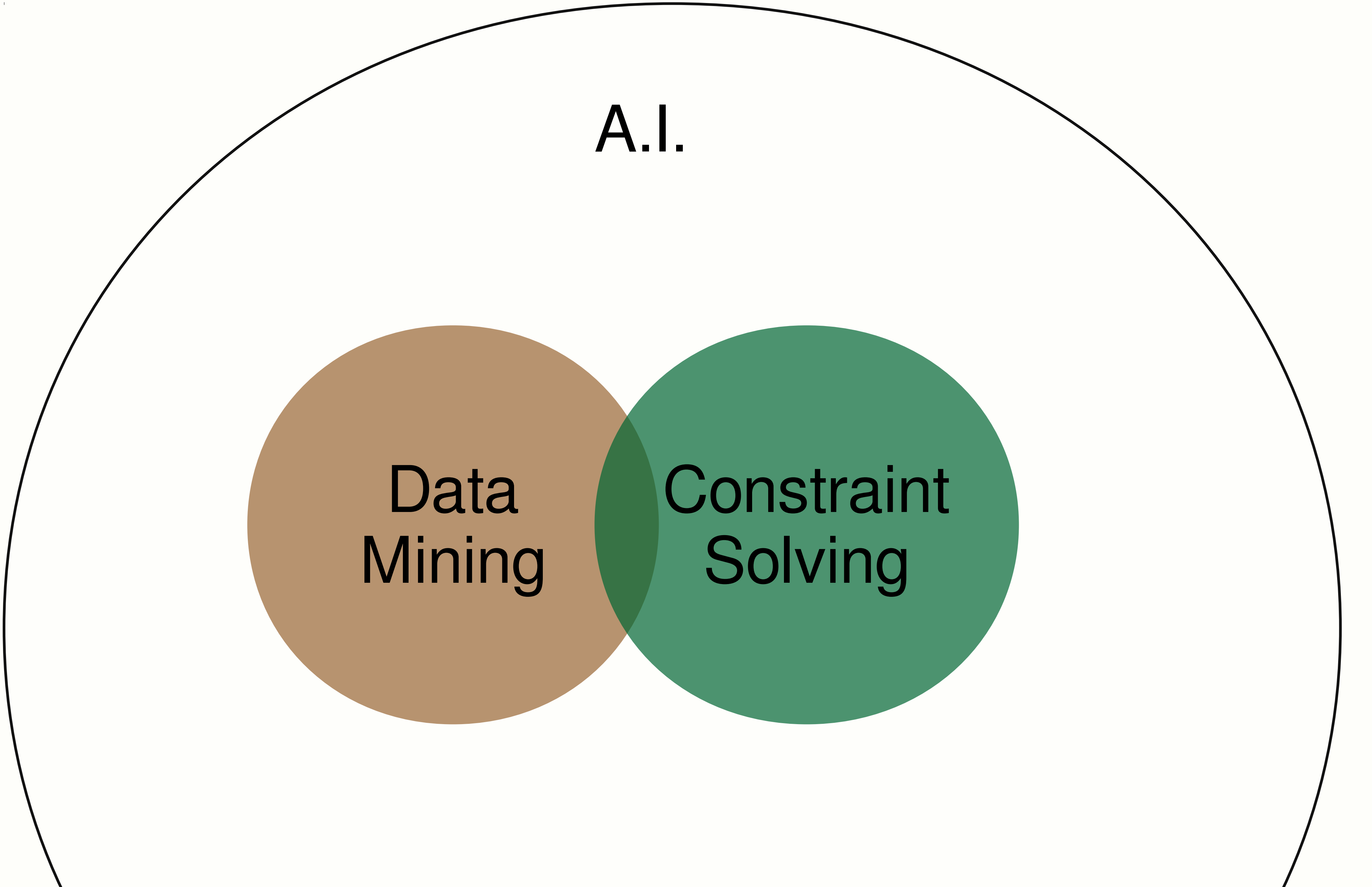
Declarative constraint-based pattern mining: from modeling to solving

Tias Guns

BUTO, VUB Brussel
Declarative Languages & AI lab, KU Leuven
http://homepages.vub.ac.be/~tiasguns/

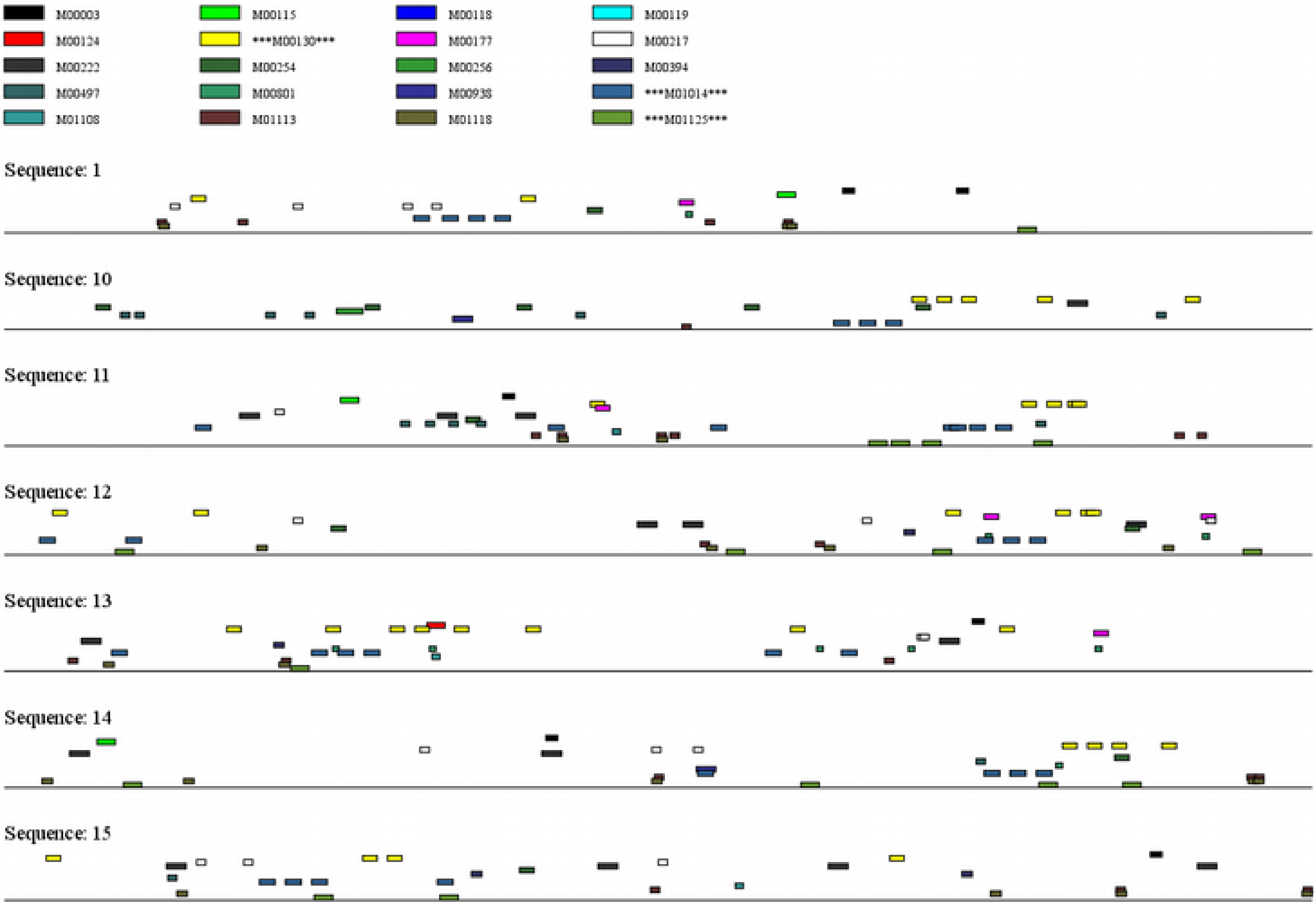
DeLBP workshop @ IJCAI 2017


A.I.
Data Mining
Constraint Solving

Itemset mining

Transactions:

Biological sequence mining

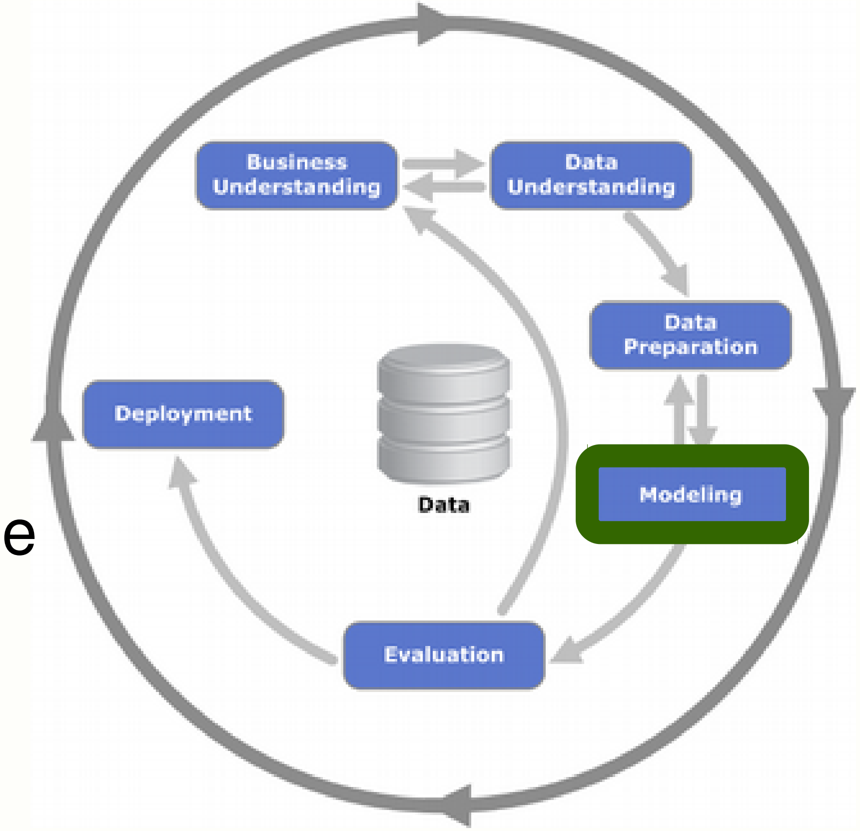


(Discrete) Data mining: methods

Usually specific algorithms for specific problems

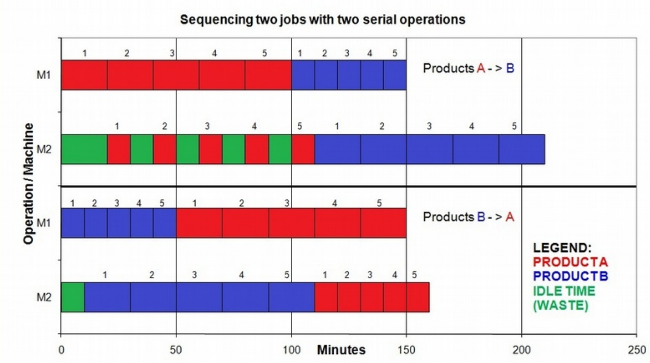
Highly scalable, but;

- New problems rarely fit existing methods well
- Tedious programming & hacks
- Refining solution methods is hard, even though typical in the knowledge discovery cycle

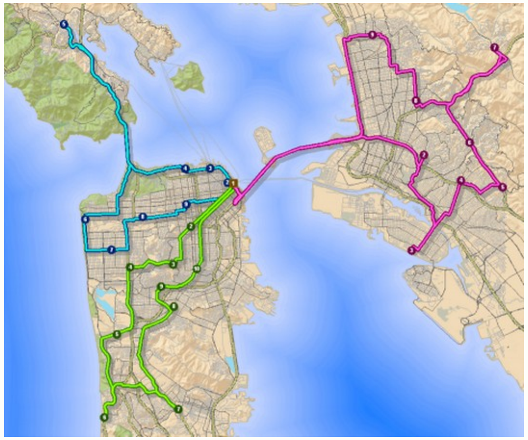


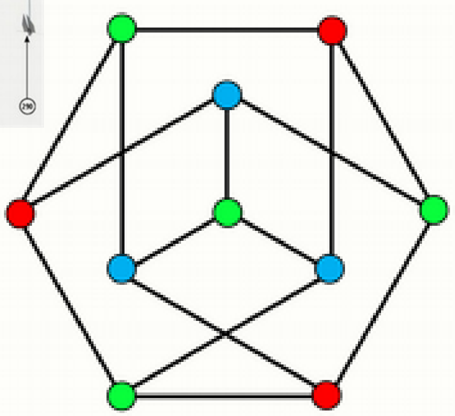
Constraint Solving

"Solving constraint satisfaction/optimization problems"



- Scheduling
- Routing
- Configuration
- Graph problems



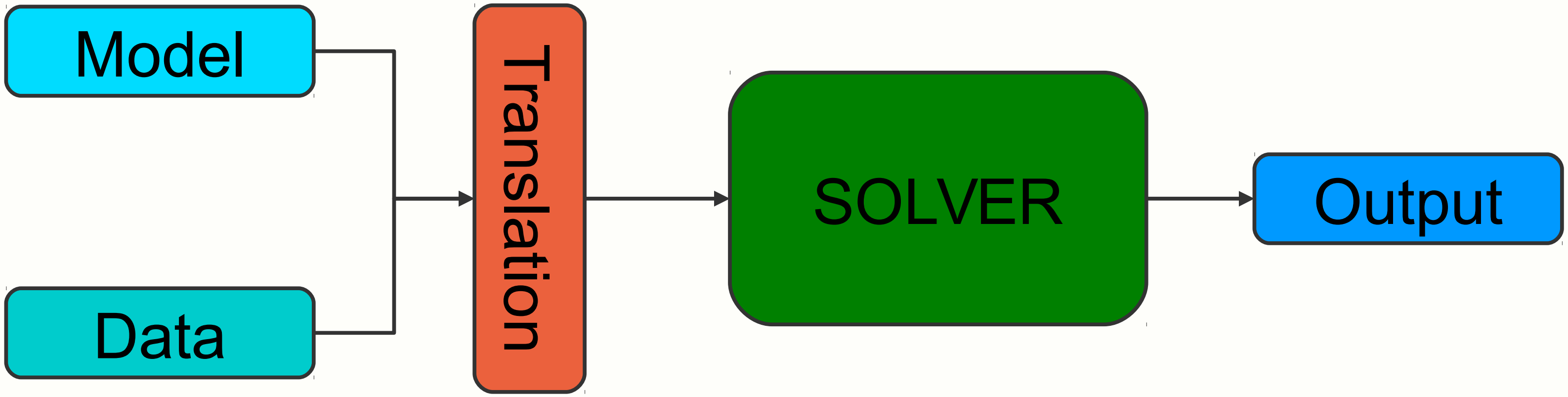


Constraint solving: methods

"Combinatorial problem = Model + Solve"

Model = specification of constraints over variables

Solve = search for satisfying/optimal solutions



Many generic and efficient solvers available

Constraint solving: why data mining?

→ many DM problems **are** combinatorial problems

Modeling

- \+ Adding/removing/combining constraints
- \+ Complex constraints
- \- Modeling choices matter

Solving

- \+ Reusing solving technology
- ± Exhaustive, optimal
- \- Scalability towards large datasets

Active research directions

- Pattern Mining B. Cremilleux, L. De Raedt, T. Guns, T. Guyet, S. Jabbour, M. Jarvisalo, A. Kemmar, S. Loudni, S. Nijssen, B. O'Sullivan, ...
- Clustering B. Babaki, I. Davidson, T.B.H. Dao, K.C. Duong, S. Gilpin, V. Grossi, P. Hansen, O. du Merle, A. Monreale, S. Nijssen, C. Vrain, ...
- Structure learning C. Bessiere, J. Cussens, O. Grinchtein, M. Heule, T. Jaakkola, M. Meila, B. O'Sullivan, D. Sontag, P. Van Beeck, S. Verwer, ...

