On the in-the-middle algorithm and heuristic and some of its properties

Dag Wedelin, Data Science Division, CSE, Chalmers

Beginning with special cases of linear programming, I will describe these algorithms, and some of their properties. I will also briefly discuss the max-sum problem and related algorithms, and discuss some of the general challenges with numerical propagation algorithms in relation to classical constraint programming.

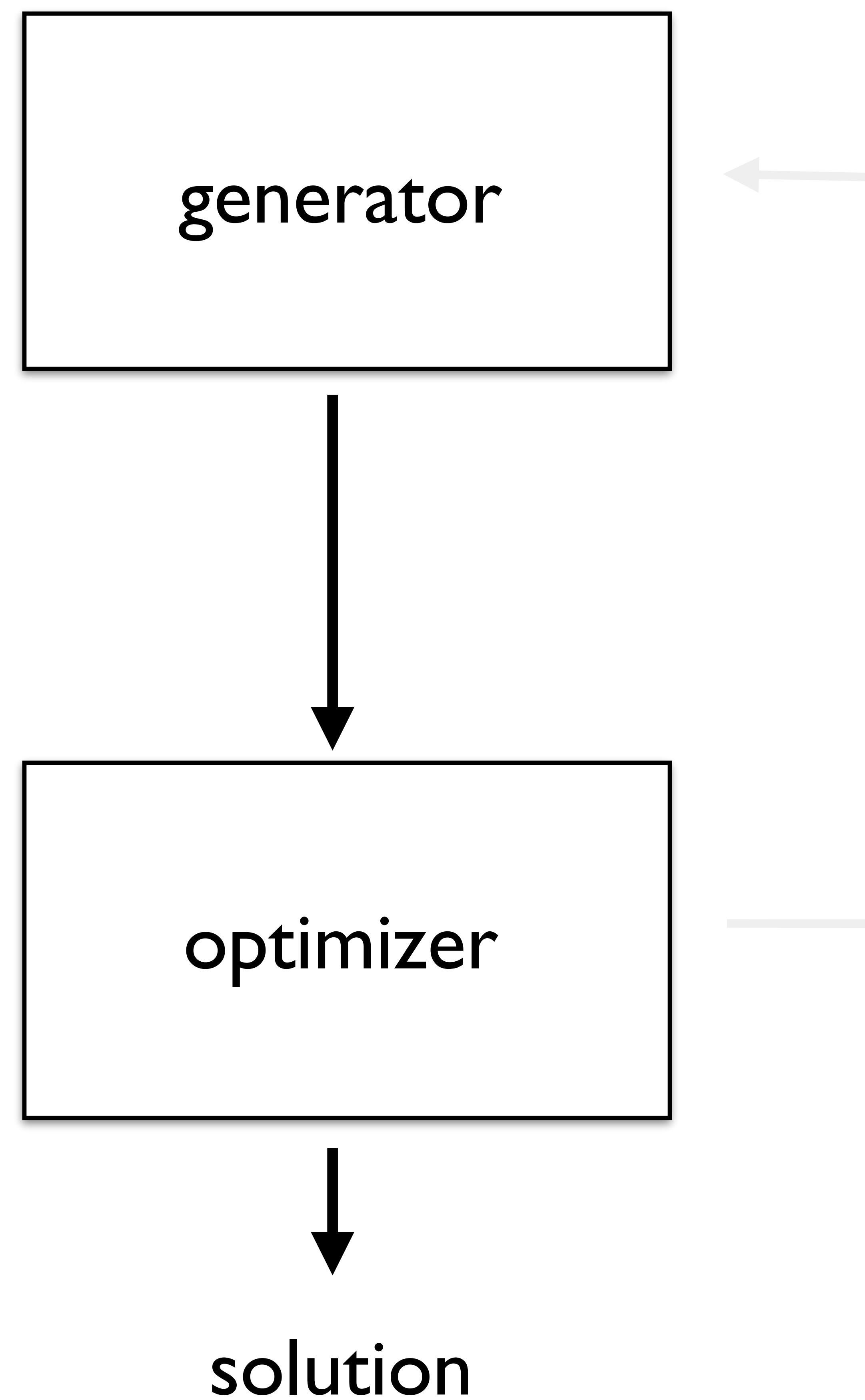
CHALMERS | UNIVERSITY OF GOTHENBURG

Crew pairing screenshot

In crew pariring system

generate many legal pairings

select an optimal subset of
these pairings



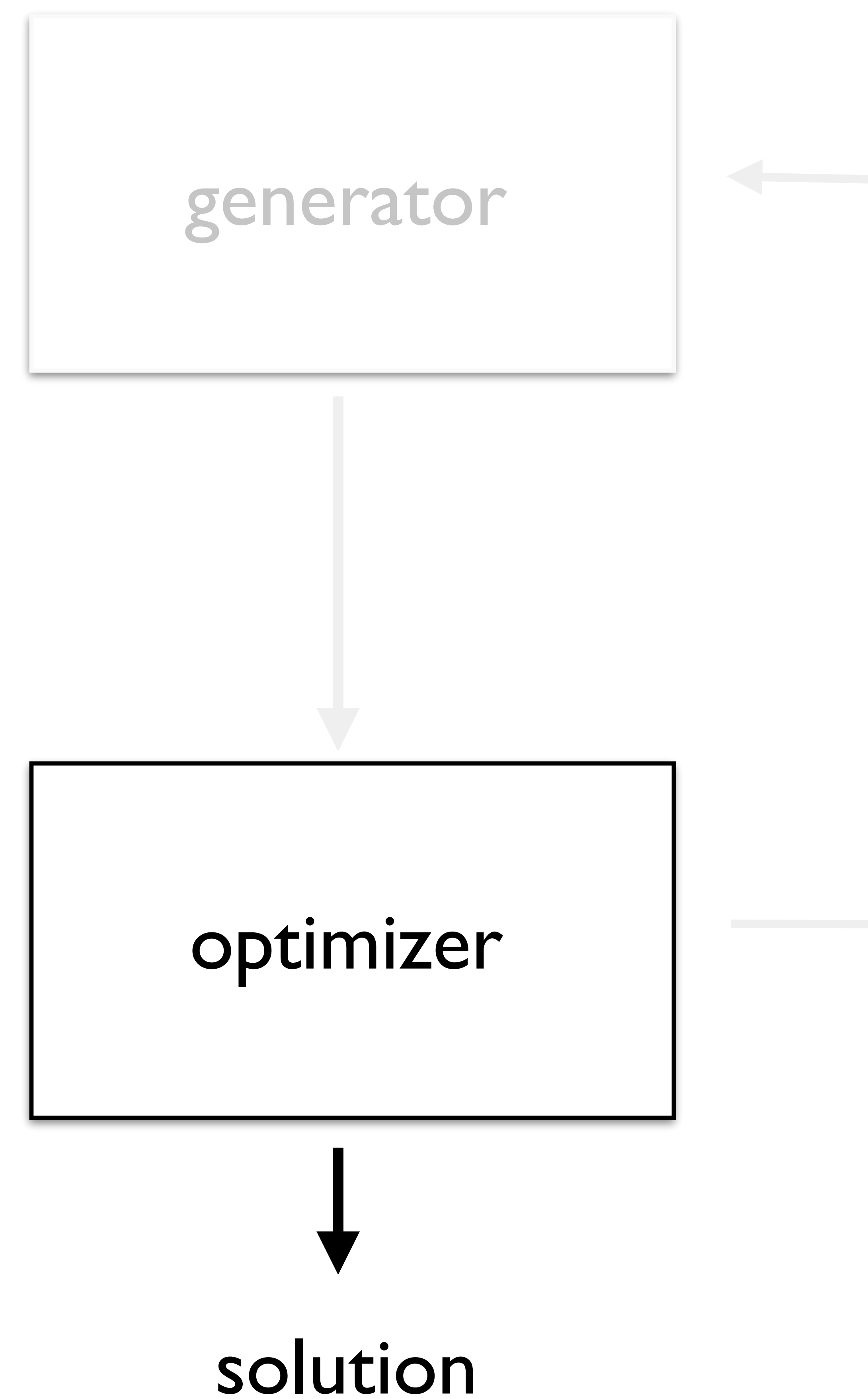
In crew pariring system

min cx

$Ax \geq 1$ (set covering)

$Cx \leq d$ (base capacity)

x binary vector



paqs optimizer

- <u> in-the-middle algorithm</u>
- <u> in-the-middle heuristic</u>

Regularly benchmarked,
continuously improved.

