# Improved Method of Cell Placement with Symmetry Constraints for Analog IC Layout Design

Shinichi Koda, Chikaaki Kodama
and Kunihiro Fujiyoshi

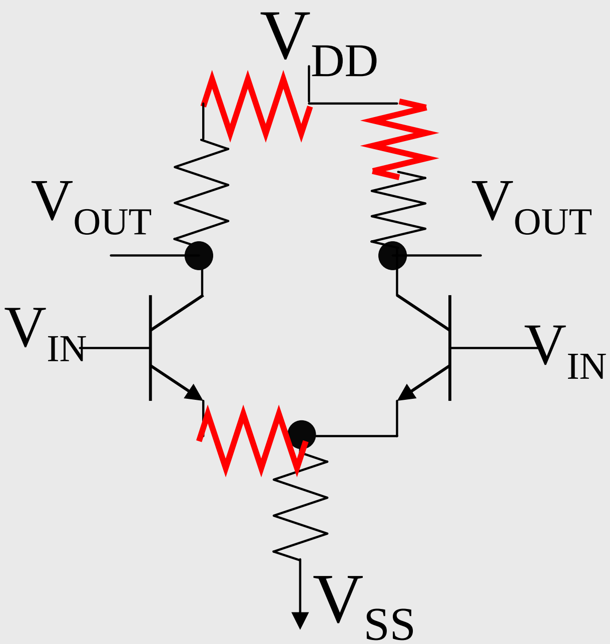
Department of Electrical and Electronic Engineering
Tokyo University of Agriculture & Technology

# Outline

1. Background
2. Conventional Method
    2-1. Balasa’s method and defects
3. Proposed Method
    3-1. Application of Okuda’s approach
    3-2. Decrease of linear expression by substitution
    3-3. Speed-up by constraint graph
4. Experiments
5. Conclusions

# Background

E.g. Differential Amp.




Problem peculiar to analog circuit layout

Parasitic elements are generated by cell placement and wiring.

Unbalanced parasitic elements.

Offset voltage : High
PSRR : Low

Necessary to balance parasitic elements.

Place cells ***symmetrically.***

# Background

The layout of analog ICs has been manually designed by experts.

Recently, Balasa et al. proposed a symmetric placement method using a sequence-pair.
（IEEE Trans.CAD 2000）

However, this method has some defects.

-We clarify these defects.
-We propose a new placement method with symmetry constraints.

# Symmetry Constraints

**Constraints of placing given cell pairs symmetrically to vertical or horizontal axes.**

Symmetry Group

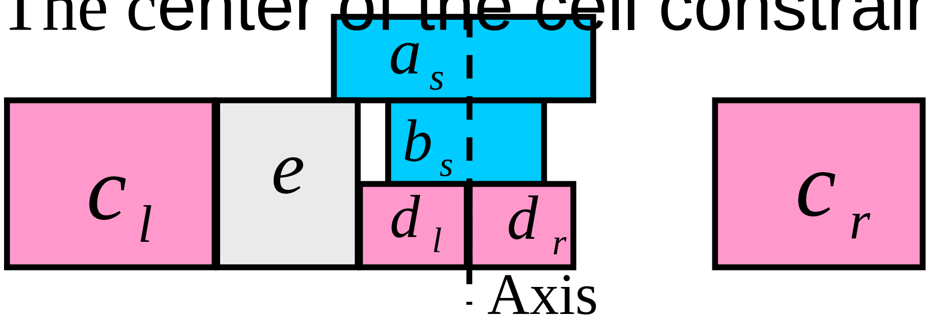
A set of cells constrained to be placed symmetrically to one axis.

**Symmetric pair**

A pair of cells constrained to be placed symmetrically to one axis.

**Self symmetric cell**

The center of the cell constrained to be placed on the axis.




symmetric pairs: $(c_l, c_r),(d_l, d_r)$

self symmetric cells: $a_s, b_s$

Symmetry constraints

$\{a_s, b_s, (c_l, c_r), (d_l, d_r)\}$

Suffix $l$ : Left of pairs

Suffix $r$ : Right of pairs

Suffix $s$ : Self symmetric cell

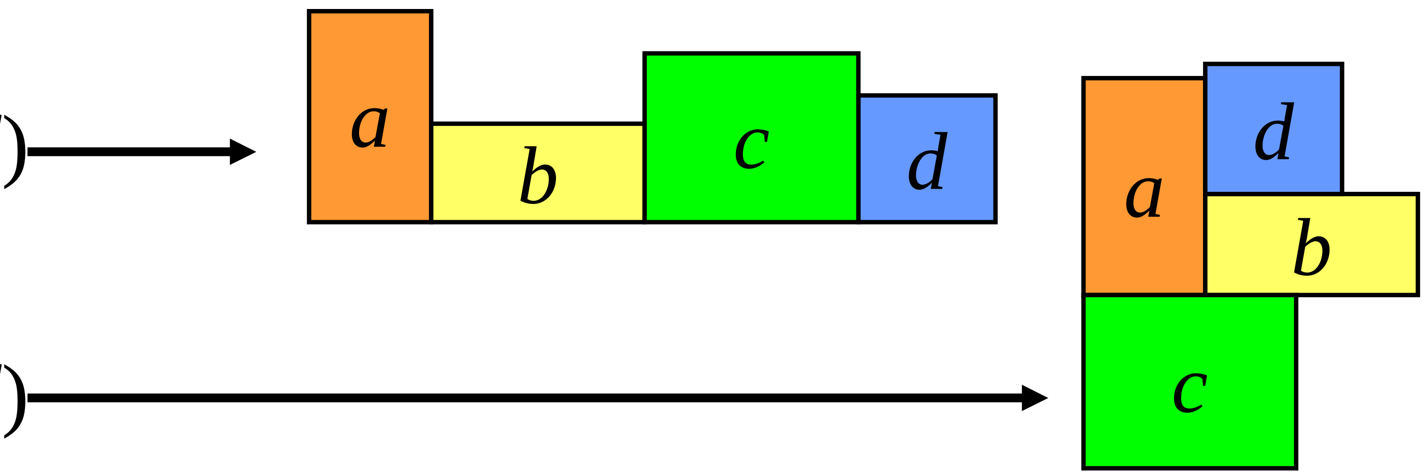
# Sequence-pair (Murata et al. in IEEE Trans.CAD 1996)

- An ordered pair of ($\Gamma_+$ ; $\Gamma_-$)
  ($\Gamma_+$ and $\Gamma_-$ each is a permutation of rectangle names.)
- To show relative position of all rectangle pairs
- Possible to represent any rectangle packing
- Decoding time: $O(n^2)$ ($n$: the number of rectangles)

| $\Gamma_+$ | $\Gamma_-$ | Relative position |
|---|---|---|
| *a* is before *b* | *a* is before *b* | *a* is left of *b* |
| *a* is before *b* | *a* is after *b* | *a* is above *b* |

(*abcd* ; *abcd*) → 

(*adbc* ; *cabd*) → 



# Balasa's method（IEEE Trans.CAD 2000）

(1) They revealed necessary and sufficient condition for symmetric feasible seq-pair.

Definition of symmetric feasible:

| Given seq-pair *S* is symmetric feasible | ⇄ | There exists a placement satisfying<br>- Constraints imposed by seq-pair *S*<br>- Symmetry constraint for one axis. |
|---|---|---|

(2) They proposed a method of obtaining the closest packing satisfying the given constraints in polynomial order time.

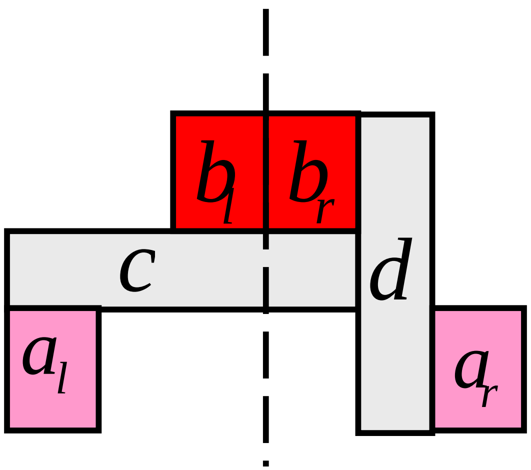
(3) They insisted that the method can be easily expanded into plural symmetry groups.

