Declarative and Interactive pattern mining

Alexandre Termier

With slides from Guns/Nijssen/Negrevergne/Cremilleux/Plantevit/Soulet

Introduction

- Problem
  - Patterns should have interest for an analyst…
  - …but it is hard to get into the analyst’s head!

- Solution 1: don’t care, let the analyst do the work on input/output
- Solution 2: help the analyst ask a more relevant question
  - Declarative pattern mining
- Solution 3: integrate the analyst in the algorithmic loop
  - Interactive pattern mining

Declarative pattern mining

Declarative pattern mining ?

- One cause of pattern explosion: expected result is weakly defined
  - Most of the time: patterns that are *frequent*

- **Declarativity**: allow the user to express simply properties of output

  User should not write pattern mining code
  - Use of constraints -> constraint-based pattern mining
  - Use of measures -> skypatterns

- Difficulty: algorithmic performance depend on good understanding of pattern space characteristics
  - Custom definitions must respect properties allowing efficient mining

Constraint-based pattern mining

- **Idea:** a pattern P is interesting if it satisfies some constraints
  - Until now: we have mainly studied the constraint « P is *frequent* »
  - Other constraints exist: « P contains X », « Weight of P is above 20 units »,…

- Study different constraints
  - Can they be of practical use?
  - Are there algorithms to mine patterns satisfying them efficiently?

- => Led to the discovery of *classes of constraints*

Anti-monotone constraints

Definition: C anti-monotone: if $Q \subseteq P$, then $C(P) \Rightarrow C(Q)$

Ex: frequency is an anti-monotone constraint

Monotone constraints

Definition: C monotone: if $P \subseteq Q$, then $C(P) \Rightarrow C(Q)$

Ex: suppose that each item is associated with a price.

Then $C(P) = \text{sum}(P.\text{price}) \geq 500$ is monotone

Convertible constraints

Definition: C convertible

- anti-monotone: there exists an order of items such that if C(P), then C(Q) for Q a prefix of P
- monotone: there exists an order of items such that if ¬C(P), then ¬ C(Q) for Q a prefix of P

Ex: C(P) = avg(P.value) ≥ v is convertible anti-monotone, and avg(P.value) ≤ v is convertible monotone

here the order chosen is the decreasing order of values of items

Loose anti-monotone constraints

Definition: C loose anti-monotone: if C(P), then there exists e $\in$P s.t. C(P \ {e})

Ex: C(P) = var(X.value) ≤ v is loose anti-monotone

More constraints

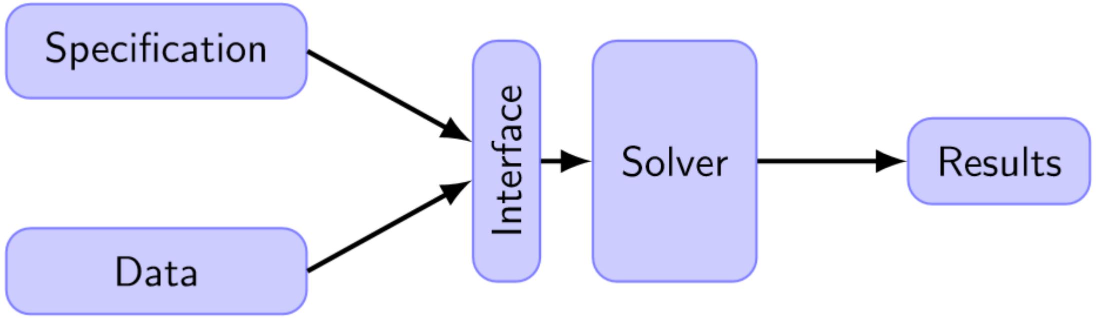
- All previous constraints led to dedicated mining algorithms

- Perhaps the most evolved approach is Music [Soulet et al, 06]:
  - Define a language of primitive constraints + combinations
  - And algorithm to mine queries in that language

- Flexibility limited by:
  - Categories of constraints envisioned by algorithms developpers
  - Algorithm efficiency
- How to be more flexible?

Back to the objectives of a declarative approach

Declarative approach



- Concise, extensible specification
- Independence: specification ⇔ solving mechanisms
  - Generic solvers
    SAT,CP,MIP...
  - Portfolio of specialised algorithms
  - Approximate or sampling solver
  - Solver specialised for a execution platform

Slide: B. Négrevergne

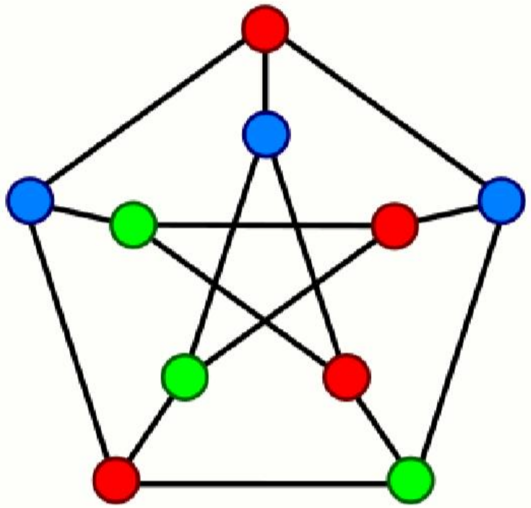
Constraint Programming (CP)

- CP = framework to solve complex combinatorial problems

- The problem is expressed declaratively through variables with domains and a set of constraints

- A **solver** takes the specification and outputs solutions (if any)
  - No code to write!

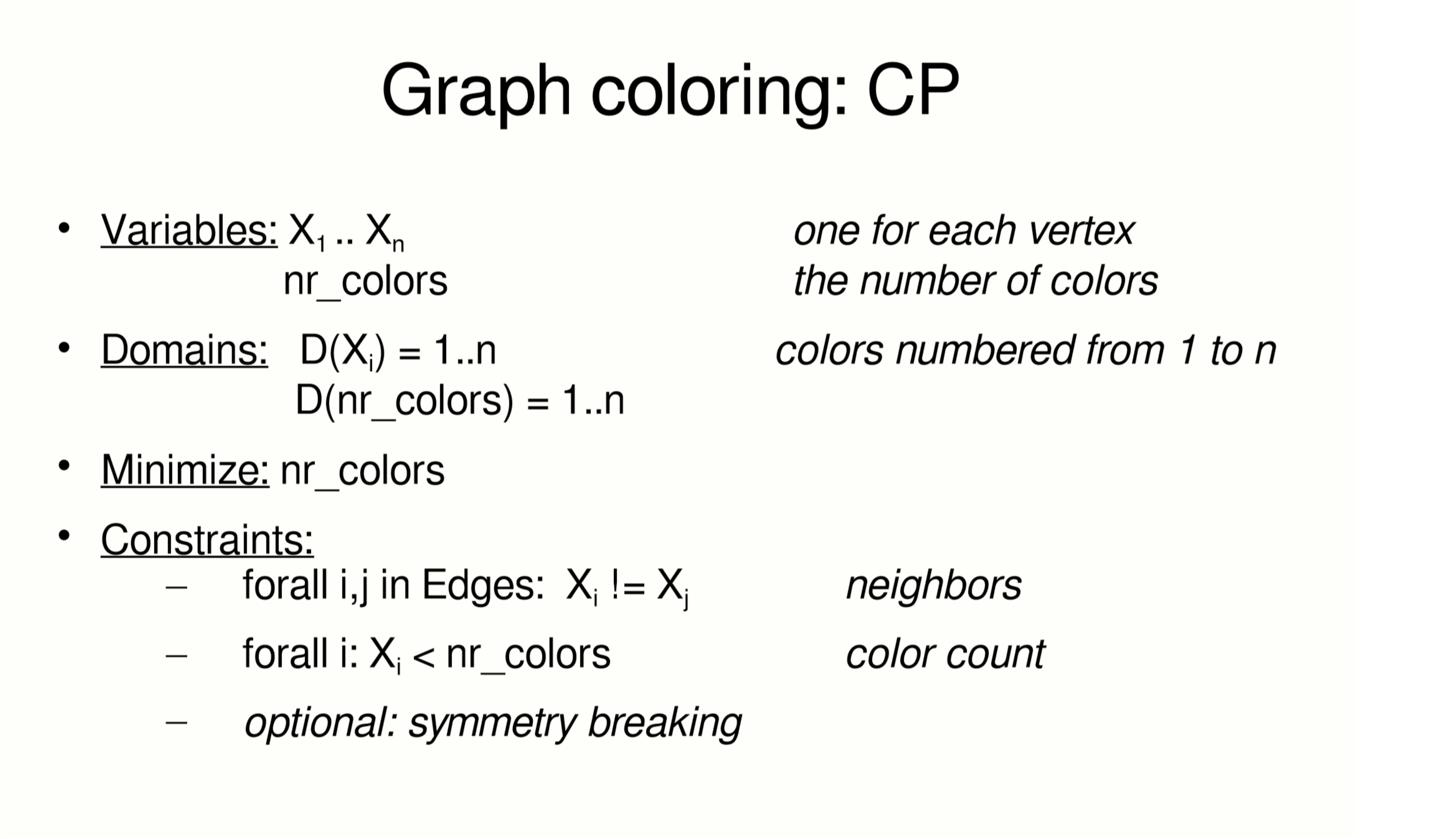
Modeling example: graph coloring



- Given: a graph G=(V,E) with vertices V and edges E
- Find: a coloring of vertices V with minimal nr. of colors

