

# **Evaluating Pattern Set Mining Strategies in a Constraint Programming Framework**

*Tias Guns* and Siegfried Nijssen and Luc De Raedt

DTAI, K.U.Leuven, Belgium

**PAKDD11**-- Shenzhen, China

# Pattern mining

Analysing a dataset  
to find patterns of *interest*

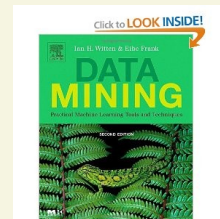
For example:

Analysing purchases (e.g. books)

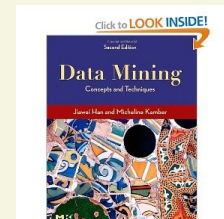
Here, patterns are sets of “items”



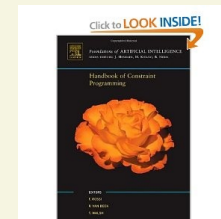
(e.g.



+



+



)

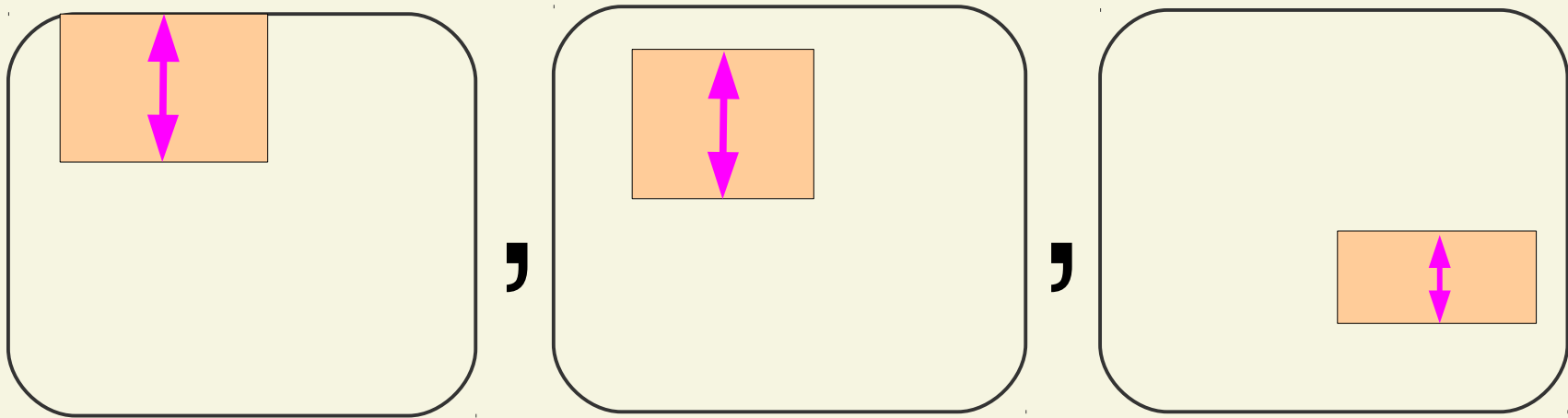
# *Interest:* frequent patterns

Given:

- Examples (here: categorical data)

Find:

- All **patterns** with **frequency**  $\geq \theta$



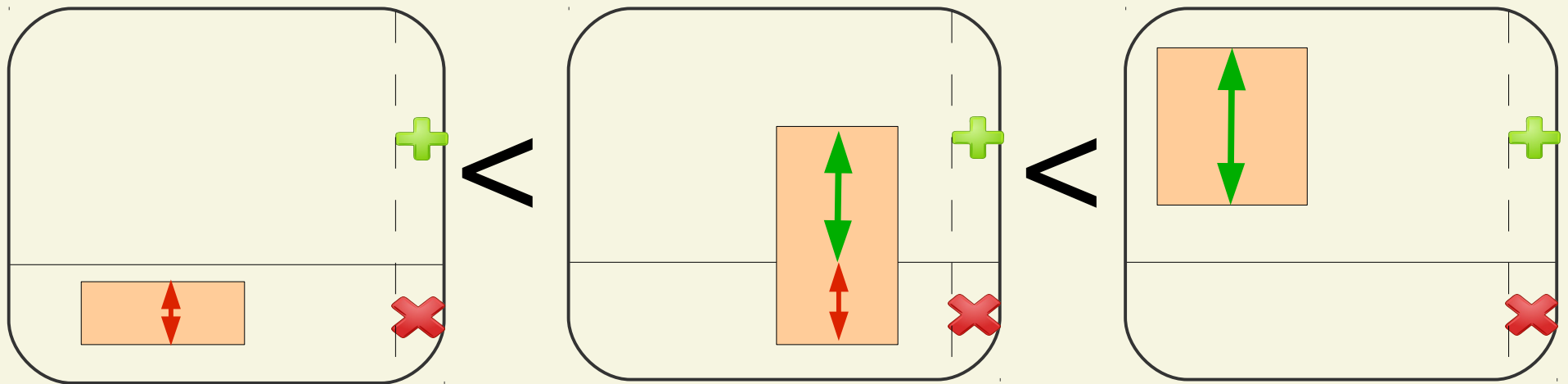
# *Interest:* discriminative patterns

Given:

- Positive and Negative examples

Find:

- The most discriminative pattern:  
**maximising accuracy** on positives (= Pos - Neg)

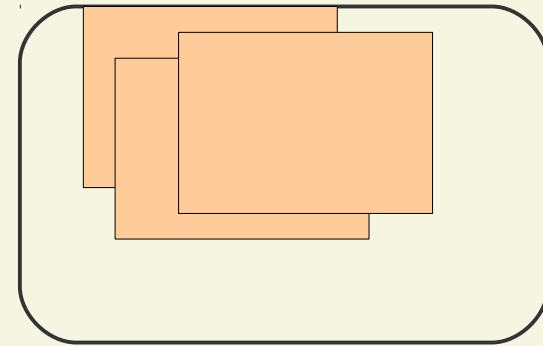


# Common pattern mining problems

- Too many patterns of *interest*:



- Many overlapping patterns:

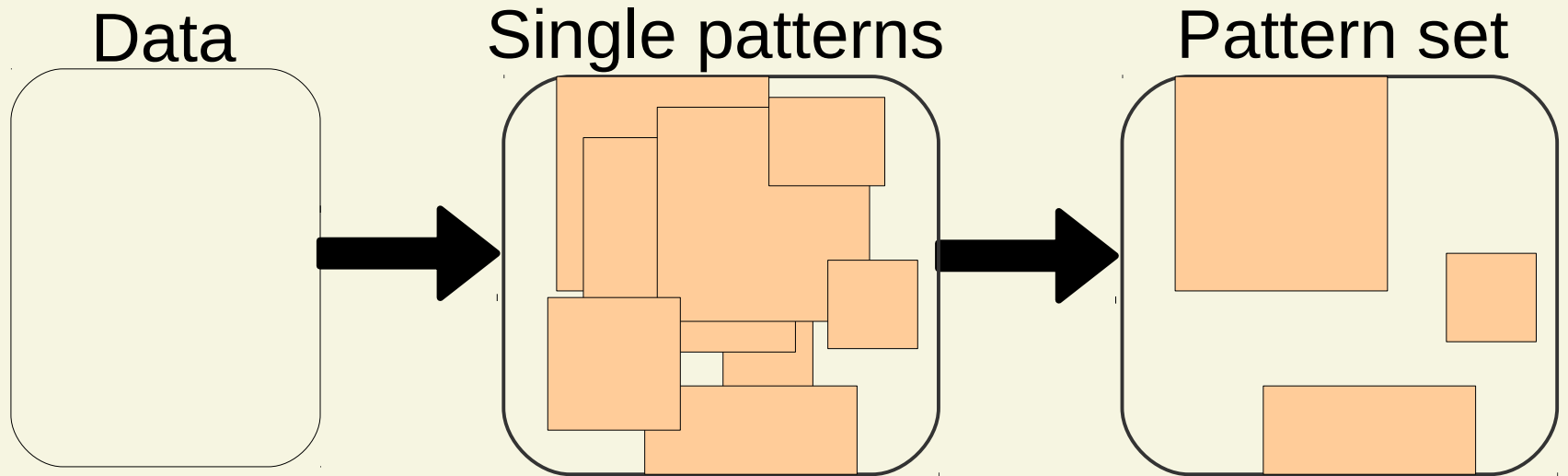


- Single patterns, just pieces of the puzzle:

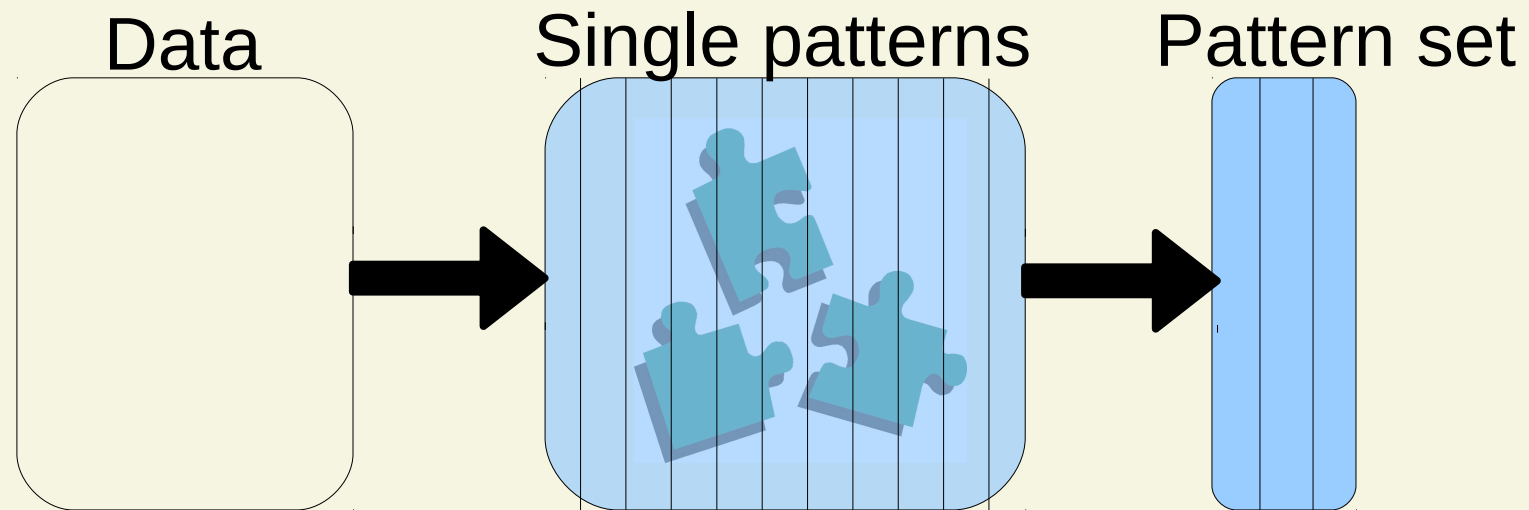


- ➔ Better understanding of data
- ➔ Outliers
- ➔ Classifier

# Putting the pieces together?



# Putting the pieces together?



Emerging research topic:

Pattern Set Mining

# Key points

## Problem:

- Many strategies proposed (*for Pattern Set Mining*)
- No clear comparison possible

## Our contribution:

- I. Framework to compare strategies
- II. Case study on *concept learning* task

# Overview

1. Motivation

**2. Pattern Set Mining strategies**

3. Constraint Programming framework

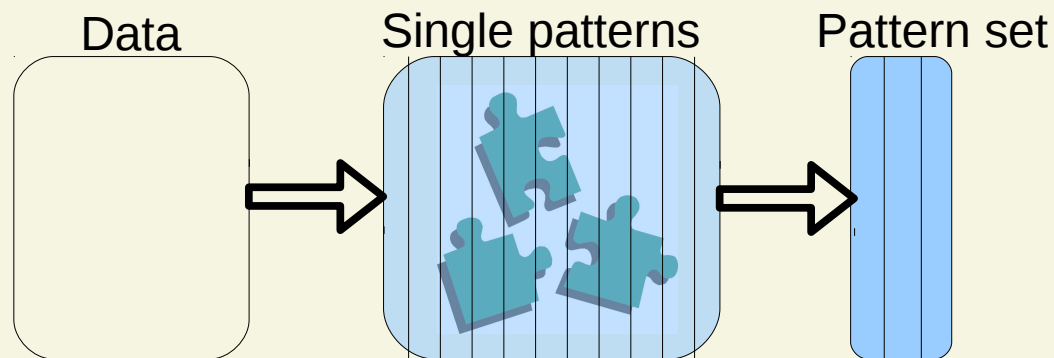
4. Case study

5. Conclusions

# Pattern Set Mining Strategies

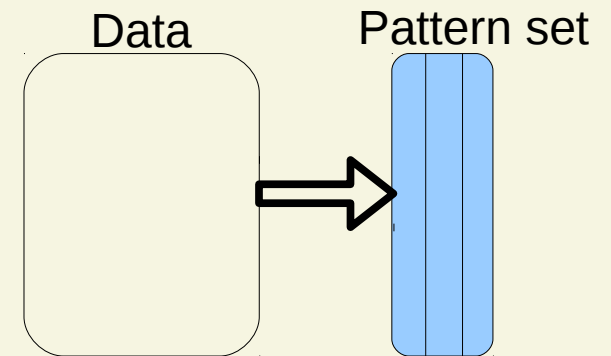
Popular choices:

2-step



vs.