In this talk

Modeling: generality

I. Itemset mining and constraints

II. A modeling language for constraint-based mining?

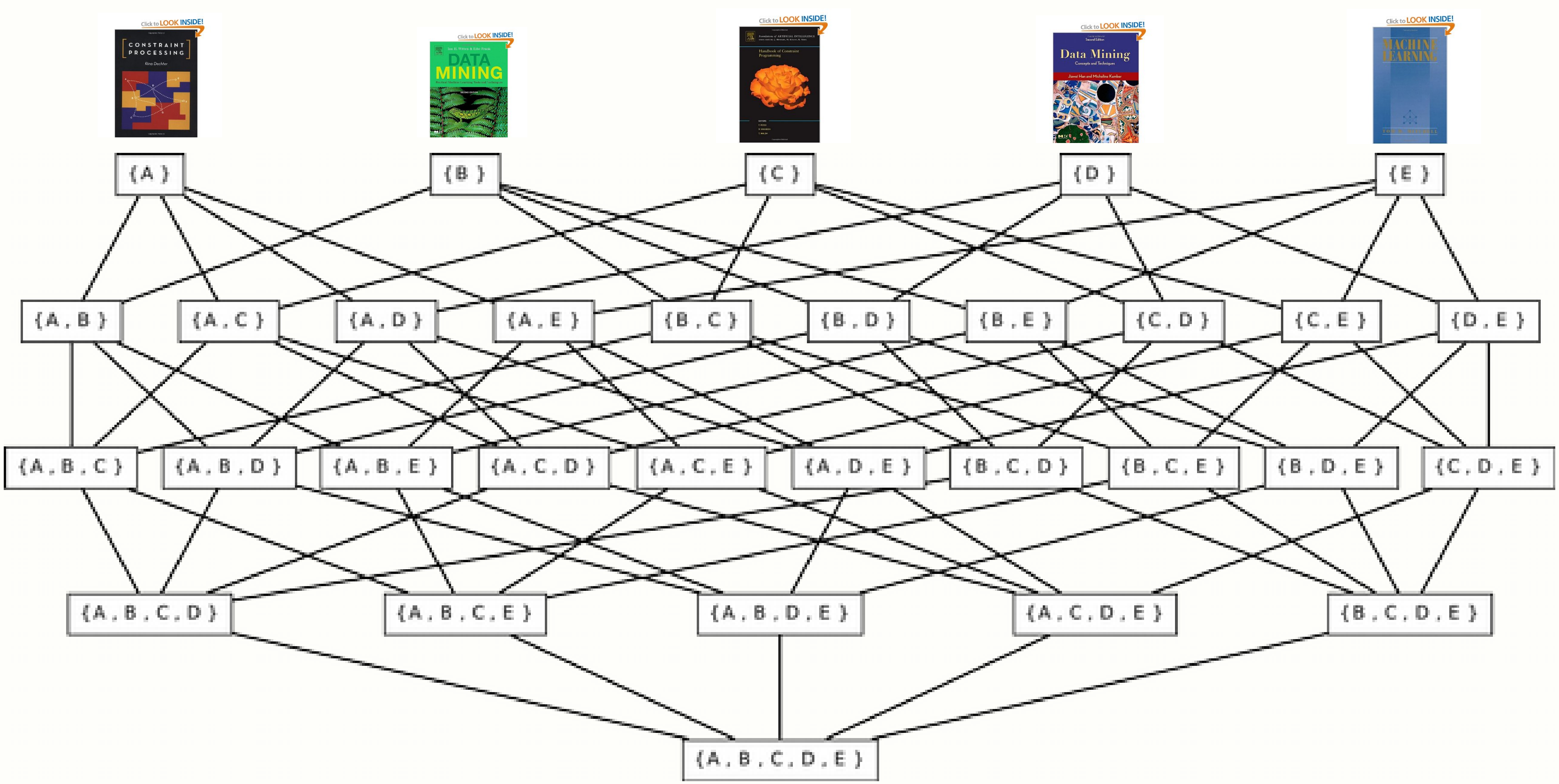
Solving: efficiency

III. Scalability of generic itemset solving

IV. Sequence mining and constraints

Constraint-based Itemset Mining

- Fundamental enumeration problem
- Well studied
- Many constraints
- Many applications



“Interesting” patterns:

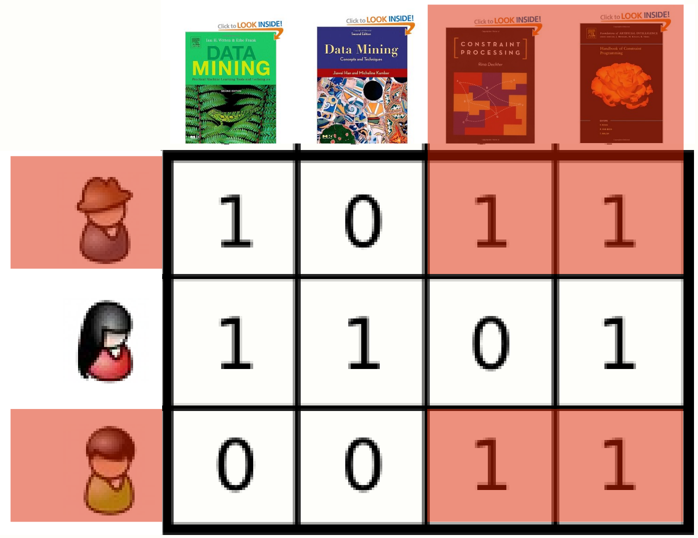
- which patterns appear <u>frequently</u> in a dataset?
- which patterns have a certain <u>structure</u>?
- which patterns have a high <u>cost or profit margin</u>?
- which patterns <u>summarize</u> a dataset?
- which patterns are frequent on one dataset and <u>infrequent on another</u>?
- which patterns are <u>significant</u> w.r.t. a background model?

→ specified by constraints

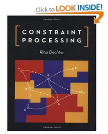
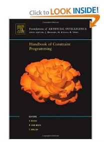
“Constraint-based mining”

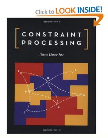
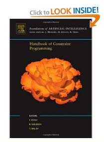
Frequent Itemset Mining

Find: all sets of *items* appearing frequently

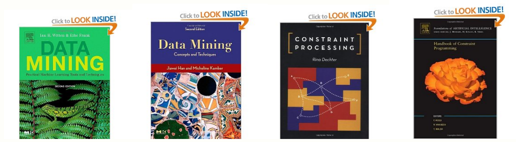


1	0	1	1
1	1	0	1
0	0	1	1

cover( , ) = {  ,  }

frequency( , ) = | {  ,  } | = 2

CP for Itemset Mining



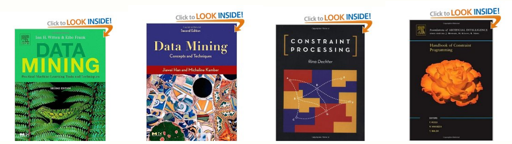
	i1	i2	i3	i4
	0/1	0/1	1	1
t1 0/1	1	0	1	1
t2 0/1	1	1	0	1
t3 0/1	0	0	1	1

One solution = one frequent itemset: enumerate all

coverage: $\forall T_t : \quad T_t = 1 \Leftrightarrow set(I_1, \ldots I_n) \subseteq \boxed{set(row_t)}$

frequency: $\sum_t T_t \geq Freq$

CP for Itemset Mining



	i1 0/1	i2 0/1	i3 0/1	i4 0/1
t1 0/1	1	0	1	1
t2 0/1	1	1	0	1
t3 0/1	0	0	1	1

coverage: $\forall T_t: \quad T_t = 1 \Leftrightarrow \sum_i I_i (1 - D_{ti}) = 0$

frequency: $\forall I_i: \quad I_i = 1 \Rightarrow \sum_t T_t D_{ti} \geq Freq$

[L. De Raedt, T. Guns, S. Nijssen, KDD 2008]

