributions, the maximal number of borders in synthetically generated random streams have been given. The number of borders corresponds linearly to the memory requirements of the algorithm.

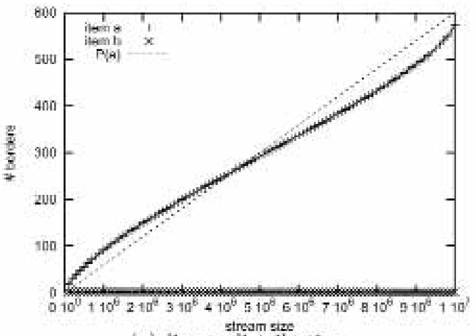
5 Summary

In the presentation, a new support measure for itemsets in streams is introduced and motivated. An algorithm to maintain a small summary based on which the support can immediately be produced is presented. Experimental evaluation shows that the memory requirements of this summary are very low.

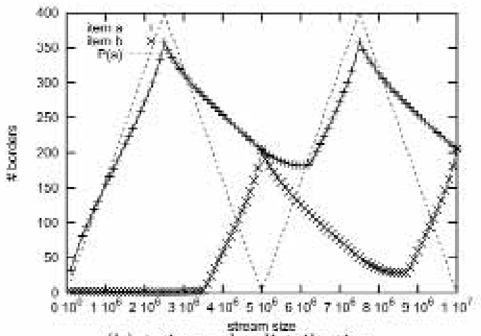
References

[1] Ruoming J. and Agrawal G.: An Algorithm for In-Core Frequent Itemset Mining on Streaming Data. in Proc. 5th IEEE Int. Conf. on Data Mining (ICDM'05), pp 210–217.

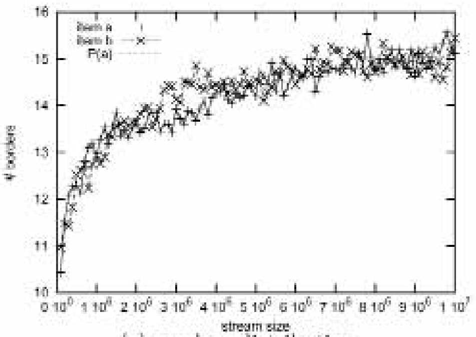
[2] Calders, T., Dexters, N., and Goethals, B.: Mining Frequent Items in a Stream Using Flexible Windows. In *Proc. of the ECML/PKDD-2006 International Workshop on Knowledge Discovery from Data Streams (IWKDDS)*; (2006)



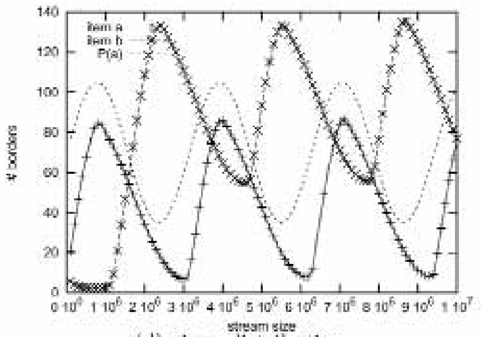
(a) linear distribution



(b) twin peaks distribution



(c) random distribution



(d) sinus distribution

Figure 1: Size of the summaries for two items a and b

[3] Demaine E.D., Lopez-Ortiz A. and Munro, J.I.: Frequency Estimation of Internet Packet Streams with Limited Space. In *Proc. of the 10th Annual European Symposium on Algorithms*, pp 348–360. (2002)

[4] Giannella C., Han J., Robertson E. and Liu C.: Mining Frequent Itemsets Over Arbitrary Time Intervals in Data Streams. In *Technical Report TR587*, Indiana University, Bloomington. (2003)

[5] Giannella C., Han J., Pei J., Yan X. and Yu P.S.: Mining Frequent Patterns In Data Streams at Multiple Time Granularities. In H. Kargupta, A. Joshi, K. Sivakumar and Y. Yesha (eds), *Next Generation Data Mining*, pp 191 212. (2003)

[6] Golab L., DeHaan D.,Demaine E.D., Lopez-Ortiz A. and Munro J.I.: Identifying Frequent Items in Sliding Windows over On-Line Packet Streams. In *Proc. of the 1st ACM SIGCOMM Internet Measurement Conference*, pp 173 178. (2003)

[7] Jin R. and Agrawal G.: An Algorithm for In-Core Frequent Itemset Mining on Streaming Data. In *Proc. of the 5th IEEE International Conference on Data Mining (ICDM)*, pp 210–217. (2005)