In production since
many years.

Parallel implementation

the in-the-middle algorithm

The simple assignment problem

tasks

persons

3	8	6	4
4	2	3	0
2	7	10	8
6	8	10	5

$$\max \quad c_1x_1 + c_2x_2 + \dots + c_{16}x_{16}$$

Subject to

$$\begin{cases} x_1 + x_2 + x_3 + x_4 = 1 \\ \vdots \end{cases}$$

$$\begin{cases} x_1 + x_5 + x_9 + x_{13} = 1 \\ \vdots \end{cases}$$

x binary

3	8	6	4
4	2	3	0
2	7	10	8
6	8	10	5

$\Rightarrow$

-4	1	-1	-3
4	2	3	0
2	7	10	8
6	8	10	5

3	8	6	4
4	2	3	0
2	7	10	8
6	8	10	5

⇒

-4	1	-1	-3
4	2	3	0
2	7	10	8
6	8	10	5

⇒… ⇒

-2	1	-3	-1
1	-3	-4	-3
-5	-2	-1	1
-1	-1	1	-2

Subtract average of largest and second largest numbers.

3	8	6	4
4	2	3	0
2	7	10	8
6	8	10	5

⇒

-4	1	-1	-3
4	2	3	0
2	7	10	8
6	8	10	5

Iterate for all rows and columns until there are no more sign changes.

⇒... ⇒

-2	1	-3	-1
1	-3	-4	-3
-5	-2	-1	1
-1	-1	1	-2

select assignments with positive cost!

Simplest possible algorithm? (just subtracting the smallest number does not work)

In-the-middle for 0-1 ILP

$$\max\ cx$$

$$Ax = b$$

$$x \text{ binary}$$

A contains {-1,0,1}
b integer
inequalities ok

Consider

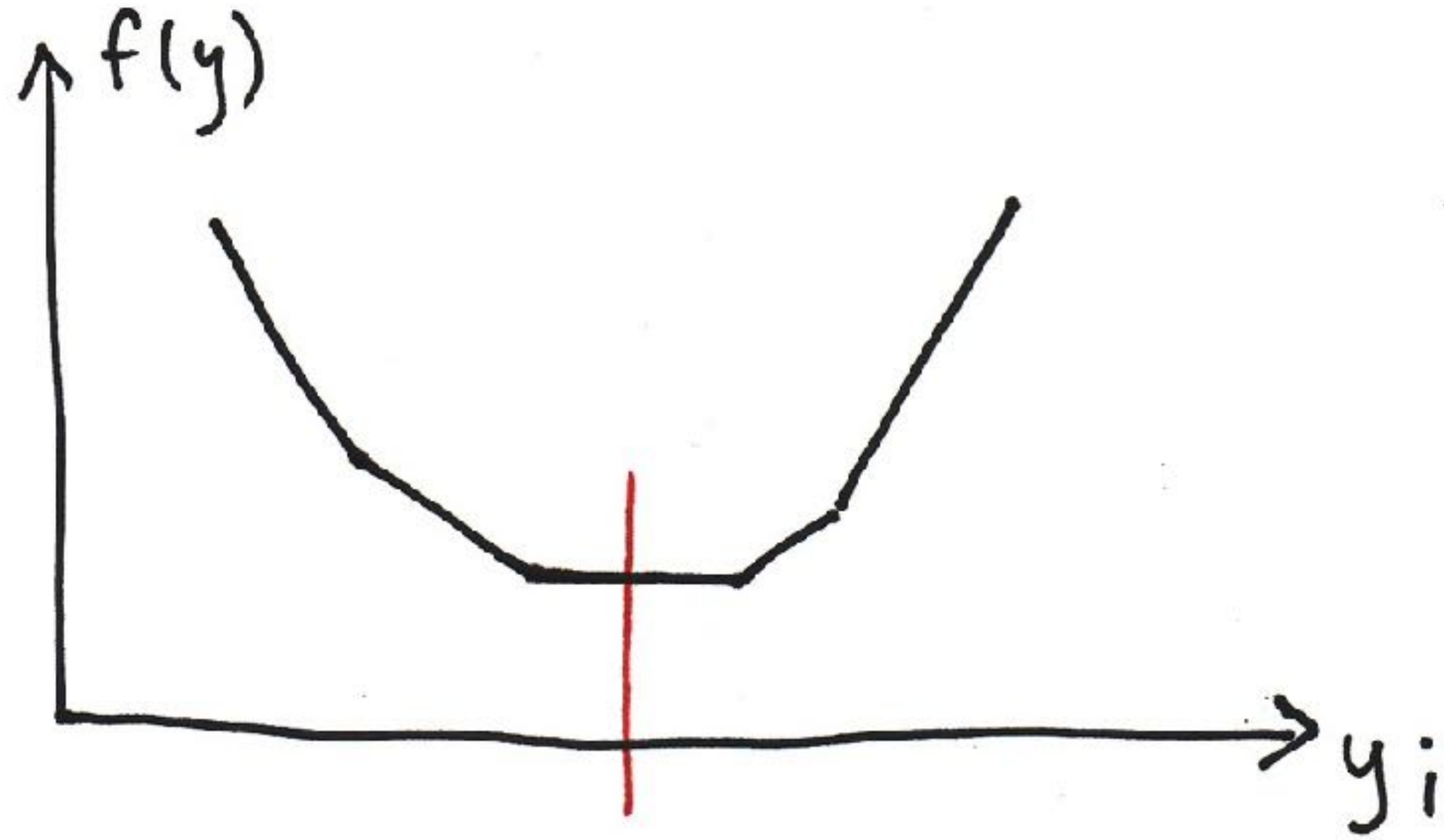
$$\bar{c} = c - yA$$

-4	1	-1	-3
4	2	3	0
2	7	10	8
6	8	10	5

iteratively make single constraints feasible by selecting the dual yi in-the-middle of the possible interval