# Defect 1

## Symmetric feasible sequence-pair

Symmetry constraints

$\{(a_l, a_r), (b_l, b_r)\}$




Unique sequence-pair

$(b_l b_r c a_l d a_r ; a_l c b_l b_r d a_r)$

Necessary and sufficient condition for symmetric feasible

$\Gamma^{-1}_{+}(a) < \Gamma^{-1}_{+}(b)$,

$\Gamma^{-1}_{-}(\mathrm{sym}(b)) < \Gamma^{-1}_{-}(\mathrm{sym}(a))$

sym($a$) is pair of $a$ ($a_s$=sym($a_s$))

$(b_l\, b_r\, c\, a_l\, d\, a_r ; a_l\, c\, b_l\, b_r\, d\, a_r)$

Not symmetric feasible

This is not necessary condition !!

Searching only for symmetric feasible sequence-pair.

Optimum solutions may be overlooked.

# Defect 2

X coordinate determining algorithm

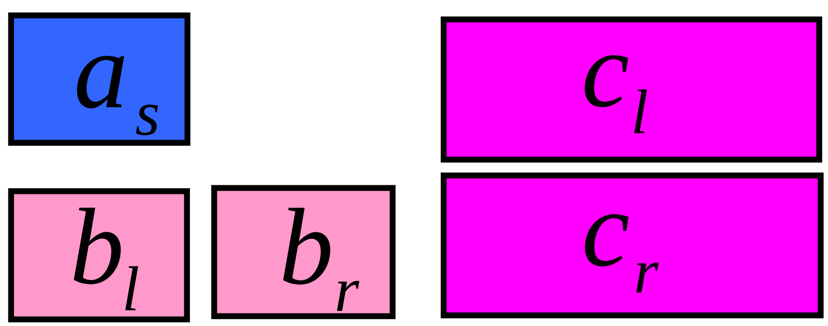
## Input

Symmetry constraints

$$\{a_s, (b_l, b_r), (c_l, c_r)\}$$

Seq-pair

$$(b_l c_l c_r a_s b_r ; b_l a_s c_l c_r b_r)$$

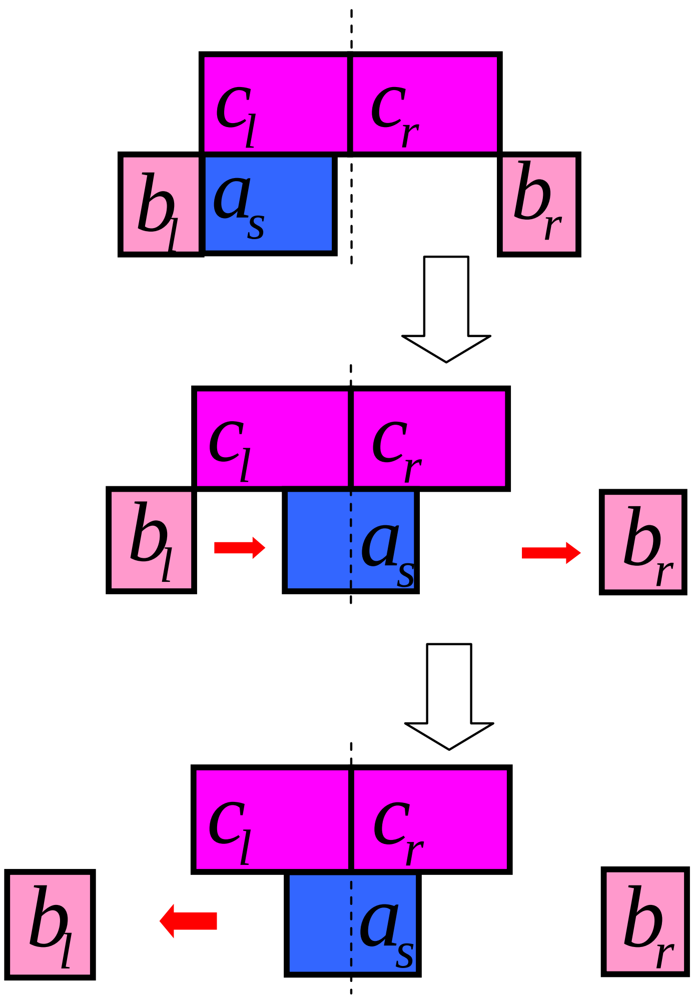


# Defect 2

## X coordinate determining algorithm

seq-pair

$(b_l c_l c_r a_s b_r ; b_l a_s c_l c_r b_r)$




・Cells are packed leftwards based on a given sequence-pair and x coordinate of the axis is determined.

If cells can be placed symmetrically, right cells and self symmetric cells are moved rightwards one by one in order of $\Gamma_+$ .
- Cells constrained to be on the right are also moved rightwards by the same distance.

Left cells are moved leftwards one by one in reverse order of $\Gamma_+$ and they are placed symmetrically.
- Cells constrained to be on the left are also moved leftwards by the same distance.

# Defect 3

## Y coordinate determining algorithm

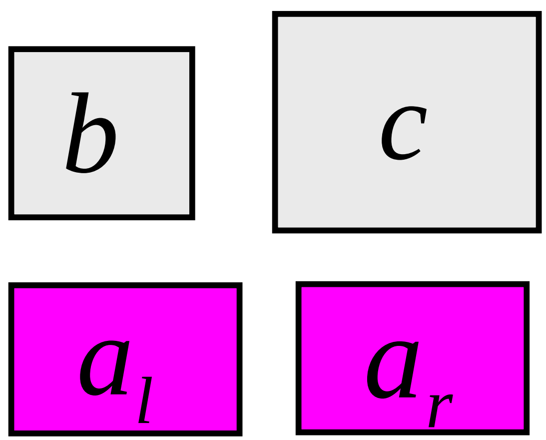
### Input

Symmetry constraints

$$\{(a_l, a_r)\}$$

seq-pair

$$(a_l b c a_r ; b a_l a_r c)$$

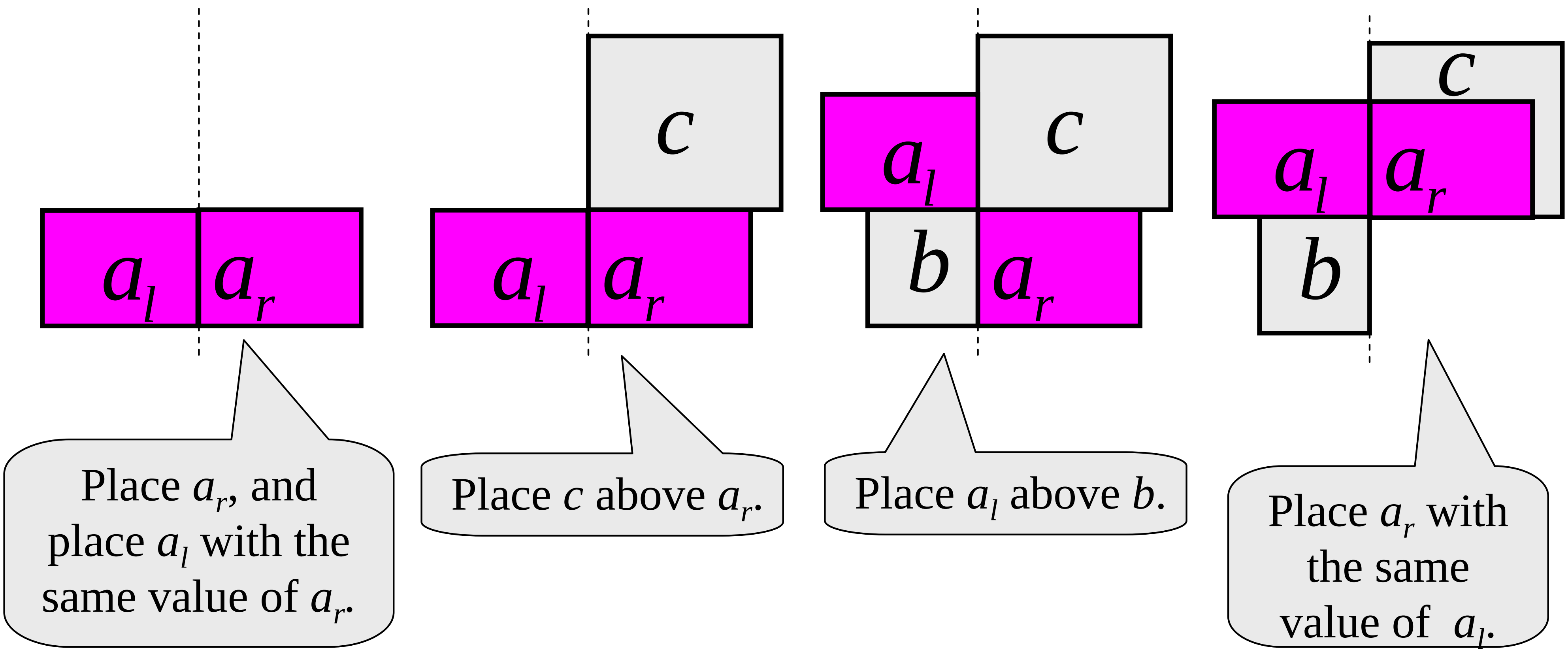


# Defect 3

## Y coordinate determining algorithm

- Determine y coordinate of each cell in reverse order of $\Gamma_+$.
- When y coordinate of one cell in a symmetric pair is determined, determine y coordinate of the other cell to the same value.

sequence-pair $(a_l b c a_r ; b a_l a_r c)$



# Defect 4

Expanding into plural symmetry groups

Group1: $\{(a_r^1, a_l^1)\}$
Group2: $\{(b_l^2, b_r^2)\}$

No placement satisfying these constraints.

