Data Mining

Andreas Karwath
Joerg Wicker
Johannes Gutenberg-Universität Mainz

APriori Algorithm (Agrawal et al., 1993)

$i := 1$
$C_i := \{\{A\} | A \text{ is an item}\}$
while $C_i \neq \{\}$ **do**
    % candidate testing (database scan)
    for each set in C_i test whether it is frequent
    let F_i be the collection of frequent sets from C_i
    % candidate formation
    let C_{i+1} be those sets of size i+1 such that all subsets are in F_i (frequent)
    $i := i + 1$
return $\cup F_j$

Candidate Formation

- By *joining*: union of pairs of frequent itemsets from the previous level
- e.g., {A,B} and {B,C} gives {A,B,C}
- However, {A, C} might still be infrequent
- Thus, additional pruning step checking whether all subsets are known to be frequent

Main Ideas of APriori

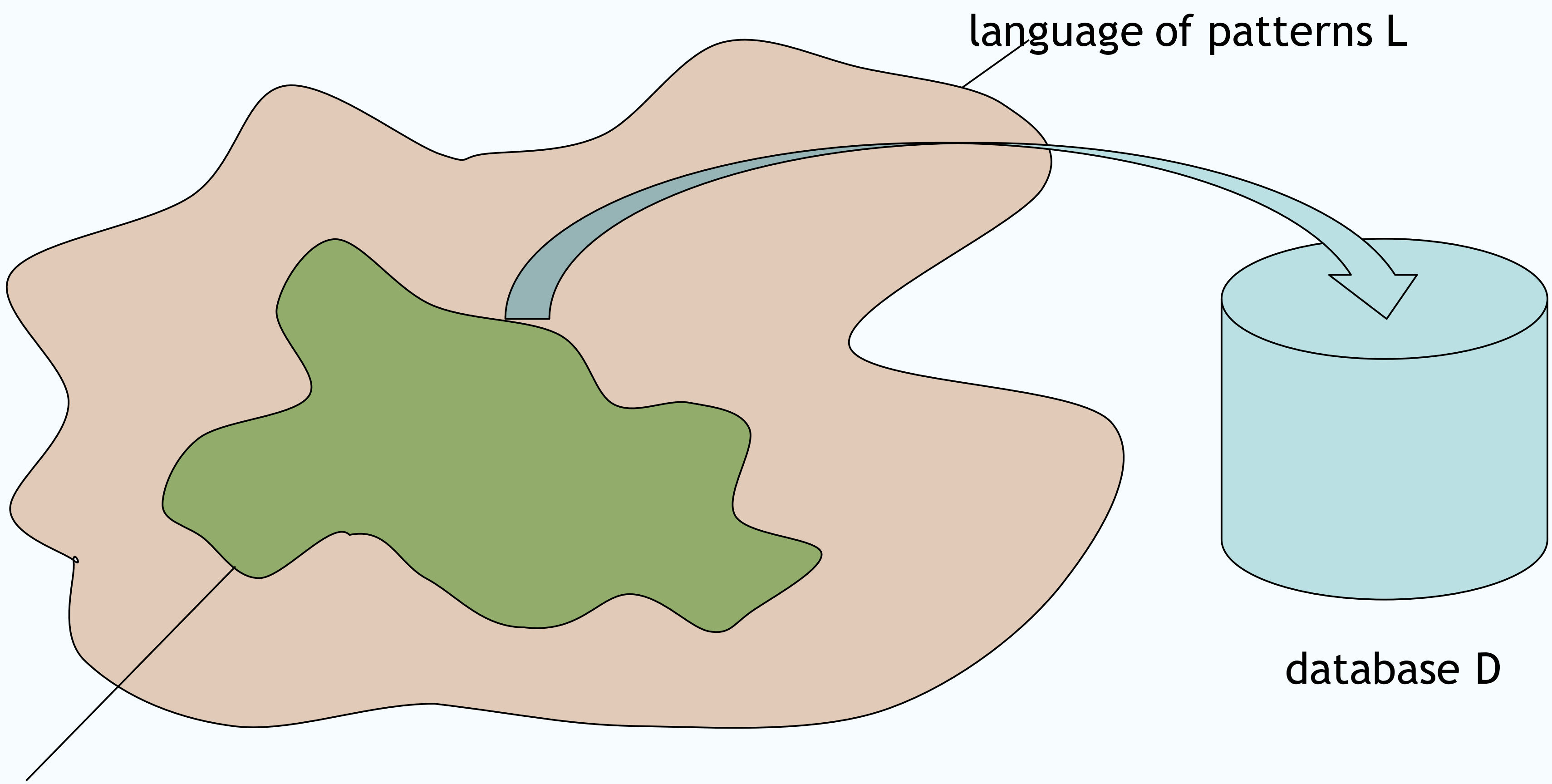
- Each iteration consists of two phases
  - candidate formation
  - candidate testing (database scan)
- Minimize database scans