[8] Karp, R. M., Papadimitriou, C. H. and Shenker, S.: A Simple Algorithm for Finding Frequent Elements in Streams and Bags. In *ACM Trans. on Database Systems* 28, pp 51–55. (2003)

[9] Lee, D. and Lee, W.: Finding Maximal Frequent Itemsets over Online Data Streams Adaptively. In *Proc. of the 5th IEEE International Conference on Data Mining (ICDM)*, pp 266–273. (2005)

[10] Lin C.-H., Chiu D.-Y., Wu Y.-H. and Chen A.L.P.: Mining Frequent Itemsets from Data Streams with a Time-Sensitive Sliding Window. In *Proc. SIAM International Conference on Data Mining*. (2005)

[11] Yu J.X., Chong Z., Lu H. and Zhou A.: False Positive or False Negative: Mining Frequent Items from High Speed Transactional Data Streams. In *Proc. of the 30th International Conference on Very Large Databases*, pp 204–215. (2004)

Mining Frequent Itemsets in a Stream

Toon Calders, TU/e

(joint work with Bart Goethals and Nele Dexters, UAntwerpen)

Outline

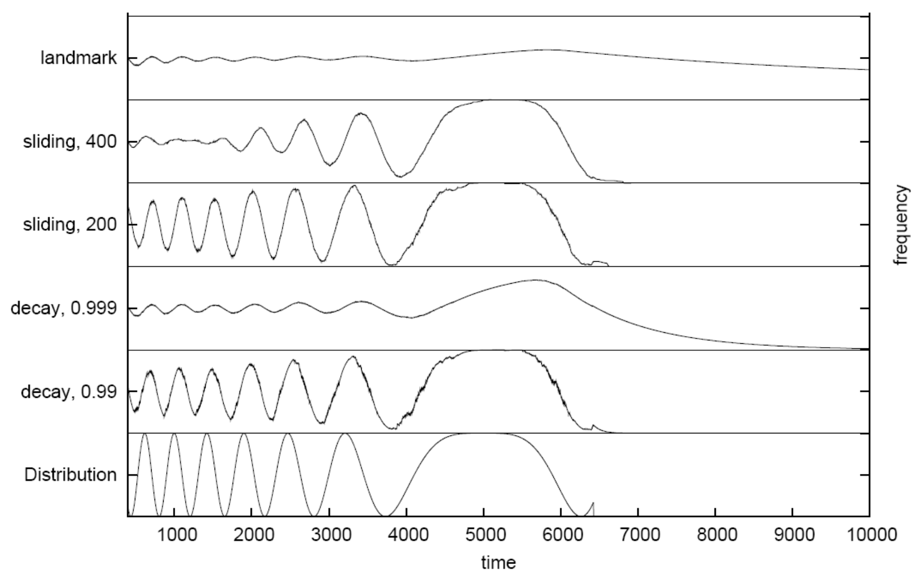
- Motivation
- Max-Frequency
- Algorithm
  - for one itemset
  - mining all Frequent Itemsets
- Experiments
- Conclusion

Motivation

- Model:
  - Every timestamp an itemset arrives
- Goal:
  - Find sets of items that frequently occur together
  - Take into account history,
  - Yet, recognize sudden bursts quickly

Motivation

- Most definitions of frequency rely heavily on the correct parameter settings
  - Sliding window length
  - Decay factor
  - …
- Correct parameter setting is hard
  - Can be different for different items (not to mention sets!)



Outline

- Motivation
- Max-Frequency
- Algorithm
  - for one itemset
  - mining all Frequent Itemsets
- Experiments
- Conclusion

Max-Frequency

Therefore, a new frequency measure:

$$\text{mfreq}(I, S) := \max_{k=1..|S|}(\text{freq}(I, \text{last}(k, S)))$$

Frequency is measured in the window where it is *maximal.*

Itemset gets the benefit of the doubt …

Example

mfreq(a, $\underrightarrow{\text{ac abc ab ac ab bc}}$)

