MONDRIAN: Annotating and querying databases through colors and blocks

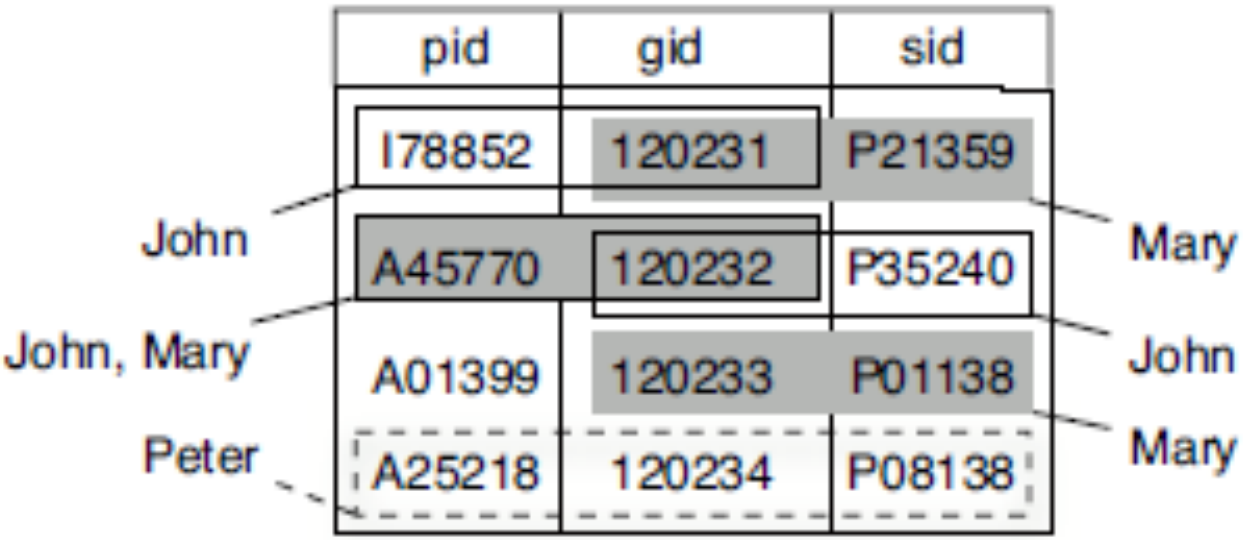
Scientific databases

- Scientific databases play a central role in the advancement of science by providing access to large collections of data.
- Interest to computer scientists due to the data management challenges
- scientific data come in a variety of formats which range from flat-formatted files to images and electronic publications

Challenges

- *integrate :Different data types*
- A*nnotate:* new data is created which are used by other scientists *comments on Data*
- *cross-reference*
- maintain *data provenance*
- DB used :GDB (a human gene database), Swissprot (a protein database),and PIR (a protein sequence database).

Annotation



(d) An integrated relation

Contribution

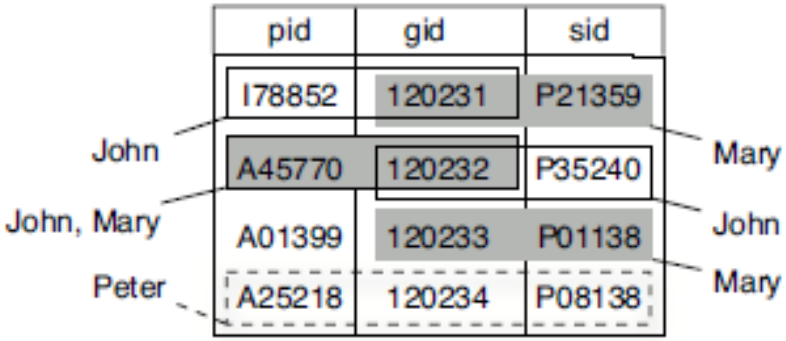
- Support annotation of *sets of values*.
- a block is used to indicate the set of values for which an annotation exists
- compared to its unannotated relations, the annotation of the database imposes minimum overhead both in terms of
  - space
  - query execution time
- *color algebra* a query language are that it is at a level of abstraction that is independent of the chosen representation of annotations

Contribution (cont')