  such that all (v_1,v_2) in E: $\text{color}(v_1)$!= $\text{color}(v_2)$

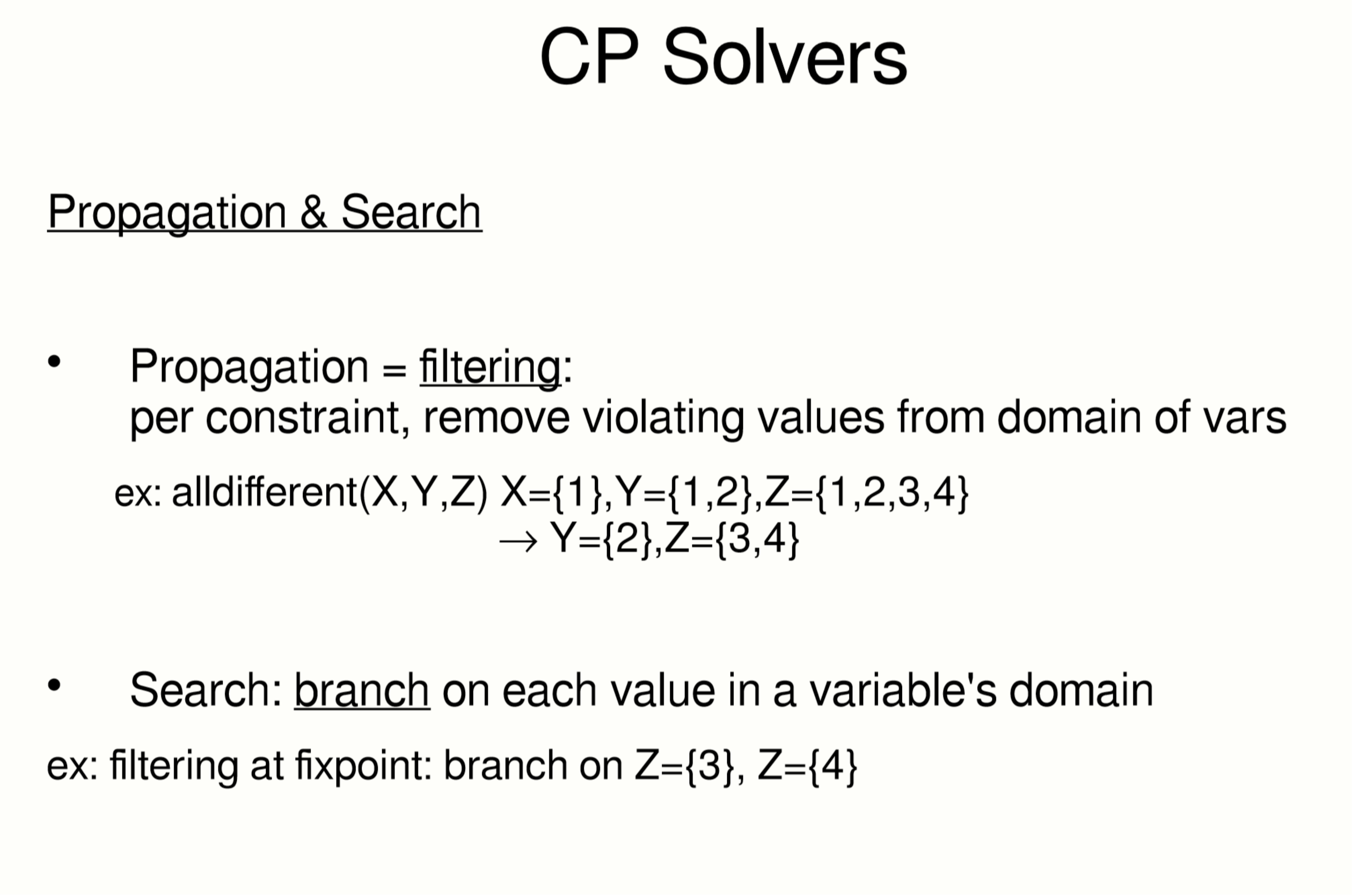
Slide: Guns, Nijssen et al.



Graph coloring: CP

- <u>Variables:</u> $X_1 .. X_n$ *one for each vertex*
  nr_colors *the number of colors*
- <u>Domains:</u> $D(X_i) = 1..n$ *colors numbered from 1 to n*
  D(nr_colors) = 1..n
- <u>Minimize:</u> nr_colors
- <u>Constraints:</u>
  - forall i,j in Edges: X_i != X_j *neighbors*
  - forall i: X_i < nr_colors *color count*
  - *optional: symmetry breaking*

Slide: Guns, Nijssen et al.



CP Solvers

Propagation & Search

- Propagation = filtering:
  per constraint, remove violating values from domain of vars

ex: alldifferent(X,Y,Z) X={1},Y={1,2},Z={1,2,3,4}
→ Y={2},Z={3,4}

- Search: branch on each value in a variable's domain

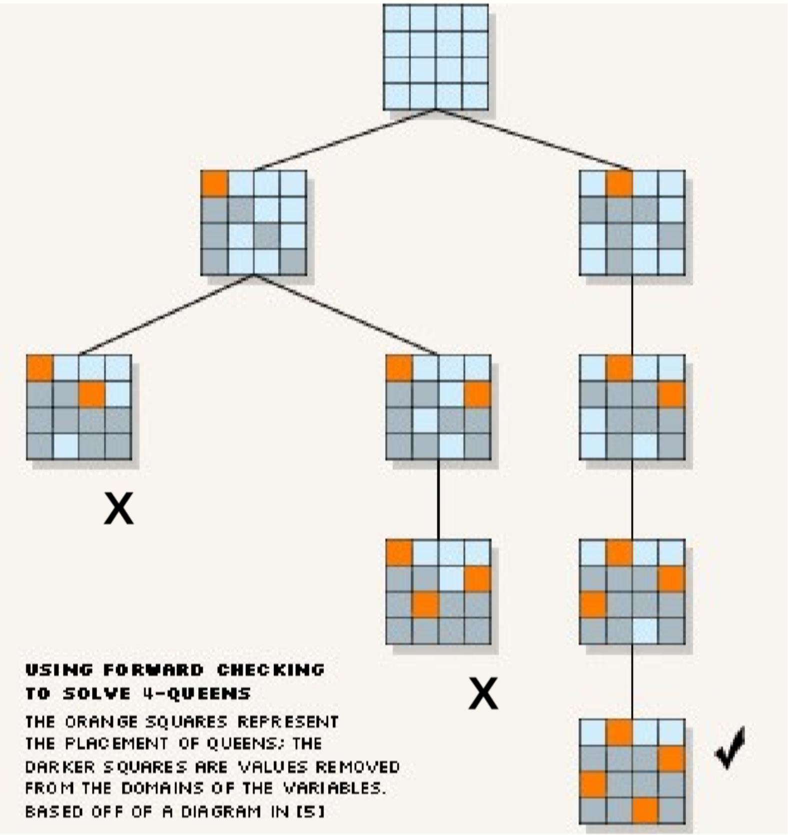
ex: filtering at fixpoint: branch on Z={3}, Z={4}

Slide: Guns, Nijssen et al.

CP: propagation and search

N-queens: one queen per row, queens can not attack each other

search tree (Boolean variables) and propagation (gray cells):


X
X
✓
USING FORWARD CHECKING
TO SOLVE 4-QUEENS
THE ORANGE SQUARES REPRESENT
THE PLACEMENT OF QUEENS; THE
DARKER SQUARES ARE VALUES REMOVED
FROM THE DOMAINS OF THE VARIABLES.
BASED OFF OF A DIAGRAM IN [5]

Slide: Guns, Nijssen et al.

Modeling Frequent Itemset Mining

One Boolean variable per item

One Boolean variable per transaction

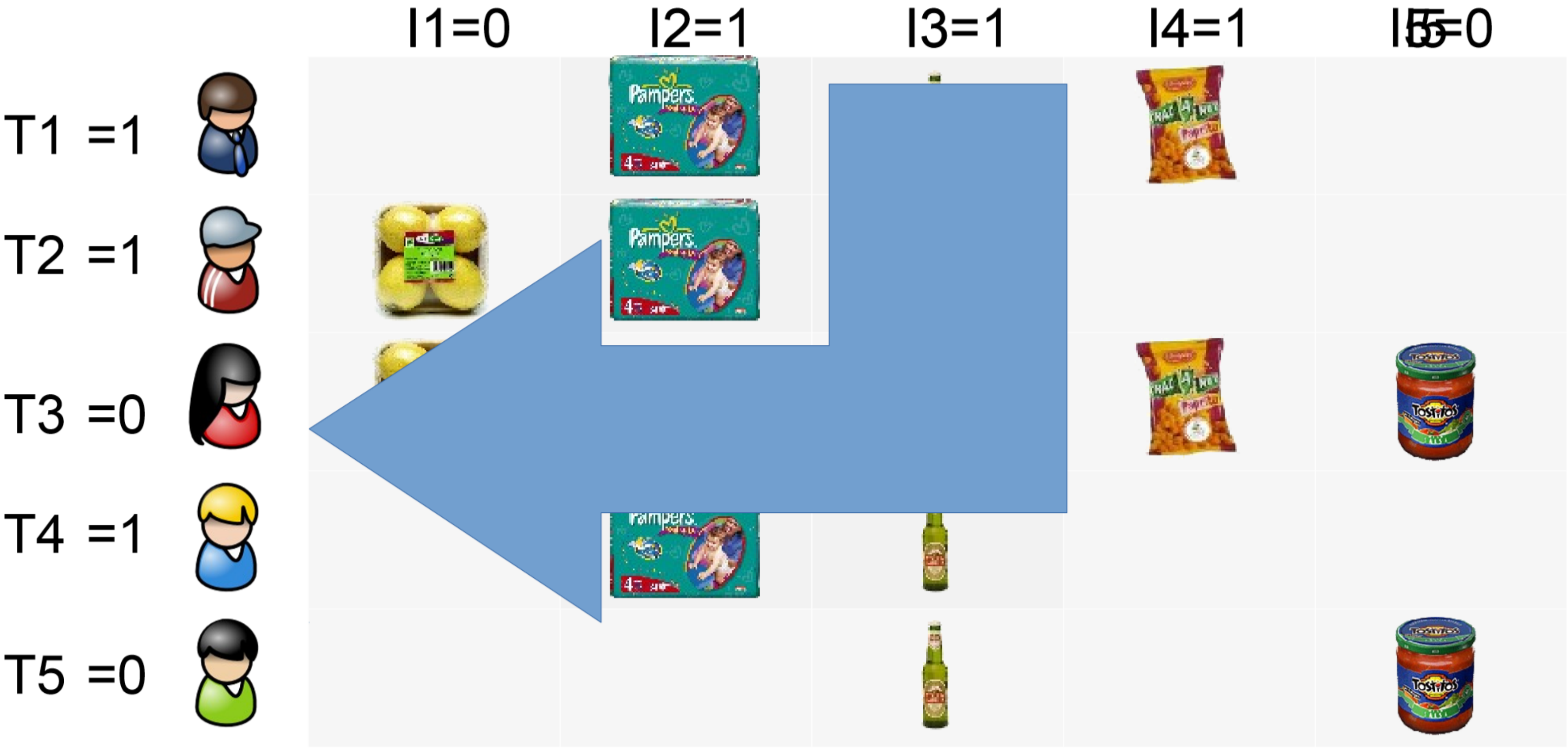
	I1	I2	I3	I4	I5 $\in \{0, 1\}$
T1					
T2					
T3					
T4					
T5					

Slide: Guns, Nijssen et al.

Modeling Frequent Itemset Mining

Two constraints:

1) A coverage constraint: (Tj = 1) iff (Itemset in Tj)



I1=0 I2=1 I3=1 I4=1 I5=0

T1 =1

T2 =1

T3 =0

T4 =1

T5 =0

Slide: Guns, Nijssen et al.