Sequence-pair
$(b_l^2\, a_l^1\, a_r^1\, b_r^2\, ;\, a_l^1\, b_l^2\, b_r^2\, a_r^1)$

$b_l^2$ is above $a_l^1$
$a_r^1$ is above $b_r^2$

Symmetric feasible seq-pair for each symmetry group.

In Balasa's method, how to handle more than one symmetry group is not clear.

# Proposed method

Feature 1.

Placement is obtained from a given seq-pair and symmetry constraints by linear programming.

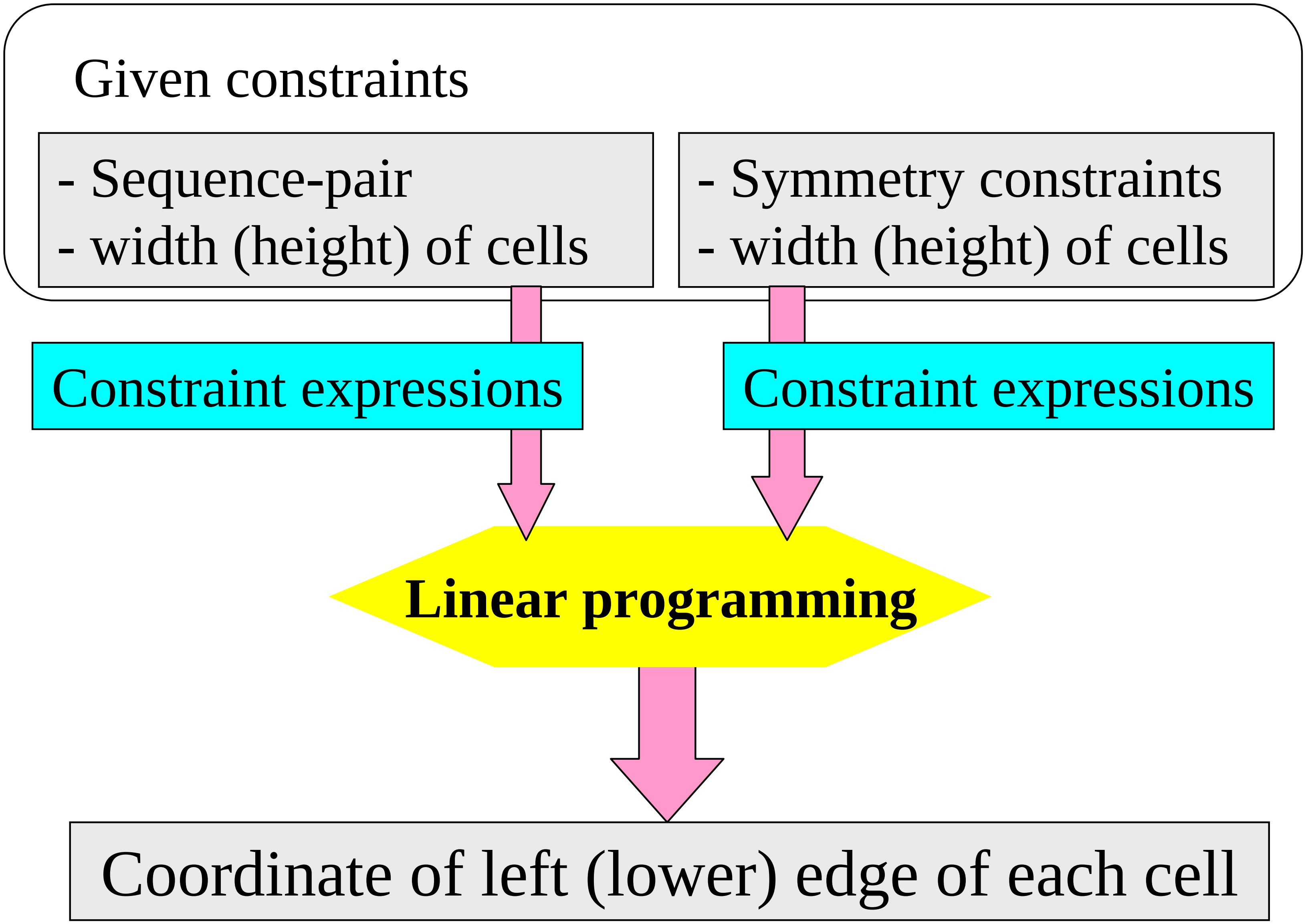
Feature 2.

Speed-up is done by reducing the number of variables and linear constraint expressions.

Feature 3.

If symmetry constraints are only for vertical (horizontal) axes, speed-up is done by determining y (x) coordinates using a graph-based algorithm.

# A simple combination



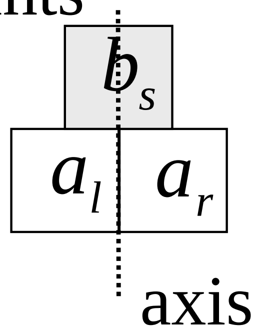
# To obtain constraint expressions from symmetry constraints

Symmetry constraints (width of cells) → constraint expressions

Symmetry constraints

Axis is vertical

$\{(a_l, a_r), b_s\}$




X coordinate of left edge of cell $a$ : $x(a)$

Y coordinate of lower edge of cell $a$ : $y(a)$

Width of cell $a$ : $w(a)$

X coordinate of axis : $axis_x$

**symmetric pair**

$$\begin{cases} axis_x - x(a_l) = (x(a_r) + w(a_r)) - axis_x \\ y(a_l) = y(a_r) \end{cases}$$

**self symmetric cell**

$$axis_x - x(b_s) = (x(b_s) + w(b_s)) - axis_x$$

# To obtain constraints expressions from sequence-pair

Sequence-pair (width of cells) → constraint expressions (Jae-Gon Kim, IEEE Trans. CAD, 2003.)

- The relation between x direction and y direction is independent.
- Explanation only for an x direction.

Sequence-pair

$(b_s a_l a_r c \;\; \text{sink} \;\; ; a_l a_r b_s c \;\; \text{sink} \;)$

$a_l$ | $a_r$ | $b_s$ | $c$

Insert a virtual cell sink ( w(sink)=0 ) in the end of $\Gamma_+$ and $\Gamma_-$.