A "dual" algorithm

Minimize piecewise linear convex function with coordinate descent



$$\min_y f(y) = yb + \max_{0 \le x \le 1} \bar{c}x$$

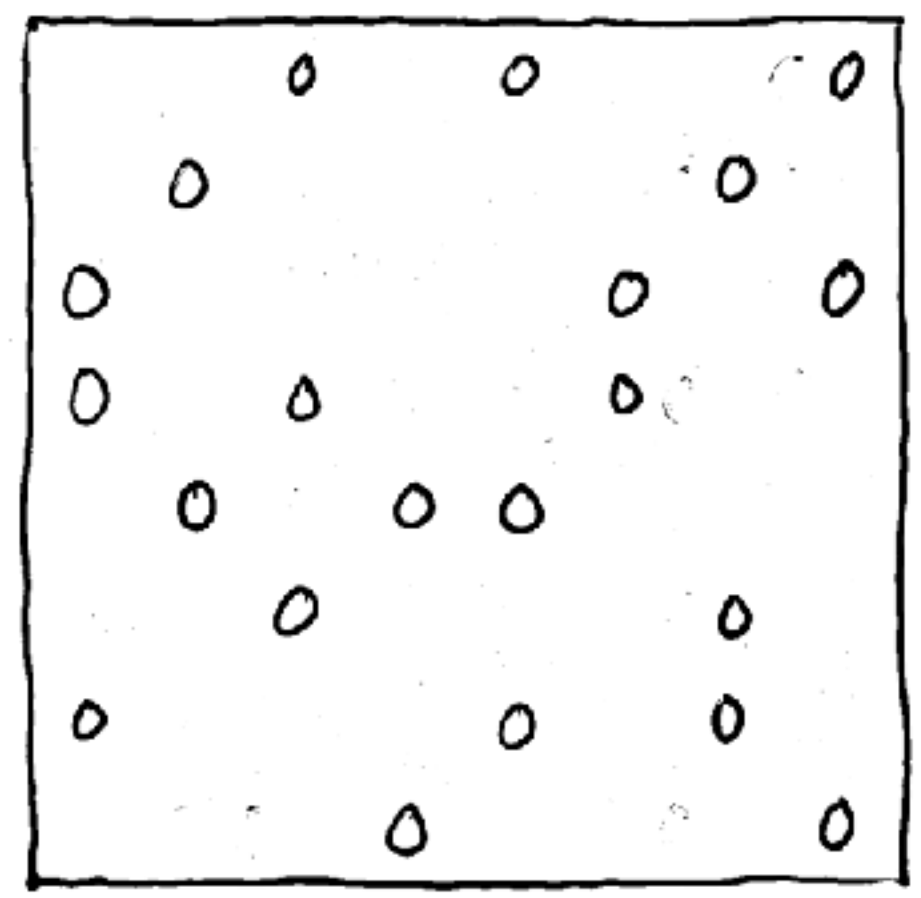
Theorem:

The in-the-middle algorithm can only solve "easy" ILP problems (the LP relaxation has a unique solution that is integer).

What about convergence?

How does in-the-middle fail for difficult ILP's?