1-step

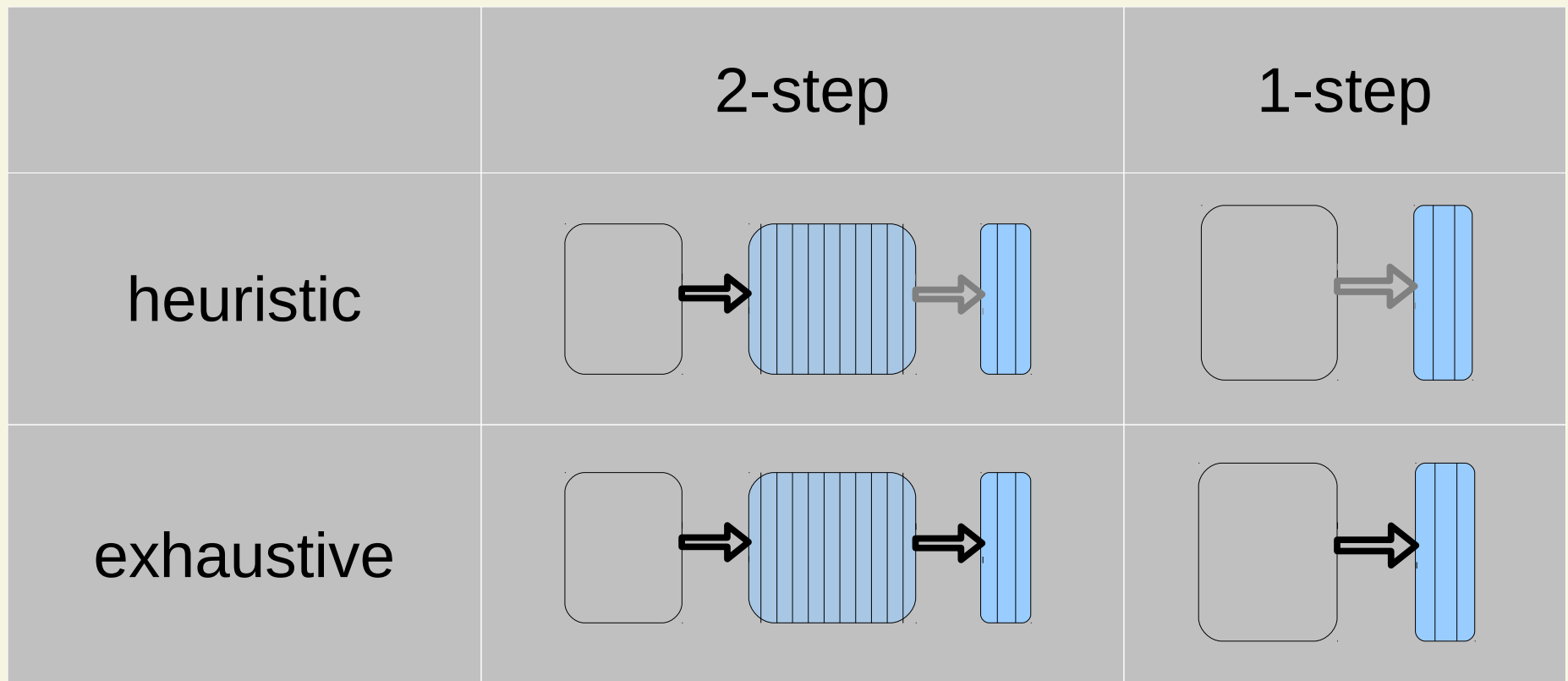


Data Mining: associative classification  
subgroup discovery, summarization, ...

Machine Learning:  
rule induction, ...

# Pattern Set Mining Strategies

More choices



# Constraint Programming framework

called 'CP4IM' (url: <http://dtai.cs.kuleuven.be/CP4IM>)

- Frequent, closed, discriminative, ... itemset mining
- Most general constraint-based system to date
- Recently extended to pattern set mining

Its **generality** allows to compare many strategies

# Constraint Programming

One of the big success stories of A.I.

- Among most general problem solving methods
- Generality: each constraint is self-sustaining
- Fast solvers with large constraint libraries  
publicly available

# Constraint Programming

For solving constraint satisfaction and optimisation problems

$$\text{CP} = \text{Model} + \text{Search}$$

- Model: declarative specification of constraints
- Search: generic DFS with efficient constraint *propagation*

# Example specification of freq. itemset mining

```
int: NrI; int: NrT;  
array [1..NrT,1..NrI] of bool: TDB;  
int: Freq;  
  
array [1..NrI] of var bool: Items;  
array [1..NrT] of var bool: Trans;  
  
constraint % coverage  
  forall(t in 1..NrT) (  
    Trans[t] <-> sum(i in 1..NrI) (bool2int(TDB[t,i] -> Items[i])) <= 0  
  );  
constraint % frequency  
  forall(i in 1..NrI) (  
    Items[i] -> sum(t in 1..NrT) (bool2int(TDB[t,i]  $\wedge$  Trans[t])) >= Freq  
  );  
  
solve satisfy;
```

# Overview

1. Motivation

2. Pattern Set Mining strategies

3. Constraint Programming framework

**4. Case study**

5. Conclusions

# Case study: Concept Learning

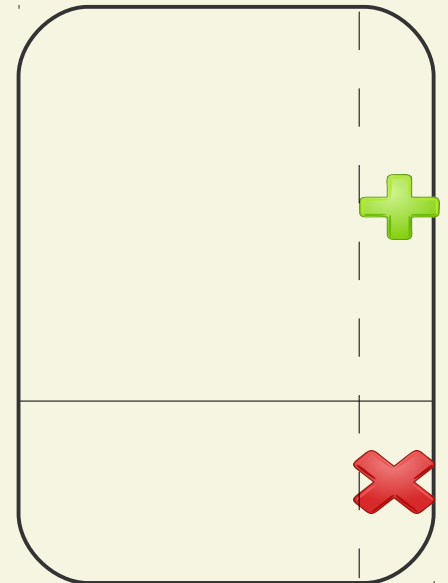
- History in computational learning theory
- Related to pattern mining and rule learning

