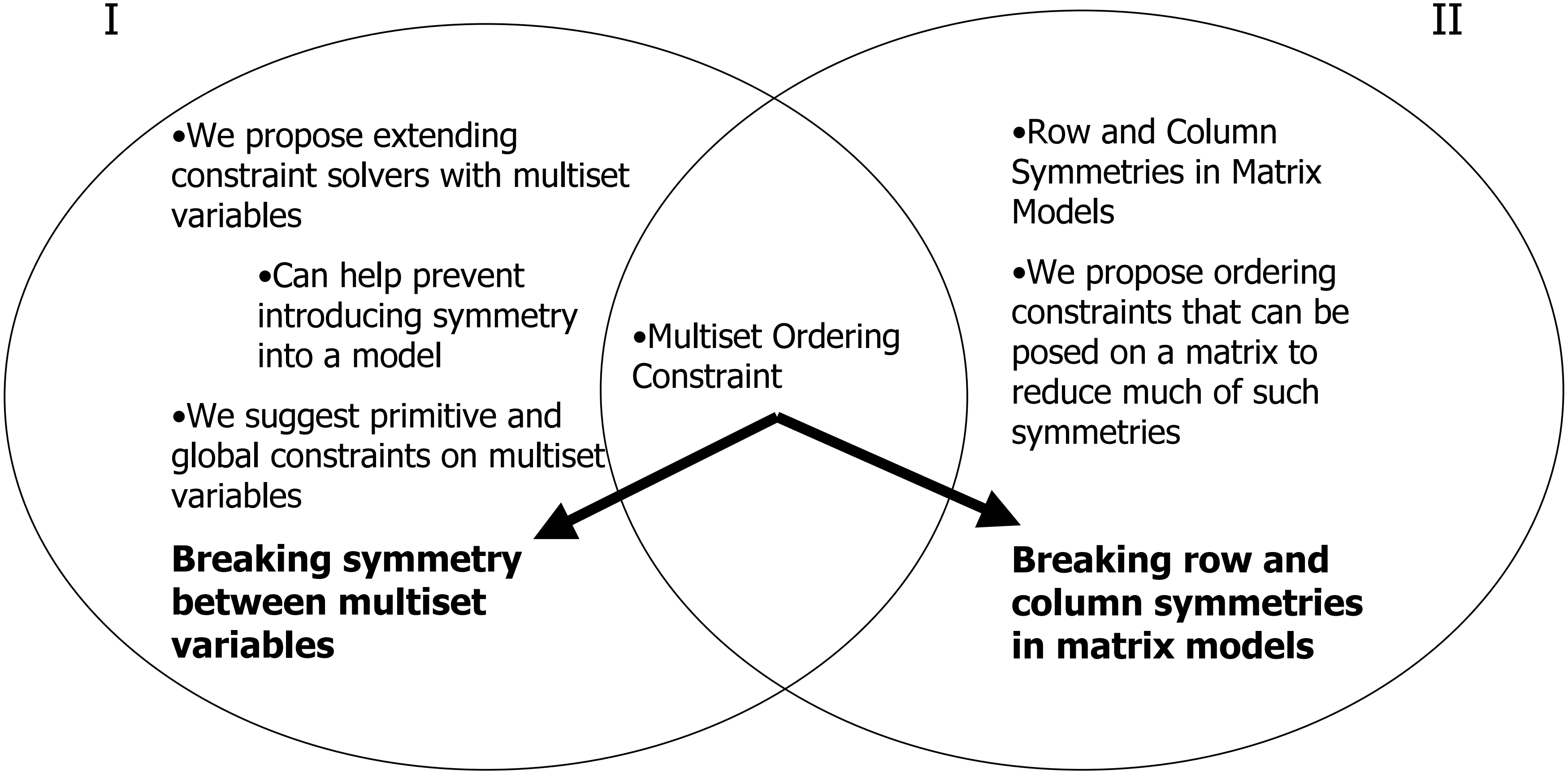
Why Joint?