Modeling Frequent Itemset Mining

Two constraints:

2) A <u>support constraint</u>: $\sum_{j} Tj \geq \mathit{minsup}$

I1=0 I2=1 I3=1 I4=1 I5=0

T1 =1

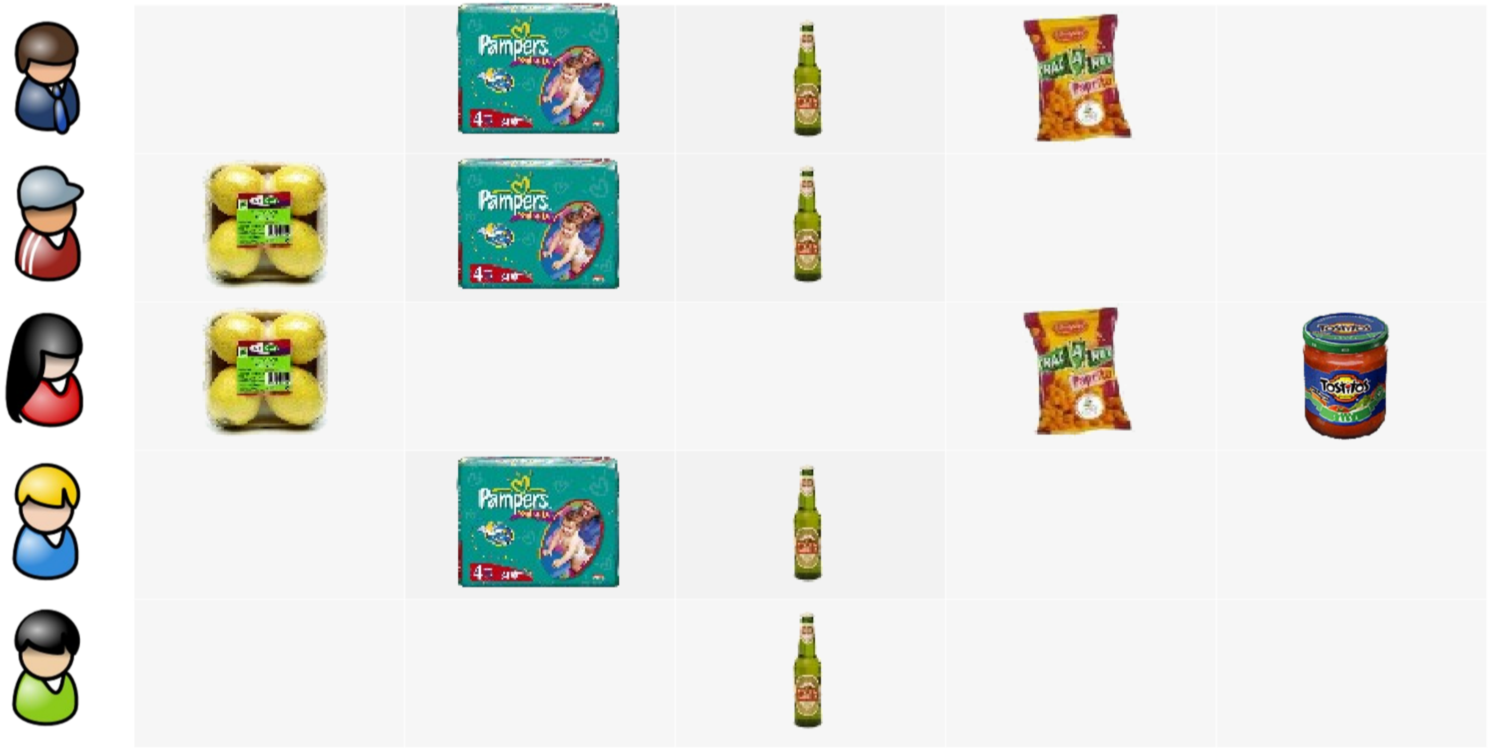
T2 =1

T3 =0

T4 =1

T5 =0

3



Slide: Guns, Nijssen et al.

FIM, MiniZinc

```
array[1..NrT] of set of int: TDB;
int: NrI; int: NrT; int: MinFreq;

var set of 1..NrI: Items;
constraint card(cover(Items, TDB)) >= MinFreq;

array [1..NrI] of int: Cost;
int: MinCost;

constraint sum(i in Items) (Cost[i]) >= MinCost

solve satisfy;
```

Slide: B. Negrevergne

Conclusion on CP + pattern mining

- New problems can be easily formulated by adding new constraints
- Also studied for mining:
  - Sequences
  - Pattern sets

- Several work to:
  - Exploit existing solvers [pb: slow]
  - Make minimal additions to solvers to be more efficient in pattern mining [Nijssen et al., 2010; Maamar et al., 2016; Schaus et al., 2017]

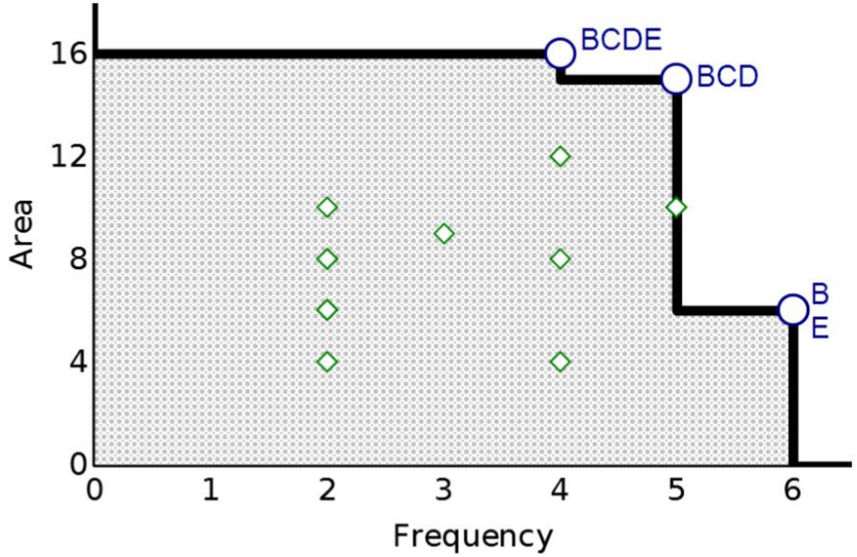
- Problems:
  - Performance (many work on that)
  - CP framework not natural for many data owners -> need a « clean » way to hide it

Skypatterns (Pareto dominance)

Notion of skylines (database) in pattern mining (Cho at al. IJDWM05, Papadopoulos et al. DAMI08, Soulet et al. ICDM11, van Leeuwen and Ukkonen ECML/PKDD13)

Tid	Items					
t_1		B			E	F
t_2		B	C	D		
t_3	A				E	F
t_4	A	B	C	D	E	
t_5		B	C	D	E	
t_6		B	C	D	E	F
t_7	A	B	C	D	E	F

Patterns	**freq**	**area**
~~*AB*~~	2	4
~~*AEF*~~	2	6
B	6	6
BCDE	4	16
~~*CDEF*~~	2	8
E	6	6
⋮	⋮	⋮



$|\mathcal{L}_{\mathcal{I}}| = 2^6$, but only 4 skypatterns

$$Sky(\mathcal{L}_{\mathcal{I}}, \{freq, area\}) = \{BCDE, BCD, B, E\}$$

Slide: Plantevit, Crémilleux, Soulet

Skypatterns: how to process?

A naive enumeration of all candidate patterns ($\mathcal{L}_{\mathcal{I}}$) and then comparing them is not feasible. . .

Two approaches:

1. take benefit from the pattern condensed representation according to the condensable measures of the given set of measures M
   - skylineability to obtain M' ($M' \subseteq M$) giving a more concise pattern condensed representation
   - the pattern condensed representation w.r.t. M' is a superset of the representative skypatterns w.r.t. M which is (much smaller) than $\mathcal{L}_{\mathcal{I}}$.

2. use of the dominance programming framework (together with skylineability)

Slide: Plantevit, Crémilleux, Soulet

Dominance programming

Dominance: a pattern is optimal if it is not dominated by another.
Skypatterns: dominance relation = Pareto dominance

1. **Principle:**
   - starting from an initial pattern s_1
   - searching for a pattern s_2 such that s_1 is not preferred to s_2
   - searching for a pattern s_3 such that s_1 and s_2 are not preferred to s_3

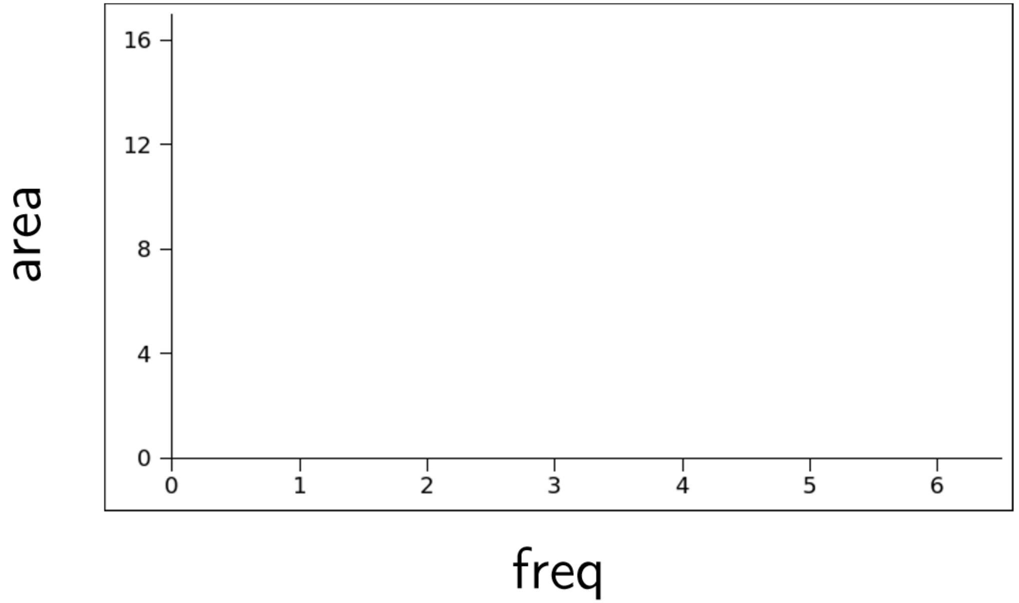
     $\vdots$
   - until there is no pattern satisfying the whole set of constraints
2. **Solving:**
   - constraints are dynamically posted during the mining step

Principle: increasingly reduce the dominance area by processing pairwise comparisons between patterns. Methods using Dynamic CSP (Negrevergne et al. ICDM13, Ugarte et al. CPAIOR14, AIJ 2017).