Given:

positive & negative examples

Find:

the **best** conceptual description  
of the positive examples



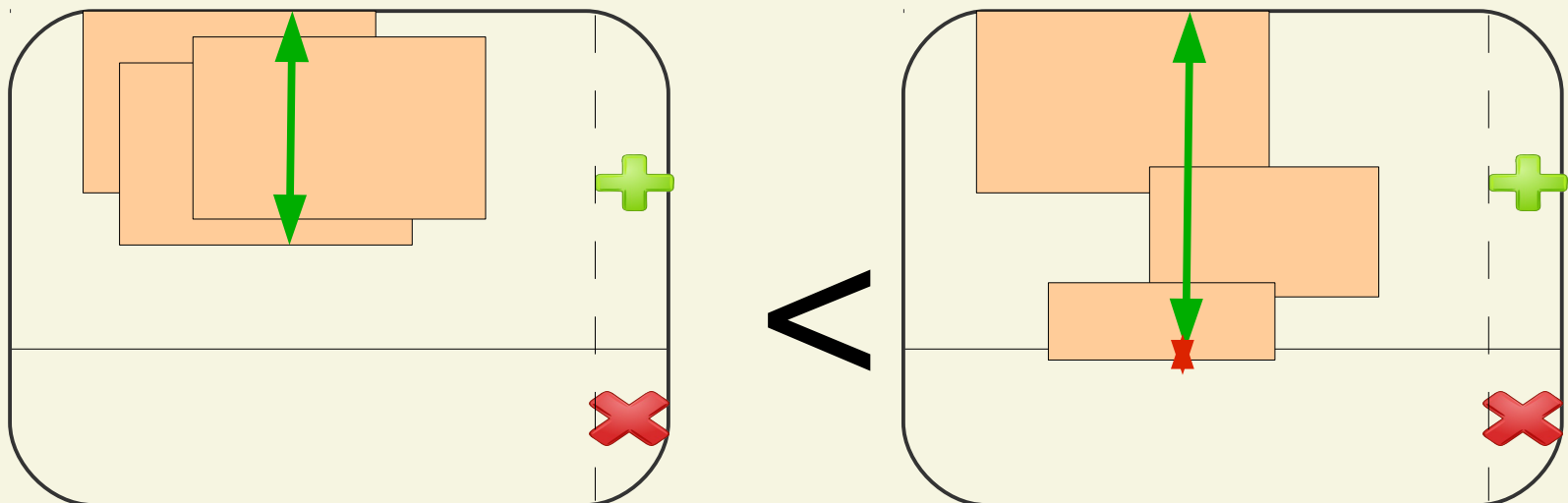
# Case study: Concept Learning

Given:

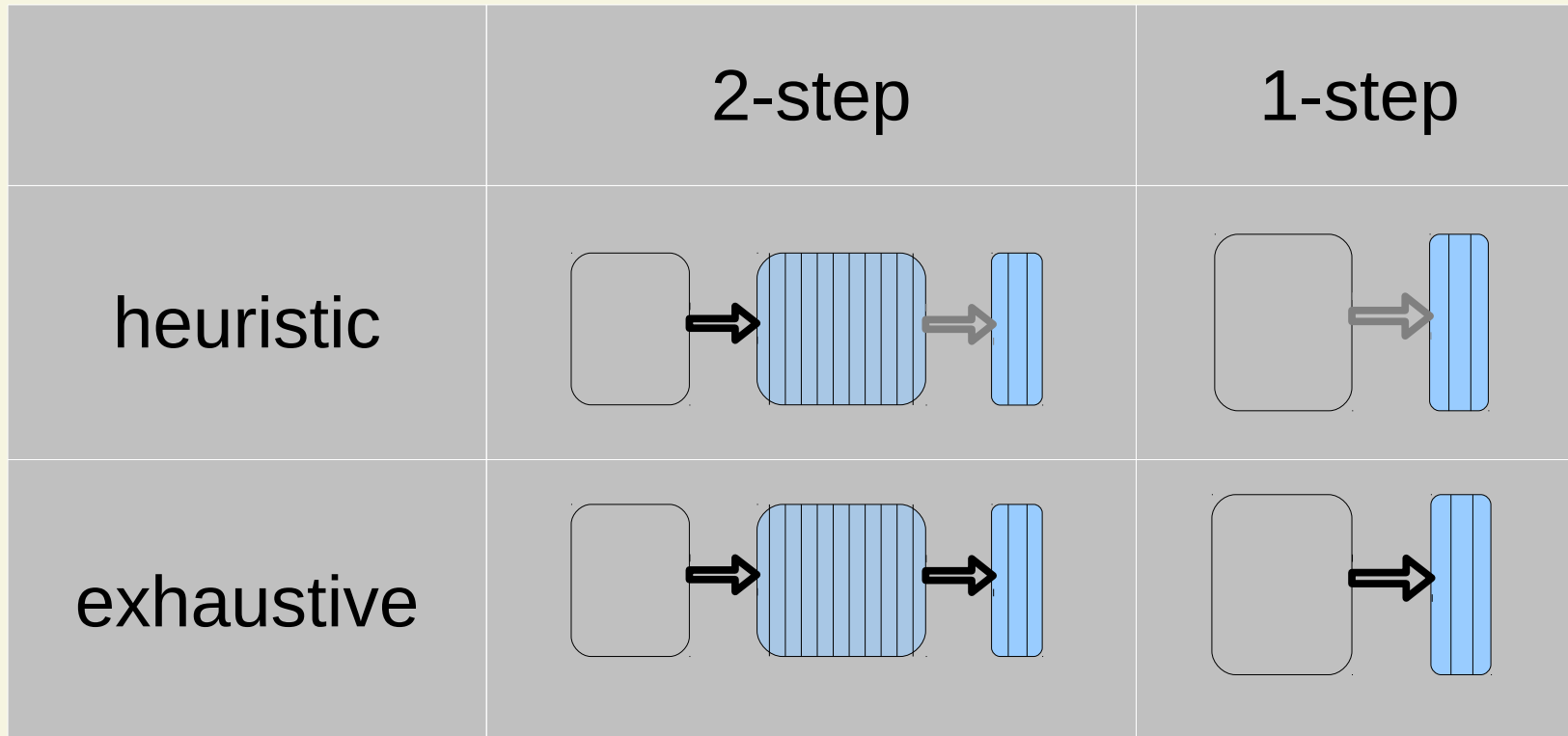
Pos. & Neg. examples, here: categorical data

Find:

Conceptual description, here: set of patterns/rules  
**maximizing** accuracy (= Pos - Neg)