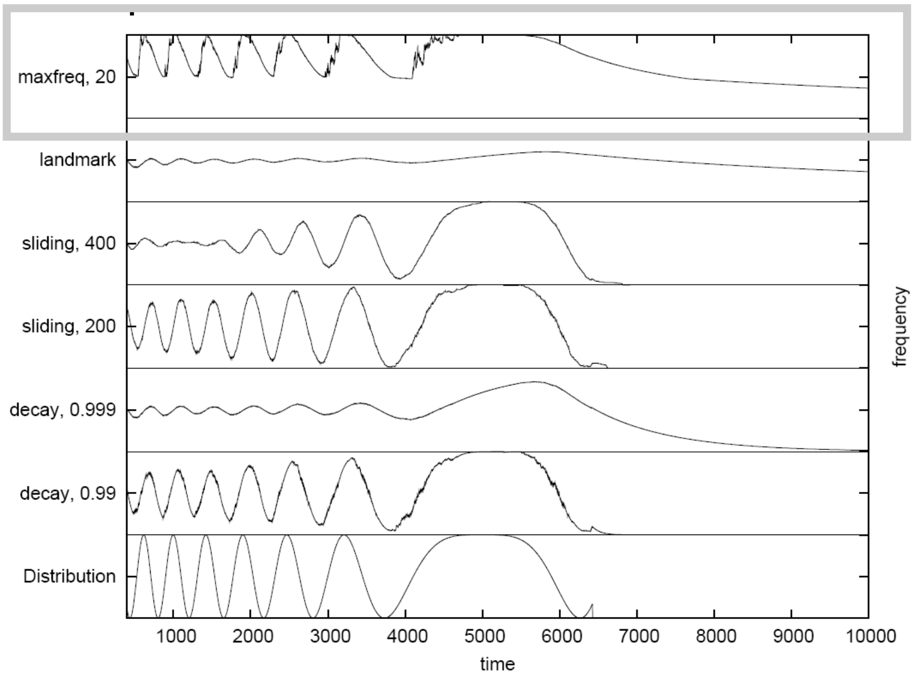
ac	bc	ab	ac	ab	bc	0	
ac	bc	ab	ac	ab	bc	1/2	
ac	bc	ab	ac	ab	bc	2/3	
ac	bc	ab	ac	ab	bc	3/4	←
ac	bc	ab	ac	ab	bc	3/5	
ac	bc	ab	ac	ab	bc	4/6	

Properties of Max-Freq

+ Detects sudden bursts
+ Takes into account the past

- When target itemset arrives: sudden jump to a frequency of 1

+ Solution: minimal window length



Outline

- Motivation
- Max-Frequency
- Algorithm
  - for one itemset
  - mining all Frequent Itemsets
- Experiments
- Conclusion

Algorithm

1. How to do it for ***one*** itemset?
2. How to do it for a ***frequent*** itemset?
3. How to do it for ***all*** frequent itemsets?

Maintain a ***summary*** of the stream that allows to find the frequencies immediately.

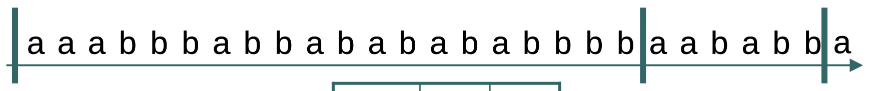
Properties (one itemset)

Checking all possible windows to find the maximal one: **infeasible**

BUT: not every point needs to be checked

↓

Only some special points = the borders



timestamp	**1**	**21**	**27**
# targets	**8**	**3**	**1**

How to find a border?

- Target set a
- Is the marked position a border?

ab ac bc | ac bc abc a b

How to find a border?

- Target set a
- Is the marked position a border?

ab ac bc | ac bc abc a b

2/3 1/3

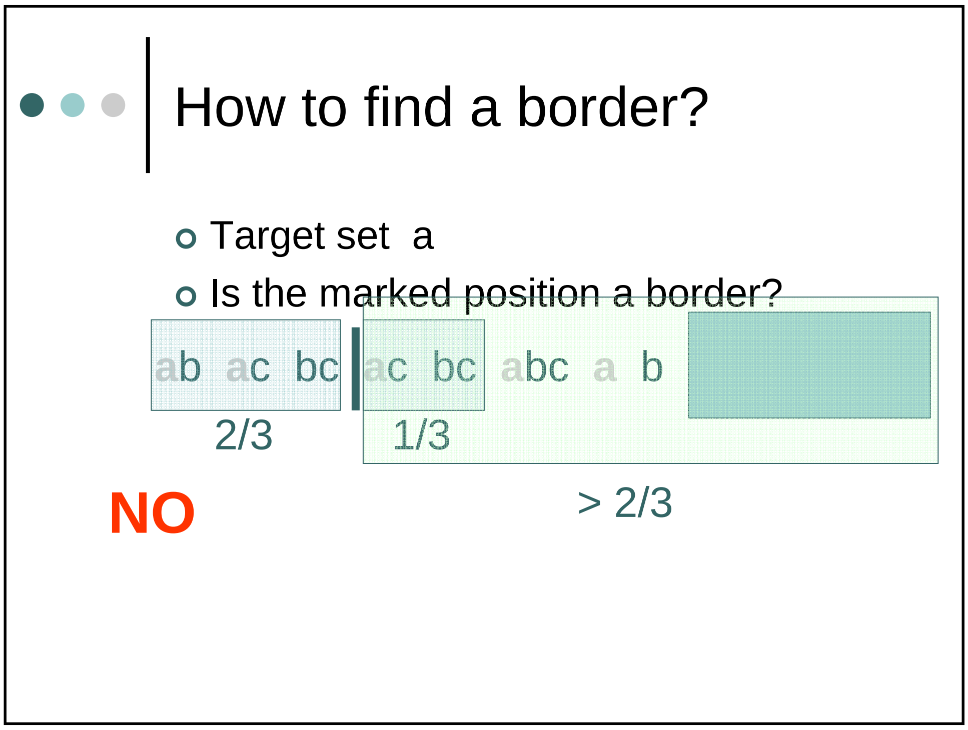
How to find a border?