$c$ is right of $b_s$ → $x(c) \geq x(b_s) + w(b_s)$

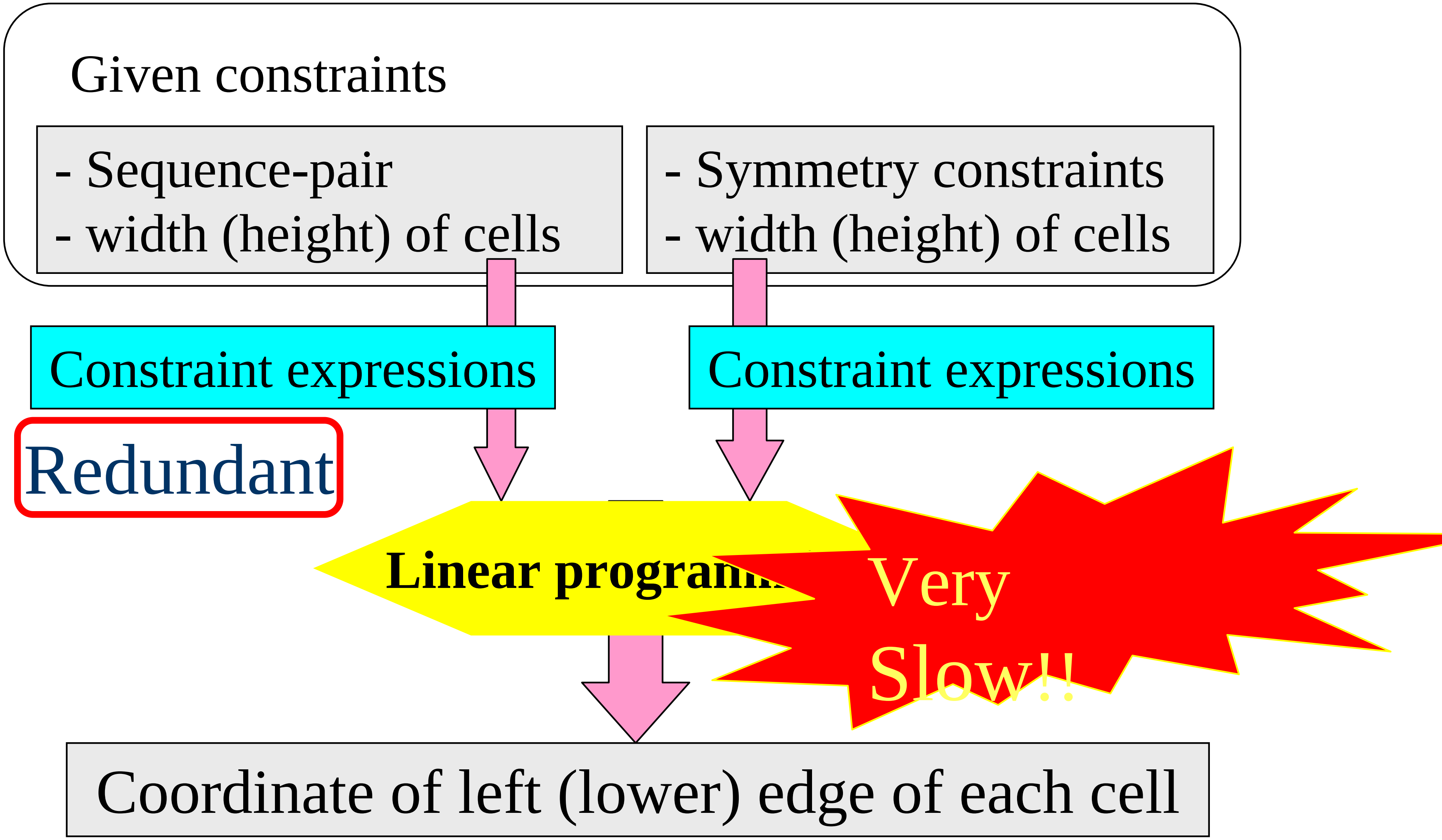
$a_r$ is right of $a_l$ → $x(a_r) \geq x(a_l) + w(a_l)$

$c$ is right of $a_r$ → $x(c) \geq x(a_r) + w(a_r)$

sink is right of $c$ → $x(\text{sink}) \geq x(c) + w(c)$

Obtained constraints expression → Linear programming → X coordinate of left edge of each cell

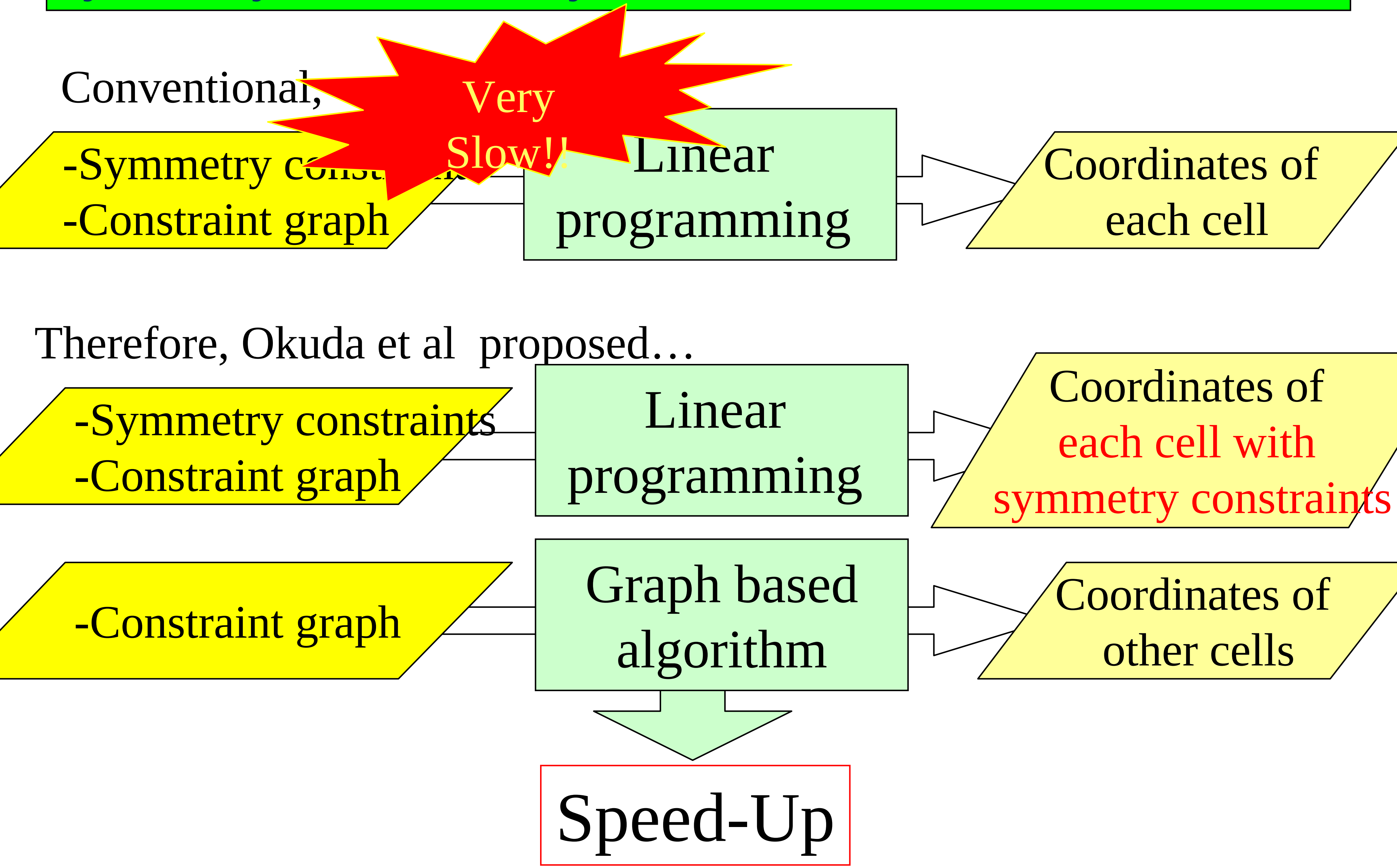
# A simple combination




Therefore, we reduce

# of variables and constraint expressions.

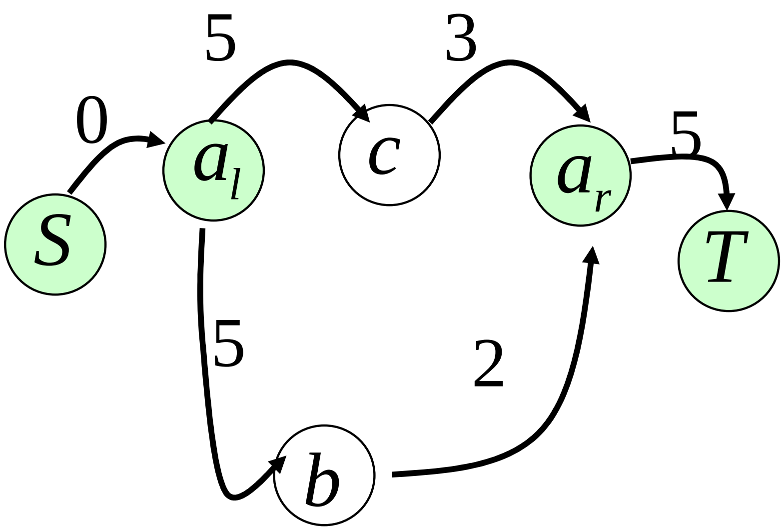
# Speed-up method for the compaction problem including symmetry constraints by Okuda et al. (IEICE Trans.(A) 1990)



# Speed-up method of the compaction problem including symmetry constraints by Okuda et al. (IEICE Trans.(A) 1990)

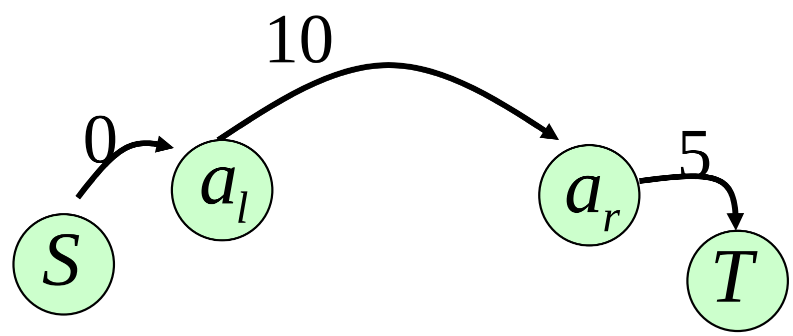
input

- Constraint graph




- Symmetry constraint
  $\{(a_l, a_r)\}$

1. Pick up source, sink and vertices with symmetry constraints from a given constraint graph
2. Set the direct edge from *a* to *b*.
   Weight = the longest path value from *a* to *b*.