a lot of zeros…



eg. 8-queens
with diagonal
constraints

A small difficult problem

$$\max \quad x_1 + x_2 + x_3$$

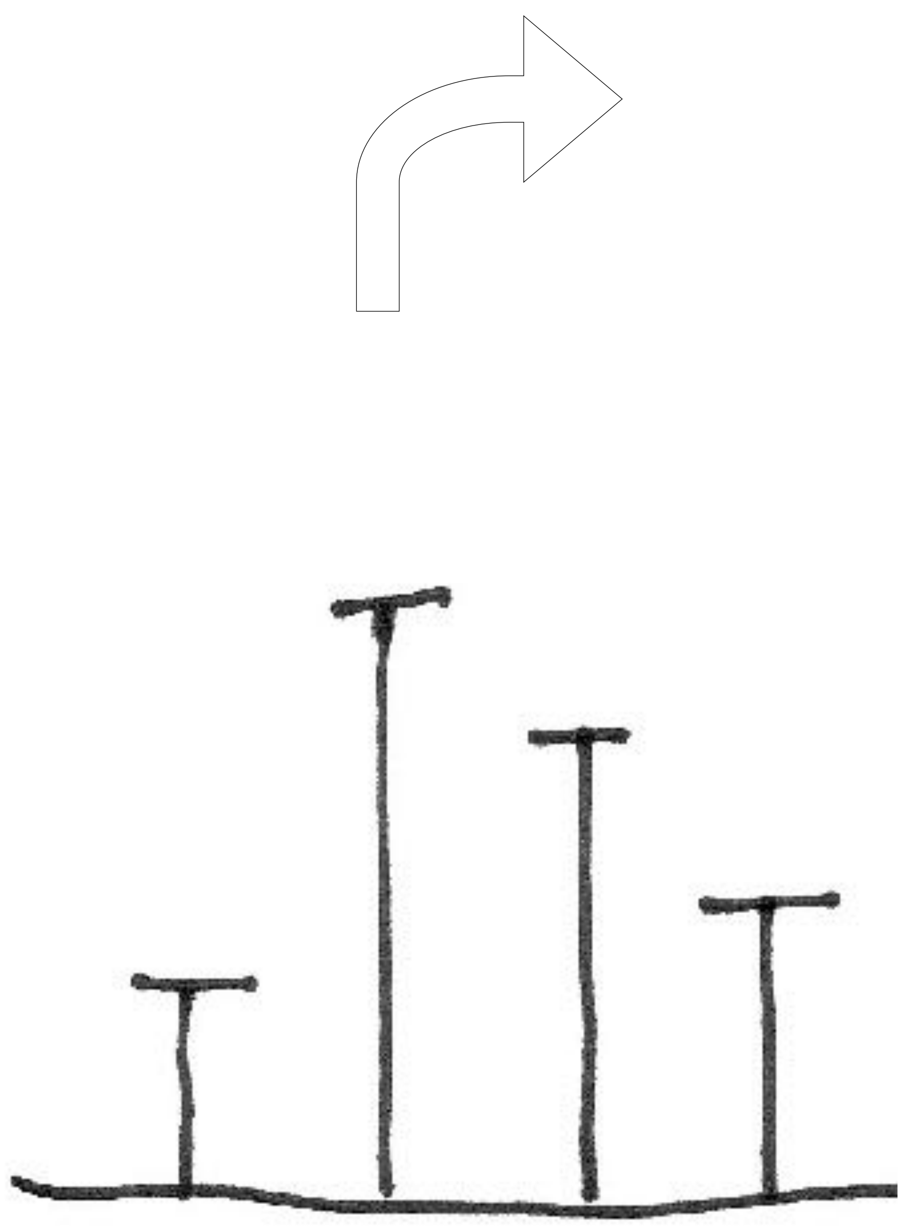
$$x_1 = x_2 + x_3$$

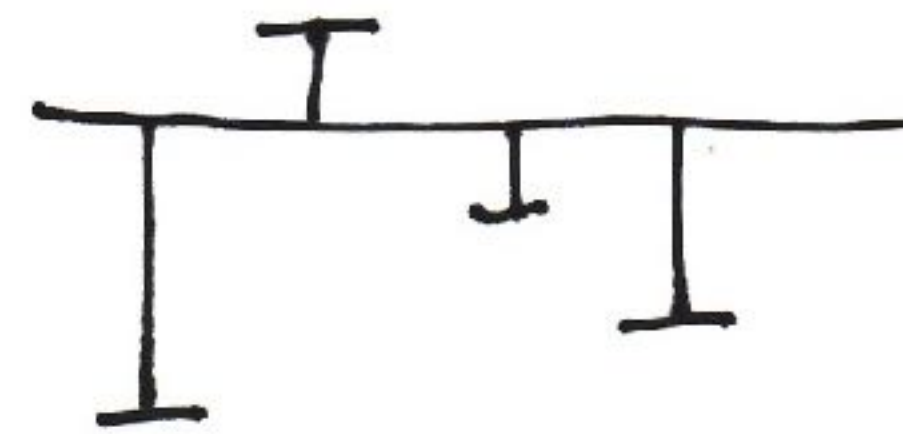
$$x_2 = x_1 + x_3$$

$$x_3 = x_1 + x_3$$

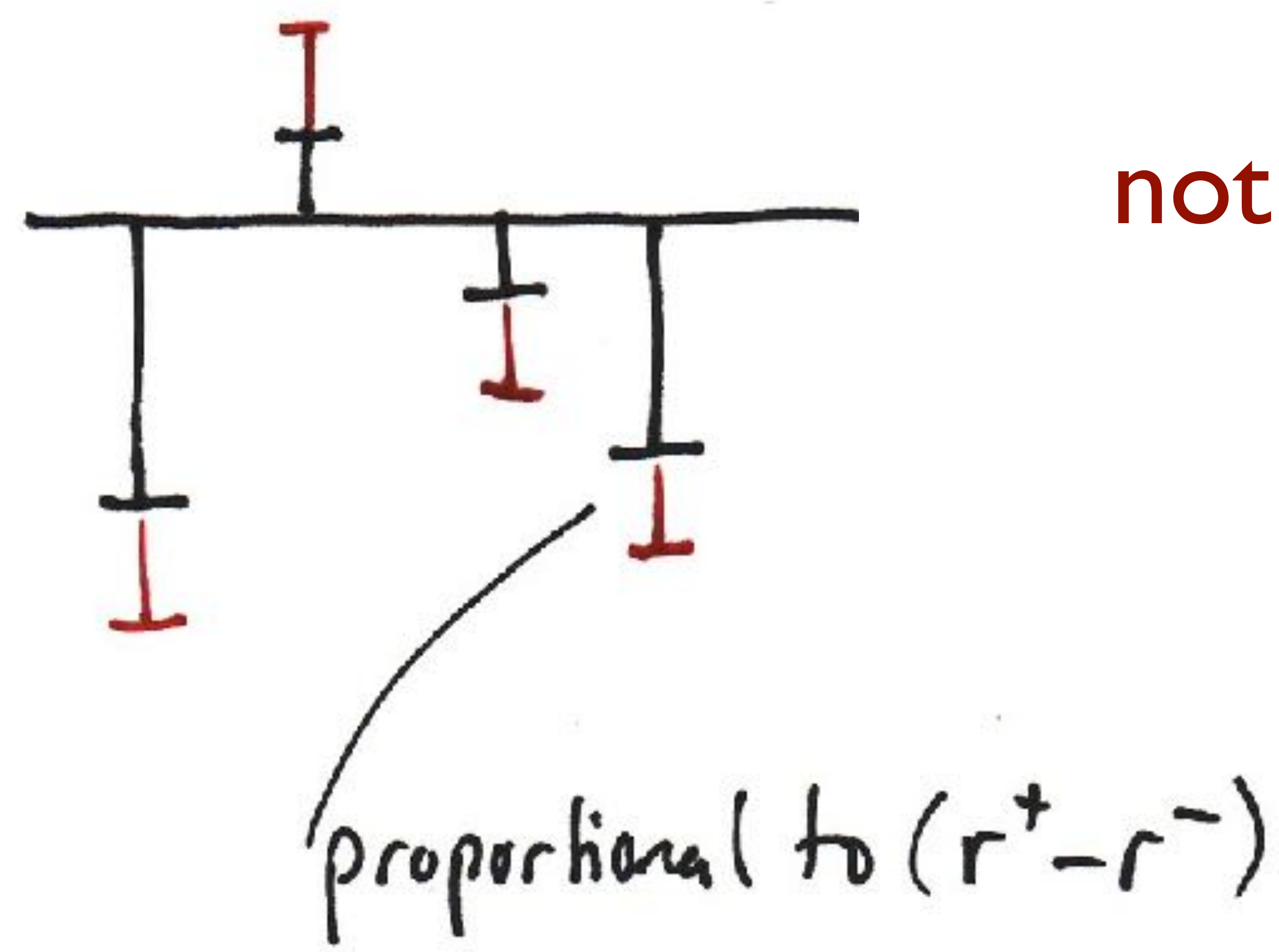
$\bar{c}_1 = \bar{c}_2 = \bar{c}_3 = 0$ is a fixpoint!

in-the-middle heuristic for difficult 0-1 ILP's





invariant



not invariant!

Now it suddenly works!

(weighted 8-queens problem)

1	3	7	5	8	3	2	9
5	3	8	7	5	1	1	9
4	3	7	5	8	2	3	8
7	1	2	4	1	4	6	2
2	6	2	8	4	4	4	1
3	5	4	5	4	5	4	5
1	2	3	4	5	6	7	8
3	2	2	2	2	2	6	8