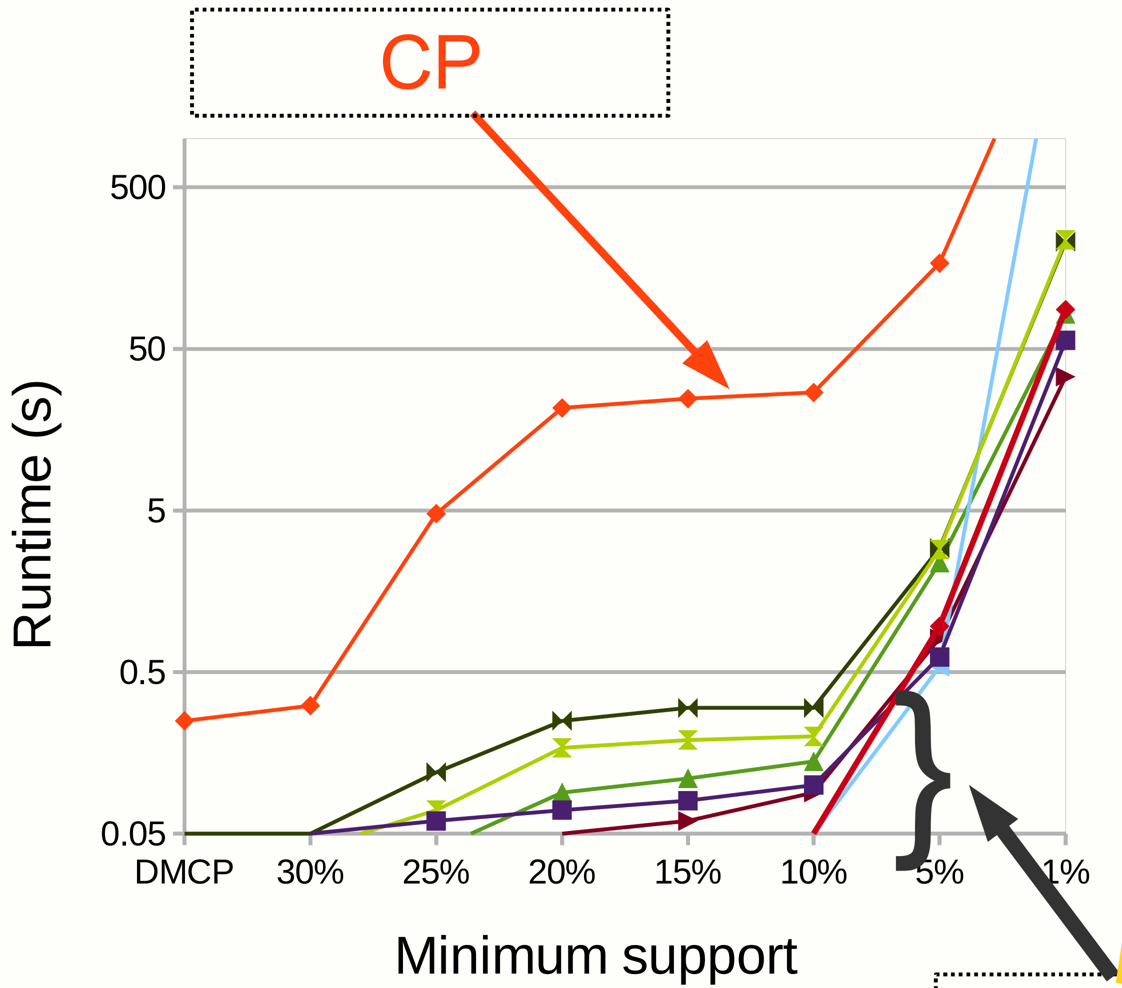
Generality

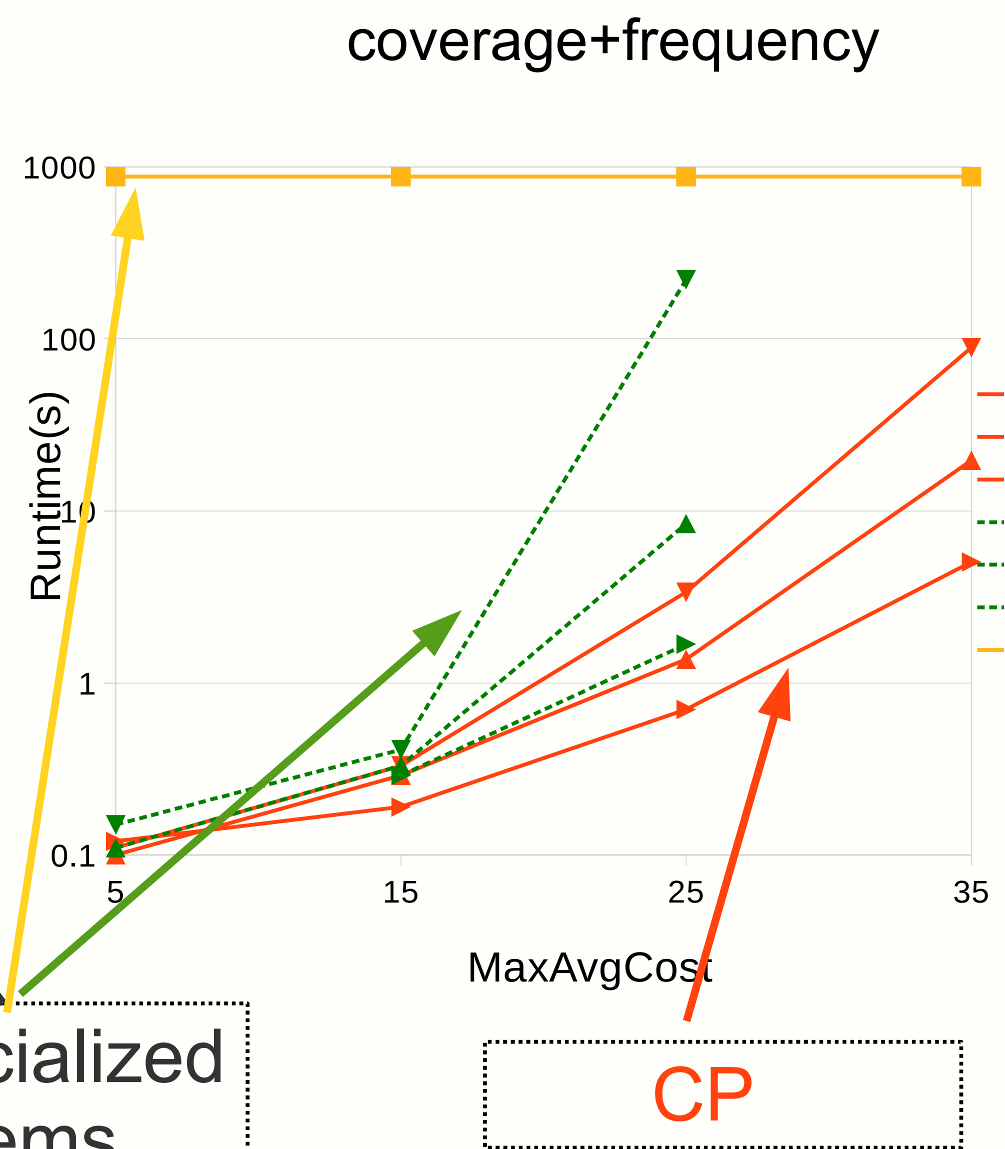
	LCM [15]	MAFIA [6]	ExAMiner [4]	DualMiner [5]	DDPrune [12]	CP4IM
Constraints on data						
Minimum frequency	X	X	X	X		X
Maximum frequency				X		X
Emerging patterns						X
Condensed Representations						
Maximal	X	X		X		X
Closed	X	X				X
$\delta-$Closed						X
Constraints on syntax						
Max/Min total cost			X	X		X
Minimum average cost			X			X
Max/Min size	X	X	X	X		X
Constraints on labelled data						
Minimum correlation					X	X
Maximum correlation						X

[L. De Raedt, T. Guns, S. Nijssen, AAAI 2010]

Few constraints

Many constraints





Specialized systems

Take away message 1.

Constraint Programming for Itemset Mining:

- Mathematical and reasonably compact encoding
- Generic: many constraints can be expressed
- Effective in case of tight constraints

Many extensions (not in this talk):

- Pattern *set* mining
  [T. Guns, S. Nijssen, L. De Raedt, TKDE 2013]
  [A. Ouali, S. Loudni, Y. Lebbah, P. Boizumault, A. Zimmermann, L. Loukil, IJCAI 2016]
- Skypatterns / multi-objective
  [W. Ugarte, P. Boizumault, S. Loudni, B. Cremilleux, ECAI 2014]
  [W. Ugarte, P. Boizumault, B. Crémilleux, A. Lepailleur, S. Loudni, M. Plantevit, C. Raïssi, A. Soulet, AIJ 2017]
- SAT, BDD, ASP solvers
  [JP. Metivier, P. Boizumault, B. Cremilleux, M. Khiari, S. Loudni, IDA 2012]
  [H. Cambazard, T. Hadzi, B. O'Sullivan, ECAI 2010]
  [M. Jarvisalo, LPNMR 2011]

In this talk

Modeling

I. Itemset mining and constraints

II. A modeling language for constraint-based mining?

Solving

III. Scalability of generic itemset solving

IV. Sequence mining and constraints

A modeling language for pattern mining?

A long standing dream...

Many have roots in '**Inductive Databases**' idea of Heikki Mannila

$\rightarrow$ a database where *data* and *patterns* are both easily queried

Projects integrating mining in SQL:

- MINE RULE [Meo et al, 1996]
- MSQL [Imielinksi & Virmani, 1999]
- Mining Views [Blockeel et al, 2012]

Mostly: mining algorithm parameters $\leftrightarrow$ query parameters

A modeling language for pattern mining?

A long standing dream...

Others looked at **constraint-based languages**

- Levelwise [Mannila & Toivonen 1997]
- MusicDFS [Soulet & Cremilleux 2005]
- ConQueSt [Bonchi & Lucchese 2007]

Mostly: based on (anti-)monotonicity of constraints

Little support for other constraints (closed, maximal, discriminative) or combinations.

Modeling languages in CP

Constraint Programming has long tradition of modeling languages

- ECLiPSe and B-prolog (Constraint Logic Programming)
- OPL [Van Hentenryck, 1999]
- COMET [Van Hentenryck and Michel, 2005]
- MiniZinc [Nethercote et al, 2007]
- Essence [Frisch et al, 2008]

→ CP languages as starting point for pattern mining language

A modeling language for pattern mining?

MiningZinc

- Based on the established MiniZinc language
  - **High-level** mathematical-like notation
  - **User-defined** constraints and functions
  - **Solver independent** (10+ CP solvers & SAT & MIP)
- Modeling: pattern mining specific constrains and functions
- Solving: generic AND specialised methods (transparently)

[T. Guns, A. Dries, S. Nijssen, G. Tack, L. De Raedt, IJCAI 2013]

Example: constrained itemset mining

library with itemset mining specific functions and predicates

```
include ''lib_itemsetmining.mzn''

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

var set of 1..NrI: Items;

constraint card(cover(Items, TDB)) >= MinFreq;

constraint card(cover(Items, TDB)) <= MaxFreq;

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

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

solve satisfy;
```

Solver independence

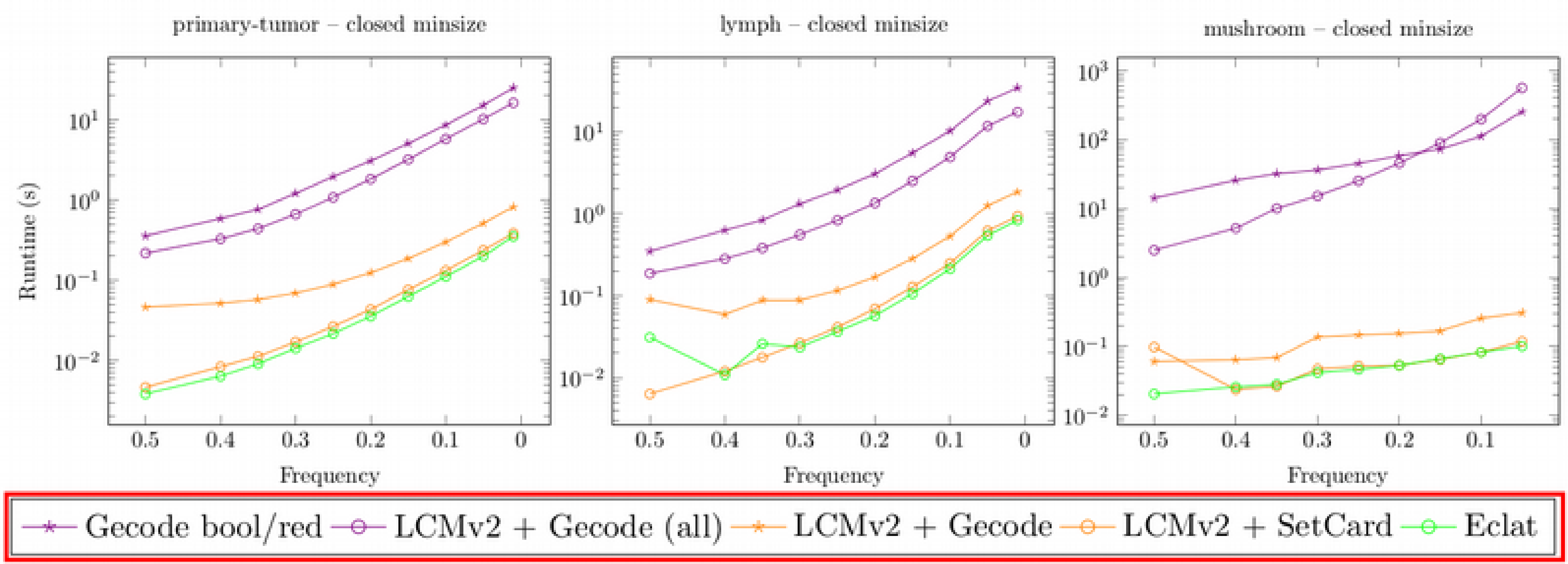
```
var set of 1..NrI: Items; array[int] of set of int: TDB;
constraint card(cover(Items, TDB)) >= 20;
constraint card(cover(Items, TDB)) =< 40;
solve satisfy;
```

