University of BRISTOL

Making Sense of (Multi-)Relational Data

Part II: Exploration through targeted patterns

Jefrey Lijffijt

Eirini Spyropoulou

Tijl De Bie

Approaches

- Safarii / “Multi-Relational Data Mining”
- RDB-Krimp
- Inductive Logic Programming

Safarii / "Multi-Relational Data Mining"

Data

- Relational database / E-R model

Knobbe (2004)

Molecule

id	name
1	CO_2
2	H_2O

Atom

id	mol_id	element
1	1	C
2	1	O
3	1	O
4	2	H
5	2	H
6	2	O

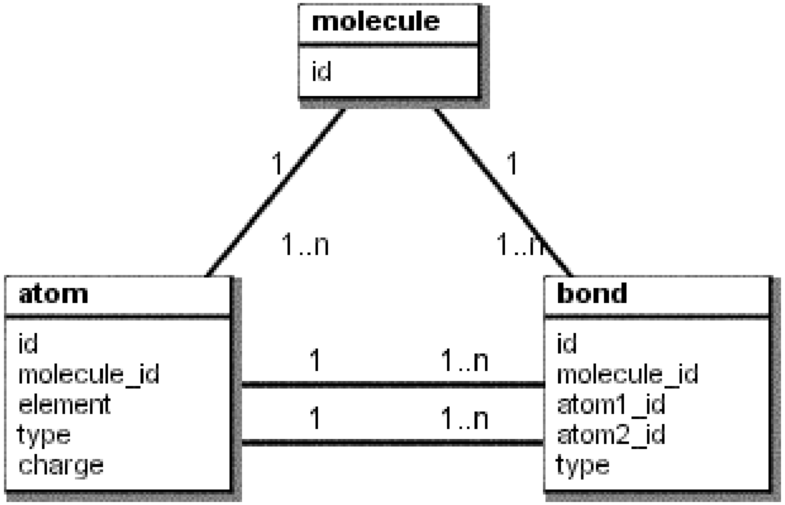
Bond

id	atom1	atom2	type
1	1	2	double
2	1	3	double
3	4	6	single
4	5	6	single

Figure 3.2 Relational representation of CO_2 and H_2O.

Data

- Relational database / E-R model



Knobbe (2004)

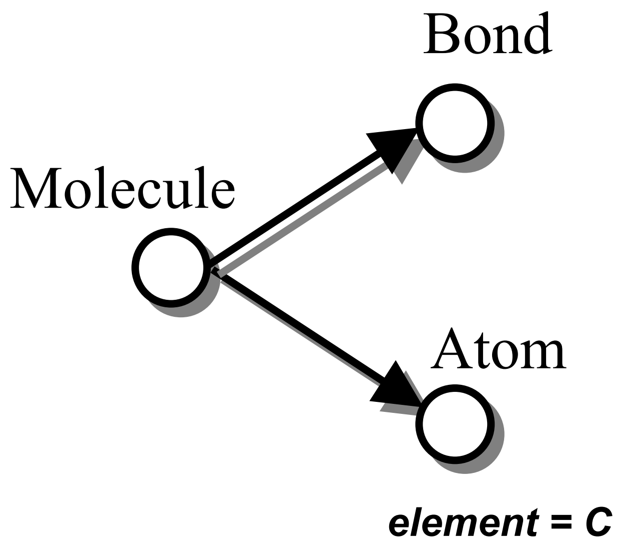
Example pattern

- Pattern =

  all molecules with

  at least one bond

  and a *C* atom



Knobbe (2004)

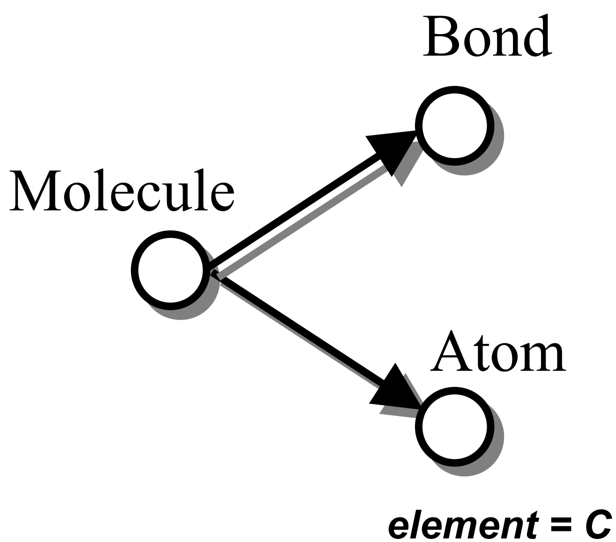
Pattern syntax

- *Individuals* are records in the *target table*, along with its associations and associated parts
  - The units which we want to predict/describe