  for each tuple *t* do
      for each candidate itemset *i* do
          …
- Avoid unnecessary tests on the database (test only those patterns that can, knowing the previous levels, be frequent)

Patterns (Itemsets) and (Association) Rules

- From frequent itemsets c and $c \cup \{i\}$ derive if c then $\{i\}$
- Start with the maximally specific frequent itemsets
- Variants possible: only one item in the RHS (very common assumption), only one item in the LHS (not very common)
- Generally: patterns and rules
  frequent patterns p, q such that $p \leq q$
  if p then q (with some confidence)

Formalization of Data Mining



q(p, D) … interestingness predicate: a pattern p from L is interesting wrt. database D
what is interesting? *frequent*, *non-redundant*, *class correlated*, *structurally diverse*, …

Formalization of Data Mining

- Simple formalization/definition of data mining (Mannila & Toivonen, 1997)
- Language L of patterns p
- Database D
- Interestingness predicate q
- Find a theory of the data:
  $Th(L, D, q) = \{p \in L \mid q(p,D) \text{ is true}\}$

Outline

- Constraint-based mining
- Condensed representations: closed and free sets

Anti-Monotonicity and Monotonicity

L: language of patterns

Constraint is *anti-monotonic* iff

$\forall \phi, \gamma \in L: \phi < \gamma \wedge \gamma \in S \rightarrow \phi \in S$

