S7303 - FINDING PARALLELISM IN GENERAL-PURPOSE LINEAR PROGRAMMING

TECHNISCHE UNIVERSITÄT DARMSTADT

Daniel Thuerck[1,2] (advisors Michael Goesele[1,2] and Marc Pfetsch[1])

Maxim Naumov[3]

[1] Graduate School of Computational Engineering, TU Darmstadt

[2] Graphics, Capture and Massively Parallel Computing, TU Darmstadt

[3] NVIDIA Research

culip

INTRODUCTION TO LINEAR PROGRAMMING

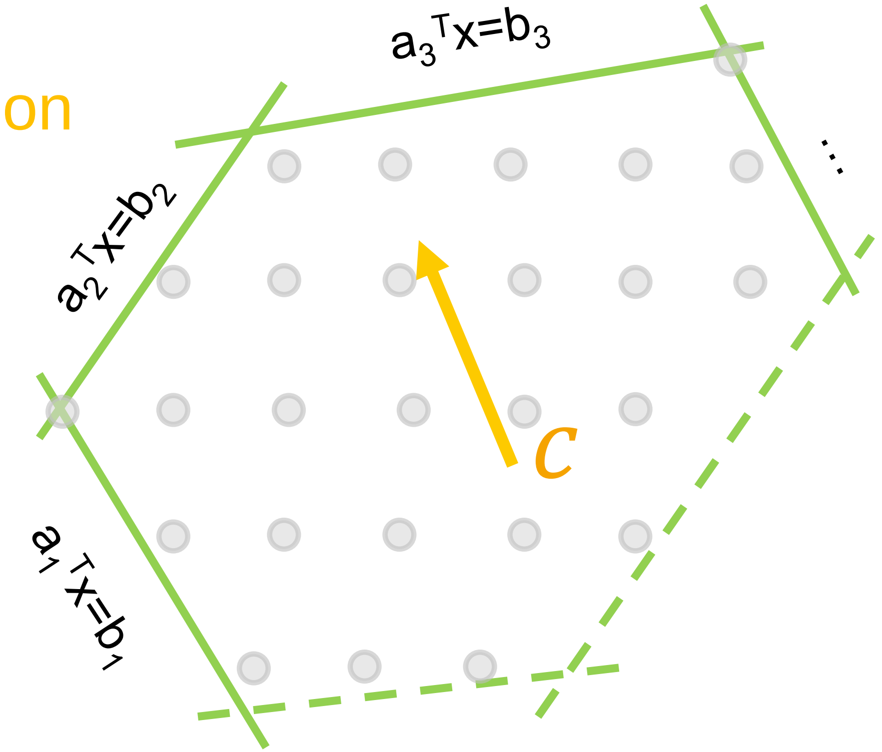
Linear Programs

$$\begin{aligned} \min \quad & c^\top x \\ \text{s.t.} \quad & Ax \le b \\ & x \ge 0 \end{aligned}$$