From text-based model to ``execution plans'':

1) specialised solvers *(if supported constraint combination)*
2) automatic post-processing:
   - use specialised solver on subset of constraints
   - post-process remaining constraints with generic solver
3) generic (CP) solvers

Experiments, hybrid solving

frequent itemset mining, with minimum size and closure constraint



[T. Guns, A. Dries, S. Nijssen, G. Tack, L. De Raedt, AIJ 2017]

Take away message 2.

Modeling:

- Can build on existing high-level CP languages
- Solver independence:
  - Automatic model rewriting [ModRef workshop, 2002 - present]
  - Automatic *chaining* of CP/DM algorithms: hybridization

Open questions

- Multiple execution strategies: algorithm selection? parallelism?
- Problems not fitting standard CP
  - Skyline patterns / multi-objective [W. Ugarte, P. Boizumault, S. Loudni, B. Cremilleux, ECAI 2014]
  - Dominance / preference over solutions [B. Negrevergne, A. Dries, T. Guns, S. Nijssen, ICDM 2013]
- Text-based language vs. embedded language

In this talk

Modeling

I. Itemset mining and constraints

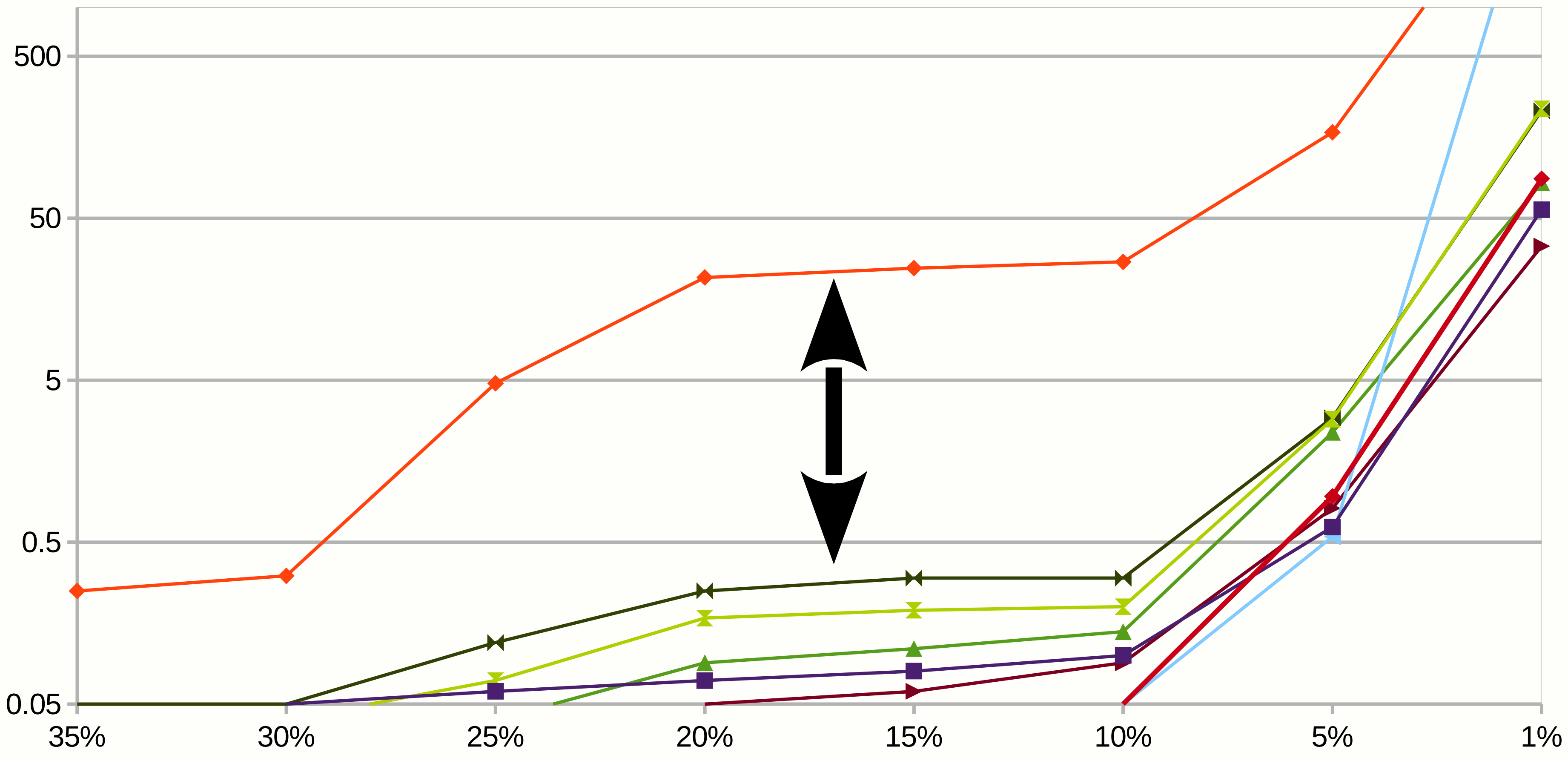
II. A modeling language for constraint-based mining?

Solving

III. Scalability of generic itemset solving

IV. Sequence mining and constraints

Improving scalability?

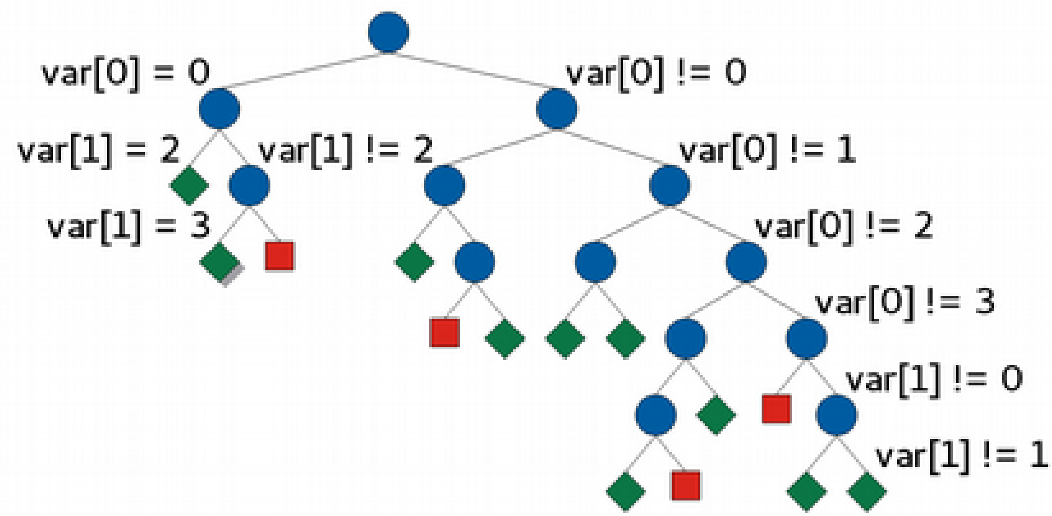


IM search vs. CP search

- Highly efficient IM algorithms do depth-first search
- CP solver is at its core a principled depth-first search framework

Algorithm 2 Constraint-Search(D)

$D :=$propagate(D)
if D is a false domain **then**
  return
end if
if $\exists v \in \mathcal{V} : |D(v)| > 1$ **then**
  $v := \arg\min_{v \in \mathcal{V}, D(v)>1} f(v)$
  $D_p :=$ split($D(v)$)
  Constraint-Search($D \cup \{v \mapsto D_p\}$)
  Constraint-Search($D \cup \{v \mapsto D - D_p\}$)
else
  Output solution
end if



What makes the difference?

Differences IM search / CP search

In pure CP model:

for each transaction a separate constraint

➔ data is split into many individual constraints

➔ CP has to do bookkeeping of constraints and its variables

in IM:

constraints are checked on entire data at once

➔ can use advanced data structures like vertical tidlists

➔ can cache computations (e.g. frequency of each item)

$\rightarrow$ keep data together in CP?