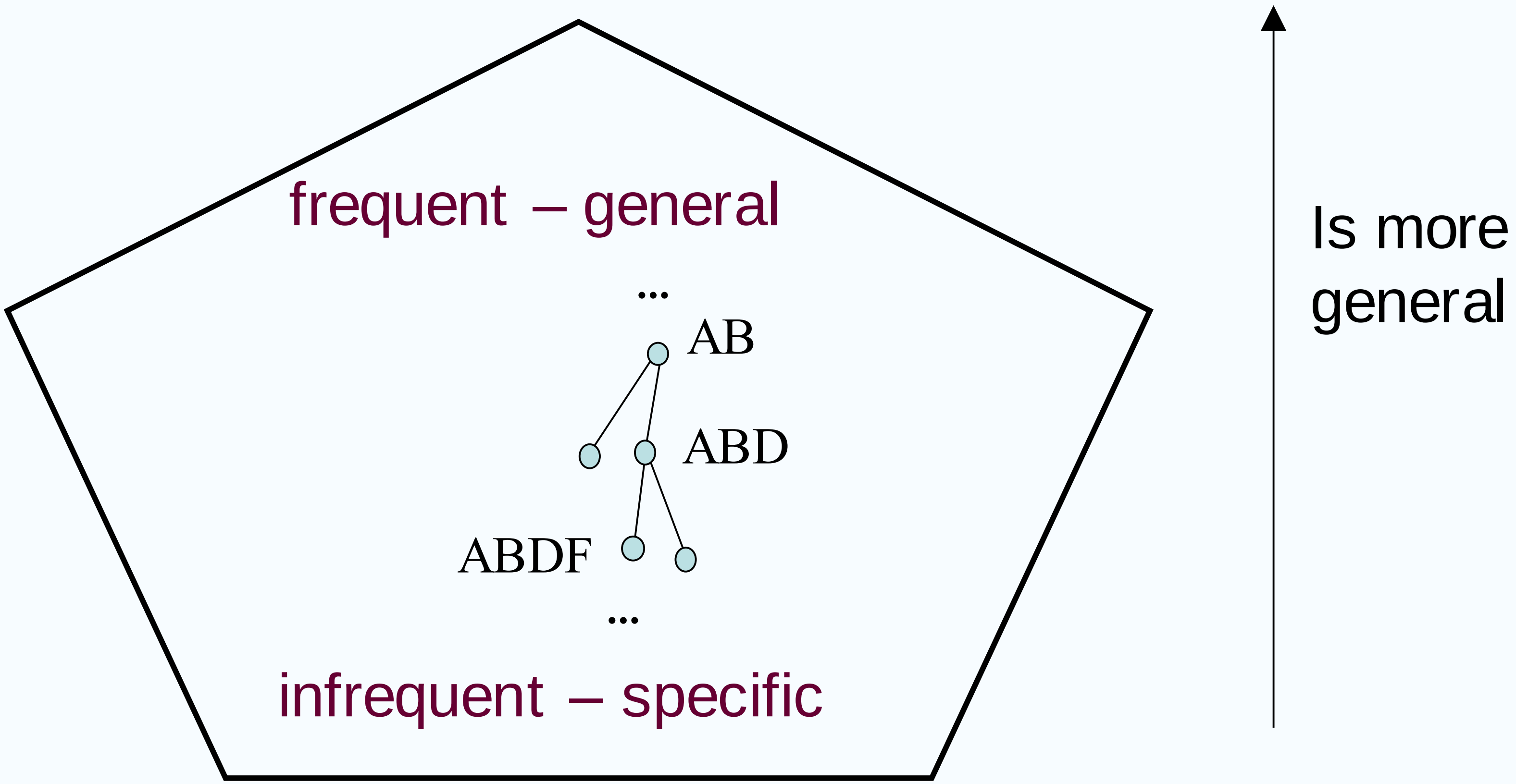
e.g., minimum frequency, $p \leq \{ABDF\}$

Constraint is *monotonic* iff

$\forall \phi, \gamma \in L: \phi < \gamma \wedge \phi \in S \rightarrow \gamma \in S$

e.g., maximum frequency, $p \geq \{AB\}$

Monotonicity and Anti-Monotonicity



Borders
(Mannila & Toivonen, 1997)