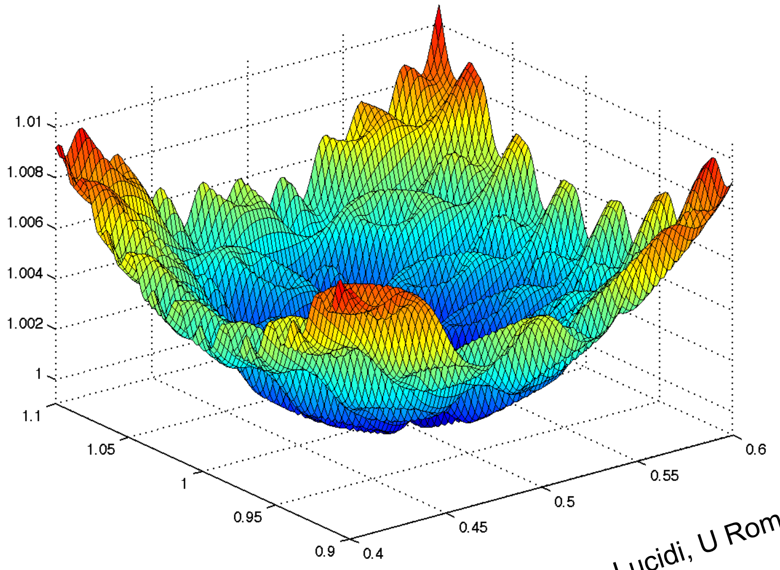
Linear objective function

Linear constraints

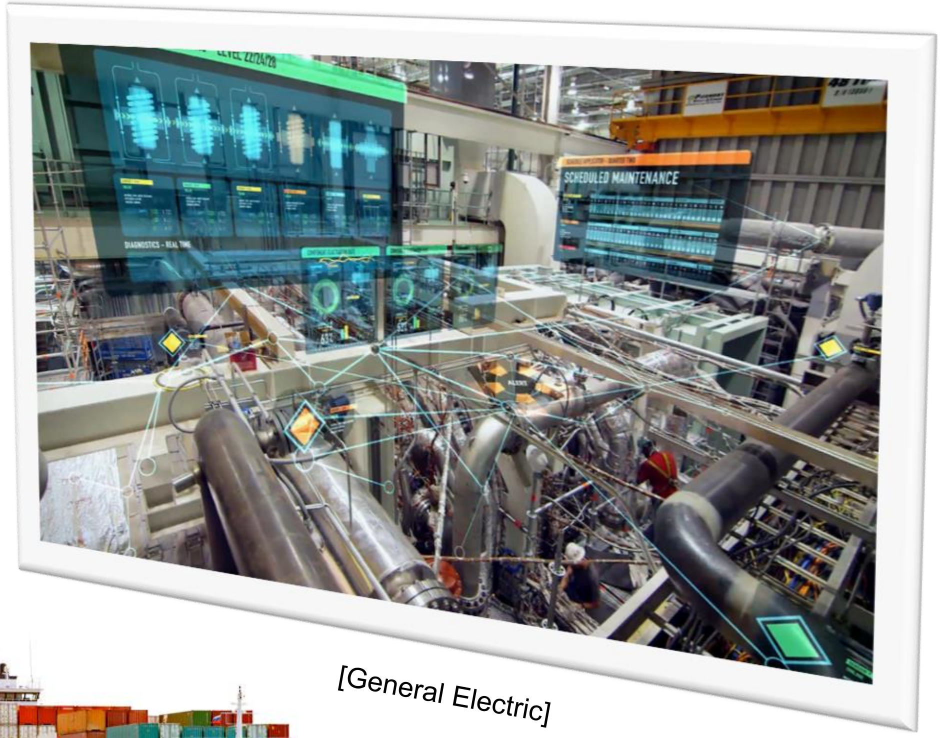
where $A = \begin{bmatrix} a_1{}^T \\ \\ a_m{}^T \end{bmatrix}$ and $b = \begin{bmatrix} b_1 \\ \\ b_m \end{bmatrix}$



Linear Programs: Applications



[Stefano Lucidi, U Roma]



[General Electric]

[3P Logistics]