- Change in the representation of annotations requires us to reformulate all our queries
- annotation mechanism is capable of annotating both single values and multiple values.
- algebra to query values and annotations is
  - *complete* (it expresses all possible queries over annotated databases)
  - *minimal* (every operator is primitive, and thus necessary).

value-based query vs not a value-based annotation query

- value-based query: might want to find which tuples are annotated by either Mary or John
- not a value-based annotation query: finding which gene-protein sequence (*gid, sid*) pairs are annotated



(d) An integrated relation

Color and blocks

- refer to a set of attribute values as a *block*
- each annotation is represented by a *color*.
- *color databases* (databases that are annotated)
- *color queries* (queries on annotated databases) written using a *color algebra* (an algebra that accounts for annotations)

Notations

- Relation *Ri*, we denote its set of attributes by *sort*(*Ri*), while we use *ri* to denote an instance of the relation.
- upper-case letters alphabet (*A,B, . . .*) to denote attribute names
- uppercase letters late in the alphabet (*X,Y, . . .*) are used to denote sets of attributes.
- lower-case letters alphabet (*a, b, . . .*) are used to denote attribute values
- alphabet (*x, y, . . .*) are used to denote sets of attributes values.
- Finally, *C* denotes a set of colors

Example -1

gid	gname
120231	NF1
120232	NF2
120233	NGFB
120234	NGFR
120235	NHS

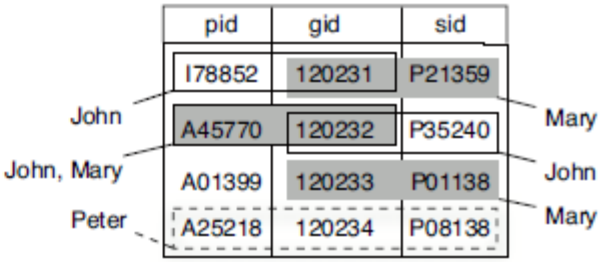
(a) GDB relation

sid	sname
P01138	Beta-NGF
P08138	TNR16
P14543	Nidogen
P21359	Neurofibromin
P35240	Merlin

(b) Swissprot relation

pid	pname
A01399	Nerve growth factor
A25218	Tumor necrosis factor
A45770	Merlin
I78852	Neurofibromatosis
Q6T45	Nancy-Horan syndrome

(c) PIR relation



(d) An integrated relation

Figure 1. Three biological sources and their integrated relation

- $\chi(t1, \{pid, gid\}) = \{John\}$, $\chi(t1, \{gid, sid\}) = \{Mary\}$
- $\chi(t2, \{pid, gid\}) = \{John, Mary\}$ $\chi(t2, \{gid, sid\}) = \{John\}$
- $\chi(t3, \{gid, sid\}) = \{Mary\}$
- $\chi(t4, \{pid, gid, sid\}) = \{Peter\}$

CA (cont')

- Block projection:

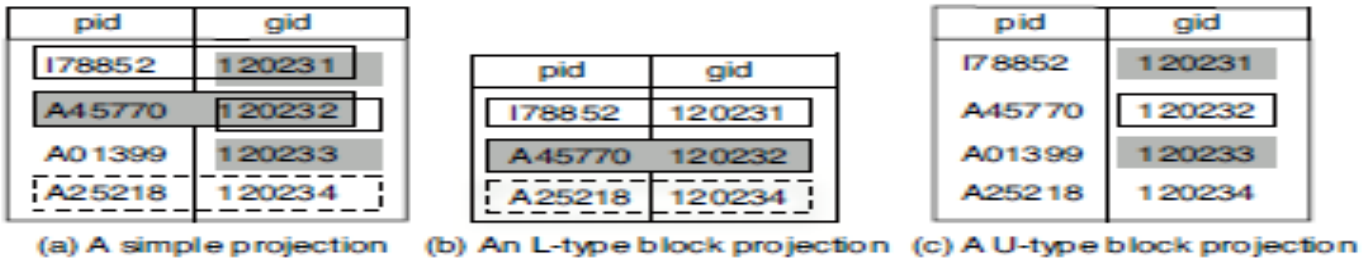


Figure 2. The projection operators

- Figure 2 (b) $\Pi^{L}_{pid}(r)$ and Figure 2 (c) $\Pi^{U}_{gid}(r)$
- The $\Pi^{L}_{gid}(\Pi^{U}_{gid}(r))$ *returns all tuples with a block on*
- *gid alone. These are the first three tuples in Figure 2(c).*