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

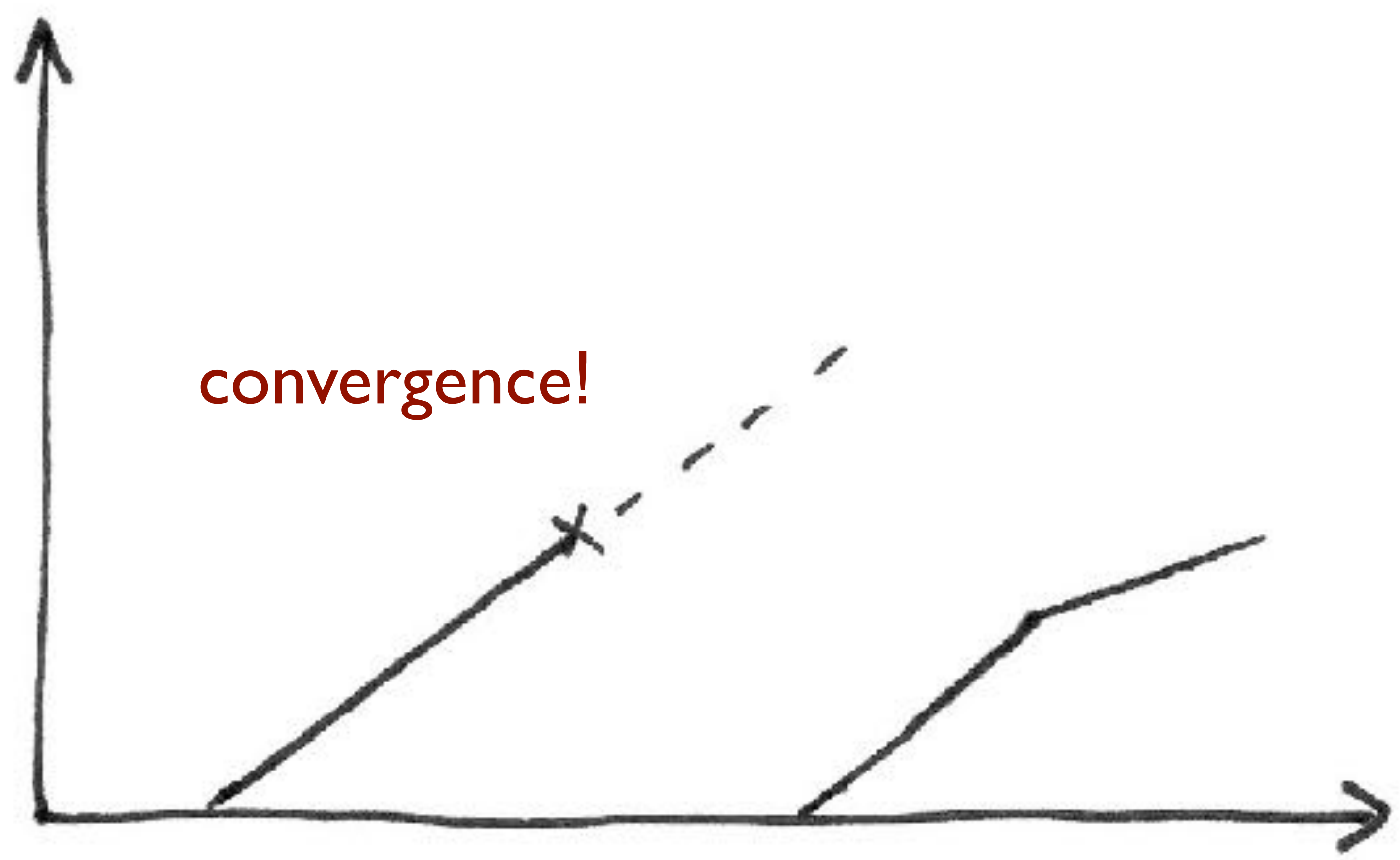
in-the-middle heuristic

$$y_i = -\frac{r^+ + r^-}{2} \pm \alpha \frac{r^+ - r^-}{2}$$

in-the-middle algorithm

in-the-middle heuristic

Sweep mechanism



best results with as low disturbance as possible

(also small random costs for resolving ties)

the max-sum problem

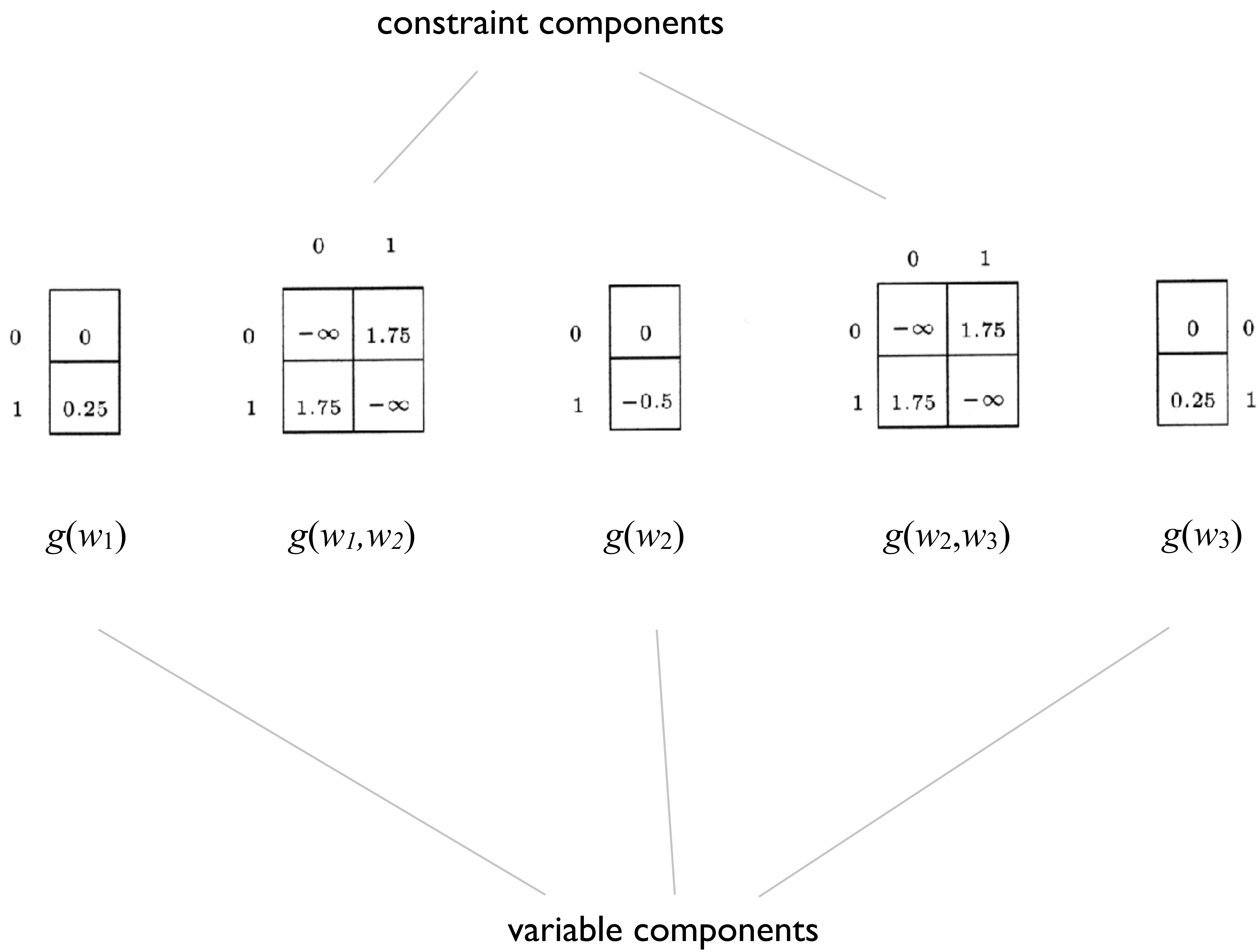
The max-sum problem

vector of discrete variables

different subsets of
the variables

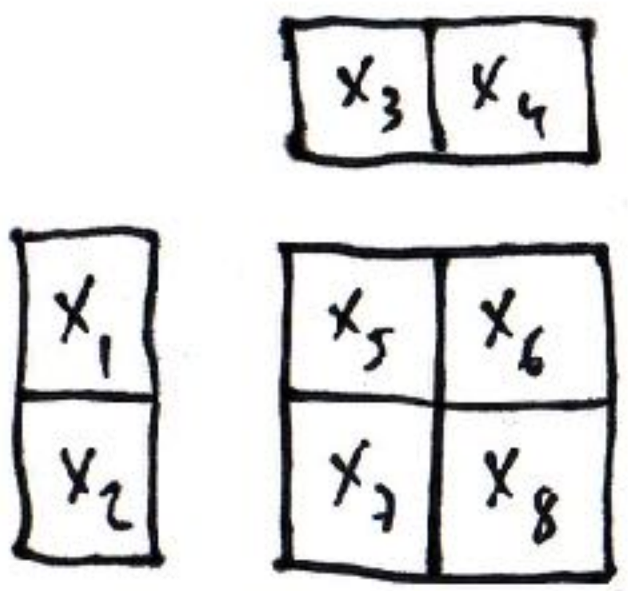
$$\max_{w} f(w) = \sum_{k} g_k(w^k) + C$$