I
II
•We propose extending constraint solvers with multiset variables
•Can help prevent introducing symmetry into a model
•We suggest primitive and global constraints on multiset variables
Breaking symmetry between multiset variables
•Multiset Ordering Constraint
•Row and Column Symmetries in Matrix Models
•We propose ordering constraints that can be posed on a matrix to reduce much of such symmetries
Breaking row and column symmetries in matrix models

Part I

Constraint Programming
with Multisets

Motivation

- Symmetry often occurs in many problem
  - Configuration
  - Scheduling
  - Assignment
  - Routing
  - ...
- Dealing efficiently and effectively with symmetry is a challange
- Sometimes symmetry is inherently present in the problem
- Sometimes modelling introduces unnecessary symmetry

Template Design Problem

- Prob002 in CSPLib
  - Cat food labels to be printed on templates
  - Several designs (tuna, chicken, …) on each template
- Assign designs to each printing template

Model 1

- Variables
  - There are a fixed number of slots on each template (s), and of templates (t)
  - t*s variables, S_{ij}
    - The design assigned to slot i on template j
- Symmetry!
  - Slots within a template are essentially indistingusihable
    - Slots of a template can be permuted
  - Templates are essentially indistingusihable
    - Two templates can be permuted

Model 2

- Variables
  - T_i (one for every template)
    - Multiset of designs assigned to template T_i
- A multiset is a set with repetitions
  - M={{ 0,1,1,2,2,3}}
- Multiset as the designs on a template are often repeated
- Symmetry between slots on a template is eliminated
- Still symmetry!
  - Templates are essentially indistingusihable
  - We can order multisets to break this symmetry

Overview Of the Talk

- Discuss how to represent multiset variables
- Compare the representations
- Suggest primitive and global constraints on multiset variables
- Introduce multiset ordering
  - Discuss how to enforce this as a constraint
- Concluding Remarks

Representing Multisets

- Bounds Representation
  - Generalises the bounds representation of set variables by Gervet
  - For each multiset variable M, keep

    greatest lower bound (glb) and least upper bound (lub)
  - Compact but cannot represent all forms of disjunction
    - M is either {{0}} or {{1}}
    - glb(M)={{ }}, lub(M)={{0,1}}
    - But {{ }} and {{0,1}} are also in the domain

Representing Multisets

- Occurrence Representation
  - Generalises the characteristic function of set variables
  - Each multiset variable M is represented by a vector **m**

    $\mathbf{m_1}$, $\mathbf{m_2}$, **…**, $\mathbf{m_d}$

    $\mathbf{m_i}$ **= #**occurrences(i,M)

  - Compact but cannot represent all forms of disjunction
    - M is either {{0}} or {{1}}
    - $\mathbf{m_0}$ in 0..1**,** $\mathbf{m_1}$ in 0..1
    - But {{ }} and {{0,1}} are in the domain

Representing Multisets

- Fixed Cardinality
  - Multisets of fixed cardinality are common
    - template design problem: every template has a fixed number of slots for designs on each template
  - Each multiset variable M is represented by a vector **M**

    $\mathbf{M_1, M_2, \ldots, M_k}$

    $\mathbf{M_i}$ represents an element of multiset M

    k = cardinality of multiset
  - Remember Model 1 of template design problem!
  - Need to post (more) complex constraints to overcome symmetry
  - Compact but cannot represent all forms of disjunction

Comparing Representations

- One representation is **as expressive** than another
  - it can represent the same sets of multiset values
- One representation is **more expressive** than another
  - it is as expressive
  - there is one set of multiset values that it can represent that the other cannot

Comparing Representations

- For set variables
  - Bounds and occurrence representations are as expressive as each other
- For multiset variables
  - Occurrence representation is more expressive than the bounds
    - M takes either {{ }} or {{0,0}}
    - $\mathbf{m_0}$ in {0,2}
    - glb(M)={{ }}, lub(M)={{0,0}} which permits M to take {{0}}
  - Fixed cardinality is incomparable to both the bounds and the occurrence representations

Multiset Constraints

- Primitive Constraints
  - $M = N$
  - $M \subset N$
  - $M \cup N$
  - $M \cap N$
  - $M - N$
  - $x \in M$
  - $|M|$
- Global Constraints
  - disjoint($[M_1,\ldots,M_n]$)
    - $Mi \cap Mj=\{\{\ \}\}$ for all $i \neq j$
  - partition($[M_1,\ldots,M_n],M$)
    - $Mi \cap Mj=\{\{\ \}\}$ for all $i \neq j$
    - $M \cup \ldots \cup M_n = M$
  - distinct($[M_1,\ldots,M_n]$)
    - $Mi \neq Mj$ for all $i \neq j$