- A *subgroup* is a set of individuals

Pattern syntax

- *Pattern = subgroup = selection graphs*



Knobbe (2004)

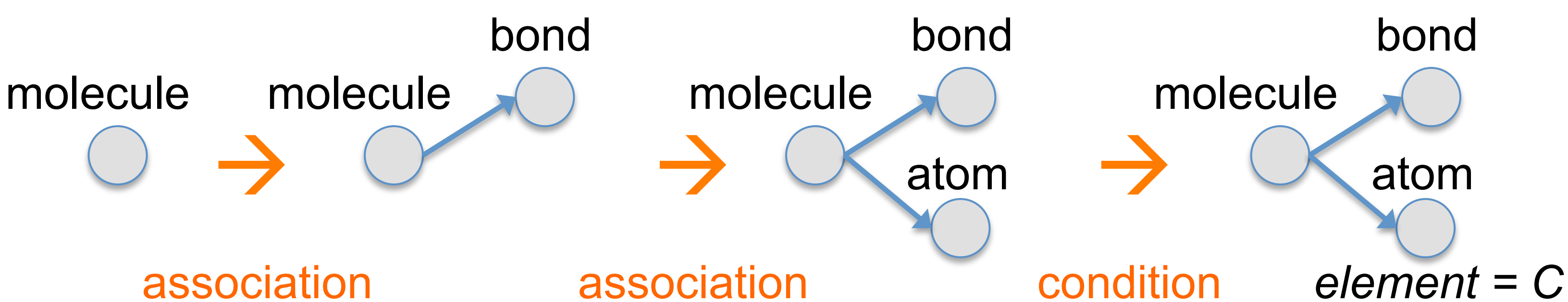
Pattern syntax

- *Pattern = subgroup = selection graphs*

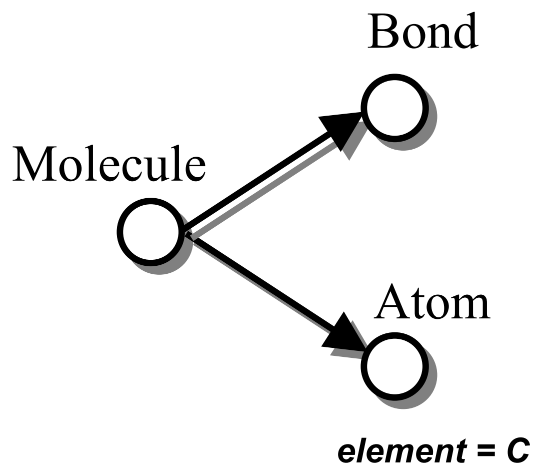
- Mining is then *refinement* of selection graphs
  - Conditioning: choose subset of values (=, ≥, ≤)
  - Association: add an association

Pattern syntax

- Simple case: condition & association refinement



Pattern syntax



Molecule

id	name
1	CO_2
2	H_2O

Atom

id	mol_id	element
1	1	C
2	1	O
3	1	O
4	2	H
5	2	H
6	2	O

Bond

id	atom1	atom2	type
1	1	2	double
2	1	3	double
3	4	6	single
4	5	6	single

Knobbe (2004)

Figure 3.2 Relational representation of CO_2 and H_2O.