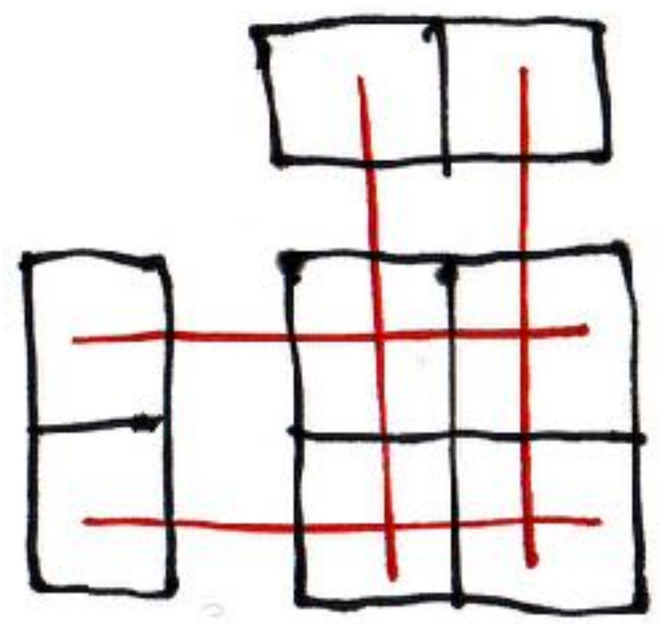
Allows a highly non-linear cost function


constraint components
0
0
1
0.25
0 1
0
−∞
1.75
1
1.75
−∞
0
0
1
−0.5
0 1
0
−∞
1.75
1
1.75
−∞
0
0
0.25
1
g(w1)
g(w1,w2)
g(w2)
g(w2,w3)
g(w3)
variable components

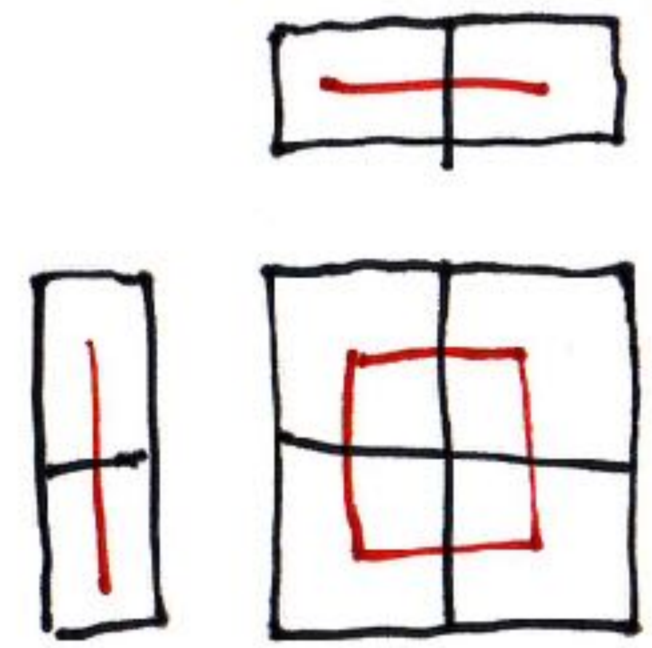
modelling and solving the max-sum problem as an ILP

The max-sum ILP





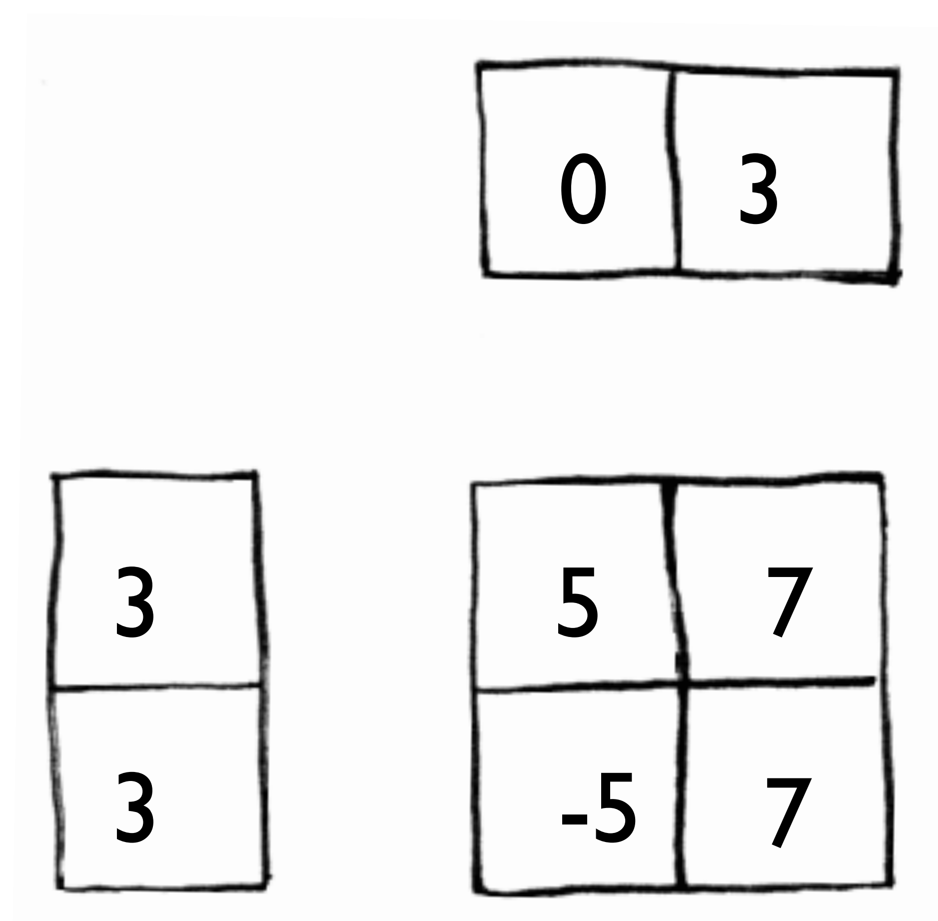
marginalization constraints



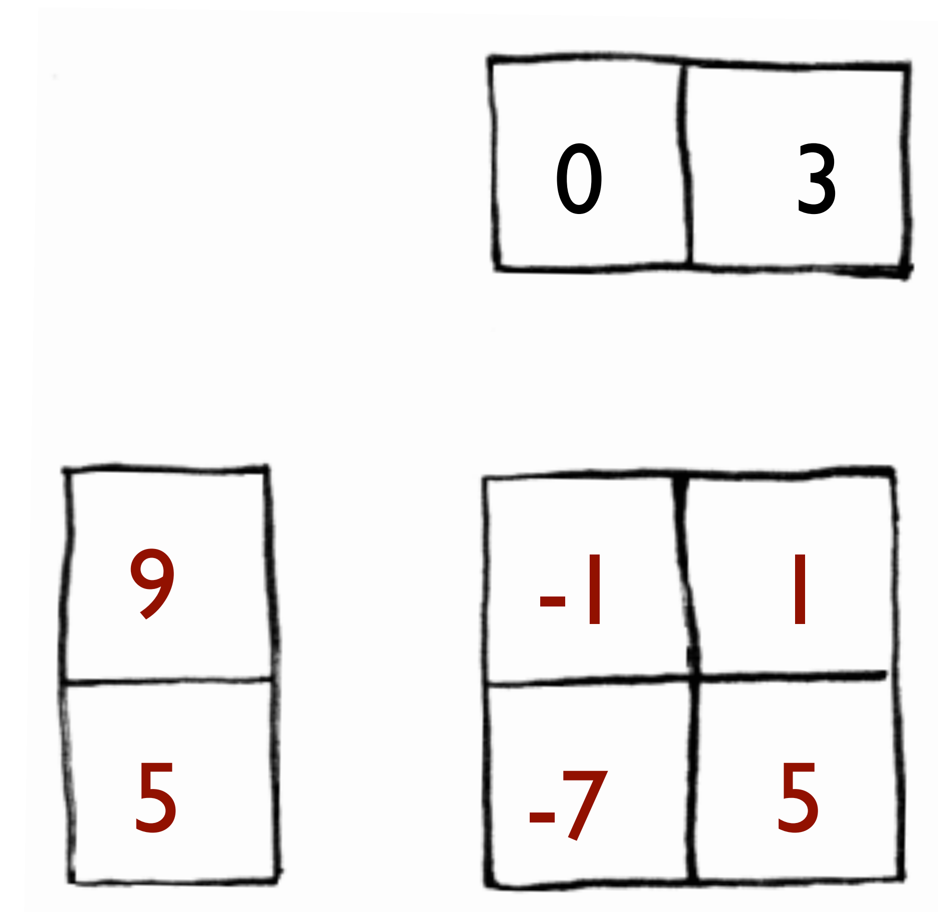
normalization constraints

For C, we introduce an extra variable x_C, together with the constraint $x_C = 1$.