- Target set a
- Is the marked position a border?

ab ac bc | ac bc abc a b

2/3 1/3

NO



How to find a border?

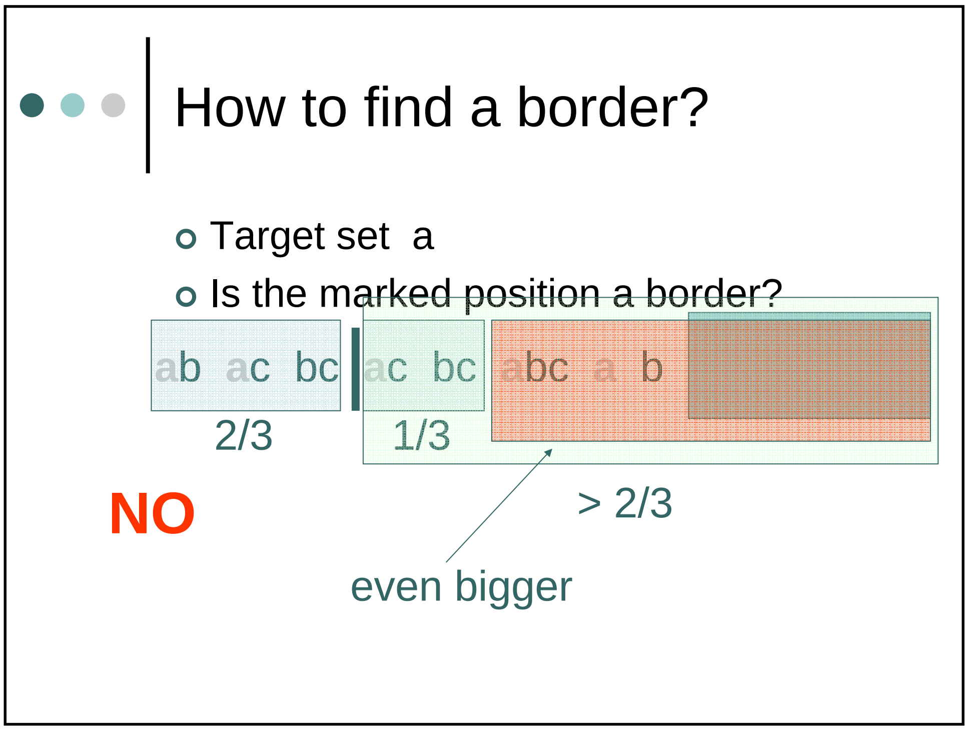
- Target set a
- Is the marked position a border?

ab ac bc | ac bc abc a b

2/3 1/3

> 2/3

NO



How to find a border?

- Target set a
- Is the marked position a border?

ab ac bc | ac bc abc a b

2/3 1/3

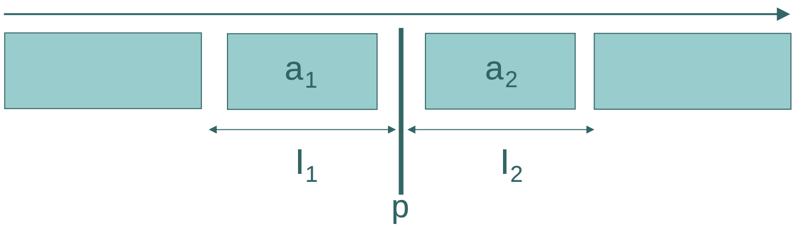
> 2/3

NO

even bigger

How to find the borders?