Implementation of MRDM: Safarii

- Relational association rule discovery

- Find refinements using *aggregation*
  - Categorical: select an attribute-value
  - Numerical: exists ≤,≥ , min ≤, max ≥
    - These are SQL primaries, there are many more possibilities

Algorithmic approach

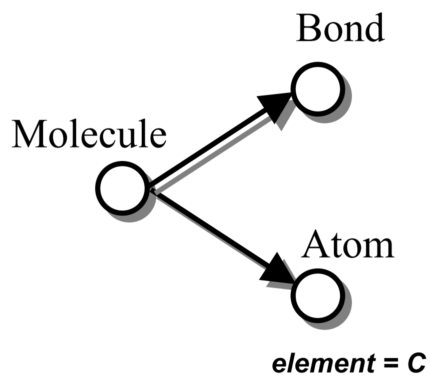
- Restrict # associations, # refinements
- Generate SQL queries, push workload to DB

- Aggregation is greedy
    - Choose only optimal split at runtime
    - Essentially a form of *local discretisation*

Knobbe (2004)

Algorithmic approach

- Patterns are also SQL queries




```
SELECT DISTINCT T0.id
FROM molecule T0, bond T1, atom T2
WHERE T0.id = T1.molecule_id and T0.id = T2.molecule_id
AND T2.element = 'C'
```

Interestingness

- Several objective interestingness measures
  - support($S \rightarrow T$) = $P(ST)$
  - coverage($S \rightarrow T$) = $P(S)$
  - accuracy($S \rightarrow T$) = $P(T \mid S)$
  - specificity($S \rightarrow T$) = $P(\neg S \mid \neg T)$
  - sensitivity($S \rightarrow T$) = $P(S \mid T)$
  - novelty($S \rightarrow T$) = $P(ST) - P(S) \cdot P(T)$

Interestingness

- Several objective interestingness measures
- Steer aggregation
- Rank rules

Predicting bad loans

Knobbe (2004)

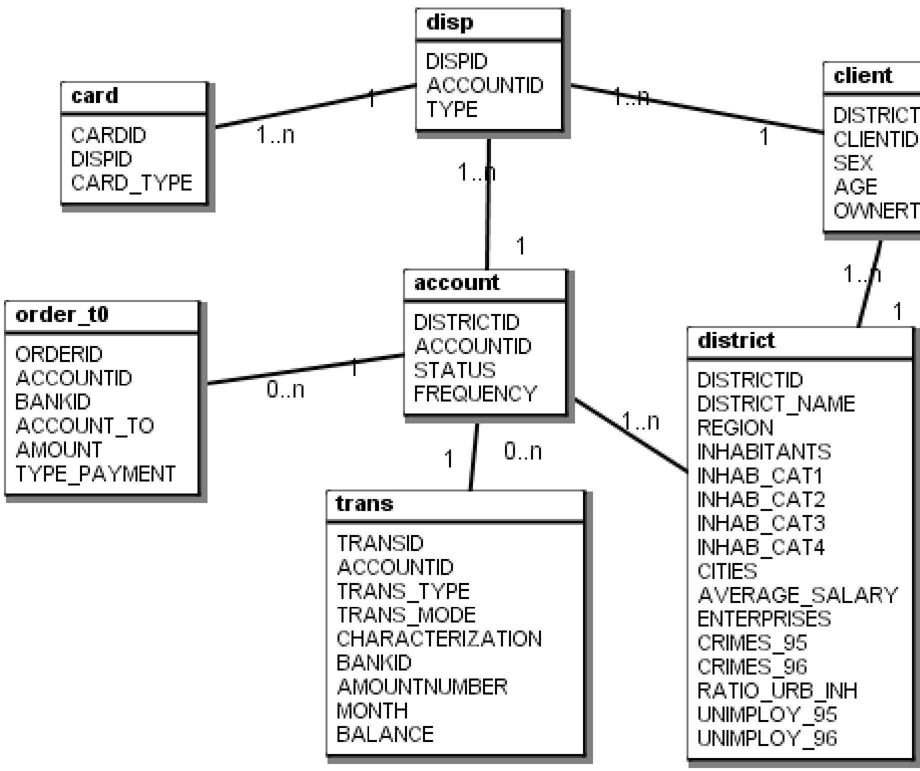
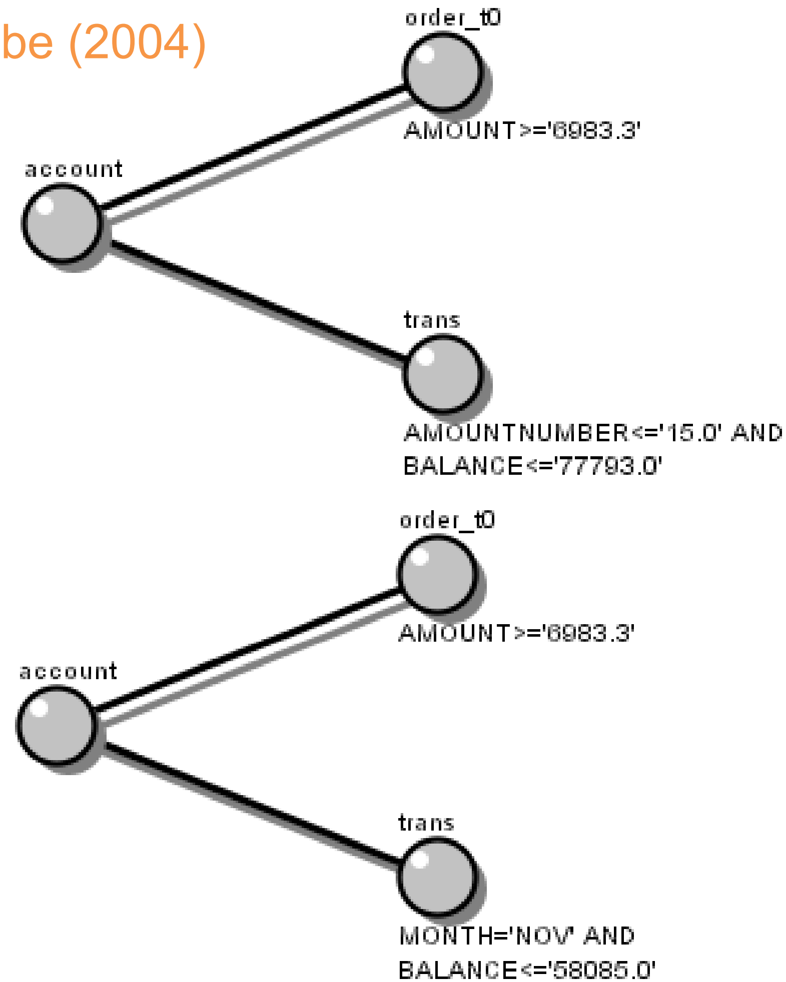