Max-sum with the in-the-middle algorithm



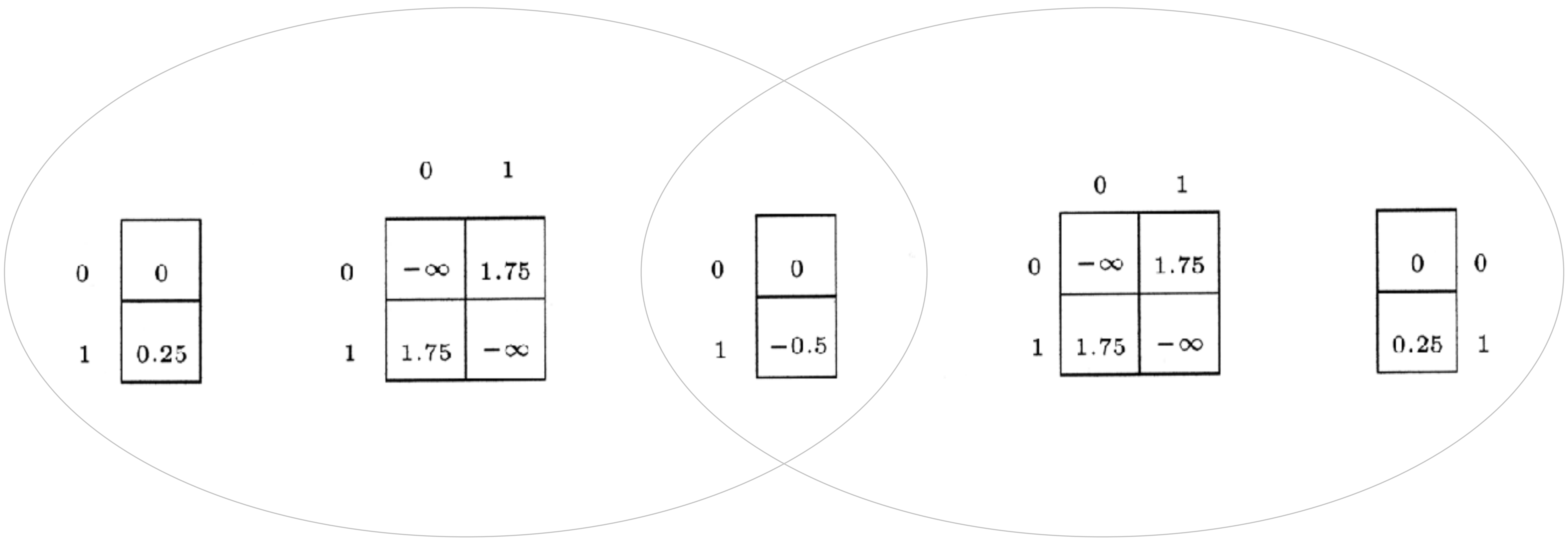
start



in-the-middle

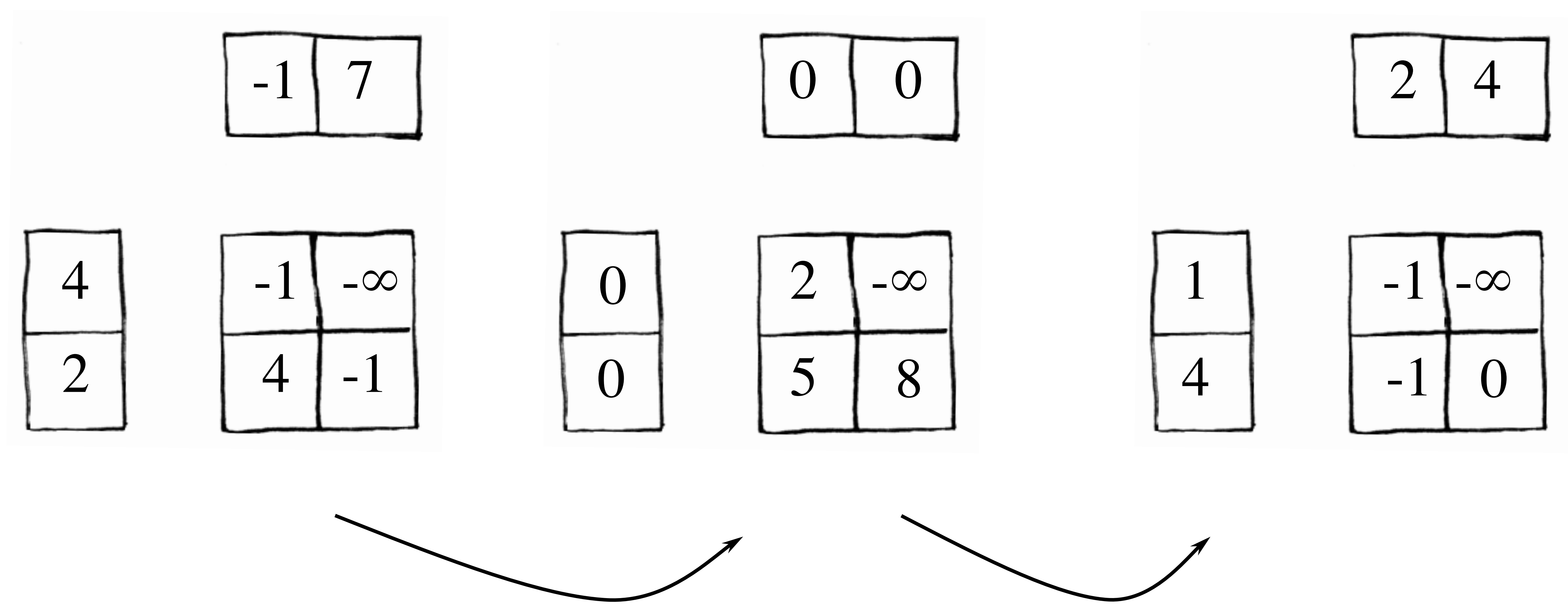
(makes linear constraint feasible)

Or solve entire subproblems with specialised algorithm!