Global Constraints

- Does decomposition hurt?
- GAC on **alldifferent**$[X_1,\ldots,X_n]$ $\longrightarrow$ AC on $X_i \neq X_j$ for all $i \neq j$
- Surprisingly there are global constraints where decompisition does not hurt
  - GAC on disjoint($[M_1,\ldots,M_n]$) $\leftrightarrow$ AC on the decomposition
  - GAC on partition($[M_1,\ldots,M_n],M$) $\leftrightarrow$ AC on the decomposition

Ordering Multisets

- For breaking symmetry between multisets
  - Model 2 of template design problem
    - Each template is a multiset
    - Templates are essentially indistingusihable
    - We can break symmetry by imposing

$$T_1 \leq_m T_2 \leq_m T_3 \leq_m T_4$$

- For breaking row and column symmetries in matrix models
  - Treat each row as a multiset
  - Insist that rows, when viewed as multisets, are ordered
  - Wait for the second half of my talk ☺

What is Multiset Ordering?

- Fixed cardinality multisets
- M $<_m$ N iff
  - x=max(M), y=max(N)
  - x<y OR (x=y AND M-{{x}} <m N-{{y}})

R_1=[1,2,3,2,3,1,1,2] → {{1,1,1,2,2,2,3,3}}

R_2=[1,1,3,1,3,1,1,3] → {{1,1,1,1,1,3,3,3}}

$R_1 <_m R_2$

Concluding Remarks

- Many constraint solvers support sets
- But very few (if any) current solvers support multisets
- Many problems can naturally be modelled using multisets
- Multiset variables can help prevent introducing symmetry into a model

Part II

Symmetry-Breaking Constraints
for Matrix Models

Motivation

- Row and column symmetries of an **n** x **m** matrix model
  - Rows and columns can be permuted
  - Super-exponential number of symmmetries (**n**! X **m**!)
- Eliminating all symmetries is not easy
  - Symmetry breaking methods have to deal with very large number of symmetries
  - The effort required could easily be exponential
- Challange: How can we reduce significant amount of symmetries with only a polynomial effort?

Previous Work

- Constrain the rows and columns to be lexicographically ordered (double-lex)
  - 0(n) symmetry-breaking constraints
  - Listen to my talk tomorrow ☺
- A subset of full SBDS functions
  - Row transpositions, column transpositions, all combinations of a row transposition and a column transposition
    - $0(n^4)$ SBDS functions
- Adding symmetry-breaking constraints + running SBDS
  - Constrain the rows to be ordered by their sums (row-sum)
  - $O(n^2)$ column transpositions (col-trans)

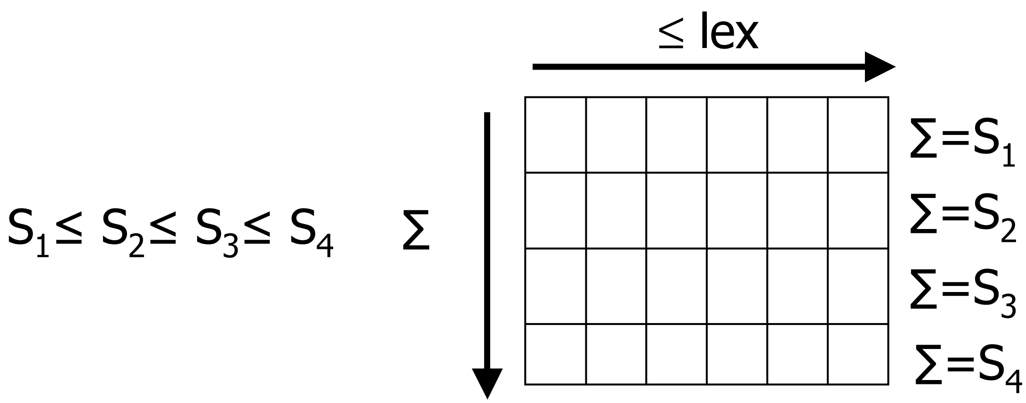
Advantages

- Many symmetries are eliminated
- The effort is polynomial
- Practical for large matrices

Outline of Rest of Talk

- Symmetry Breaking Constraints
- Experimental Comparison
- Discussions

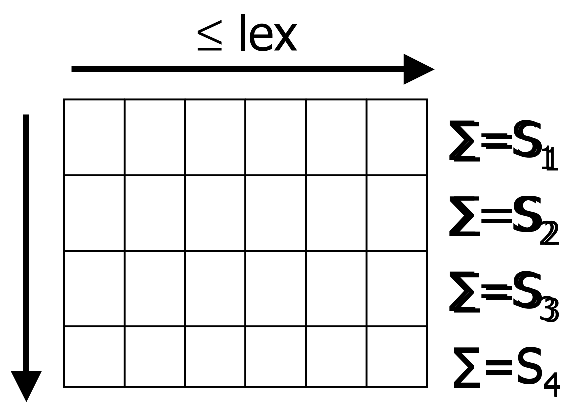
Col-lex + row-sum



- All symmetry broken if **alldifferent**(S_1, S_2, S_3, S_4)
- Symmetries remain if ¬**alldifferent**(S_1, S_2, S_3, S_4)