Simple constraint graph

Constraint expressions obtained

# 1. Application of speed-up method by Okuda

Given constraints

- Sequence-pair
- width (height) of cells

- Symmetry constraints
- width (height) of cells

Constraint expressions **of simple constraint graph**

Constraint expressions

**linear programming**

- Coordinate of **each cell with symmetry constraints is determined by LP.**

- Coordinates of other cells are determined by using graph.

# 1. Application of speed-up method by Okuda

**Naive method**

*Seq-pair*

⇓ $O(n^2)$

Constraint graph

⇓ $O(s^2n^2)$

Simple Constraint graph

⇓

**Set of linear constraint expressions**

**Proposed method**

*Seq-pair*

⇓ $O(sn\log\log n+se)$

**Set of constraint expressions**

$s$ : #cells with symmetry constraints

$n$ : #all cells

$e$ : #constraint expressions

We obtain a set of constraint expressions faster.

# 2. Speed-up by substitution

Symmetry constraints

$\{a_s, (b_l, b_r)\}$

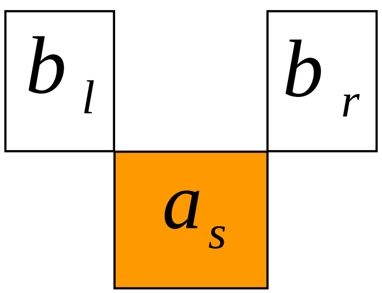
$$axix_x - x(a_s) = x(a_s) + w(a_s) - axix_x$$
$$axix_x - x(b_r) = x(b_l) + w(b_l) - axix_x$$

$$x(a_s) = axix_x - w(a_s) / 2$$
$$x(b_r) = 2\, axix_x - x(b_l) - w(b_l)$$

Substitute.

*Seq-pair*

$(b_l a_s b_r ; a_s b_l b_r)$




$$x(a_s) + w(a_s) \leq x(b_r)$$
$$x(b_l) + w(b_l) \leq x(b_r)$$

$$axix_x - x(b_l) \leq w(b_l) - w(a_s) / 2$$
$$axix_x - x(b_l) \leq w(b_l)$$

Remove redundant constraint expressions.

$$axix_x - x(b_l) \leq w(b_l) - w(a_s) / 2$$

2 variables and
3 expressions are reduced

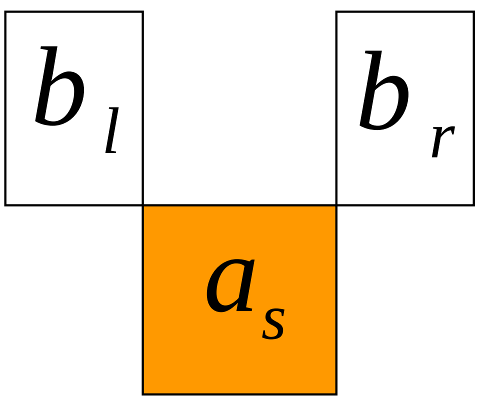
# 3. Speed-up by determining y coordinates by using graph.

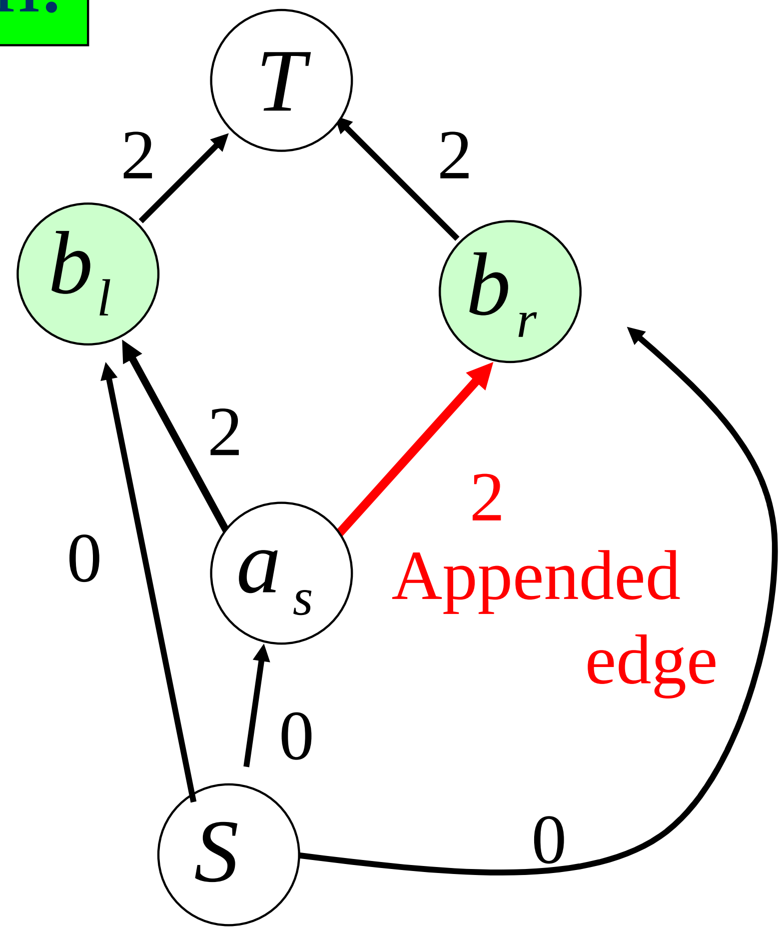
*Seq-pair*

$(b_l a_s b_r ; a_s b_l b_r)$

Symmetry constraint

$\{a_s, (b_l, b_r)\}$




1. Obtain constraint graph from sequence-pair.
2. If an edge is inputted to one of symmetric pair, append an edge with the same weight from the same vertex to the other.

Can be decoded in $O(n^2)$ time

# Experiments

To confirm improvement of the proposed methods.

(1) Experimental comparison in CPU time

To confirm whether the proposed method can obtain nearly optimum solutions or not.

(2) Placement experiments

Note: Linear constraint expressions are solved by simplex method.

MOVE operation:
Choose two elements randomly from a given seq-pair ($\Gamma_+$ ;$\Gamma_-$ ) and exchange each other in both $\Gamma_+$ and $\Gamma_-$ or in either of them.

# (1) Experimental comparison of CPUtime

Method 0. A simple combination.(only using LP)

Method 1. Using simple constraint graph.

Method 2. Using substitution with Method 1.