CA (cont')

- Selection:

$$\chi'(t,Y) = \begin{cases} \chi(t,Y) & A,B \notin Y; \\ \chi(t,Y) \cap \beta(t,A) \cap \beta(t,B) & \text{otherwise.} \end{cases}$$

- Where $\beta(t,A)$ (resp. $\beta(t,B)$) is the set of colors of all blocks in t containing attribute A (resp. B).
- The operator Σc, where $c \in C$, takes as input any instance r, χ and returns the instance r, χ of the same sort defined by $r = \{t \mid t \in r$ and there exists a block in t of color $c\}$,
- and for any $t \in r$ and any set of attributes $Y \subseteq sort(R)$,

$\chi(t, Y) = \chi(t, Y) \cap \{c\}$.

CA (cont')

- Product:

$$\chi'(t, Y) = \begin{cases} \chi_r(\pi_{sort(R)}(t), Y) & \text{if } Y \subseteq sort(R); \\ \chi_s(\pi_{sort(S)}(t), Y) & \text{if } Y \subseteq sort(S); \\ \emptyset & \text{otherwise.} \end{cases}$$

- Figure 4 (c) Product of R in 4(a) and R` in 4(b) followed by selection on gid and gid`

pid	gid
I78852	120231
A45770	120232
A25218	120234

(a) A relation

gid	sid
120231	P21359
120232	P35240
120233	P01138
120234	P08138

(b) Another relation

pid	gid	gid'	sid'
I78852	120231	120231	P21359
A45770	120232	120232	P35240
A25218	120234	120234	P08138

(c) Cartesian-product followed by selection

Figure 4. Selection and product operators

CA (cont')

- If we deciide to remove one of the repeted columns , will have 2 different structure as in figure (5) a & b
- Merge: $\mu_{Y,Z}$

$$\chi'(t, X) = \chi(t, X_1) \cap \chi(t, X_2),$$

pid	gid	sid'
I78852	120231	P21359
A45770	120232	P35240
A25218	120234	P08138

(a) Projecting out gid'

pid	gid'	sid'
I78852	120231	P21359
A45770	120232	P35240
A25218	120234	P08138

(b) Projecting out gid

pid	gid'	sid'
I78852	120231	P21359
A45770	120232	P35240
A25218	120234	P08138

(c) Projection after the merge operator is applied

Figure 5. The merge operator

Theorem 1 (Minimality)

- *The set of operators in the color algebra is minimal.*
- color query results in the same set of tuples as relational algebra query on an unannotated database.
- join identifies attributes based on both their values and block structure and merges the "common" blocks.

$$\langle r, \chi_r \rangle \bowtie \langle s, \chi_s \rangle$$

$$\pi_{sort(r) \cup sort(s) \setminus \{A_j\}} (\mu_{sort(r), sort(s)} (\Pi^L_{A_i} (\sigma_{A_i = A_j} (r \times s)) \cup (\Pi^L_{A_j} \sigma_{A_i = A_j} (r \times s))))$$

Connection with relational model

pid	*gid*	*sid*	*bpid*	*bgid*	*bsid*	γ
$I78852$	120231	$P21359$	0	0	0	c
$I78852$	120231	$P21359$	1	1	0	$John$
$I78852$	120231	$P21359$	0	1	1	$Mary$

- Pros:
  - no re-structuring of the existing schemas is necessary.
- cons
  - waste of space

Color relational algebra

- Recall the *CRep is the set of relational databases which* represent a color database through the rep mapping.
- **Theorem 2 (Soundness). For every color database D*, χ and every CA expression Q, there exists a color relational*** algebra (CRA) expression *P such that*
- rep(*Q(**D, χ)) = P(rep(D, χ)).***

Color relational algebra

- CA expression $\sigma Ai=Aj\ (r)$ equivalent to the following CRA expression
- $\sigma Ai=Aj \wedge Bi=0 \wedge Bj=0(rep(r)) \cup \sigma Ai=Aj \wedge Bi=1 \wedge Bj=1(rep(r)) \cup$