Col-lex + row-sum(+ row-lex)

$S_1 \leq S_2 \leq S_3 \leq S_4$ Σ



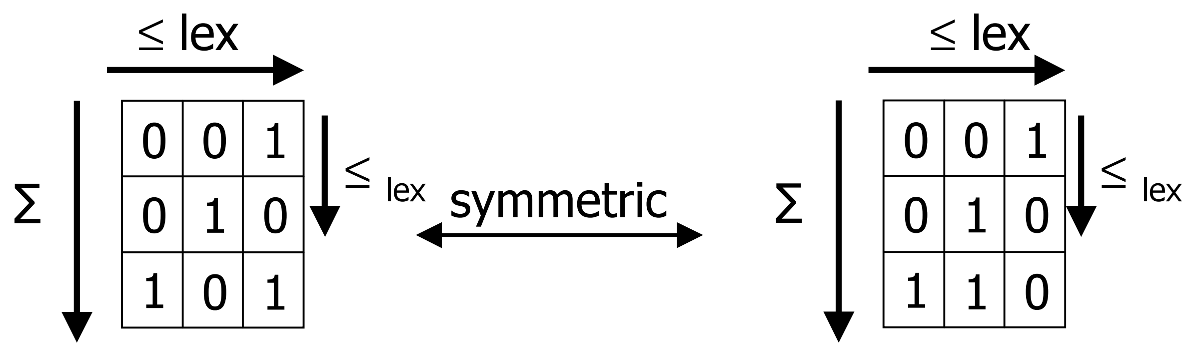
$S_1 = S_2 \rightarrow S_1 \leq_{lex} S_2$

$S_3 = S_4 \rightarrow S_3 \leq_{lex} S_4$

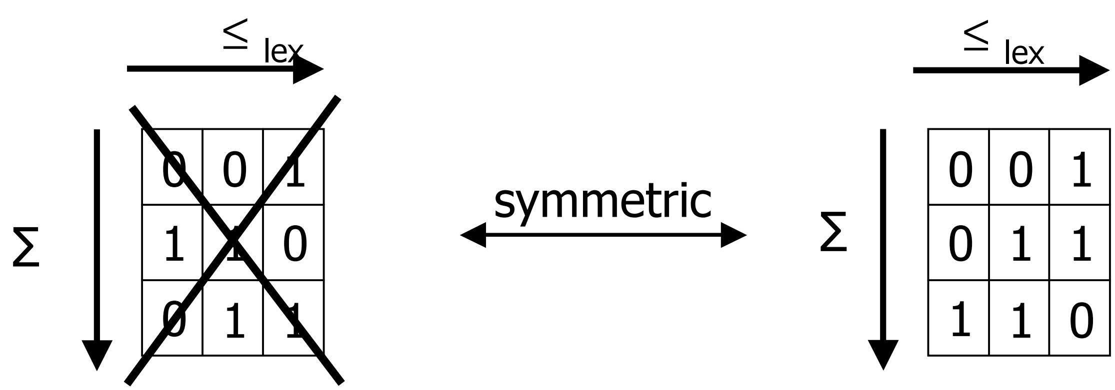
- $S_1 = S_2 = S_3 = S_4$: double-lex
- **alldifferent**(S_1, S_2, S_3, S_4): all symmetry broken
- Combines double-lex and row-sum

Col-lex + row-sum(+ row-lex)