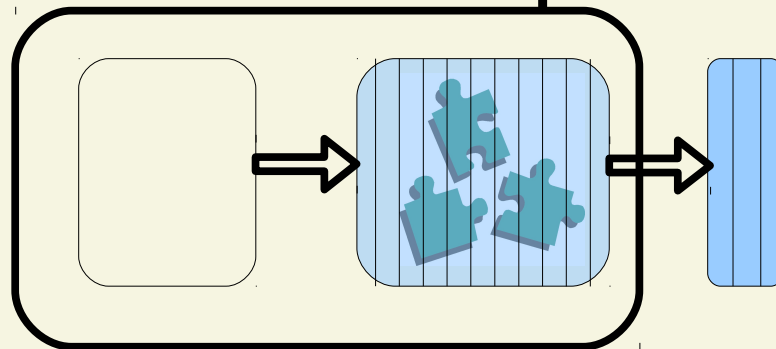


# Case study, strategies



- Each strategy uses the CP framework
- Compare strategy conditions and trade-offs

# step 1 of 2: Local pattern Mining

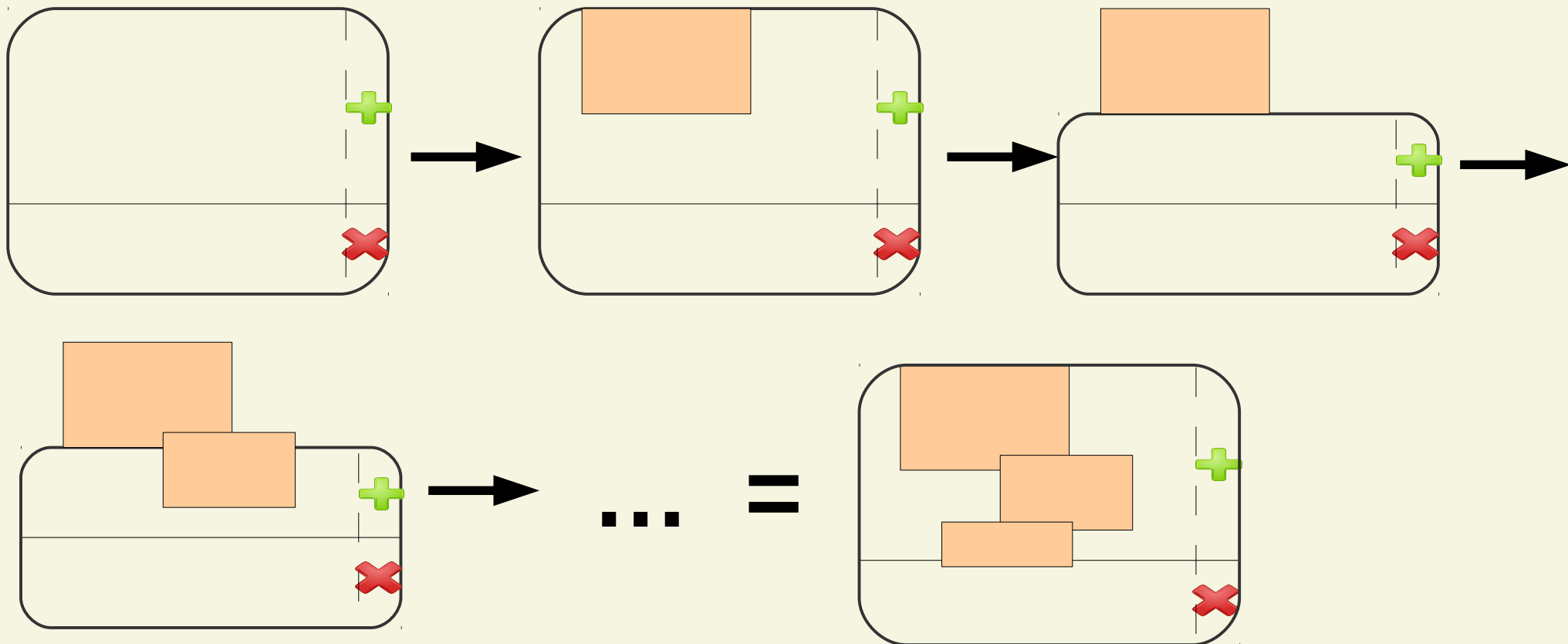


	Mushroom	Vote	Hepatitis	German-credit	Austr.-credit	Kr-vs-kp
Transactions	8124	435	137	1000	653	3196
Items	119	48	68	112	125	73
Class distr.	52%	61%	81%	70%	55%	52%
Total patterns	221524	227032	3788342	25M+	25M+	25M+
Pattern poor/rich	poor	poor	poor	rich	rich	rich
frequency $\geq 0.7$	12	1	137	132	274	23992
frequency $\geq 0.5$	44	13	3351	2031	8237	369415
frequency $\geq 0.3$	293	627	93397	34883	257960	25M+
frequency $\geq 0.1$	3287	35771	1827264	2080153	24208803	25M+
accuracy $\geq 0.7$	197	193	361	2	11009	52573
accuracy $\geq 0.6$	757	1509	3459	262	492337	2261427
accuracy $\geq 0.5$	11673	9848	31581	6894	25M+	25M+
accuracy $\geq 0.4$	221036	105579	221714	228975	25M+	25M+

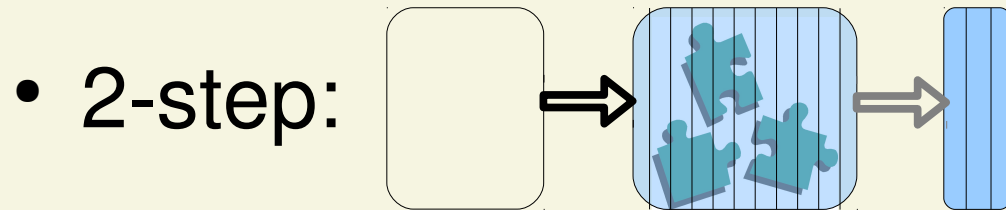
# Sequential covering, the basics

Repeat:

find pattern, add to set, remove covered examples



# Sequential covering, 2-step vs 1-step



Mine patterns + sequentially select best

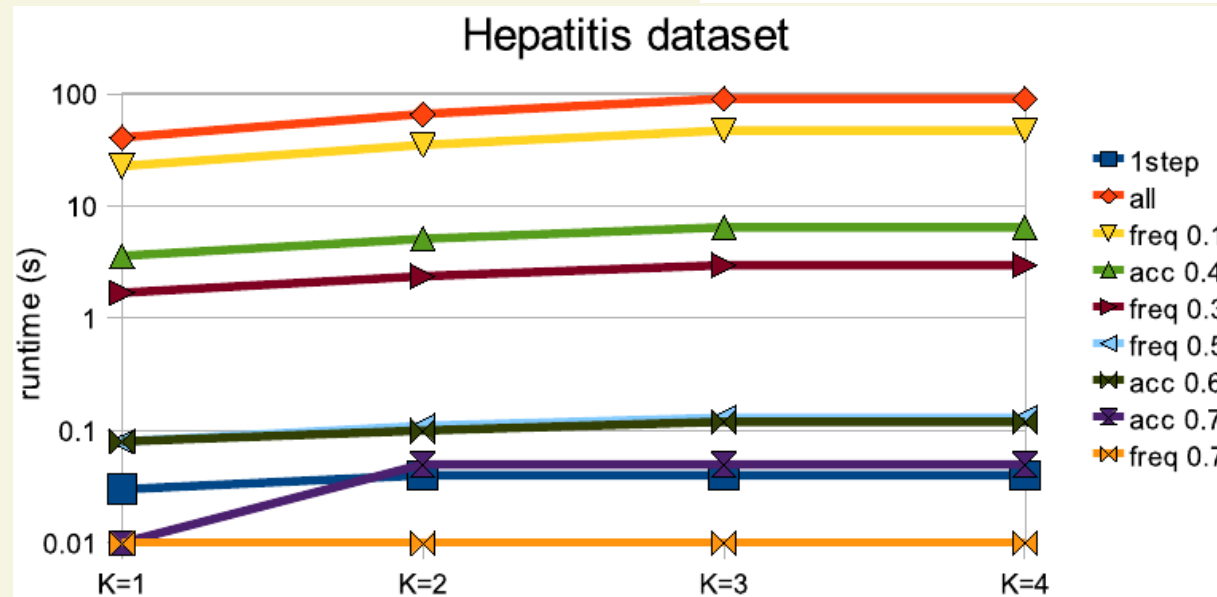
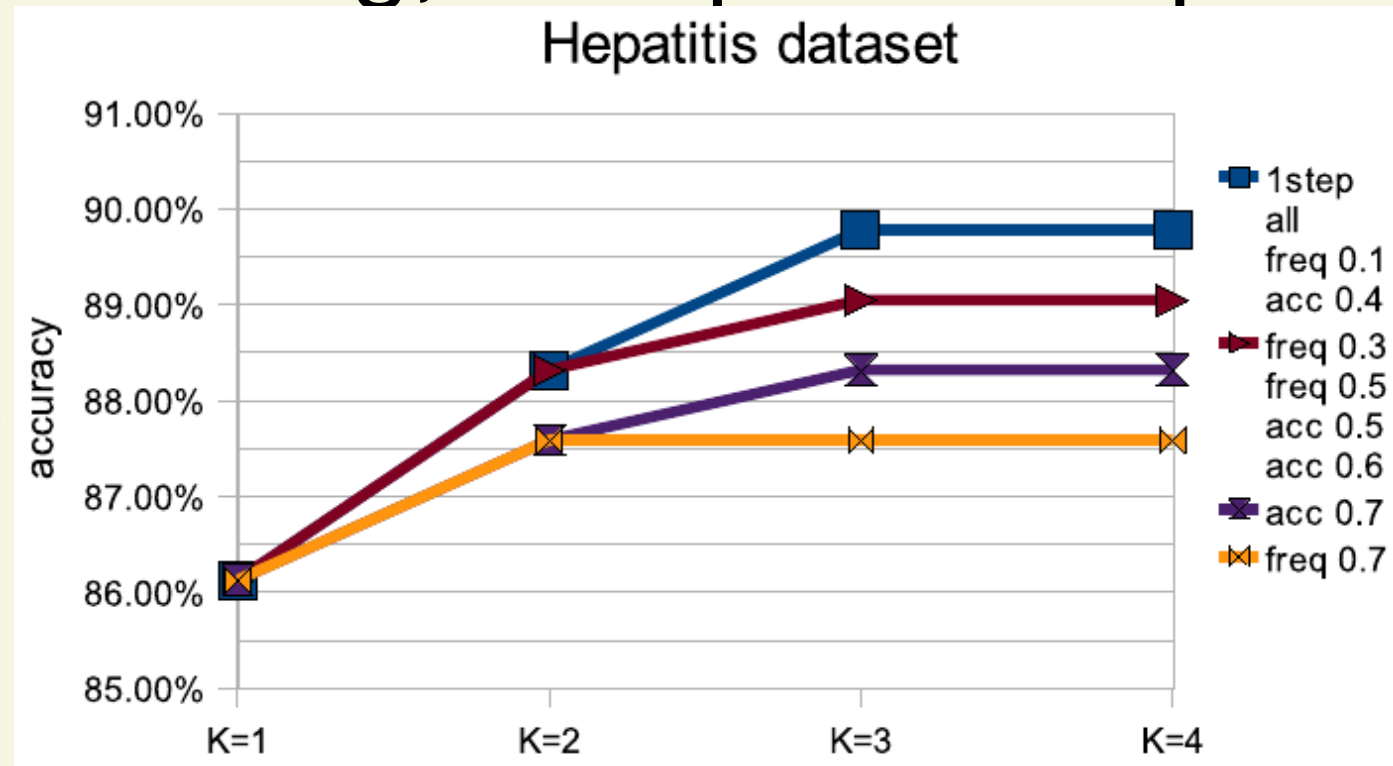
Mining: all, min. frequency or min. accuracy