Method 3. Y coordinate determined by using graph with Method 2.

| #cells | | | #symmetry groups | Method 0 | Method 1 | Method 2 | Method 3 |
|---|---|---|---|---|---|---|---|
| all | pair | self | | time[s] | time[s] | time[s] | time[s] |
| 8 | 1 | 0 | 1 | 181.8 (1.00) | 83.1 (0.45) | 66.8 (0.37) | 26.1 (0.14) |
| 10 | 4 | 0 | 1 | 546.9 (1.00) | 358.3 (0.67) | 256.4 (0.48) | 69.8 (0.13) |
| 9 | 3 | 2 | 2 | 275.3 (1.00) | 167.7 [illegible] | 133.6 (0.48) | 46.2 (0.17) |
| 9 | 1 | 1 | 1 | 166[illegible] (1.00) | 47.8 (0.29) | 39.1 (0.22) | [illegible] (0.12) |
| 16 | 4 | 2 | 2 | 1466.7 (1.00) | 581.2 (0.40) | 397.9 (0.27) | 133.0 (0.07) |

7/100

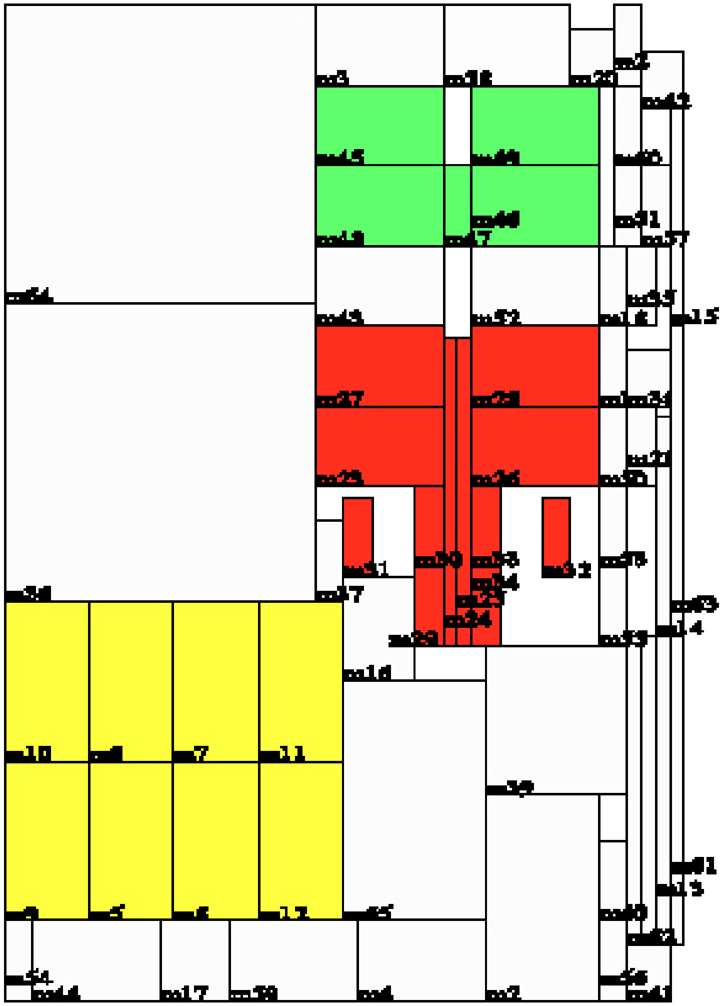
# (2) Placement experiment

## Method 3 is used.

#Cells: 65 (from IEEE Trans. CAD'04 Balasa)

#Symmetry groups: 3

| | | |
|---|---|---|
| Red : | #Pair 6 | #Self 0 |
| Yellow : | #Pair 4 | #Self 0 |
| Green : | #Pair 2 | #Self 1 |



Packing ratio: 106.53%

Time: 6.68[min]

(Pentium4 3.2GHz)

# (2) Placement experiment

## Method 3 is used.

#Cell:110 (from IEEE Trans. CAD'04 Balasa)

#Symmetry groups:5

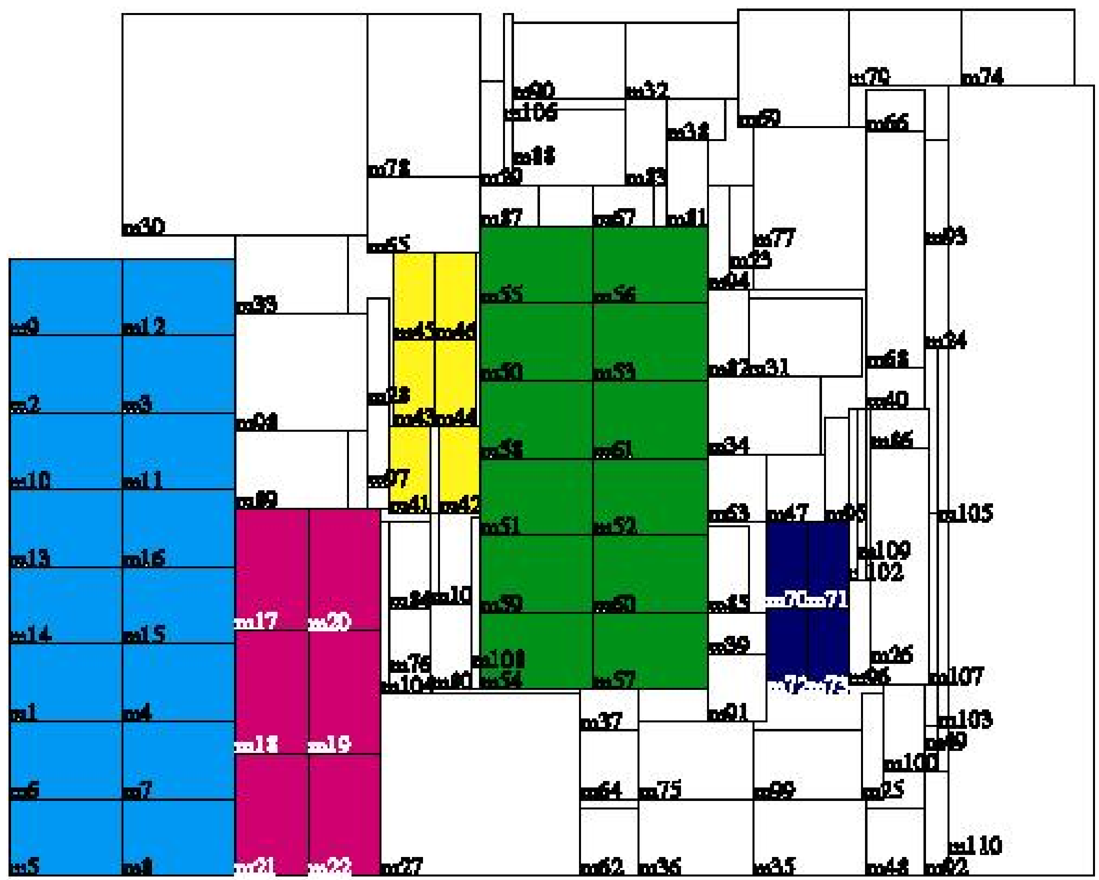
Cyan :#Pair 8 #Self 0

Magenta :#Pair 3 #Self 0

Yellow : #Pair 3 #Self 0

Green :#Pair 6 #Self 0

Blue :#Pair 2 #Self 0



Packing ratio: 108.58%

Time: 54.2[min]

(Pentium4 3.2GHz)

## (2) Placement experiment

Experimental comparisons between the results of proposed method (Pentium4 3.2GHz) and Balasa's results (Sun Blade 100).

| Design | #Cell | #Symmetry groups | Balasa's results | | Proposed method | |
|---|---|---|---|---|---|---|
| | | | Time [min] | Area [%] | Time [min] | Area [%] |
| biasynth 2p4g | 65 | 8+12+5 | 13.00 | 115.00 | 6.68 | 106.53 |
| lnamixbias 2p4g | 110 | 16+6+6 +12+4 | 47.07 | 109.36 | 54.20 | 108.58 |

Column 3 shows the #cells (in symmetric pairs or self-symmetric) in each group.

# Conclusions

We proposed an efficient method of obtaining cell placement satisfying both the given symmetry constraints and the topology constraints imposed by a given sequence-pair.

- In order to shorten the time required by linear programming, the number of variables and constraint expressions are reduced by substituting expressions for dependent variables.
- If the symmetry axes are only vertical, we obtain the placement more quickly by using vertical constraint graph based on a seq-pair.

Future problems

- Experiments based on industrial data of analog circuits.
- Further speed-up of the proposed method.
- Handling other constraints of analog circuits.

Thank you !!

# Balasa's method defect

Balasa's Method is …

1. The closest cells placement satisfying the symmetry and topology constraints cannot be obtained.
2. Some cells overlap each other.
3. It is unclear how to handle more than one set of cells with given symmetry constraints.
4. A placement which cannot be represented by the symmetric-feasible seq-pair exists.