Figure 3.4 The Financial database.



RDB-Krimp

Data

- Relational database / E-R model
- Table defined as: key, foreign keys, attributes
  - Categorical attributes only

Koopman & Siebes (2009)

T^i			
K^i	F^i_j	A^i_1	A^i_2
k	k_j	v_1	v_2

T^1 = ACCOUNT		
$account_{ID}$	Frequency	Date
10	2	06/2007
11	3	03/2006
12	3	08/2006
13	2	03/2006
14	1	05/2008

T^2 = LOAN					
$loan_{ID}$	$account_{ID}$	Date	Amount	Duration	Payment
30	10	06/2008	10245	12	A
31	10	09/2008	13722	24	B
32	11	08/2006	27313	36	B
33	12	09/2006	27147	12	B
34	12	05/2008	27194	36	D
35	13	09/2008	30289	12	B
36	13	06/2008	18203	12	C

T^3 = ORDER					
$order_{ID}$	$account_{ID}$	Bank-To	Amount-To	Amount	Type
20	10	ST	141	1000	UVER
21	10	QR	359	2000	SIPO
22	11	YZ	850	1000	SIPO
23	13	ST	283	1000	NULL
24	13	OP	850	2000	SIPO

T^4 = DISPOSITION		
$disp_{ID}$	$account_{ID}$	Type
40	10	OWNER
41	11	DISPONENT
42	11	OWNER
43	12	DISPONENT
44	12	OWNER

Pattern syntax

- Given a target table T with key K

- Pattern =

  selection of attribute values (conj + disj) of T

  & selection of attribute values (conj + disj) for tables with K as foreign key

Koopman & Siebes (2009)

P_1 : ACCOUNT({ Frequency = 2 })
[[ORDER({ Bank-To=ST, Amount=1000 }),
ORDER({ Amount=2000, Type=SIPO })] ,
[[LOAN({ Date='06/2008', Duration=12 }),
LOAN({ Date='09/2008', Payment=B })]]