CP scalability

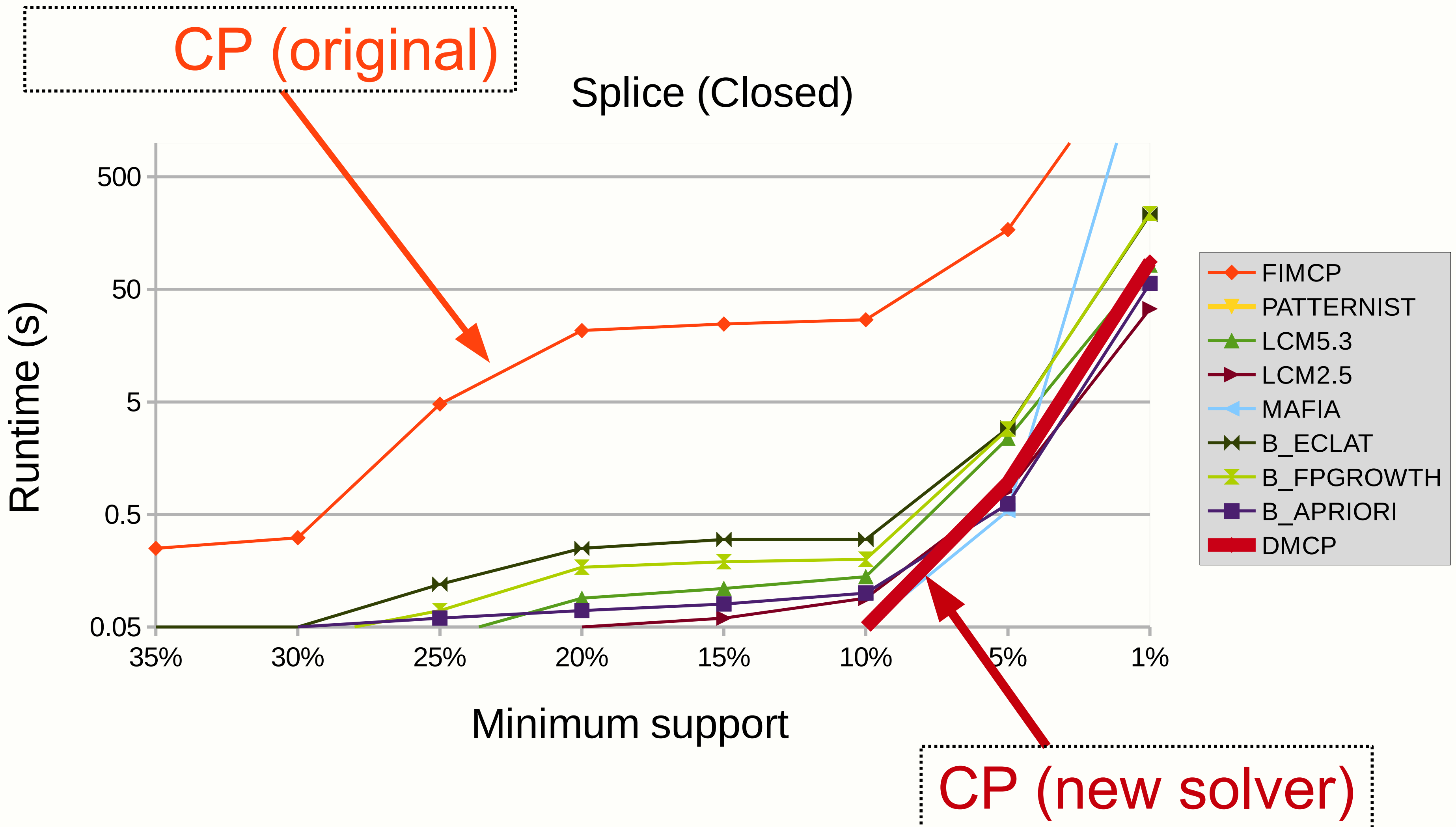
Wrote minimalistic CP solver that:

- implements standard generic CP search
- implements BoolVector variable type (bitwise computations)
- supports the generic global constraint $\boldsymbol{X} \,\square\, (D\,\boldsymbol{A} \simeq B)$
  (other constraints could be added as in any CP solver)

Within global constraint:

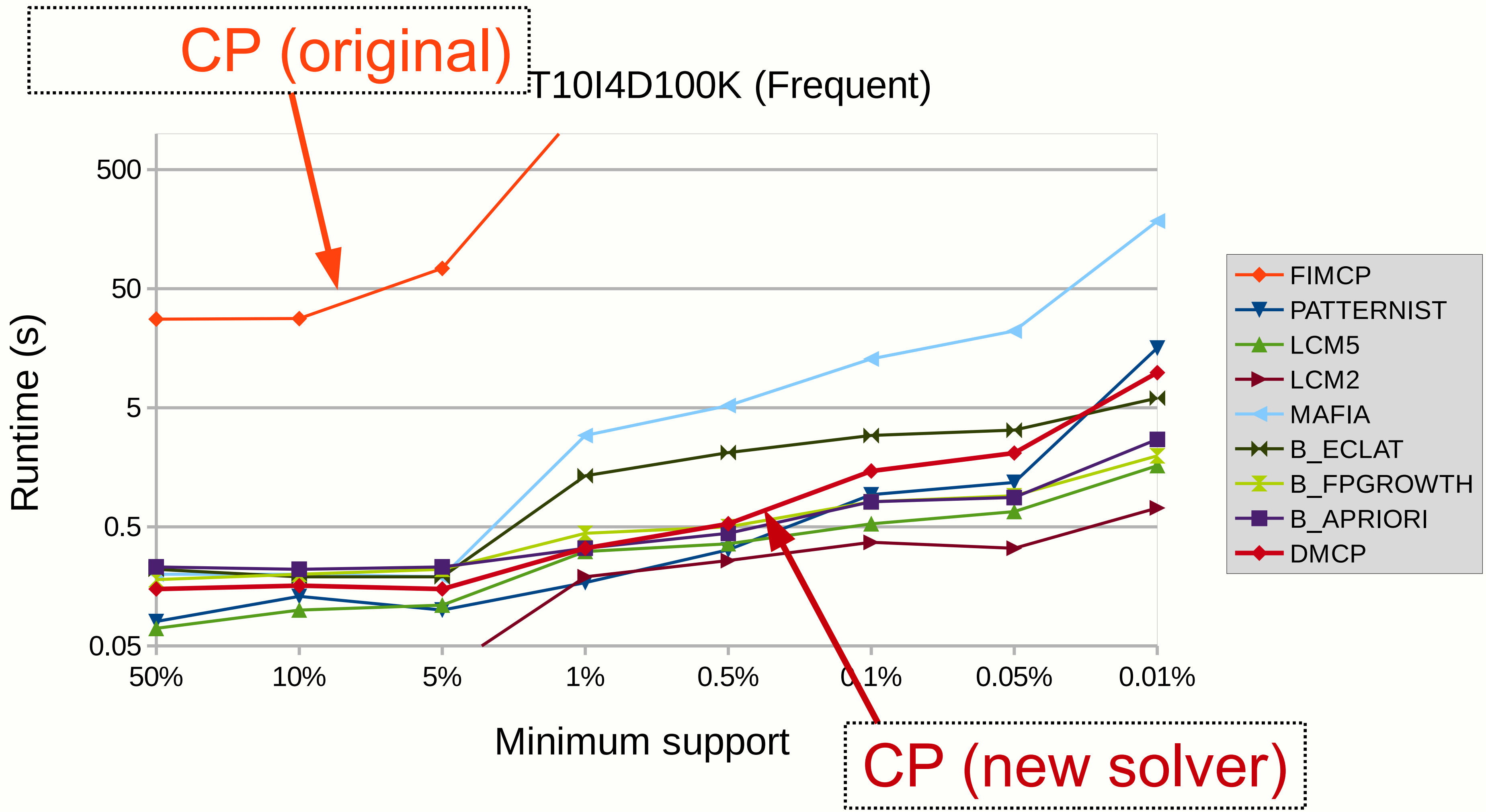
- can use (core of) same algorithms as specialised methods
- can do efficient bitwise computations and caching

Integrated solver



[S. Nijssen, T. Guns, ECMLPKDD 2010]

Frequent Itemset Mining, scaling



[S. Nijssen, T. Guns, ECMLPKDD 2010]

Within standard CP solver

- Standard CP solver
- One global constraint for:
  - computing the cover **and** enforcing minimum frequency
  - no need to expose transaction variables 'T'

Time (s)			
(1)	(2)	(3)	LCM
7.26	**3.21**	10.97	0.32
26.37	**12.27**	40.85	1.44
109.36	**45.92**	136.31	6.07
480.52	**187.89**	467.52	27.55
2 287.98	**969.40**	1 950.51	141.55
0.10	0.59	22.59	0.04
0.18	**0.14**	25.54	0.07
3.39	**3.23**	138.54	1.46
502.77	**400.10**	1 652.41	53.59
0.46	0.64	0.77	0.00
0.92	**0.56**	2.36	0.01
0.38	**0.15**	5.32	0.04

	LCM [15]	MAFIA [6]	ExAMiner [	DualMiner [	DDPmine [	CP4IM [
Constraints on data						
Minimum frequency	X	X	X	X		X
Maximum frequency				X		X
Emerging patterns						X
Condensed Representations						
Maximal	X	X		X		X
Closed	X	X				X
δ−Closed						X
Constraints on syntax						
Max/Min total cost			X	X		X
Minimum average cost			X			X
Max/Min size	X	X	X	X		X
Constraints on labelled data						
Minimum correlation					X	X
Maximum correlation						X

=> can use (and hide) same datastructures as specialised methods

[N. Lazaar, Y. Lebbah, S. Loudni, M. Maamar, V. Lemière, C. Bessiere, P. Boizumault:, CP 2016]
[P. Chaus, J. Aoga, T. Guns, CP 2017]

Take away message 3.