# Algorithm to obtain a constraints expressions set of a simple constraint graph from seq-pair efficiently

*Seq-pair*

focus

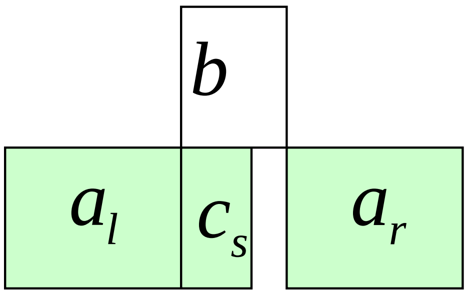
$(a_l b c_s a_r$ sink $; a_l c_s b a_r$ sink $)$

Symmetry constraints

$\{(a_l, a_r), c_s\}$

Width of cells

$a_l$:5 $b$:3 $c_s$:2 $a_r$:5




| | $a_l$ | $c_s$ | $a_r$ | sink |
|---|---|---|---|---|
| $a_l$ | | | | |
| $c_s$ | | | | |
| $a_r$ | | | | |
| sink | | | | |

Focus to inverse order from the last cell of $\Gamma_-$

# Algorithm to obtain a constraints expressions set of a simple constraint graph from seq-pair efficiently

*Seq-pair*

focus

$(a_l b c_s a_r$ sink ; $a_l c_s b a_r$ sink )

5

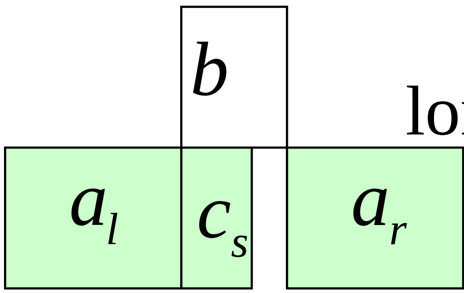
longest path value

Symmetry constraint

$\{(a_l, a_r), c_s\}$

Width of cells

$a_l$:5 $b$:3 $c_s$:2 $a_r$:5




| | $a_l$ | $c_s$ | $a_r$ | sink |
|---|---|---|---|---|
| $a_l$ | | | | |
| $c_s$ | | | | |
| $a_r$ | | | | |
| sink | | | | |

Calculate all longest paths length
from focused cell to cell which can arrive

# Algorithm to obtain a constraints expressions set of a simple constraint graph from seq-pair efficiently

*Seq-pair*

focus

$(a_l b c_s a_r \text{ sink} ; a_l c_s b a_r \text{ sink})$

5

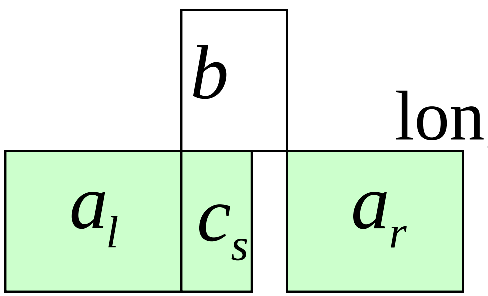
longest path value

Symmetry constraint

$\{(a_l, a_r), c_s\}$

Width of cell

$a_l$:5 $b$:3 $c_s$:2 $a_r$:5




$$x(a_r) + 5 \leq \text{sink}$$

| | $a_l$ | $c_s$ | $a_r$ | sink |
|---|---|---|---|---|
| $a_l$ | | | | |
| $c_s$ | | | | |
| $a_r$ | | | | 5 |
| sink | | | | |

Register the longest path value with a matrix and get constraint expression.

# Algorithm to obtain a constraints expressions set of a simple constraint graph from seq-pair efficiently

focus

*Seq-pair*

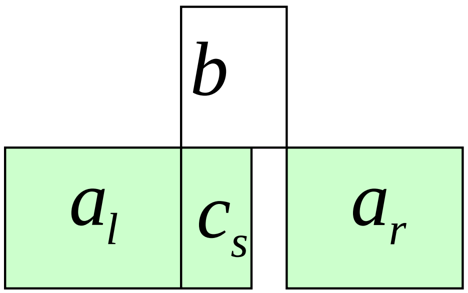
$(a_l b c_s a_r \text{ sink} ; a_l c_s b a_r \text{ sink} )$

Symmetry constraint

$\{(a_l, a_r), c_s\}$

Width of cells

$a_l$:5 $b$:3 $c_s$:2 $a_r$:5




$x(a_r) + 5 \leq \text{sink}$

| | $a_l$ | $c_s$ | $a_r$ | sink |
|---|---|---|---|---|
| $a_l$ | | | | |
| $c_s$ | | | | |
| $a_r$ | | | | 5 |
| sink | | | | |

If focused cell does not have symmetry constraints, I hold nothing

# Algorithm to obtain a constraints expressions set of a simple constraint graph from seq-pair efficiently

*Seq-pair* focus

$(a_l b c_s a_r$ sink ; $a_l c_s b a_r$ sink $)$

2 7

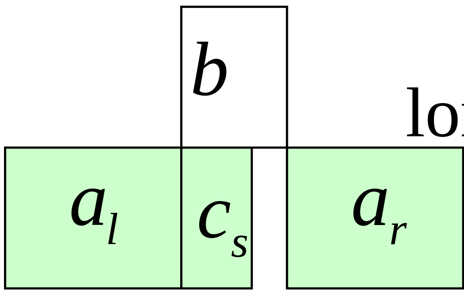
longest path value

Symmetry constraint

$\{(a_l, a_r), c_s\}$

Width of cells

$a_l$:5 $b$:3 $c_s$:2 $a_r$:5




$x(a_r) + 5 \leq$ sink

| | $a_l$ | $c_s$ | $a_r$ | sink |
|---|---|---|---|---|
| $a_l$ | | | | |
| $c_s$ | | | | |
| $a_r$ | | | | 5 |
| sink | | | | |

As follows likewise

# Algorithm to obtain a constraints expressions set of a simple constraint graph from seq-pair efficiently

*Seq-pair*

focus

$(a_l b c_s a_r \text{ sink} ; a_l c_s b a_r \text{ sink})$

2 7

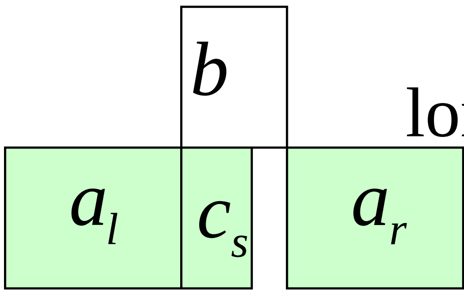
longest path value

Symmetry constraint

$\{(a_l, a_r), c_s\}$

Width of cells

$a_l$:5 $b$:3 $c_s$:2 $a_r$:5




$$x(a_r) + 5 \leq \text{sink}$$

| | $a_l$ | $c_s$ | $a_r$ | sink |
|---|---|---|---|---|
| $a_l$ | | | | |
| $c_s$ | | | 2 | |
| $a_r$ | | | | 5 |
| sink | | | | |

As follows likewise

# Algorithm to obtain a constraints expressions set of a simple constraint graph from seq-pair efficiently

*Seq-pair*

focus

$(a_l b c_s a_r \ \text{sink} \ ; a_l c_s b a_r \ \text{sink})$

2 7

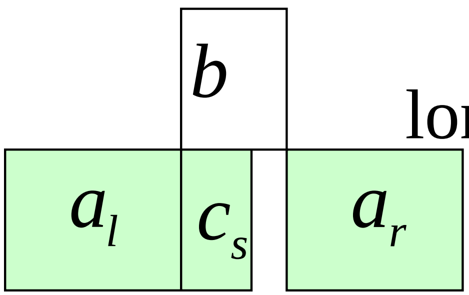
longest path value

Symmetry constraint

$\{(a_l, a_r), c_s\}$

Width of cells

$a_l$:5 $b$:3 $c_s$:2 $a_r$:5




$$x(a_r) + 5 \le \text{sink}$$

$$x(c_s) + 2 \le x(a_r)$$

| | $a_l$ | $c_s$ | $a_r$ | sink |
|---|---|---|---|---|
| $a_l$ | | | | |
| $c_s$ | | | 2 | |
| $a_r$ | | | | 5 |
| sink | | | | |

As follows likewise

# Algorithm to obtain a constraints expressions set of a simple constraint graph from seq-pair efficiently

*Seq-pair*

focus

$(a_l b c_s a_r \text{ sink} ; a_l c_s b a_r \text{ sink})$

2 7

longest path value

Symmetry constarint

$\{(a_l, a_r), c_s\}$

Width of cells

$a_l$:5 $b$:3 $c_s$:2 $a_r$:5

$$x(a_r) + 5 \le \text{sink}$$

$$x(c_s) + 2 \le x(a_r)$$

LPV $c_s \sim a_r \sim$ sink = LPV $c_s \sim$ sink.
Therefore,
constraint expression does not make
It's unnecessary transitive constraint.