Slide: Plantevit, Crémilleux, Soulet

Dominance programming: example of the skypatterns

Trans.	Items					
t_1		B			E	F
t_2		B	C	D		
t_3	A				E	F
t_4	A	B	C	D	E	
t_5		B	C	D	E	
t_6		B	C	D	E	F
t_7	A	B	C	D	E	F



$$M = \{freq, area\}$$

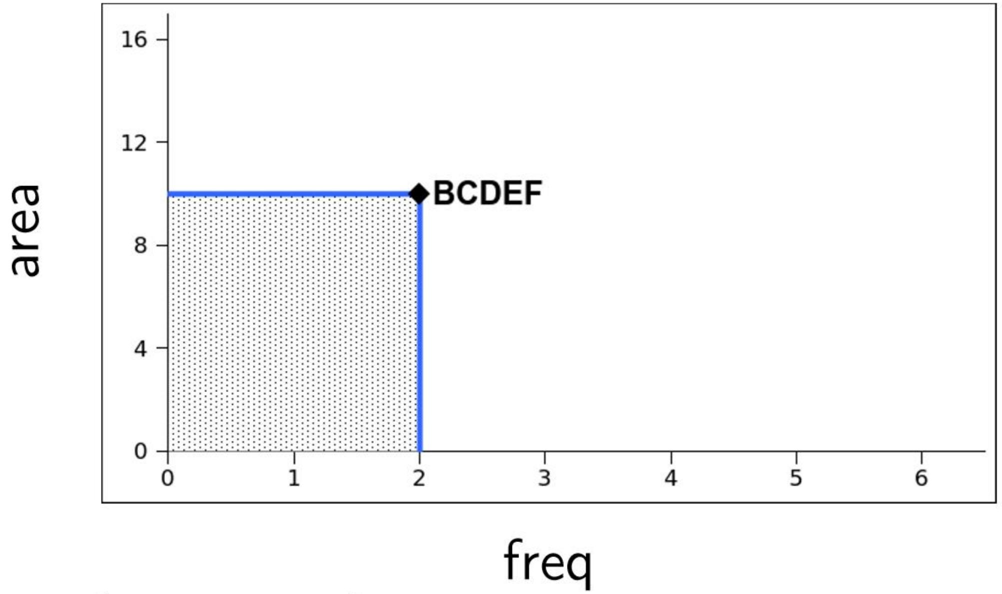
$$q(X) \equiv closed_{M'}(X)$$

$$Candidates =$$

Slide: Plantevit, Crémilleux, Soulet

Dominance programming: example of the skypatterns

Trans.	Items					
t_1		B			E	F
t_2		B	C	D		
t_3	A				E	F
t_4	A	B	C	D	E	
t_5		B	C	D	E	
t_6		B	C	D	E	F
t_7	A	B	C	D	E	F



$$M = \{freq, area\}$$

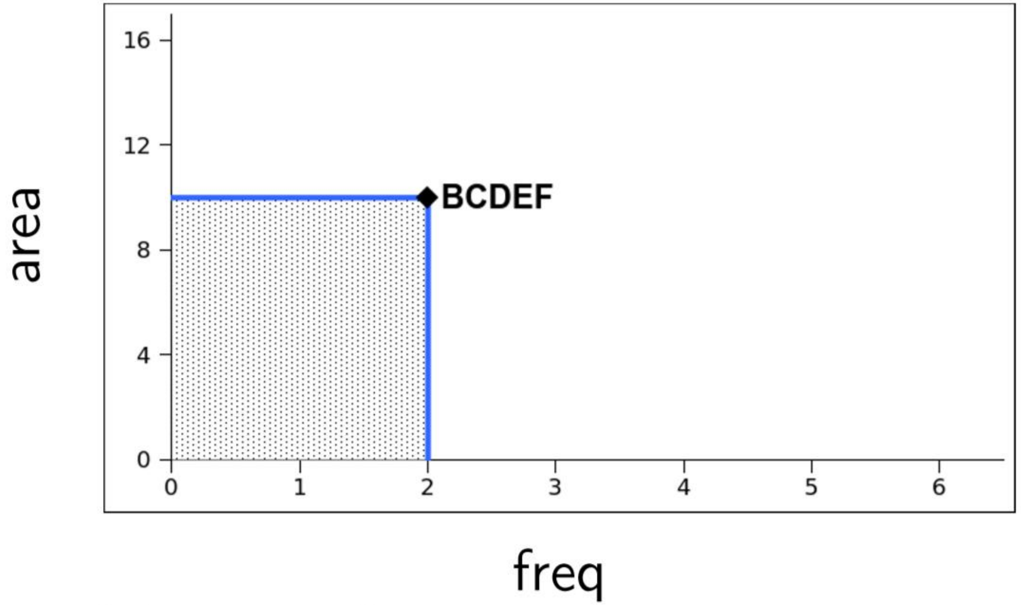
$$q(X) \equiv closed_{M'}(X)$$

$$Candidates = \{\underbrace{\text{BCDEF}}_{s_1},$$

Slide: Plantevit, Crémilleux, Soulet

Dominance programming: example of the skypatterns

Trans.	Items					
t_1		B			E	F
t_2		B	C	D		
t_3	A				E	F
t_4	A	B	C	D	E	
t_5		B	C	D	E	
t_6		B	C	D	E	F
t_7	A	B	C	D	E	F



$$M = \{freq, area\}$$

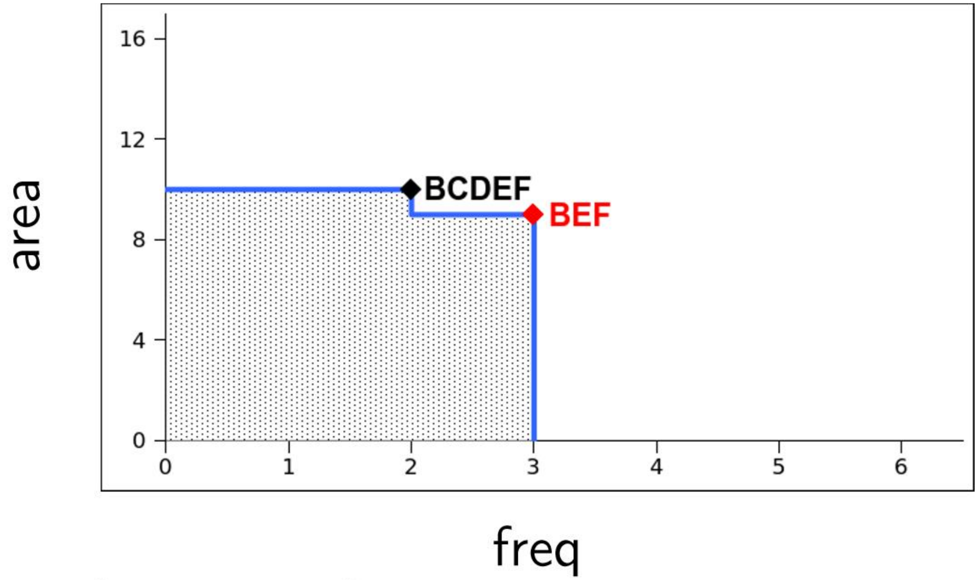
$$q(X) \equiv closed_{M'}(X) \wedge \neg(s_1 \succ_M X)$$

$$Candidates = \{\underbrace{\text{BCDEF}}_{s_1},$$

Slide: Plantevit, Crémilleux, Soulet

Dominance programming: example of the skypatterns

Trans.	Items					
t_1		B			E	F
t_2		B	C	D		
t_3	A				E	F
t_4	A	B	C	D	E	
t_5		B	C	D	E	
t_6		B	C	D	E	F
t_7	A	B	C	D	E	F



$$M = \{freq, area\}$$

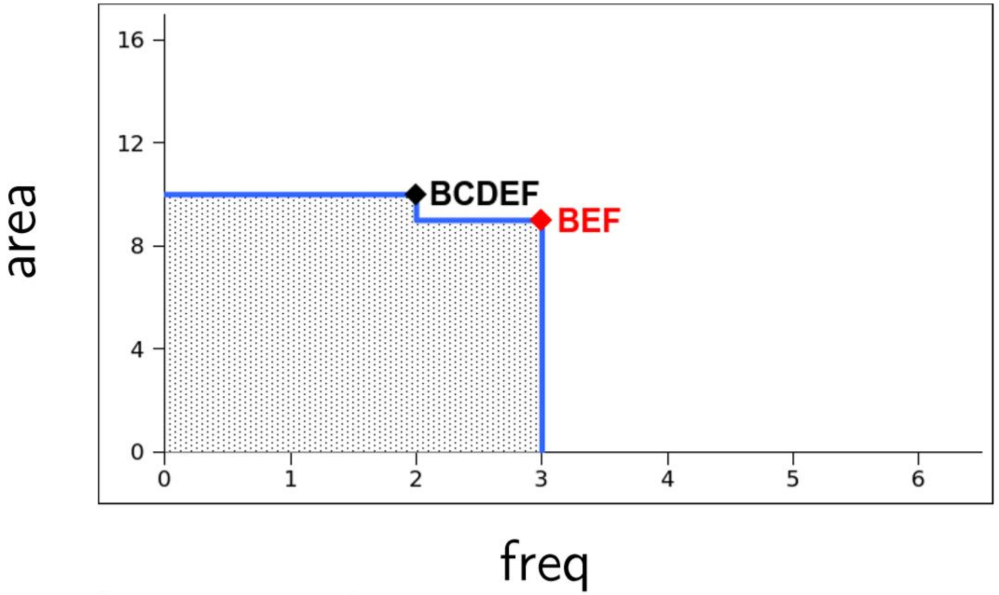
$$q(X) \equiv closed_{M'}(X) \wedge \neg(s_1 \succ_M X)$$

$$Candidates = \{\underbrace{\text{BCDEF}}_{s_1}, \underbrace{\text{BEF}}_{s_2},$$

Slide: Plantevit, Crémilleux, Soulet

Dominance programming: example of the skypatterns

Trans.	Items					
t_1		B			E	F
t_2		B	C	D		
t_3	A				E	F
t_4	A	B	C	D	E	
t_5		B	C	D	E	
t_6		B	C	D	E	F
t_7	A	B	C	D	E	F



$$M = \{freq, area\}$$

$$q(X) \equiv closed_{M'}(X) \wedge \neg(s_1 \succ_M X) \wedge \neg(s_2 \succ_M X)$$