Difference IM search / CP search

- Both use depth-first search
- IM is one big complicated algorithm
- CP decomposes problem in separate constraints

Increasing efficiency:

- Keep data together in a *global* constraint
- Bonus: advanced data structures and indexing
- Efficiency versus generality trade-off!

In this talk

Modeling

I. Itemset mining and constraints

II. A modeling language for constraint-based mining?

Solving

III. Scalability of generic itemset solving

IV. Sequence mining and constraints

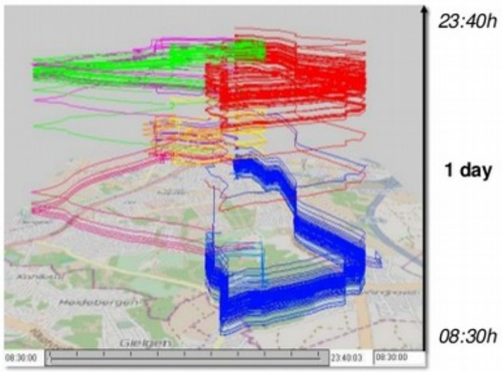
Sequential data

Example:

<Home, Work, Restaurant, Work, Home>

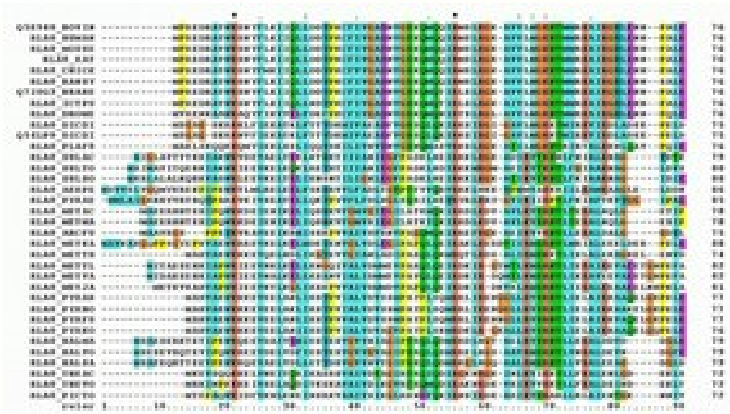
<Home, Work, Shops, Restaurant, Home>

...



Many applications:

- User mobility mining
- Web usage mining
- Event monitoring
- Biological sequence mining (DNA, Amino acids)
- ...



Sequence mining

Cover

= subsequence relation

= ordered matching

Pattern: <H, G, ?, ?, ?>

T1: <S,B,H,R,G,H,M>

T2: <S,G,H,W,L,W,M>

T3: <R,H,W,H,D,G,H>

multiple embeddings possible:

T3: <R,**H**,W,**H**,D,G,H>

Why CP?

Many constraints, we identify four categories:

- Constraints on syntax:

  size, regular expr., ...

- Constraints on data:

  min_freq, max_freq, discriminative, ...

- Preferences over the solution set

  closed, maximal, relevant, multi-objective, ...

- **new** Constraints on inclusion relation:

  max_gap, min_gap, max_span

Hard-coded in specialised algorithms...

Standard sequences

X = <Home, Work, Restaurant, Gym, Work, Home>

Example sequence P=<Home,Home>

- can have arbitrary symbols before/between/after
- ex: <Home, Work, Restaurant, Gym, Work, Home>
- Formally: $T=1 \Leftrightarrow \exists(e_1..e_n):\ e_1<\ldots<e_n \wedge \forall_j\, P[j]=X[e_j]$
- In CP: One variable for each *e_j* , multiple constraints?

What specialized algorithms do

P: <**H**, **H**, ?, ?, ?>

T1: <R,**H**,W,**H**,G,D,H>

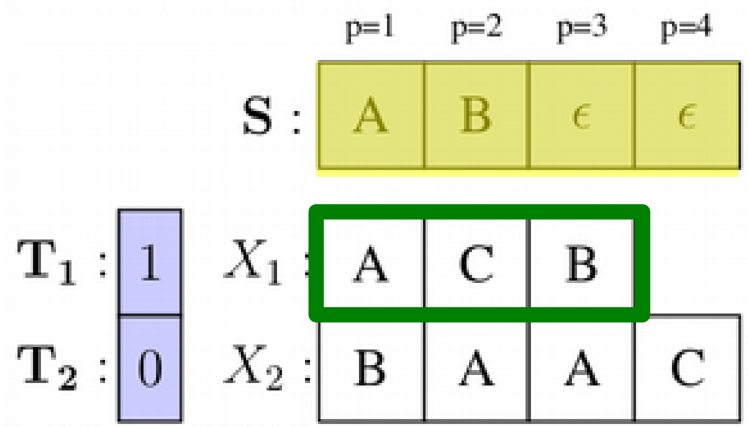
T2: <S,B,**H**,R,G,**H**,M>

PrefixSpan:

- Linear scan of each transaction, keep only pointer to *first* match of *last* symbol
- When symbol added to P, continue from pointer (incremental)
- O(1) space, O(n) algorithm

In CP, O(n) *e_j* variables and multiple constraints over them?

in CP: add global constraint



$$\forall T_t: \quad T_t = 1 \Leftrightarrow exist\text{-}embedding(S, \boxed{X_t})$$

$$\sum_t T_t \geq Freq$$

Global constraint with filtering algorithm:

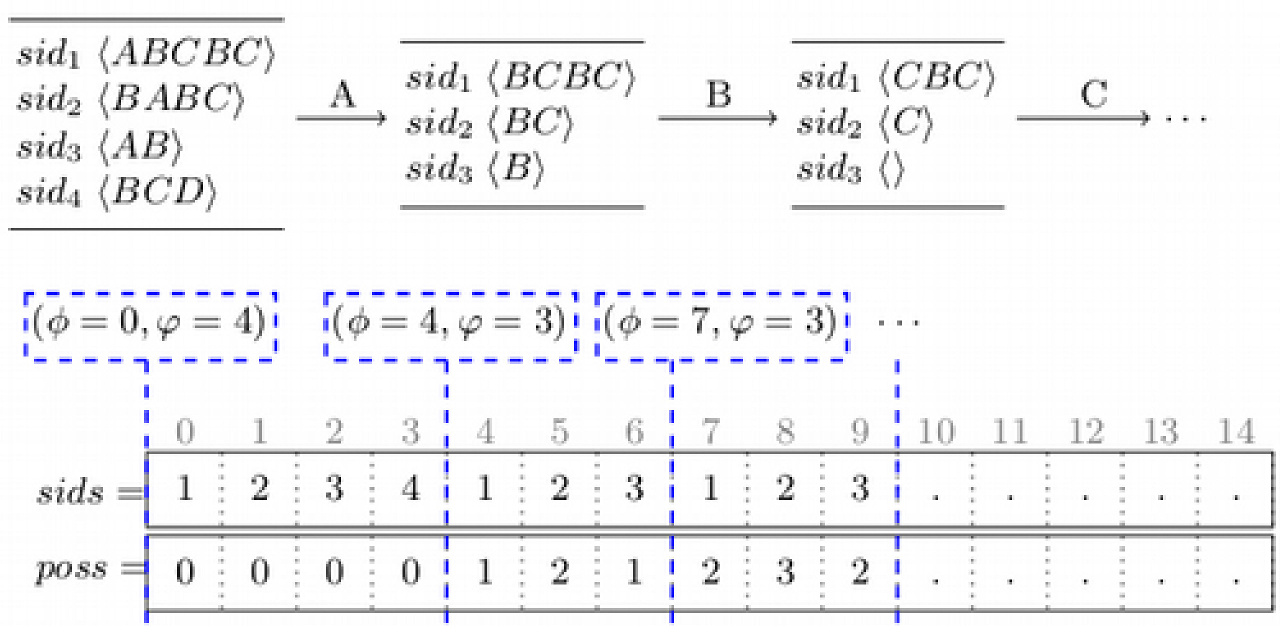
- $T_t = 1 \Leftrightarrow \exists (e_1 .. e_n): \; e_1 < \ldots < e_n \wedge \forall_j \, S[j] = X_t[e_j]$
- incremental: keep one pointer to *last assigned match e_j*

[B. Negrevergne, T. Guns, CPAIOR 2015]