- Not all symmetries are eliminated



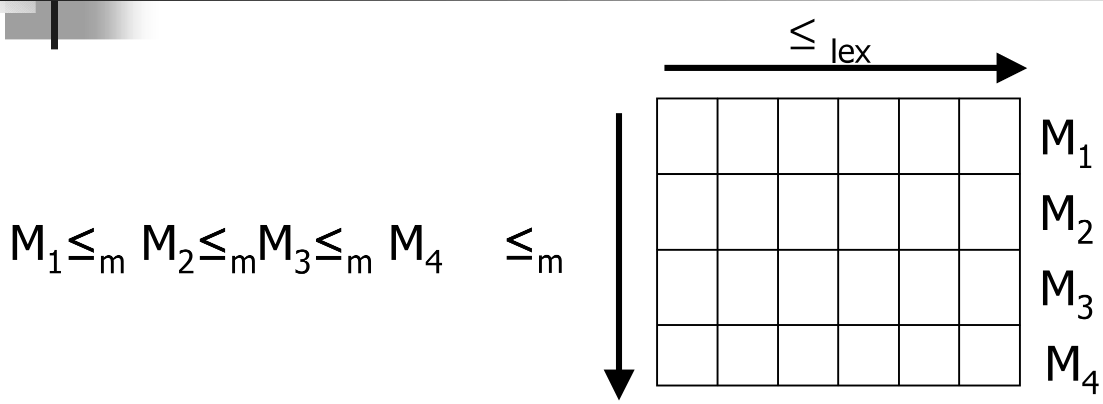
- Stronger than col-lex+row-sum



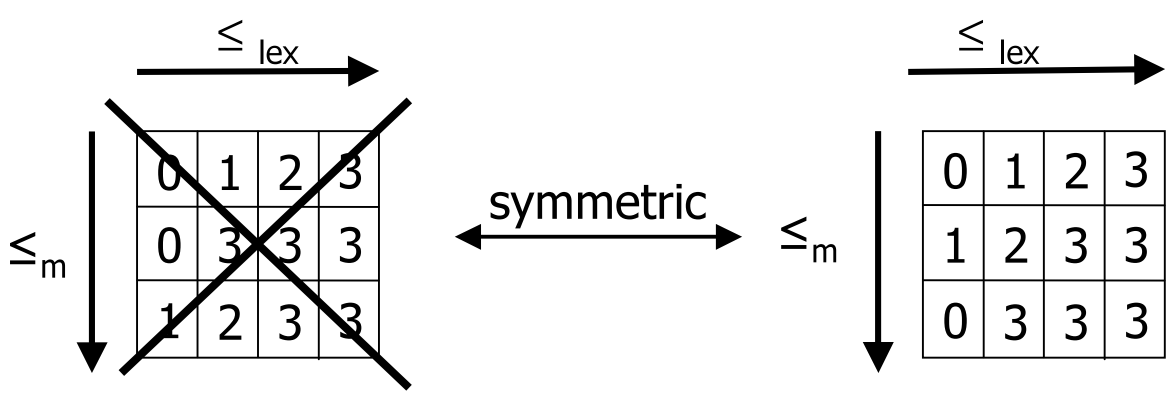
Rows as Multisets

- Treat each row as a multiset represented by fixed cardinality representatin
- Insist that rows, when viewed as multisets, are ordered
- Replace row-sum ordering constraints by multiset ordering constraints

Col-lex + row-multiset



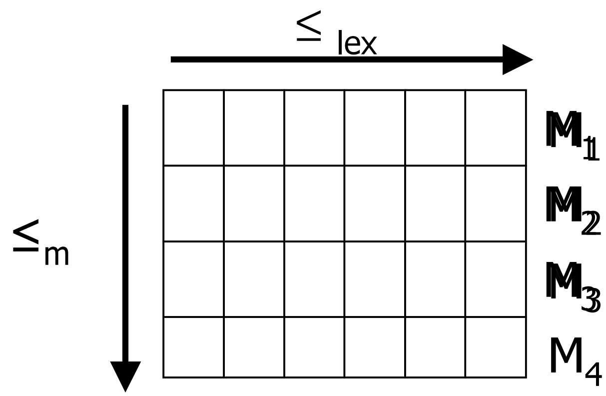
- Stronger than col-lex+row-sum



- Equivalent to col-lex+row-sum for 0/1 variables

Col-lex + row-multiset(+ row-lex)