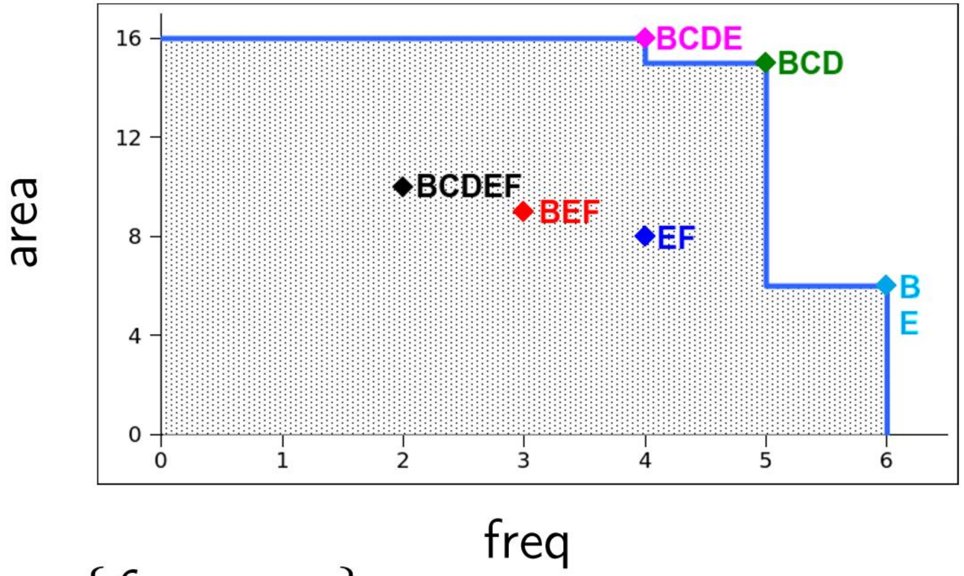
$$Candidates = \{\underbrace{BCDEF}_{s_1}, \underbrace{BEF}_{s_2},$$

Slide: Plantevit, Crémilleux, Soulet

Dominance programming: example of the skypatterns

Trans.	Items					
t_1		B			E	F
t_2		B	C	D		
t_3	A				E	F
t_4	A	B	C	D	E	
t_5		B	C	D	E	
t_6		B	C	D	E	F
t_7	A	B	C	D	E	F

$| \mathcal{L}_\mathcal{I} | = 2^6 = 64$ patterns
4 skypatterns



$$M = \{freq, area\}$$

$$q(X) \equiv closed_{M'}(X) \wedge \neg(s_1 \succ_M X) \wedge \neg(s_2 \succ_M X) \wedge \neg(s_3 \succ_M X) \wedge \neg(s_4 \succ_M X) \wedge \neg(s_5 \succ_M X) \wedge \neg(s_6 \succ_M X) \wedge \neg(s_7 \succ_M X)$$

$$Candidates = \{\underbrace{\cancel{BCDEF}}_{s_1}, \underbrace{\cancel{BEF}}_{s_2}, \underbrace{\cancel{EF}}_{s_3}, \underbrace{\underbrace{BCDE}_{s_4}, \underbrace{BCD}_{s_5}, \underbrace{B}_{s_6}, \underbrace{E}_{s_7}}_{\text{Sky}(\mathcal{L}_\mathcal{I}, M)}\}$$

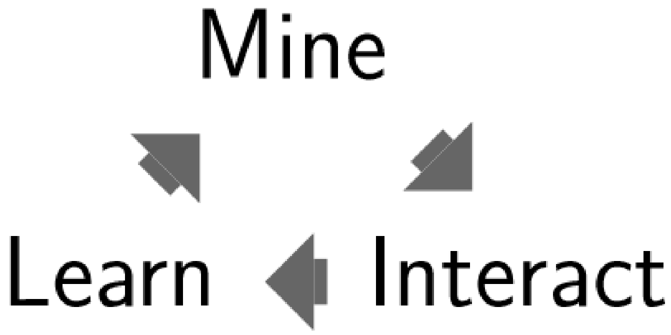
Slide: Plantevit, Crémilleux, Soulet

Interactive pattern mining

Main idea

- Users often do not know in advance what they are looking for
  - -> impossible to write a proper specification => ~~declarative approaches~~
- But if they see what they want, they can recognize it

- Goal of interactive approaches is to progressively learn “user preferences”
  - Show patterns to users
  - Get feedback
  - Learn what the users want to see



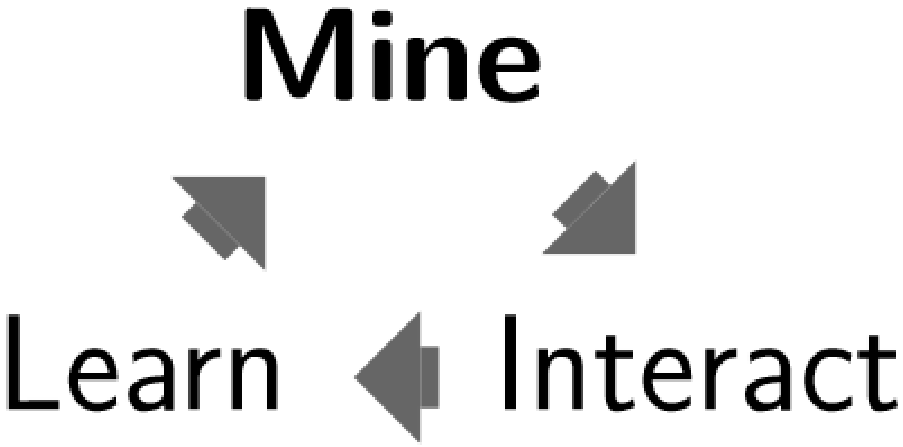
Mine

Learn

Interact

Interactive pattern mining: overview

Interactive data exploration using pattern mining. (van Leeuwen 2014)



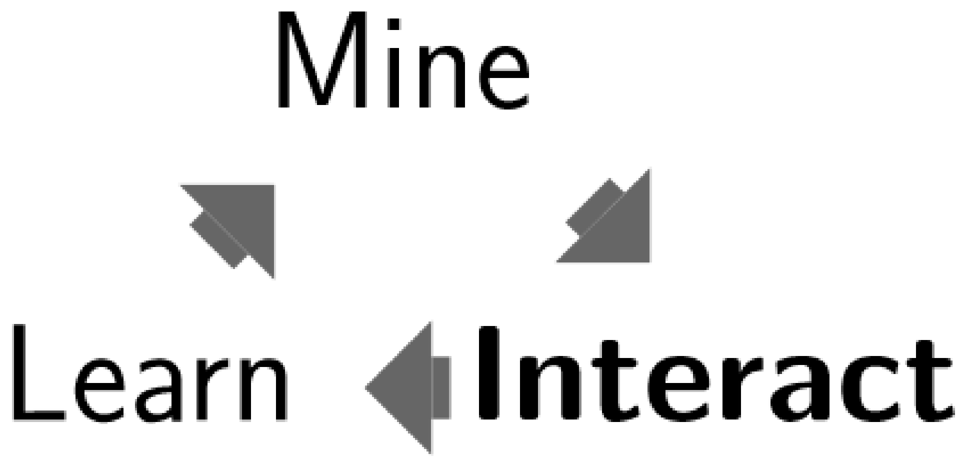
Mine

- Provide a sample of k patterns to the user (called the query $\mathcal{Q}$)

Slide: Plantevit, Crémilleux, Soulet

Interactive pattern mining: overview

Interactive data exploration using pattern mining. (van Leeuwen 2014)



Mine

Learn

Interact

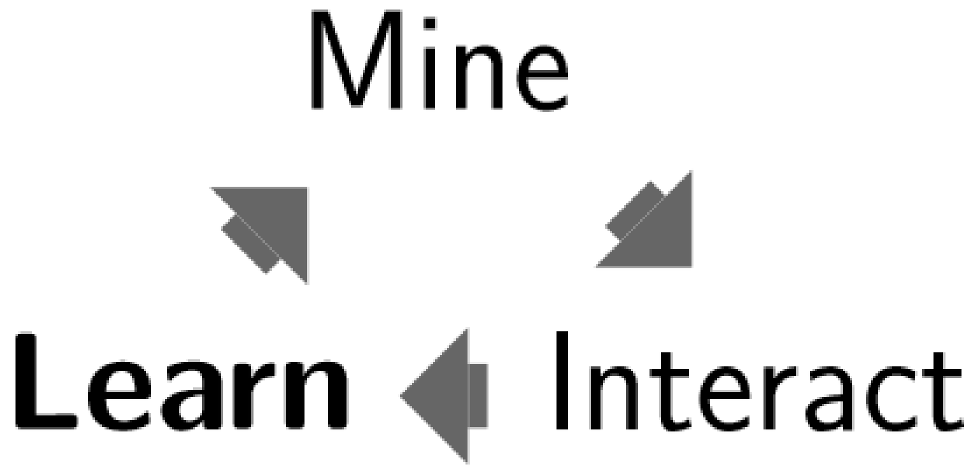
Interact

- Like/dislike or rank or rate the patterns

Slide: Plantevit, Crémilleux, Soulet

Interactive pattern mining: overview

Interactive data exploration using pattern mining. (van Leeuwen 2014)