Keeping data together

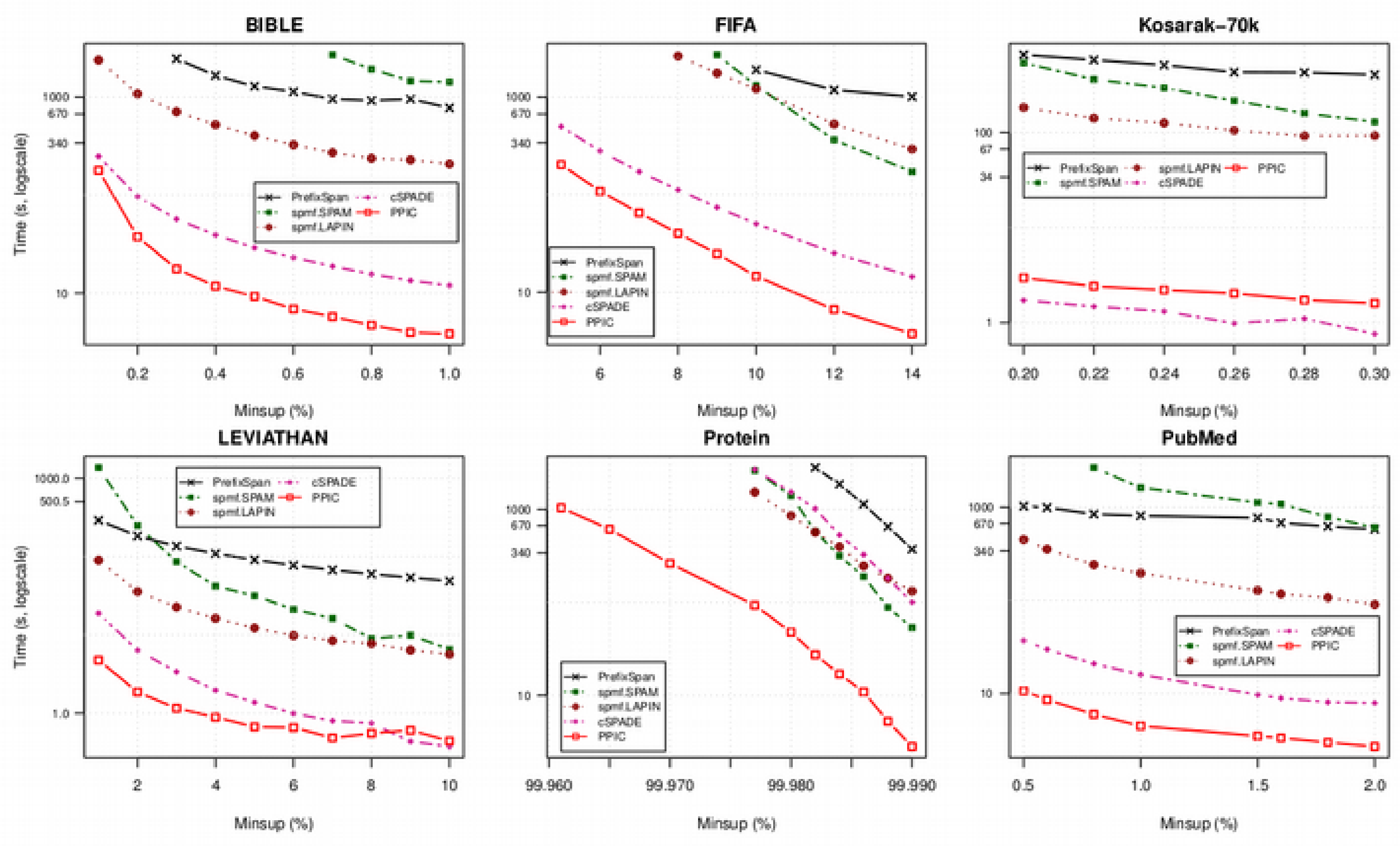
One global constraint for **all** sequences:

- algorithmic improvements: last position map, last position list
  $\rightarrow$ precomputed and cached, speedups
- use <u>backtracking-aware datastructure</u>
  $\rightarrow$ stores cover and prefix point in reversible vector



[J. Aoga, T. Guns, P. Chaus, ECMLPKDD 2016]

Efficiency: outperforms specialised!



Improving generality

Constraints:

- Constraints on sequence: compatible
- Constraints on cover set: compatible
- Preferences over the solution set: compatible
- Constraints on inclusion relation: not compatible

=> best known: min/max gap and span

gap: 1 gap: 2

<Home, Work, Restaurant, Gym, Work, Home, Bar>

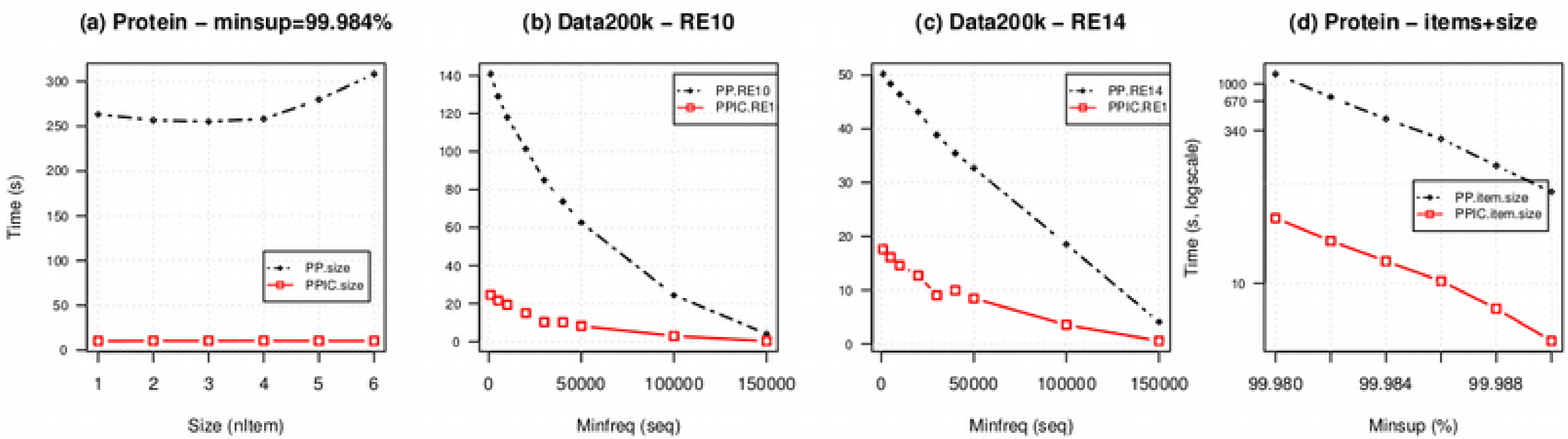
span: 4

can **modify the *global cover* constraint** to also enforce gap/span

→ improves state-of-the-art (with backtrack-aware datastructure)

[J. Aoga, T. Guns, P. Chaus, CPAIOR 2017]

Constraints



Take away message 4.

Sequence mining: more complex *coverage* relation

Global constraint:

- hides complexity of *coverage* relation
- fast (incremental, PrefixSpan-like)
- good way to hybridize with data mining techniques
  => necessary to be **efficient**

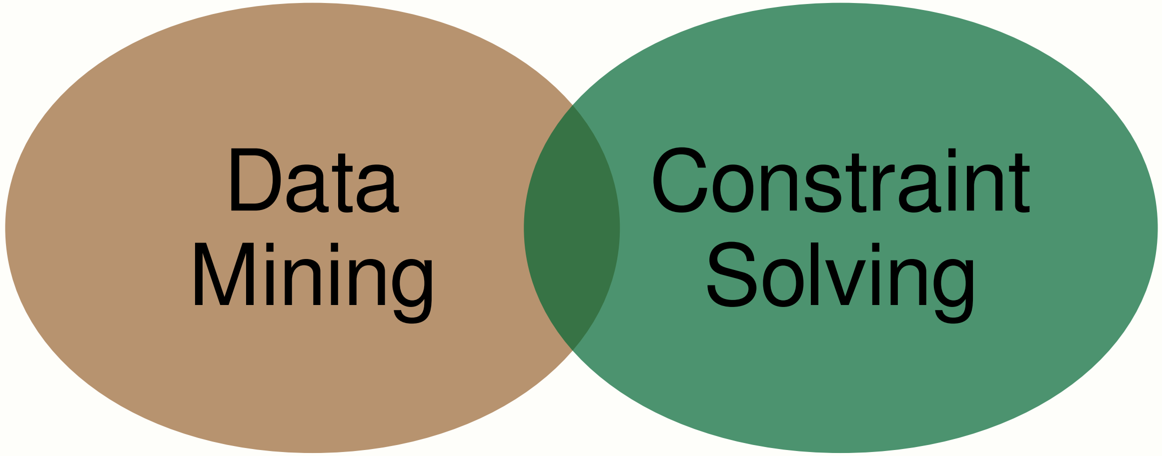
Even more efficient with backtracking-aware datastructures

Wrap-up

Finding the right level of abstraction:

- Modeling:
  - not query but set of constraints
  - solver/algorithm independence
  - automatic rewrite rules

- Solving:
  - each constraint can be made independent
  - vectorize constraints to increase efficiency (internal data structures)
  - CP as framework for pattern mining



Thanks to collaborators:

- L. De Raedt
- S. Nijssen
- A. Dries
- B. Negrevergne
- J. Aoga
- P. Chaus
- T. Le Van
- S. Paramonov
- B. Babaki
- A. Zimmermann
- G. Tack
- K. Marchal
- H. Sun
- A. Jiminez

Questions?

For more pointers, see:
AIJ Special Issue March 2017: Combining Constraint Solving with Mining and Learning
IJCAI 2017 tutorial: Data Mining and Machine Learning using Constraint Programming Languages