Sequentially mine single best pattern

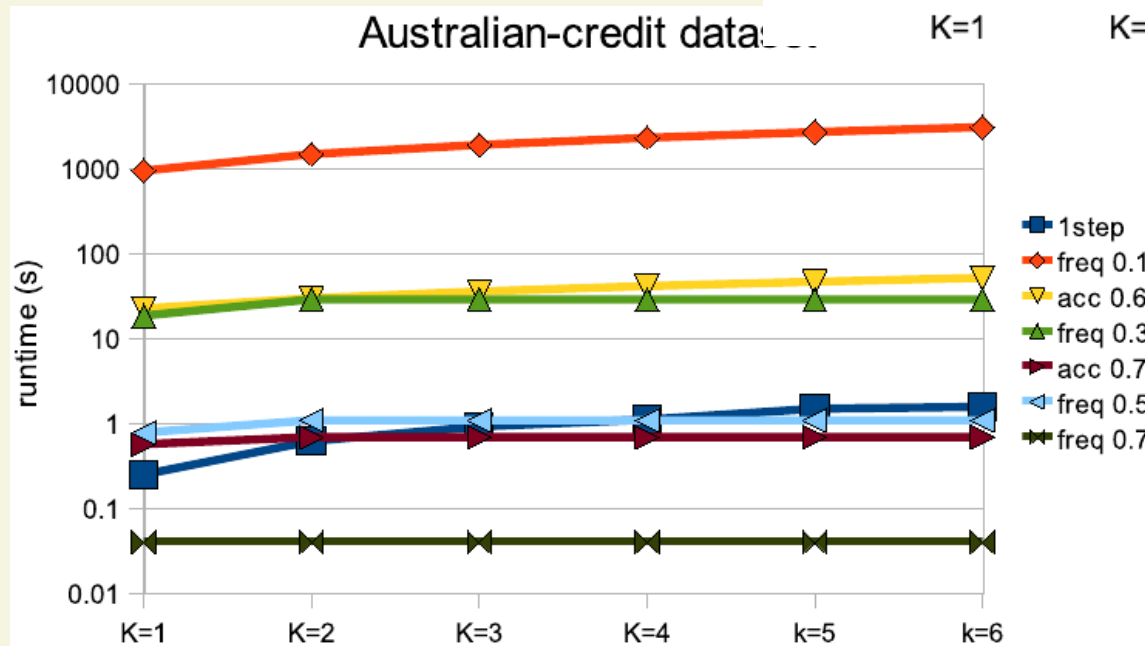
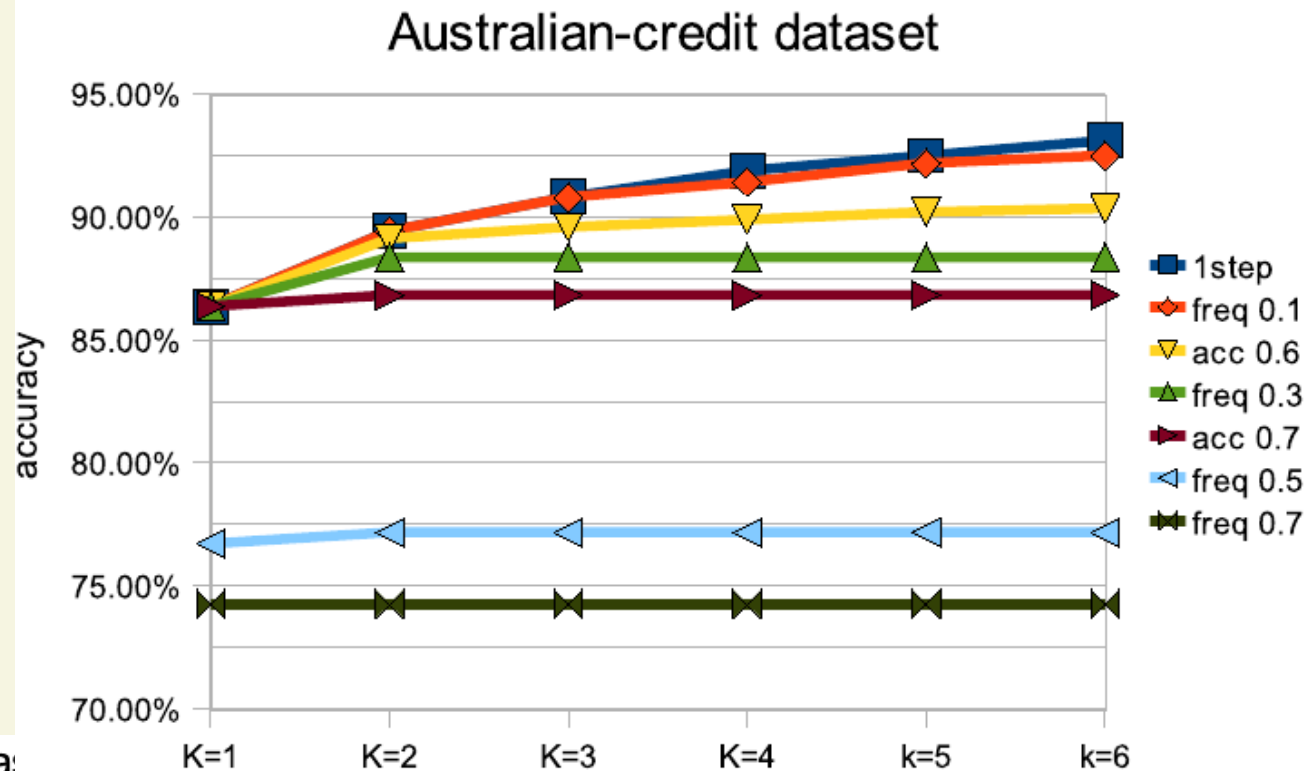
# Sequential covering, 2-step vs 1-step

Pattern poor data:

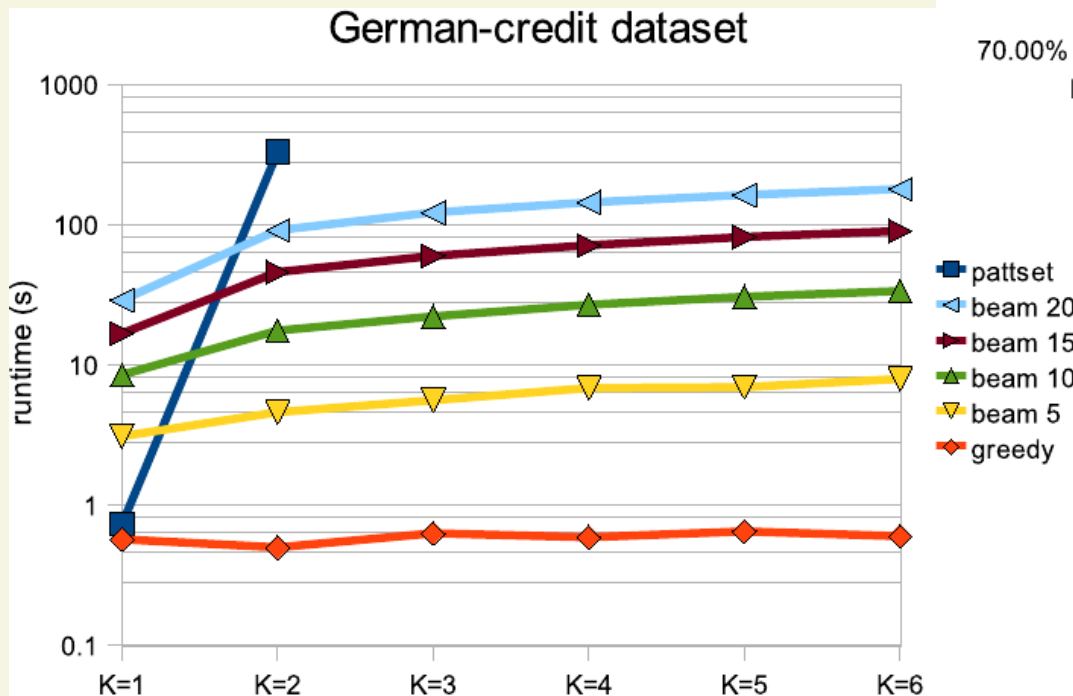
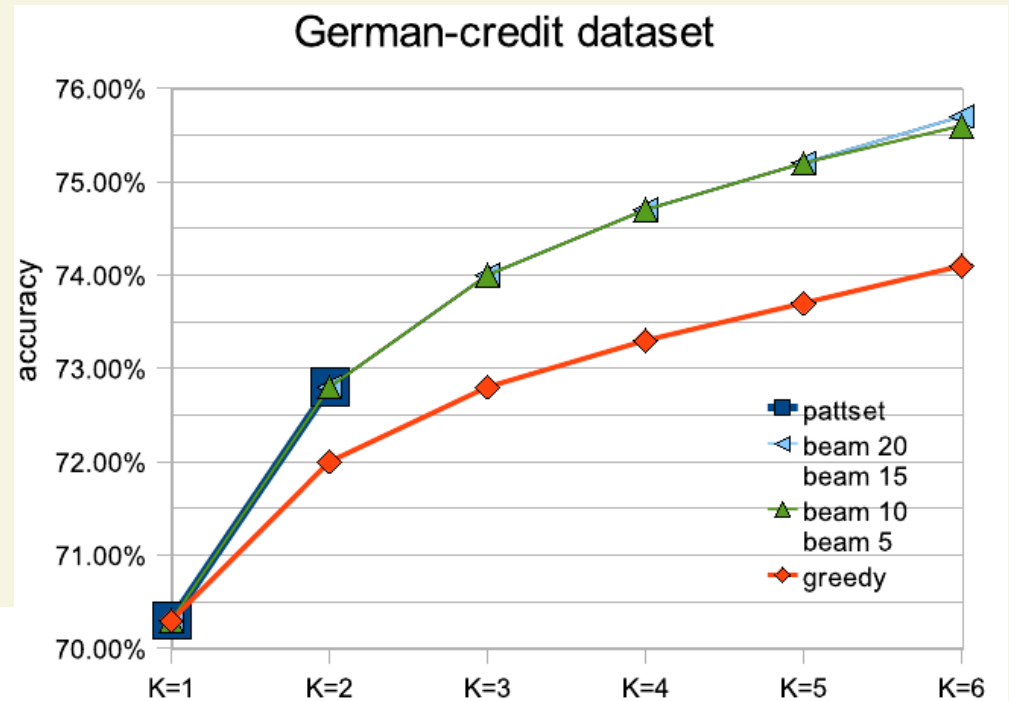