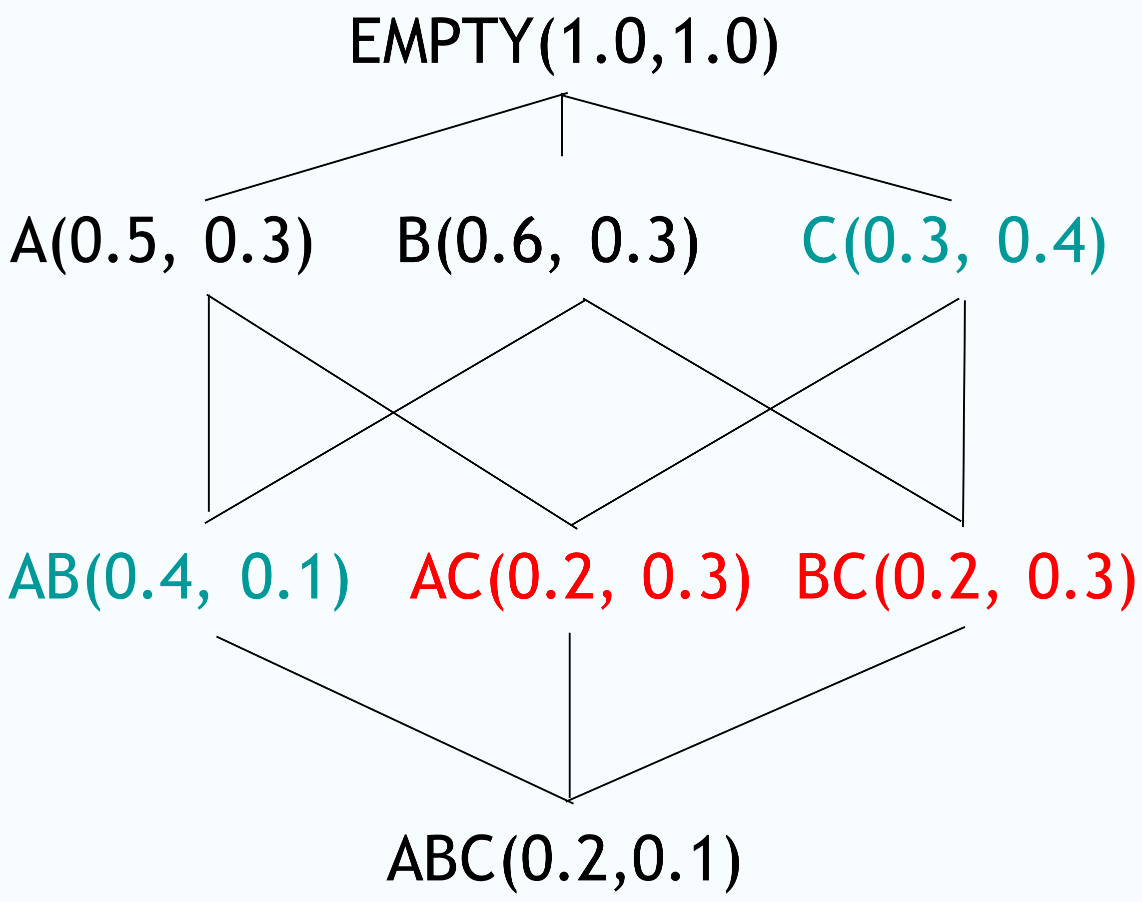
- **Positive Border** for minimum frequency constraint:
  most specific solution patterns in L
- S: set of solution patterns

  $Bd^{+}(S) = \{\phi \in S \mid \forall \gamma \in L\colon \phi < \gamma \rightarrow \gamma \notin S\}$
- **Negative Border:**
  most general non-solution patterns in L

  $Bd^{-}(S) = \{\phi \in L \setminus S \mid \forall \gamma \in L\colon \gamma < \phi \rightarrow \gamma \in S\}$

Example

freq(t, D1) $\geq$ 0.3



From APriori Output to Borders

Positive border:

- either: collect *all* frequent patterns in F and then maximize:

  $Bd^+ = \{\phi \in F \mid \neg \exists \gamma \in F: \phi < \gamma\}$

- or: collect *only those from the transition* from frequent to infrequent and then maximize

Negative border:

- just keep track of the *candidates* that turn out to be infrequent

From APriori Output to Borders

Example:

- F1= {A, B, **C**, **D**}
- C2 = {AB, AC, AD, BC, BD, **CD**}
- F2 = {**AB**, AC, **AD**, BC, **BD**}
- C3 = {ABC, **ABD**}
- F3 = {**ABC**}

Consequently:

- Bd^- = {CD, ABD}
- max({C, D, AB, AD, BD, ABC}) = Bd^+ = {AD, BD, ABC}

Summary: Conjunctions of Anti-/Monotonic Constraints

- For anti-monotonic (e.g., minimum frequency) constraint:
  run levelwise search and compute positive border
- For monotonic (e.g., maximum frequency) constraint: negate it, run levelwise search and compute negative border
- Postprocessing (pruning patterns in candidate borders)