- This is true in general:



If $a_1/l_1 \geq a_2/l_2$, position p is never the border again!

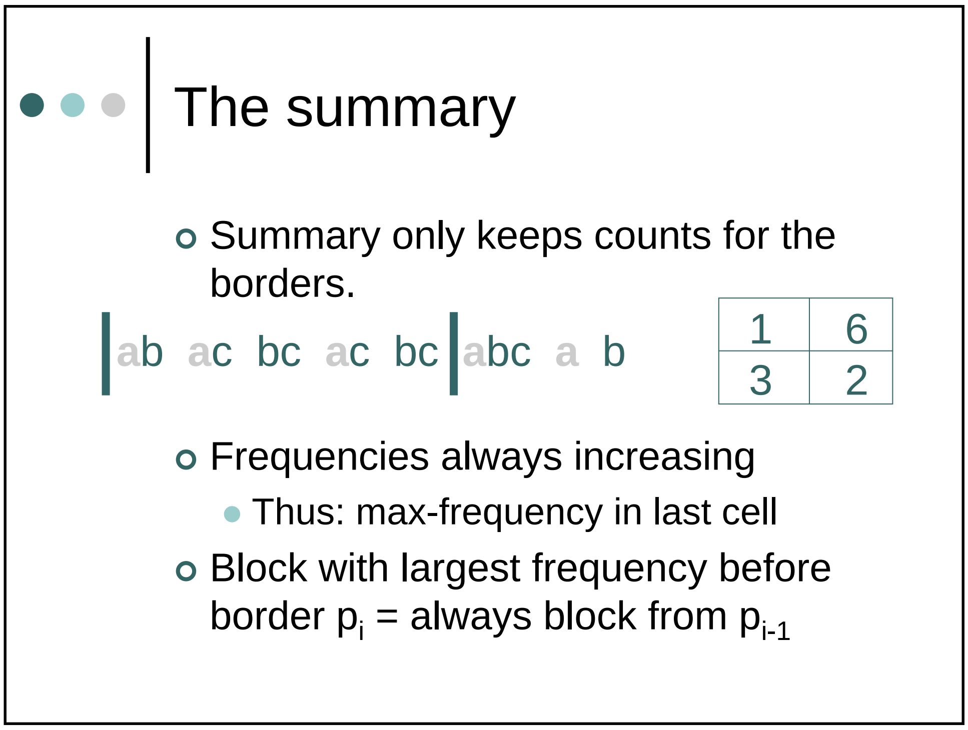
Very powerful pruning criterion!

The summary

- Summary only keeps counts for the borders.

| ab ac bc ac bc | abc a b

1	6
3	2



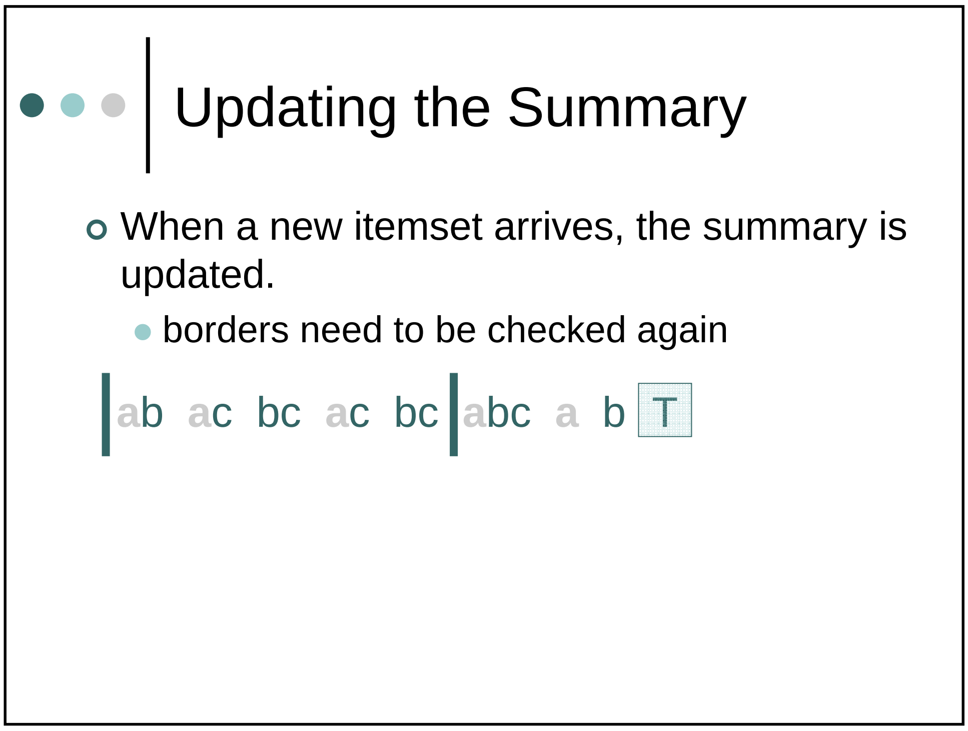
The summary

- Summary only keeps counts for the borders.