Iteratively update one subproblem at a time

Move in and move out…



move in variable components

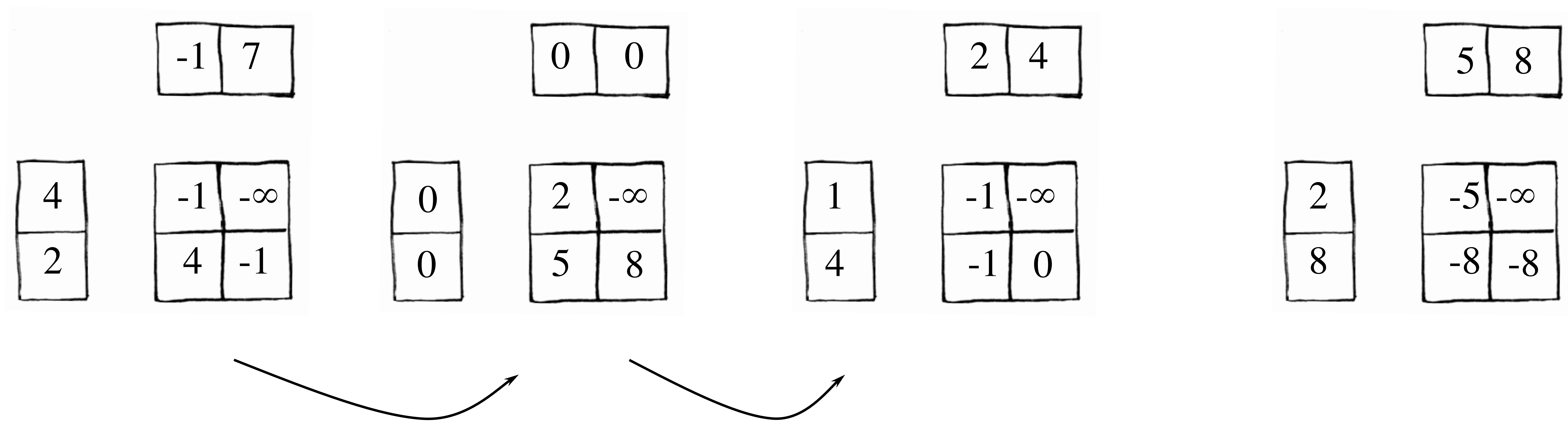
move out, but not too much

We want to keep the best combination still largest in constraint!

"non-conflicting"

minimizes f(y)

What if we move out "too much"?

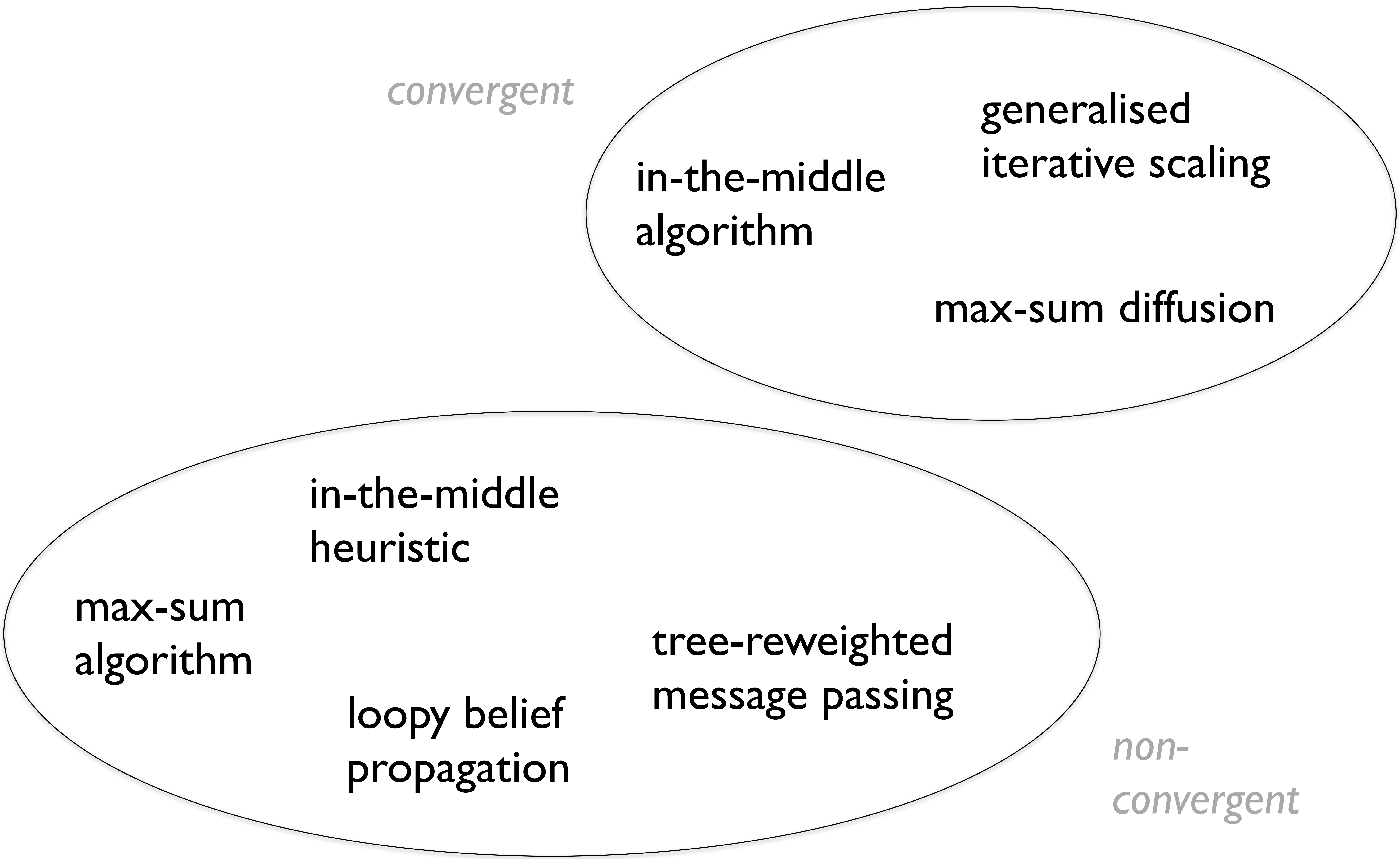


here we have moved out even more, we get a "conflict"!

(max-sum algorithm!)

We can easily explain why the *__max-sum algorithm__* *does not guarantee optimality!*

Many interesting relationships between algorithms!



Can we unify the models?

$$\max_{w} f(w) = \sum_{k} g_k(w^k) + C$$

$$\max cx$$

$$Ax = b$$

$$x_j \in \{0, 1\}$$

Let's model the other way around: ILP to max-sum!

$$\max\{2x_1 + 3x_2 + 2x_3 \mid x_1 + x_2 = 1,\ x_2 + x_3 = 1,\ x_j \text{ binary}\}.$$

$$\max \quad g_1(x_1) + g_2(x_2) + g_3(x_3) + g_4(x_1, x_2) + g_5(x_2, x_3).$$

$g_1(x_1)$

0	0
1	2

$g_4(x_1, x_2)$

	0	1
0	$-\infty$	0
1	0	$-\infty$

$g_2(x_2)$

0	0
1	3

$g_5(x_2, x_3)$

	0	1
0	$-\infty$	0
1	0	$-\infty$

$g_3(x_3)$

0	0
2	1

$$\max_{w} f(w) = \sum_{k} g_k(w^k) + C$$

$$\max cx$$
$$Ax = b$$
$$x_j \in \{0, 1\}$$

if we move out as much as possible but not too much… $\Rightarrow$ same as in-the-middle algorithm for the original ILP

max-sum algorithm $\Rightarrow$ same as in-the-middle heuristic (for α=1)

The distinction between an ILP model and a max-sum model is blurred!

The in-the-middle updates can be seen as fast specialized max-sum constraint updates!

To note from a constraint programming perspective

Numerical propagation can solve many non-trivial problems with propagation only!

(no combinatorial search!)

Summary

in-the-middle algorithm
in-the-middle heuristic

unify
algorithms!

unify
models!

END