Mine

Learn

Interact

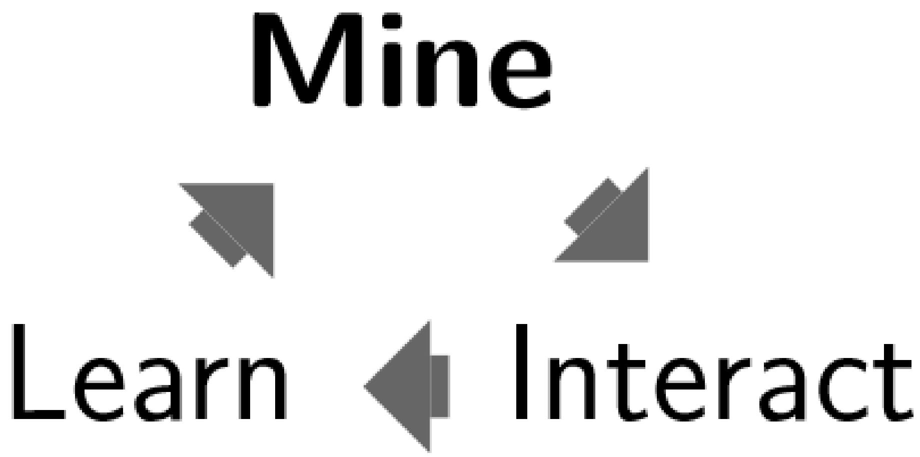
Learn

- Generalize user feedback for building a preference model

Slide: Plantevit, Crémilleux, Soulet

Interactive pattern mining: overview

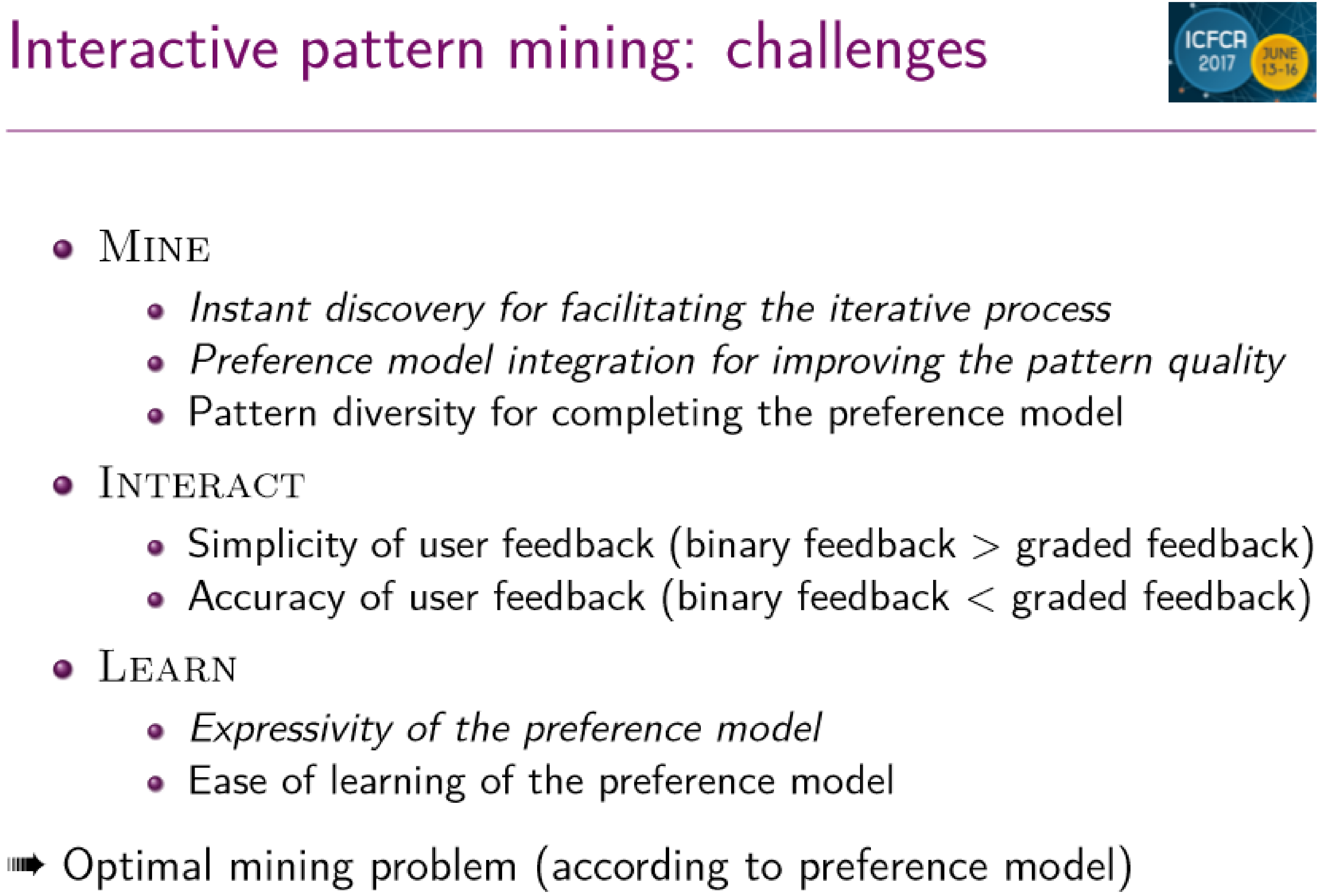
Interactive data exploration using pattern mining. (van Leeuwen 2014)



Mine (again!)

- Provide a sample of k patterns **benefiting from the preference model**

Slide: Plantevit, Crémilleux, Soulet



Interactive pattern mining: challenges

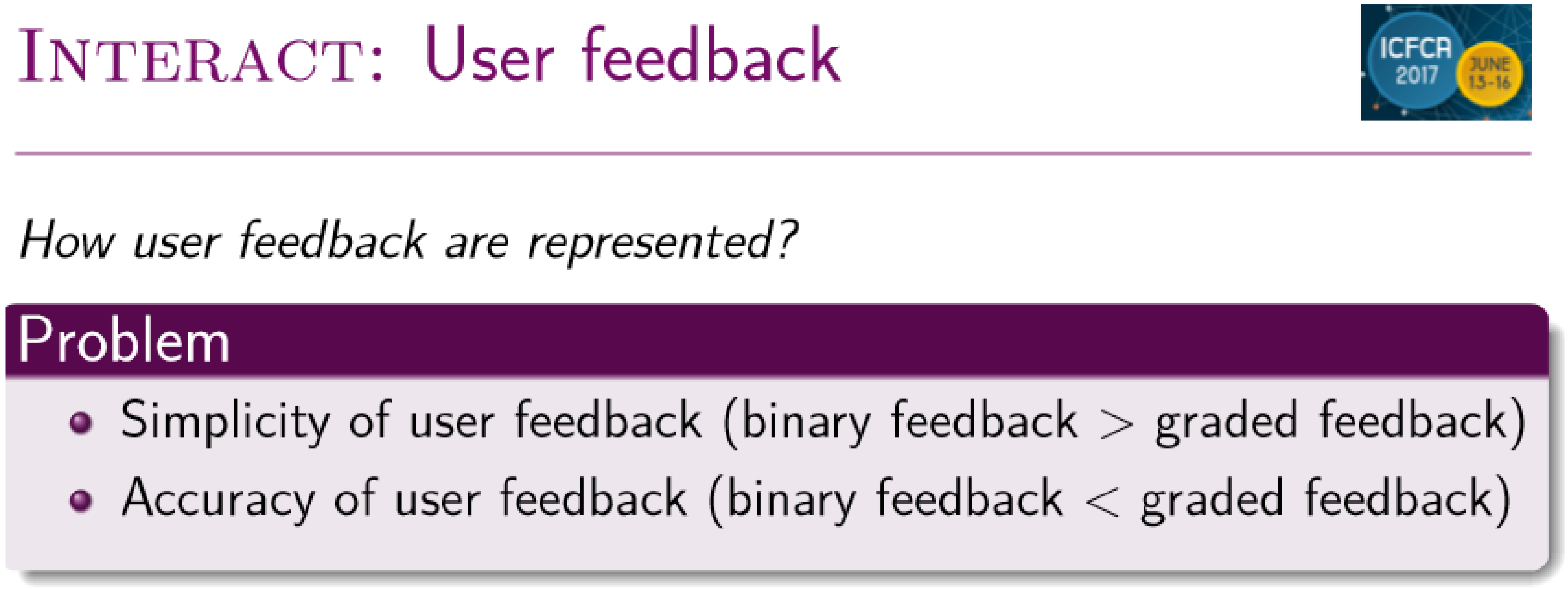
- MINE
  - *Instant discovery for facilitating the iterative process*
  - *Preference model integration for improving the pattern quality*
  - Pattern diversity for completing the preference model
- INTERACT
  - Simplicity of user feedback (binary feedback > graded feedback)
  - Accuracy of user feedback (binary feedback < graded feedback)
- LEARN
  - *Expressivity of the preference model*
  - Ease of learning of the preference model

➠ Optimal mining problem (according to preference model)

Slide: Plantevit, Crémilleux, Soulet

Learn: preference model

- Pattern-based preference model ()
  - Model based on the elements constituting the patterns
  - Ex: weigthed product model [Bhuiyan et al., 2012 ; Dzyuba et al., 2017]
    - Learn weights on items
    - Score of pattern = product of weitghts of its items
  - Ex: feature space model [Xin et al., 2006 ; Dzyuba et al., 2013]
    - Set of features for patterns (ex: attributes, coverage, lenght)
    - Weights learned on this features
- Algorithm-based preference model
  - For approaches combining multiple algorithms [Boley et al., 2013]
  - Learn which algorithm produces the pattern most liked by users (-> weights)



INTERACT: User feedback

How user feedback are represented?

Problem

- Simplicity of user feedback (binary feedback > graded feedback)
- Accuracy of user feedback (binary feedback < graded feedback)

Slide: Plantevit, Crémilleux, Soulet

Interact: User feedback

ICFCA 2017 JUNE 13-16

How user feedback are represented?

Problem

- Simplicity of user feedback (binary feedback > graded feedback)
- Accuracy of user feedback (binary feedback < graded feedback)

Weighted product model

- Binary feedback (like/dislike) (Bhuiyan et al. CIKM12, Dzyuba et al. PAKDD17)

pattern	feedback
A	like
AB	like
BC	dislike

Slide: Plantevit, Crémilleux, Soulet

INTERACT: User feedback

How user feedback are represented?

Problem

- Simplicity of user feedback (binary feedback > graded feedback)
- Accuracy of user feedback (binary feedback < graded feedback)

Feature space model

- Ordered feedback (ranking) (Xin et al. KDD06, Dzyuba et al. ICTAI13)

$$A \succ AB \succ BC$$

- Graded feedback (rate) (Rueping ICML09)