# Sequential covering, 2-step vs 1-step

Pattern rich data:



# 1-step, greedy vs. beam vs. exhaustive



# Overview

1. Motivation

2. Pattern Set Mining strategies

3. Constraint Programming framework

4. Case study

**5. Conclusions**

# Key points

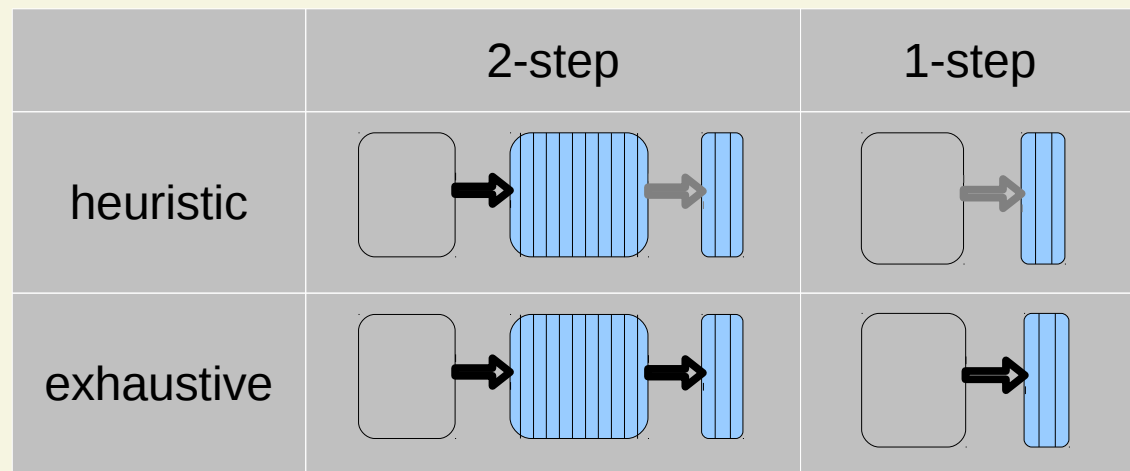
## Problem:

- Many strategies proposed (*for Pattern Set Mining*)
- No clear comparison possible

## Our contribution:

- I. Framework to compare strategies
- II. Case study on *concept learning* task

# Conclusions, case study



Each method has strong and weak points

- exhaustive: 1-step: optimality guarantees!  
unsurprisingly, low scalability
- 1-step heuristic: no thresholds!  
best runtime/accuracy trade-off  
use beam to adapt trade-off

# Conclusions, framework

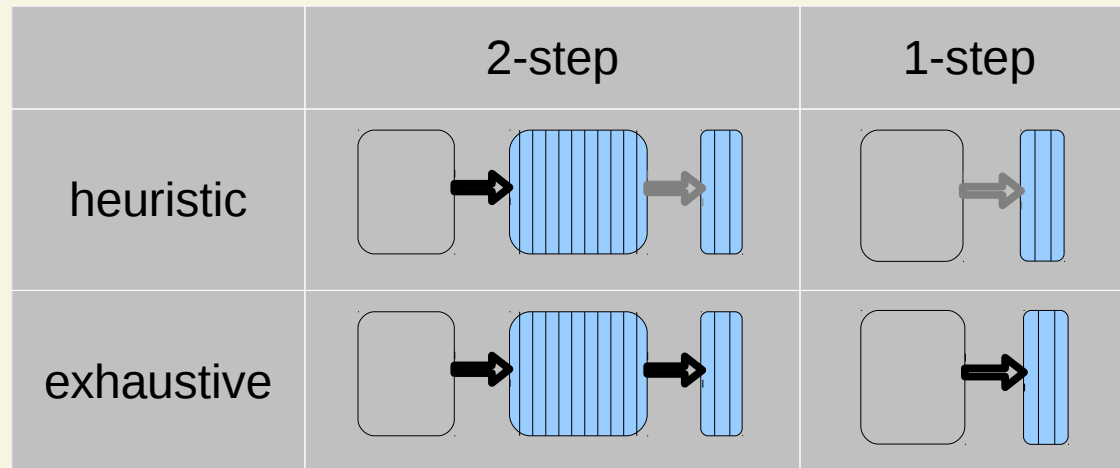
Constraint Programming framework:

- Very general (even 1-step exhaustive)
- Can encompass many strategies
- Well-suited for strategy evaluation

# Outlook

- More and different strategies
- Possible to do comparison of different measures in same framework
- Investigate other tasks

# Thank you for listening



Questions?