$\pi sort(rep(r))(\sigma Ai=Aj \wedge Bi=1 \wedge Bj=0(rep(r))$

$\delta(\sigma Ai=Aj \wedge Bi=0 \wedge Bj=1(rep(r)) \cup$

$\pi\delta(sort(()r))(\sigma Ai=Aj \wedge Bi=1 \wedge Bj=0(rep(r))$

$\delta(\sigma Ai=Aj \wedge Bi=0 \wedge Bj=1(rep(r))$

The MONDRIAN System

- experiments, we used real biological data from the Swissprot database
- two relations for our experiments, namely, relation
- *Protein containing 200,000 protein tuples* (560MB in size), and
- relation *Public that contained four* million tuples that concern publications related to proteins (750MB in size).
- Relation *Protein has eight attributes in its* schema, while relation *Public has five.*

The MONDRIAN System

- two relations as pools
- three different experimental data sets.
- Each set contained five unannotated relations
- The sizes of these five relations varied from 10,000 to 50,000 tuples (in 10,000 tuple increments).
- Thus, the total number of created relations was 30.

The MONDRIAN System

- The annotation process is influenced by three user-specified parameters.
  - The first parameter, called *MaxNo, limits the number of blocks* that can appear in each annotated tuple.
  - The second parameter, called *AvgNo, specifies the average size of each* generated block.
  - The last parameter is the cardinality of *C (the number of available colors)*