|ab ac bc ac bc |abc a b

1	6
3	2

- Frequencies always increasing
  - Thus: max-frequency in last cell
- Block with largest frequency before border p_i = always block from p_{i-1}



Updating the Summary

- When a new itemset arrives, the summary is updated.
  - borders need to be checked again

|ab ac bc ac bc |abc a b T

Updating the Summary

- When a new itemset arrives, the summary is updated.
  - borders need to be checked again

| ab ac bc ac bc | abc a b T

  - no new « before » - blocks
  - only one new « after » - block
  - maximal block before: always previous border

Updating the Summary

- When a new itemset arrives, the summary is updated.
  - borders need to be checked again

| ab ac bc ac bc | abc a b T

  - no new « before » - blocks
  - only one new « after » - block
  - maximal block before: always previous border

Updating the Summary

- The new position is a border if and only if it contains the target itemset.

| ab ac bc ac bc | abc a b | ab

1	6	9
3	2	1

| ab ac bc ac bc abc a b b

1
5

Summary: the Summary

- Only keep entries for borders
- Get Max-frequency = access last cell only
- Update summary:
  - if target: add new entry
  - if non-target: check borders
    - only one check required: still in ascending order?
    - most recent border always drops first
    - no need to check at every timestamp

Mining Frequent Itemsets

- Only interested in itemsets that are frequent.
- We can throw away any border with a frequency lower than the minimal frequency.

ab ac bc ac bc | abc a b | ab

6	9
2	1

minfeq = 2/3

Mining All Frequent Itemsets

- We only need to maintain the summaries for the frequent itemsets
- Can still be a lot, though …
  - every subset of the most recent transaction …
  - minimal window length reduces this problem
- FUTURE WORK: reduce this number; rely, e.g., on approximate counts

Outline

- Motivation
- Max-Frequency
- Algorithm
  - for one itemset
  - mining all Frequent Itemsets
- Experiments
- Conclusion

Experiments

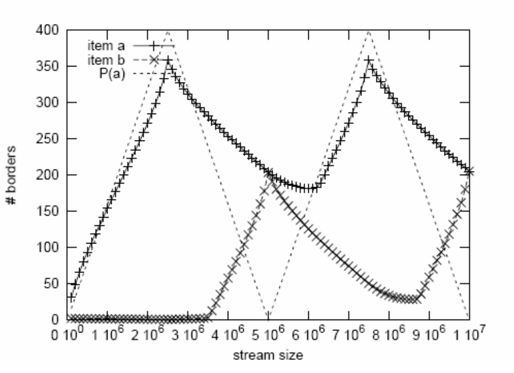
- Size of the summaries
  - number of borders for random data
  - average, maximal number of borders in real-life data
- Theoretical worst case

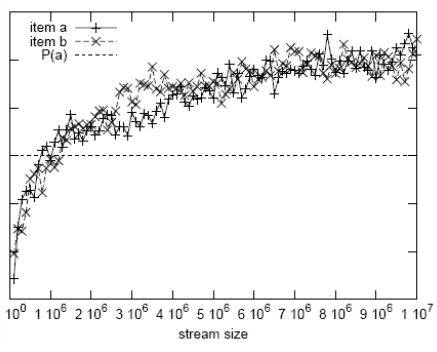
$$N = \left(\frac{\pi^2 L}{2}\right)^{2/3} \frac{3}{\pi^2}$$