P_2 : ACCOUNT({ Frequency = 3 })
[[DISPOSITION({ Type = Disponent }),
DISPOSITION({ Type = Owner })]]

frequency(P_1) = 2, count(P_1) = 10, size(P_1) = 9

frequency(P_2) = 2, count(P_2) = 6, size(P_2)=3

Partially Covered Database

T^1 = ACCOUNT

account$_{ID}$	Frequency	Date
10	2	06/2007
11	3	03/2006
12	3	08/2006
13	2	03/2006
14	1	05/2008
15	3	03/2006
16	2	06/2007

T^2 = LOAN

loan$_{ID}$	account$_{ID}$	Date	Amount	Duration	Payment
30	10	06/2008	10245	12	A
31	10	09/2008	13722	24	B
32	11	08/2006	27313	36	B
33	12	09/2006	27147	12	B
34	12	05/2008	27194	36	D
36	13	06/2008	18203	12	C
35	13	09/2008	30289	12	B

ReOrderDB

T^3 = ORDER

ordern$_{ID}$	account$_{ID}$	Bank-To	Amount-To	Amount	Type
20	10	ST	141	1000	UVER
21	10	QR	359	2000	SIPO
22	11	YZ	850	1000	SIPO
23	13	ST	283	1000	NULL
24	13	OP	850	2000	SIPO

T^4 = DISPOSITION

disp$_{ID}$	account$_{ID}$	Type
40	10	OWNER
41	11	DISPONENT
42	11	OWNER
43	12	DISPONENT
44	12	OWNER

Algorithmic (enumeration) approach

- Run FARMER for every table in DB as target
- FARMER (Nijssen & Kok, 2003) is an ILP algorithm for enumeration of frequent 'queries'
- Exhaustive search with *minsup* threshold

Interestingness

- Main contribution of RDB-Krimp

- Find **concise** set of local patterns that together describe the DB well

  → Minimum Description Length principle

    - Two part code L(H) + L(D|H)