Costs of using colors and blocks

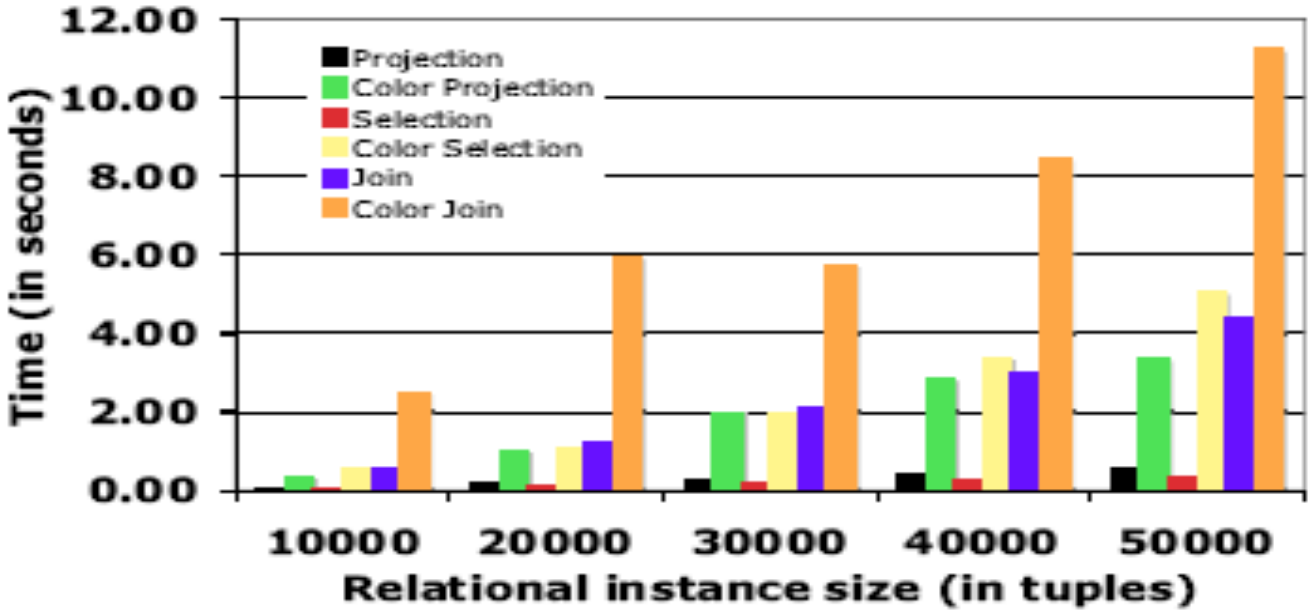


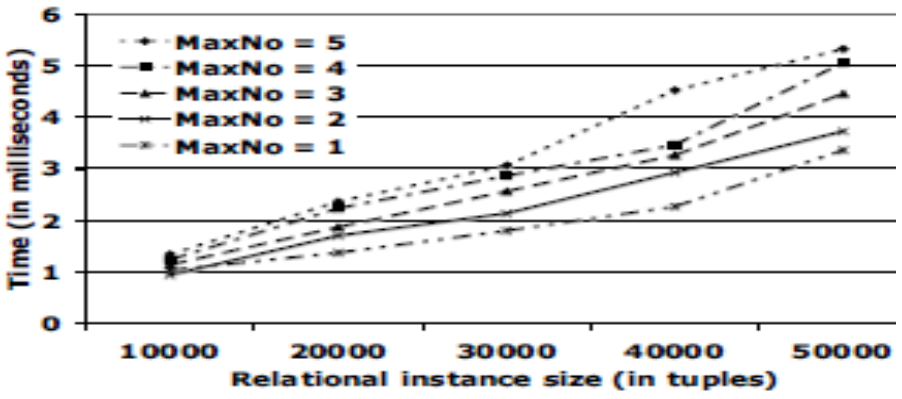
Figure 6. Color vs. normal algebra

Costs of using colors and blocks

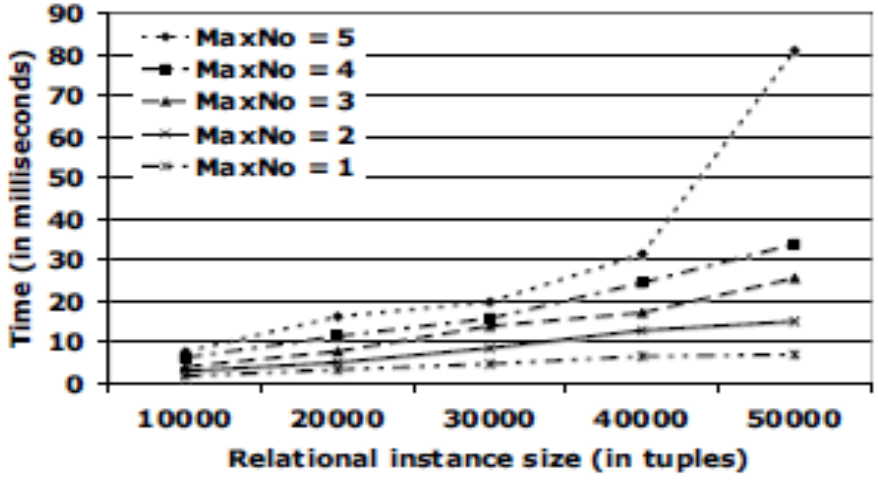
- With *MaxNo equal to three, annotating* a relation with 10,000 tuples results in a relation that is close to 30,000 tuples (assuming single colored blocks).
- Remember that color operators perform extra processing since they also consider Boolean attributes and operate on them.

Query evaluation cost parameters

1. we annotated the relations in our experimental data sets once for each value of *MaxNo* between one and five. The *AvgNo* parameter was set to three and the *C* was 100.



(a) Selection on a data value



(b) Block selection and projection

Query evaluation cost parameters

- selections on data values, is heavily influenced by the maximum number of blocks per tuple.
- As the number blocks increases tends to increases the number of tuples in the underlying representation.
- single-colored blocks, for an annotated relation with X tuples, there are $(MaxNo + 1) \times X$ representation tuples

Query evaluation cost parameters

2- second configuration, we annotated relations of various sizes by varying, this time, the AvgNo parameter between the values of one and five. Parameter MaxNo was set to three and C to 100.

- *AvgNo* parameter had negligible effects on the running times of various queries, for a fixed number of annotated tuples.
- The only change, representation-wise, is that more Boolean attributes are set to 1, instead of 0.