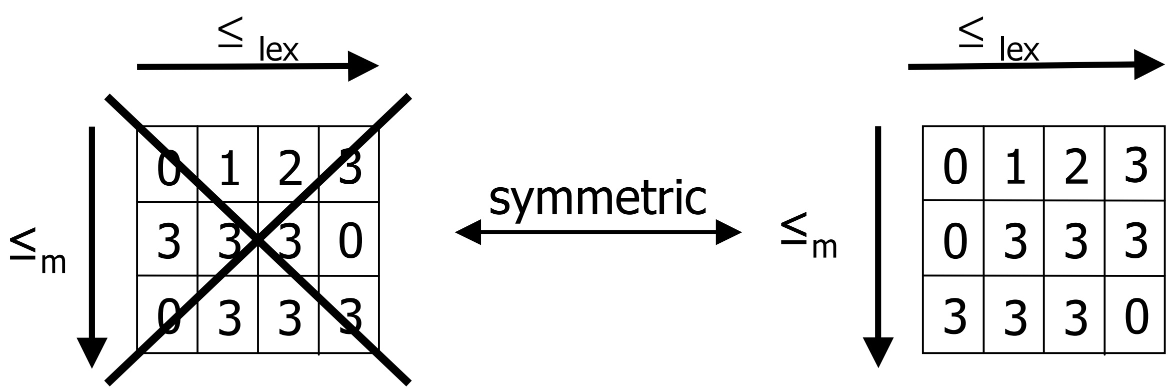
$M_1 \leq_m M_2 \leq_m M_3 \leq_m M_4$



$M_1 = M_2 \rightarrow M_1 \leq_{lex} M_2$

$M_3 = M_4 \rightarrow M_3 \leq_{lex} M_4$

- Stronger than col-lex+row-multiset



Multiset Ordering on the Rows

- Given

$A=[a_1,\ldots,a_n]$

$B=[b_1,\ldots,b_n]$

$A \leq_m B \leftrightarrow$

$(n^{a_1}+ \ldots + n^{a_n}) \leq (n^{b_1}+ \ldots + n^{b_n})$

- $BC(\leq) \leftrightarrow GAC(\leq_m)$
- Feasible for small n

Multiset Ordering on the Rows

- Rows are seen as multisets represented by fixed cardinality
- Given
  $A=[a_1,\ldots,a_n]$
  $B=[b_1,\ldots,b_n]$
  taking values from {1,...d}

- Construct occurrence representation via Regin's **gcc**
  $M=[m_1,\ldots,m_d]$
  $N=[n_1,\ldots,n_d]$
  where m_i=**occurrences**(i,A) and n_i=**occurrences**(i,B)

- $A \leq_m B \leftrightarrow M \leq_{lex} N$
- $GAC(\leq_m) \rightarrow$ GAC(**gcc**) /\ $GAC(\leq_{lex})$

Experimental Evaluation

- We ignore any constraints on the matrix
- We specify
  - the size of the matrix
  - domain size
- We want to find a set of matrices satisfying the symmetry-breaking constraints
- We evaluate the techniques wrt #matrices returned
- Results:
  - Col-lex + row-sum > double-lex
  - Col-lex + row-sum(+ row-lex) > double-lex
  - Col-lex + row-sum(+ row-lex) > col-lex + row-sum
  - Col-lex + row-sum(+ row-lex) **X** col-lex+row-multiset
  - Col-lex + row-multiset(+ row-lex) seems to be the best

Experimental Evaluation (Cont'd)

- Compared to a subset of full SBSD functions:
  - Col-lex + row-multiset(+ row-lex) X

    Column trans, row trans, combination of a row and a column trans
- Which one to prefer?
  - $O(n^4)$ SBDS functions
  - Efficient implementation of multiset ordering
    - Arithmetic constraint is expensive
    - Channelling to occurrences of values

Discussions

- According to our experiments:
  - Col-lex + row-multiset(+ row-lex)
- Same results are obtained by posing row-multiset(+row-lex) and then running SBDS on the columns
- Which one to prefer?
  - SBDS suits to any search tree
  - Efficient implementation of lexicographic ordering constraint
    - GAC($\leq_{lex}$) is O(n)

Discussions

- The experiments omit any problem constraints
- The effectiveness of the methods are problem constraints dependent
  - Row-sum constraints (e.g. BIBD)
    - Col-lex + row-sum = col-lex
    - Col-lex + row-sum(+ row-lex) = double-lex
  - Occurrence constraints (e.g. Sports Scheduling, golfers)
    - Col-lex + row-multiset = col-lex
    - Col-lex + row-multiset(+ row-lex) = double-lex

Work in Progress

- Test the methods on realistic problems
  - More reliable comparison
- Investigate how problem constraints and symmetry-breaking constraints interact
  - Evaluate the symmetry-breaking constraints from inference point of view
- Efficient implementation of multiset ordering
  - Is channelling into occurrence representation a good idea?
- Study the effect of symmetry-breaking constraints from a structural viewpoint
  - Listen to Michela Milano ☺