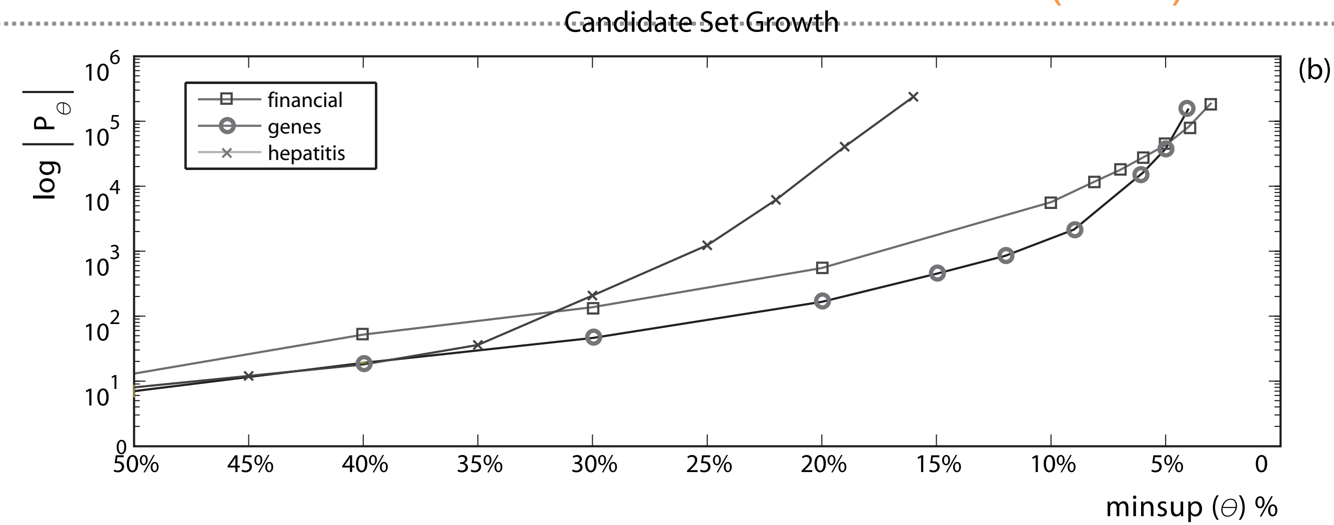
Interestingness

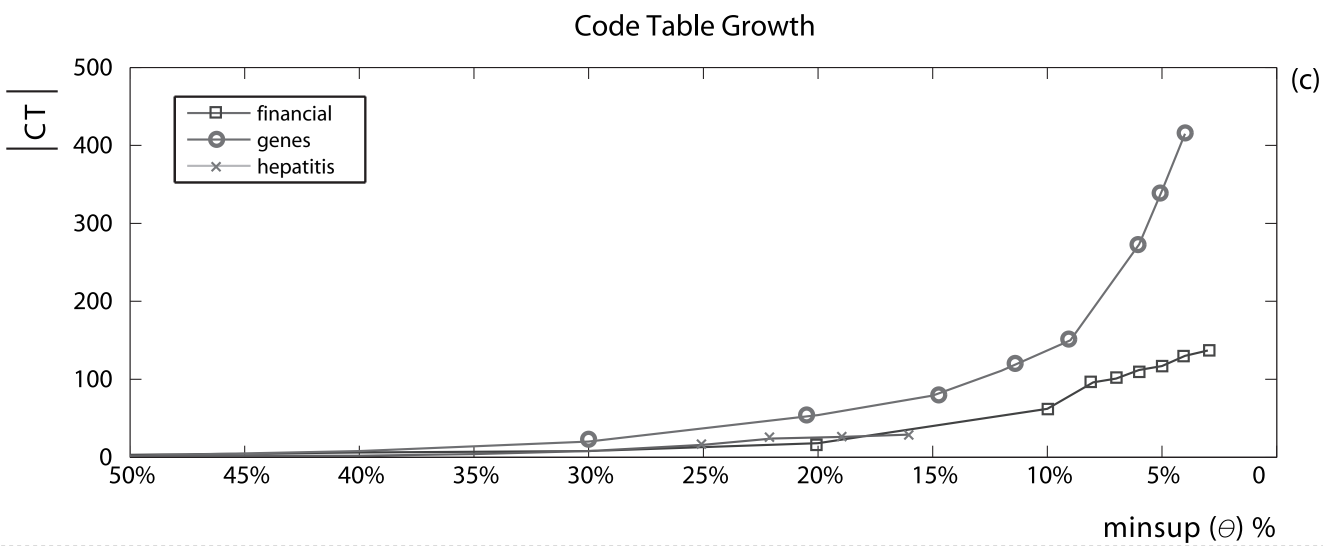
- Greedy approximation algorithm:

1. Initialise pattern set as all singletons
2. Try insert patterns one by one
    - Keep if total description length decreases

- No guarantees on optimality

Koopman & Siebes (2009)





Inductive Logic Programming

(Probabilistic) Inductive Logic Programming

- Field of research
  - Also related to / equivalent with *probabilistic logic learning*, *statistical relational learning*, *logical and relational learning*
- Ultra brief review
  - We are not experts
  - Too much to cover

Data

- Logical representation
  - Also an E-R model ?

Pattern syntax

- Generalises all pattern mining syntaxes discussed here
- Can derive predicates (rules)
  - Can have no antecedent → association but not ‘rule’
  - Terms can be variables rather than constants

daughter(C, P) :− female(C), mother(P, C)

[De Raedt & Kersting, 2008]