Query evaluation cost parameters

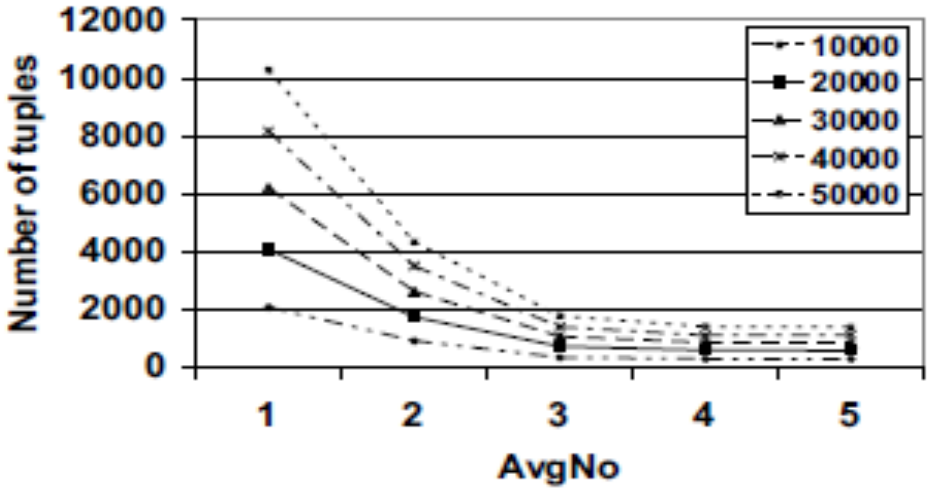


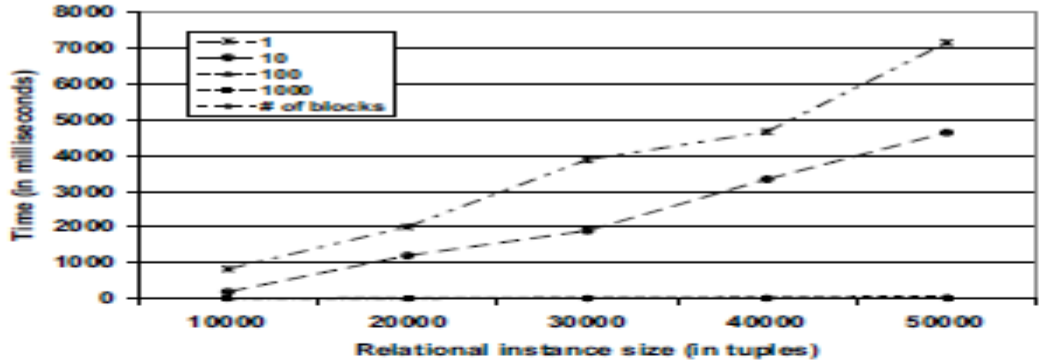
Figure 8. Result size of the Π^U operator

Query evaluation cost parameters

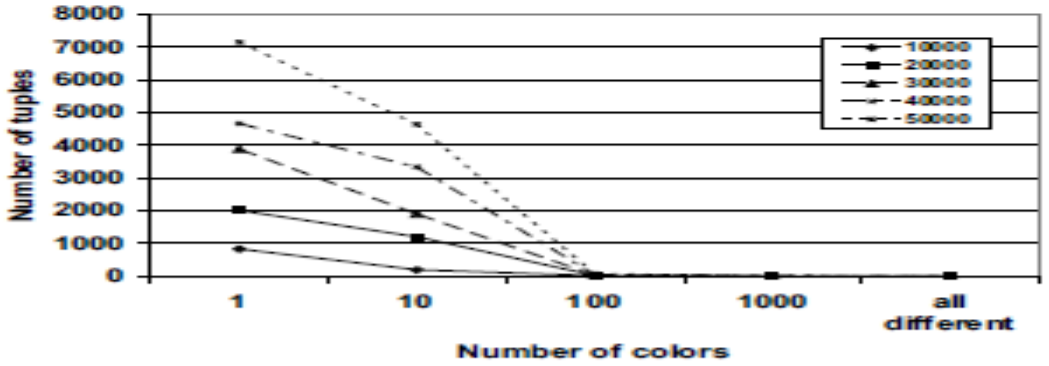
3- last configuration considered annotated relations where both MaxNo and AvgNo were set to three. Here, five different values where considered for C, namely, 1, 10, 100, 1000 and as many colors as there are blocks.

-Figure 9(a) shows, when C is 10, there is a sharp increase on the evaluation time of the operator. The reason for this is shown in Figure 9(b). With less available colors, there is a large number of blocks sharing a color.

Query evaluation cost parameters



(a) Evaluation time of block selection



(b) Result tuples of block selection

Figure 9. Block selection as C varies

Conclusions and future work

- it allows annotating not only values but also sets of values
- introduced a color algebra, and we proved that it is both complete and minimal
- colored databases provide the right framework to answer data provenance questions.