pattern	feedback
A	0.9
AB	0.6
BC	0.2

Slide: Plantevit, Crémilleux, Soulet

LEARN: Preference learning method

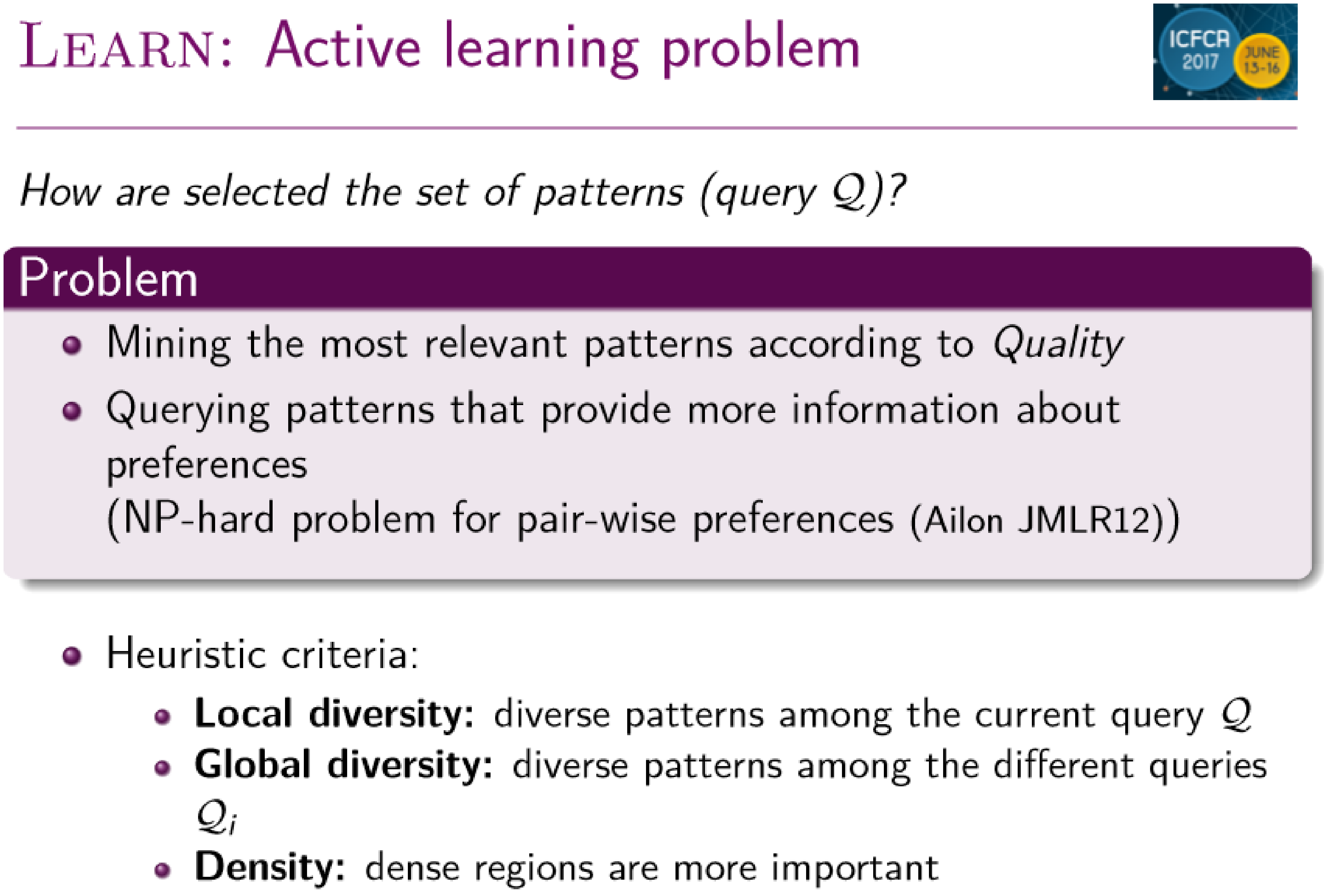
How user feedback are generalized to a model?

- **Weighted product model**
  - Counting likes and dislikes for each item: $\omega = \beta^{(\text{\#like - \#dislike})}$ (Bhuiyan et al. ICML12, Dzyuba et al. PAKDD17)

pattern	feedback	A	B	C
A	like	1		
AB	like	1	1	
BC	dislike		-1	-1
		$2^{2-0} = 4$	$2^{1-1} = 1$	$2^{0-1} = 0.5$

- **Feature space model**
  - $=$ learning to rank (Rueping ICML09, Xin et al. KDD06, Dzyuba et al. ICTAI13)

Slide: Plantevit, Crémilleux, Soulet



Learn: Active learning problem

How are selected the set of patterns (query $\mathcal{Q}$)?

Problem

- Mining the most relevant patterns according to *Quality*
- Querying patterns that provide more information about preferences
  (NP-hard problem for pair-wise preferences (Ailon JMLR12))

- Heuristic criteria:
  - **Local diversity:** diverse patterns among the current query $\mathcal{Q}$
  - **Global diversity:** diverse patterns among the different queries $\mathcal{Q}_i$
  - **Density:** dense regions are more important

Slide: Plantevit, Crémilleux, Soulet

LEARN: Active learning heuristics

(Dzyuba et al. ICTAI13)

What is the interest of the pattern X for the current pattern query $\mathcal{Q}$?

- **Maximal Marginal Relevance:** querying diverse patterns in $\mathcal{Q}$

$$\alpha \mathit{Quality}(X) + (1-\alpha) \min_{Y \in \mathcal{Q}} \mathit{dist}(X, Y)$$

- **Global MMR:** taking into account previous queries

$$\alpha \mathit{Quality}(X) + (1-\alpha) \min_{Y \in \bigcup_i \mathcal{Q}_i} \mathit{dist}(X, Y)$$

- **Relevance, Diversity, and Density:** querying patterns from dense regions provides more information about preferences

$$\alpha \mathit{Quality}(X) + \beta \mathit{Density}(X) + (1-\alpha-\beta) \min_{Y \in \mathcal{Q}} \mathit{dist}(X, Y)$$

Slide: Plantevit, Crémilleux, Soulet

Mine: Mining strategies

What method is used to mine the pattern query $\mathcal{Q}$?

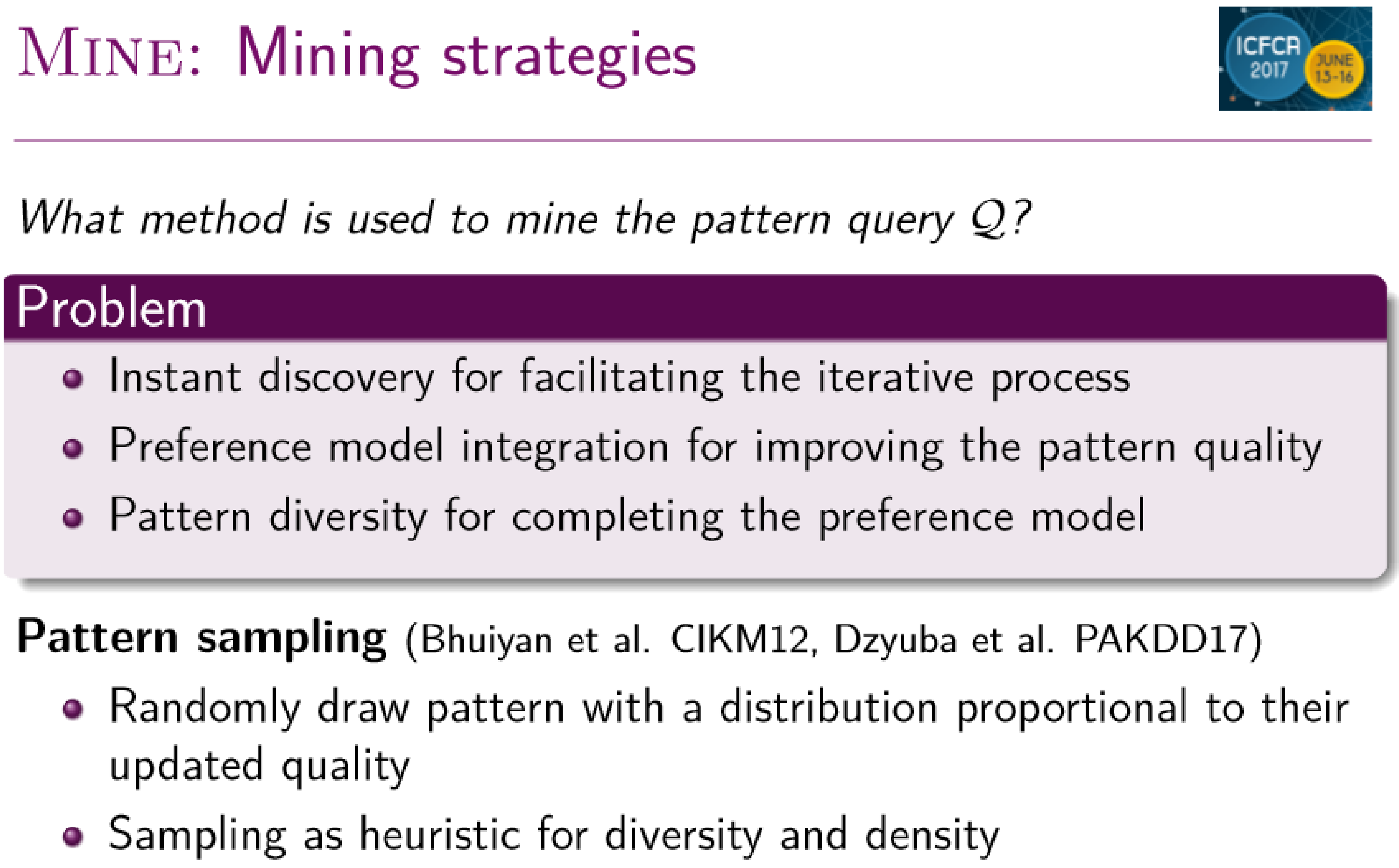
Problem

- Instant discovery for facilitating the iterative process
- Preference model integration for improving the pattern quality
- Pattern diversity for completing the preference model

Optimal pattern mining (Dzyuba et al. ICTAI13)

- Beam search based on reweighing subgroup quality measures for finding the best patterns
- Previous active learning heuristics (and more)

Slide: Plantevit, Crémilleux, Soulet



MINE: Mining strategies

What method is used to mine the pattern query $\mathcal{Q}$?

Problem

- Instant discovery for facilitating the iterative process
- Preference model integration for improving the pattern quality
- Pattern diversity for completing the preference model

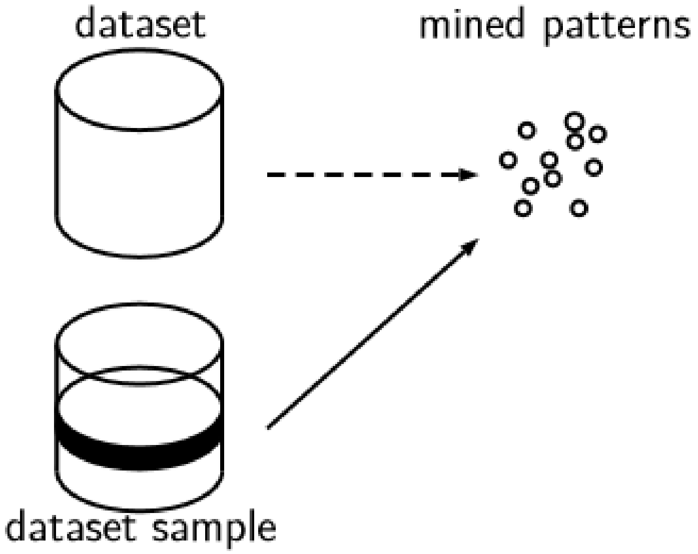
Pattern sampling (Bhuiyan et al. CIKM12, Dzyuba et al. PAKDD17)

- Randomly draw pattern with a distribution proportional to their updated quality
- Sampling as heuristic for diversity and density