Algorithmic approach

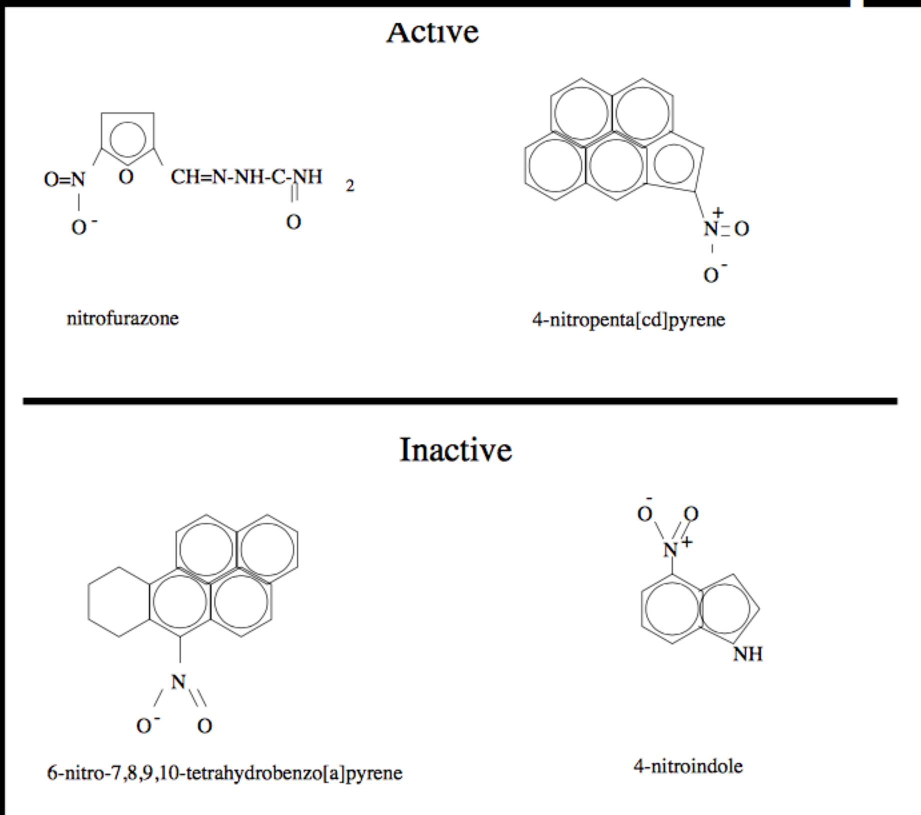
- Very different terminology
  - Logic, but
  - Various frameworks (entailment, interpretations, proofs)
- Also based on ‘generality’ (= monotonicity)
- Search can easily become very costly

Interestingness

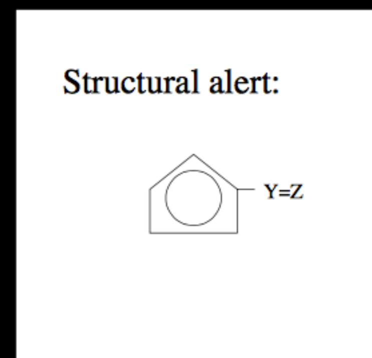
- Objective interestingness measures have been employed
  - Frequency
  - Confidence
  - …

De Raedt (2008)

Case 1: Structure Activity Relationship Prediction



[Srinivasan et al. AIJ 96]



General Purpose
Logic Learning System

Data = Set of Small Graphs

Uses and Produces
Knowledge

De Raedt (2007)

Dehaspe's Warmr ~ Apriori

PARTICIPANT Table

NAME	JOB	COMPANY	PARTY	R_NUMBER
adams	researcher	scuf	no	23
blake	president	jvt	yes	5
king	manager	ucro	no	78
miller	manager	jvt	yes	14
scott	researcher	scuf	yes	94
turner	researcher	ucro	no	81

SUBSCRIPTION Table

NAME	COURSE
adams	erm
adams	so2
adams	srw
blake	cso
blake	erm
king	cso
king	erm
king	so2
king	srw
miller	so2
scott	erm
scott	srw
turner	so2
turner	srw

COMPANY Table

COMPANY	TYPE
jvt	commercial
scuf	university
ucro	university

COURSE Table

COURSE	LENGTH	TYPE
cso	2	introductory
erm	3	introductory
so2	4	introductory
srw	3	advanced

- Knobbe, Arno (2004). *Multi-Relational Data Mining*. PhD Thesis, Utrecht University.
- Koopman, Arne & Siebes, Arno (2009). "Characteristic Relational Patterns". In *Proc. of KDD 2009*, pp 437–446, ACM, New York.
- De Raedt, Luc & Kersting, Kristian (2008). *Probabilistic Inductive Logic Programming*, Springer.
- De Raedt, Luc (2007). "Logic, Probability and Learning". Tutorial at ACAI.
- De Raedt, Luc (2008). "Logical and Relational Learning Revisited". Tutorial at ICML.