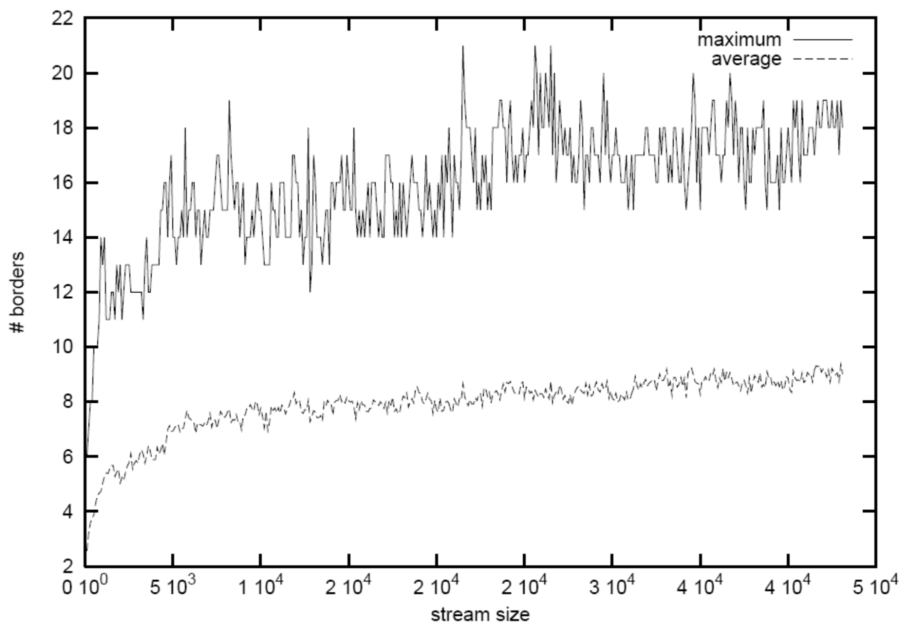
Experiments

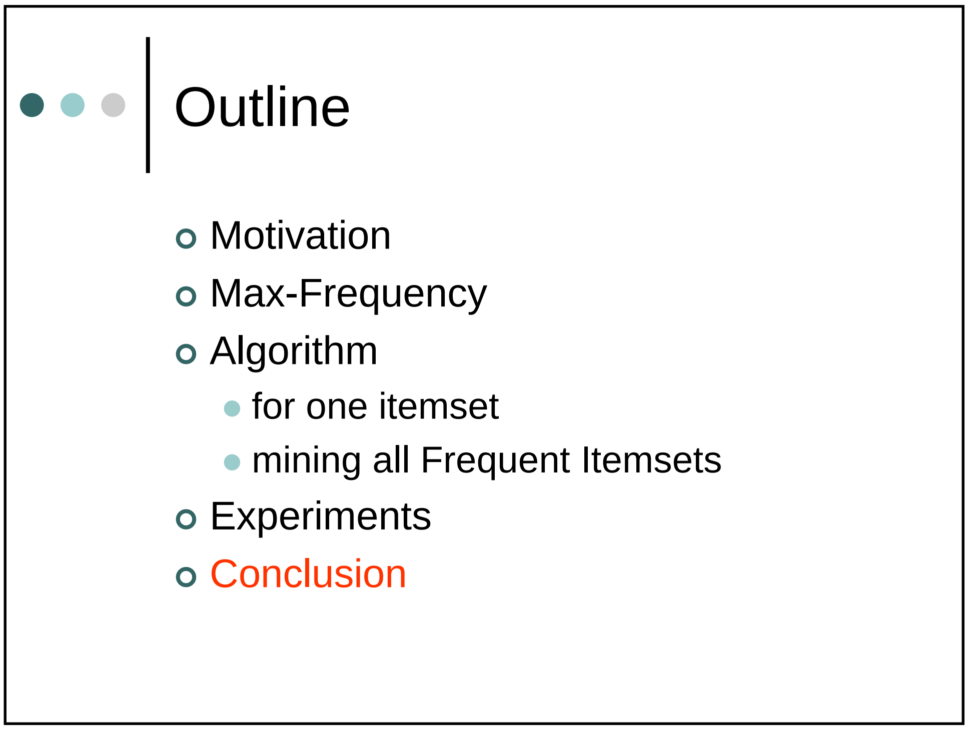
Twin Peaks distribution

Uniform Distribution



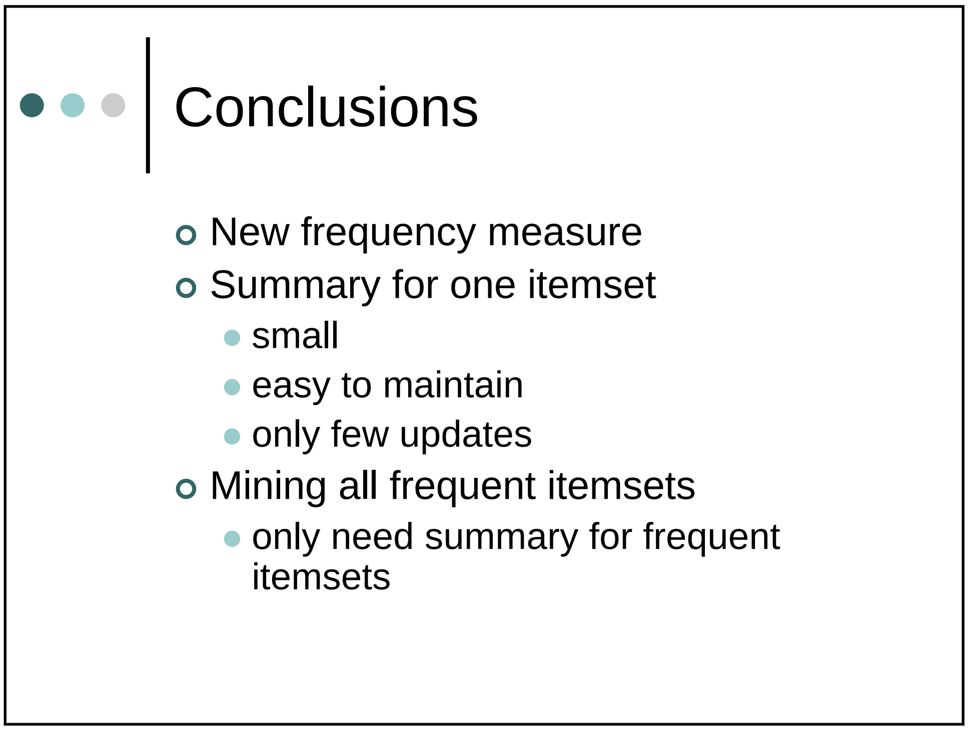






Outline

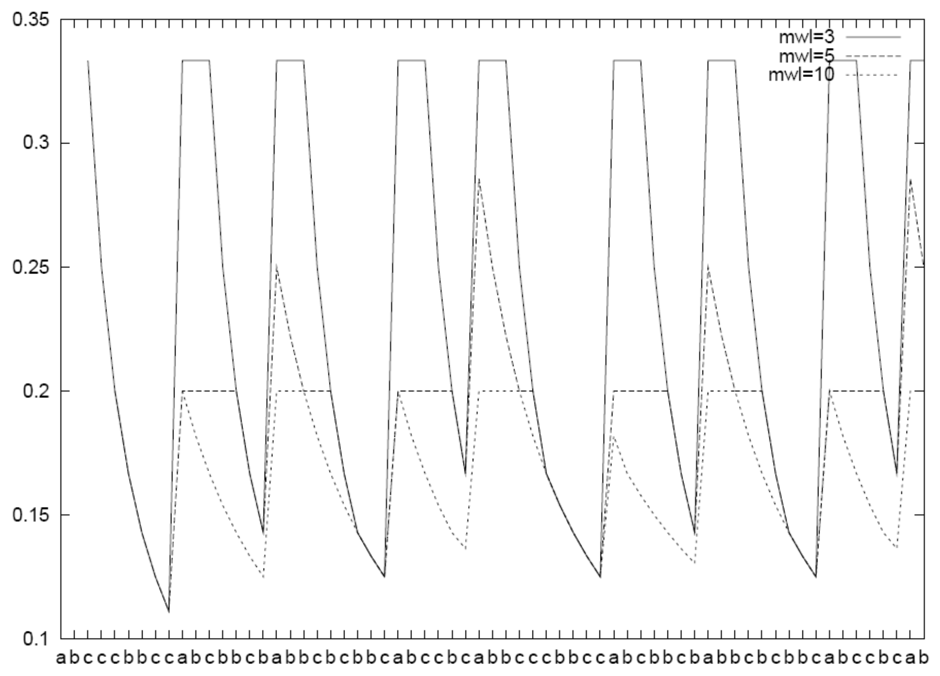
- Motivation
- Max-Frequency
- Algorithm
  - for one itemset
  - mining all Frequent Itemsets
- Experiments
- Conclusion



Conclusions

- New frequency measure
- Summary for one itemset
  - small
  - easy to maintain
  - only few updates
- Mining all frequent itemsets
  - only need summary for frequent itemsets

Thank you !



0.35
0.3
0.25
0.2
0.15
0.1
mwl=3
mwl=5
mwl=10
a b c c c b b c c a b c b b c b a b b c b c b b c a b c c b c a b b c c c b b c c a b c b b c b a b b c b c b b c a b c c b c a b