Slide: Plantevit, Crémilleux, Soulet

Dataset sampling vs Pattern sampling

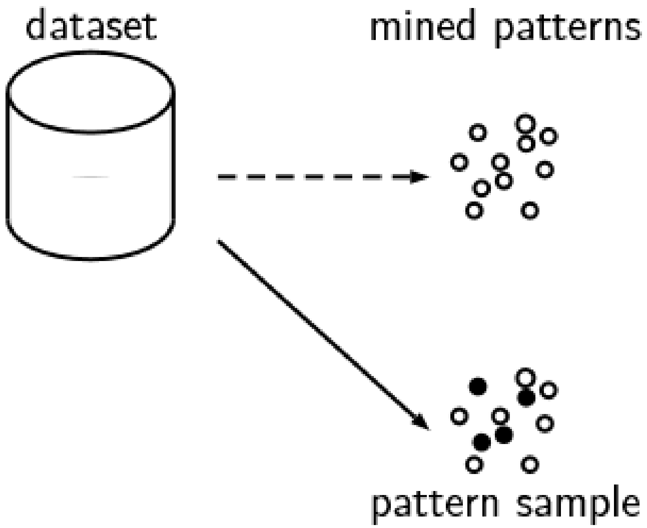
Dataset sampling



Finding all patterns from a transaction sample

➠ input space sampling

Pattern sampling



Finding a pattern sample from all transactions

➠ output space sampling

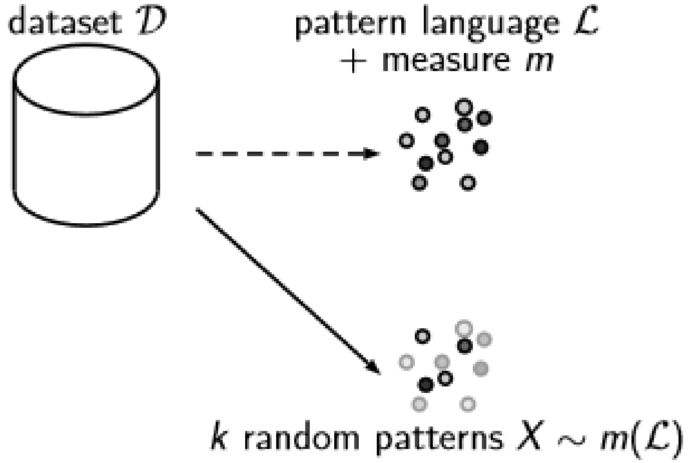
Slide: Plantevit, Crémilleux, Soulet

Pattern sampling: Problem

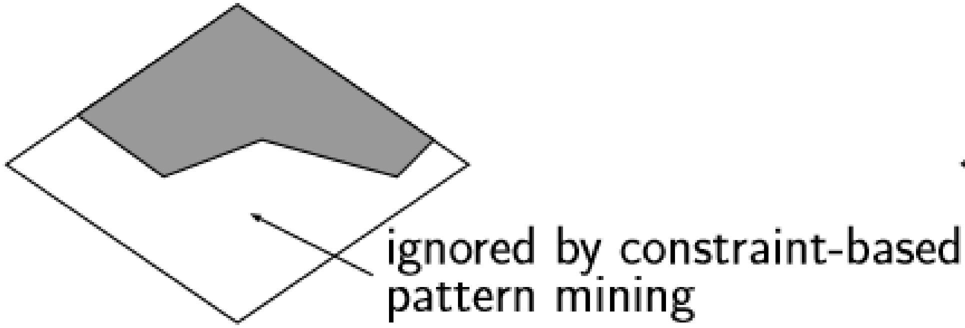
ICFCA 2017 JUNE 13-16

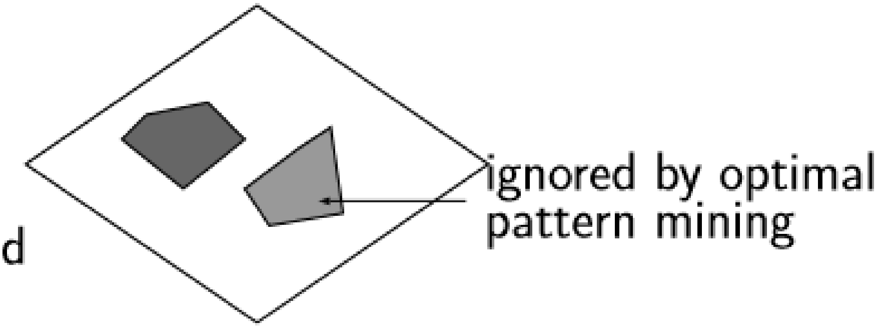
Problem

- **Inputs:** a pattern language $\mathcal{L}$ + a measure $m : \mathcal{L} \rightarrow \Re$
- **Output:** a family of k realizations of the random set $R \sim m(\mathcal{L})$



dataset $\mathcal{D}$
pattern language $\mathcal{L}$
+ measure m
k random patterns $X \sim m(\mathcal{L})$





Pattern sampling addresses the full pattern language $\mathcal{L}$ ➠ **diversity!**

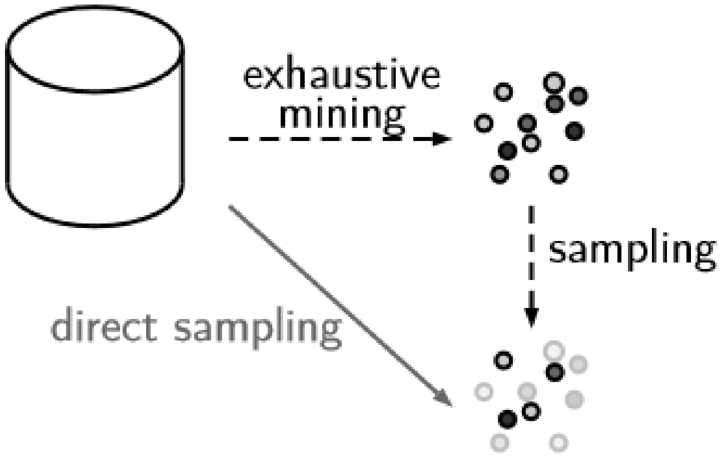
Slide: Plantevit, Crémilleux, Soulet

Pattern sampling: Challenges

ICFCA 2017 JUNE 13-16

Naive method

1. Mine all the patterns with their interestingness m
2. Sample this set of patterns according to m



exhaustive mining

sampling

direct sampling

➠ Time consuming / infeasible

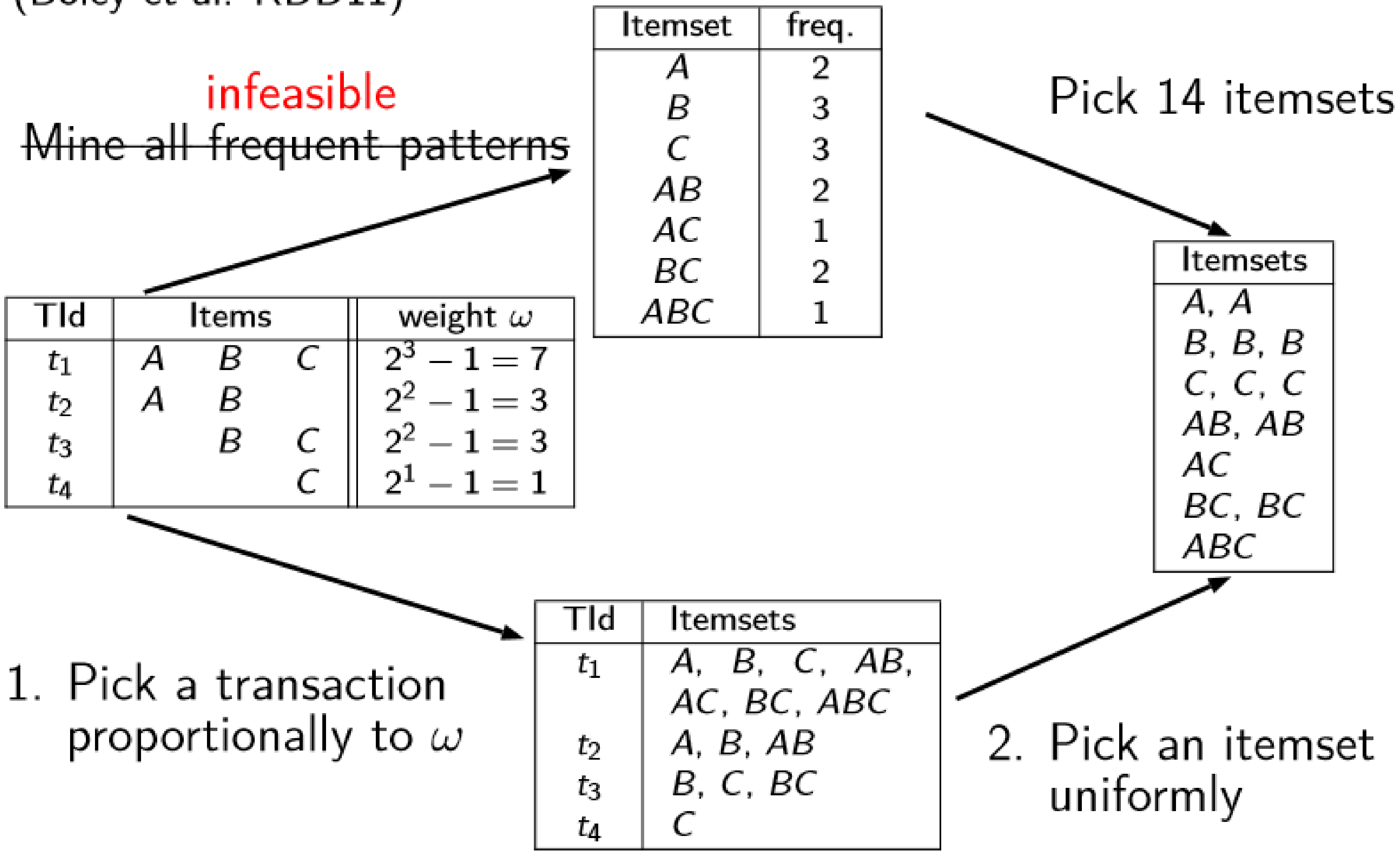
Challenges

- Trade-off between pre-processing computation and processing time per pattern
- Quality of sampling

Slide: Plantevit, Crémilleux, Soulet

Two-step procedure: Toy example

Direct local pattern sampling by efficient two-step random procedures. (Boley et al. KDD11)



infeasible

~~Mine all frequent patterns~~

Itemset	freq.
A	2
B	3
C	3
AB	2
AC	1
BC	2
ABC	1

Pick 14 itemsets

Itemsets
A, A
B, B, B
C, C, C
AB, AB
AC
BC, BC
ABC

TId	Items			weight ω
t_1	A	B	C	$2^3 - 1 = 7$
t_2	A	B		$2^2 - 1 = 3$
t_3		B	C	$2^2 - 1 = 3$
t_4			C	$2^1 - 1 = 1$

1. Pick a transaction proportionally to ω

TId	Itemsets
t_1	A, B, C, AB, AC, BC, ABC
t_2	A, B, AB
t_3	B, C, BC
t_4	C

2. Pick an itemset uniformly

Slide: Plantevit, Crémilleux, Soulet