| | $a_l$ | $c_s$ | $a_r$ | sink |
|---|---|---|---|---|
| $a_l$ | | | | |
| $c_s$ | | | 2 | 2+5 |
| $a_r$ | | | | 5 |
| sink | | | | |

# Algorithm to obtain a constraints expressions set of a simple constraint graph from seq-pair efficiently

*Seq-pair*

focus

$(a_l b c_s a_r \text{ sink} ; a_l c_s b a_r \text{ sink})$

5 8 13

longest path value

Symmetry constraint

$\{(a_l, a_r), c_s\}$

Width of cells

$a_l$:5 $b$:3 $c_s$:2 $a_r$:5

$$x(a_r) + 5 \le \text{sink}$$

$$x(c_s) + 2 \le x(a_r)$$

As follows likewise

| | $a_l$ | $c_s$ | $a_r$ | sink |
|---|---|---|---|---|
| $a_l$ | | | | |
| $c_s$ | | | 2 | 7 |
| $a_r$ | | | | 5 |
| sink | | | | |

# Algorithm to obtain a constraints expressions set of a simple constraint graph from seq-pair efficiently

*Seq-pair*

focus

$(a_l b c_s a_r \text{ sink} ; a_l c_s b a_r \text{ sink})$

5 8 13

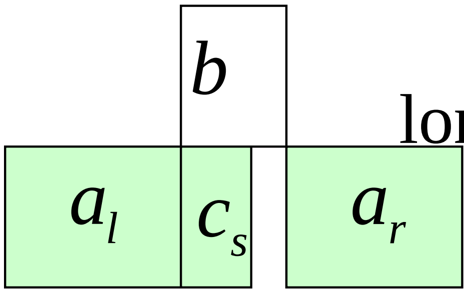
longest path value

Symmetry constraint

$\{(a_l, a_r), c_s\}$

Width of cells

$a_l$:5 $b$:3 $c_s$:2 $a_r$:5




$$x(a_r) + 5 \leq \text{sink}$$

$$x(c_s) + 2 \leq x(a_r)$$

$$x(a_l) + 5 \leq x(c_s)$$

As follows likewise

| | $a_l$ | $c_s$ | $a_r$ | sink |
|---|---|---|---|---|
| $a_l$ | | 5 | | |
| $c_s$ | | | 2 | 7 |
| $a_r$ | | | | 5 |
| sink | | | | |

# Algorithm to obtain a constraints expressions set of a simple constraint graph from seq-pair efficiently

*Seq-pair*

focus

$(a_l b c_s a_r \ \text{sink} \ ; a_l c_s b a_r \ \text{sink} \ )$

5 8 13

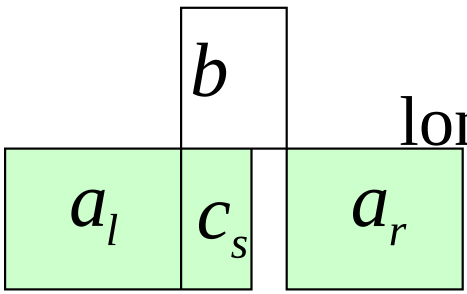
longest path value

Symmetry constraint

$\{(a_l, a_r), c_s\}$

Width of cell

$a_l$:5 $b$:3 $c_s$:2 $a_r$:5




$x(a_r) + 5 \le \text{sink}$

$x(a_l) + 8 \le x(a_r)$

$x(c_s) + 2 \le x(a_r)$

$x(a_l) + 5 \le x(c_s)$

| | $a_l$ | $c_s$ | $a_r$ | sink |
|---|---|---|---|---|
| $a_l$ | | 5 | 5+2 | |
| $c_s$ | | | 2 | 7 |
| $a_r$ | | | | 5 |
| sink | | | | |

LPV $a_l$~ $a_r$ < LPV $a_l$~$c_s$~$a_r$

This constraint expression was made.

# Algorithm to obtain a constraints expressions set of a simple constraint graph from seq-pair efficiently

*Seq-pair*

focus

$(a_l b c_s a_r$ sink ; $a_l c_s b a_r$ sink )

5 8 13

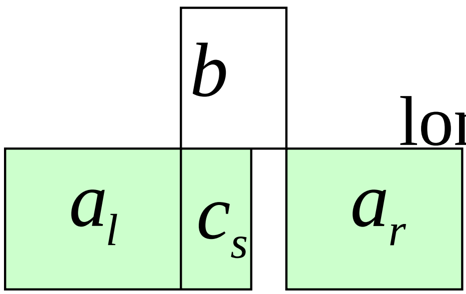
longest path value

Symmetry constaints

$\{(a_l, a_r), c_s\}$

Width of cells

$a_l$:5 $b$:3 $c_s$:2 $a_r$:5




$x(a_r) + 5 \leq \text{sink}$  $x(a_l) + 8 \leq x(a_r)$

$x(c_s) + 2 \leq x(a_r)$

$x(a_l) + 5 \leq x(c_s)$

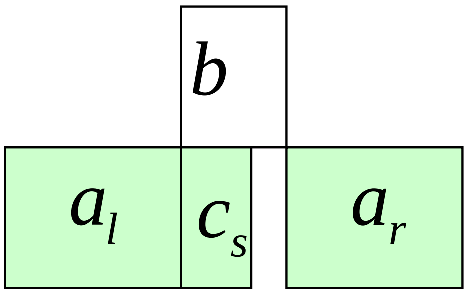
As follows likewise

| | $a_l$ | $c_s$ | $a_r$ | sink |
|---|---|---|---|---|
| $a_l$ | | 5 | 8 | 8+5 |
| $c_s$ | | | 2 | 7 |
| $a_r$ | | | | 5 |
| sink | | | | |

# Algorithm to obtain a constraints expressions set of a simple constraint graph from seq-pair efficiently

*Seq-pair*

$$(a_l b c_s a_r \text{sink}; a_l c_s b a_r \text{ sink})$$




Symmetry constraint $\{(a_l, a_r), c_s\}$

Width of cell

$a_l$:5 $b$:3 $c_s$:2 $a_r$:5

$$x(a_r) + 5 \le \text{sink} \qquad x(a_l) + 8 \le x(a_r)$$

$$x(c_s) + 2 \le x(a_r)$$

$$x(a_l) + 5 \le x(c_s)$$

| | $a_l$ | $c_s$ | $a_r$ | sink |
|---|---|---|---|---|
| $a_l$ | | 5 | 8 | 13 |
| $c_s$ | | | 2 | 7 |
| $a_r$ | | | | 5 |
| sink | | | | |

The constraint expressions of simple constraint graph was obtained from seq-pair directly.

# Balasa's method defect 4

When there are plural symmetry groups, they insist that can expand it easily

There is not placement satisfy these constraints

Axis1: $a_{l1}, a_{r1}$ is pair $b_{l2}, b_{r2}$

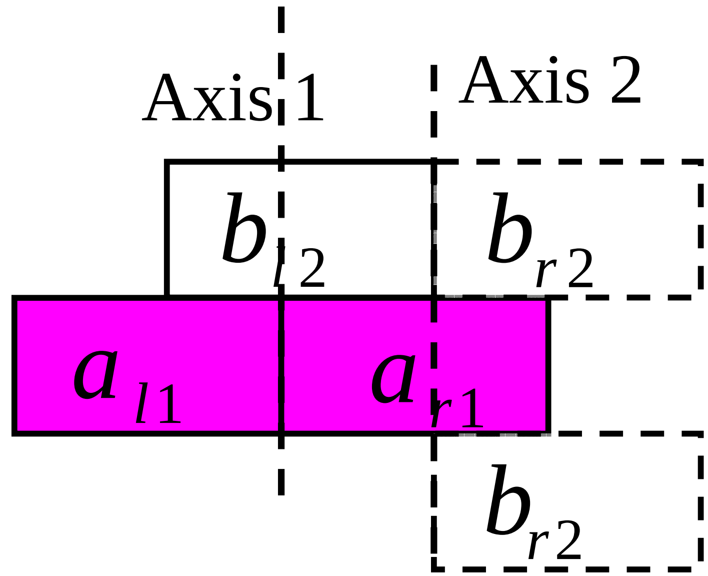
Sequence-pair

$(b_{l2}a_{l1}a_{r1}b_{r2};a_{l1}b_{l2}b_{r2}a_{r1})$

$b_{l2}$ On the $a_{l1}$

$a_{r1}$ On the $b_{r2}$

It is symmetric-feasible about each symmetry groups.




It is unclear how to handle more than one set of cells with given symmetry constraints.