Lower-Level Parallelism in LP

INTERNALS OF AN LP SOLVER

Solving LPs

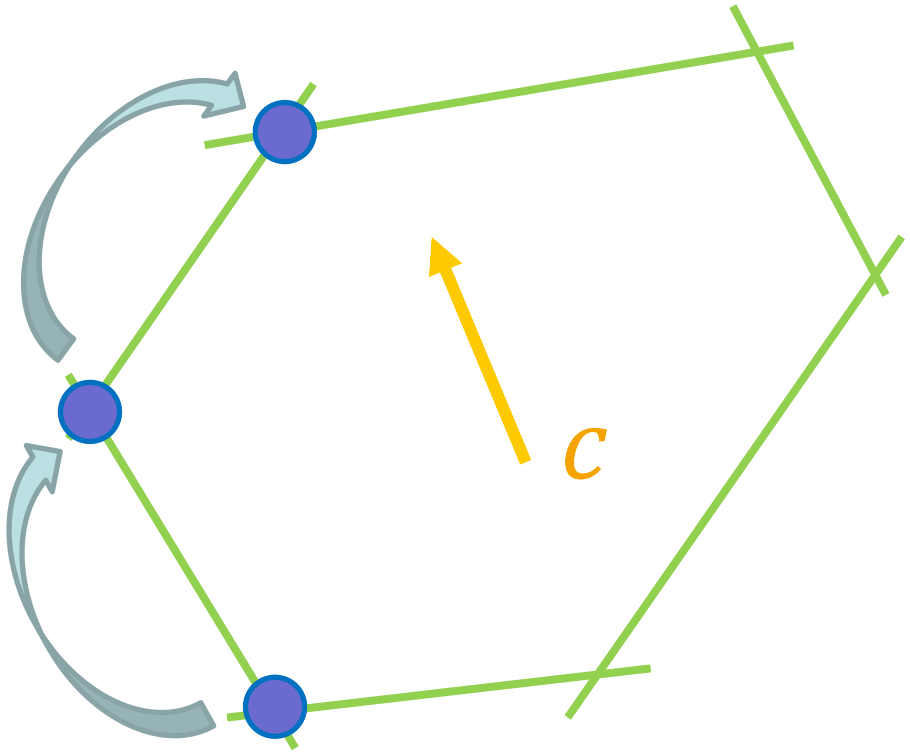
$$\begin{aligned} \min \quad & c^\top x \\ \text{s.t.} \quad & Ax = b \\ & x \geq 0 \end{aligned}$$

"Standard" LP format

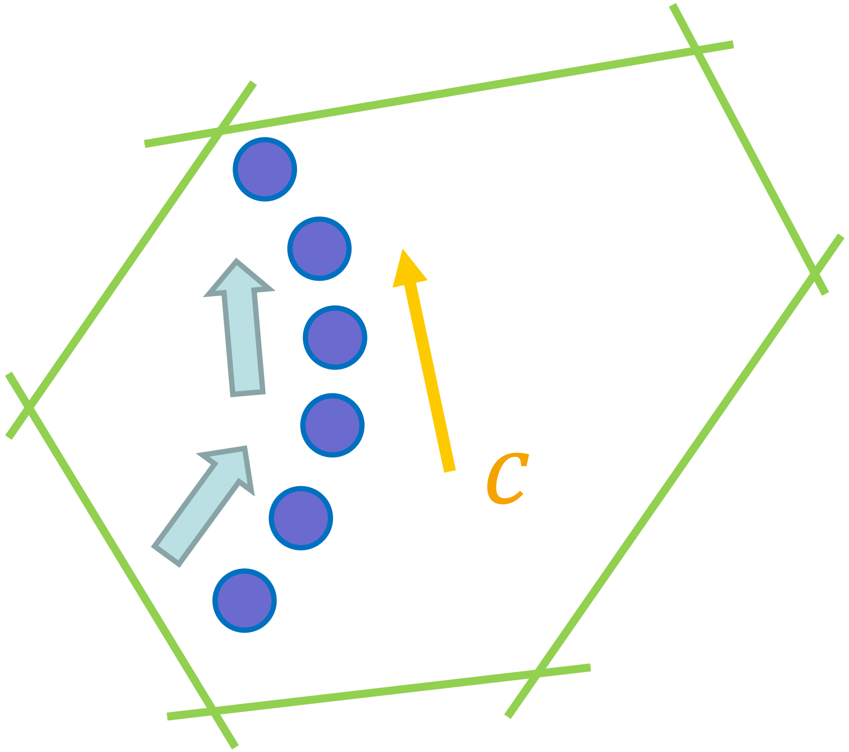
- A is $m \times n$ matrix, with $m \ll n$
- A is sparse and has full row-rank
- Variables may be bounded: $l \leq x \leq u$

Solving LPs

Simplex

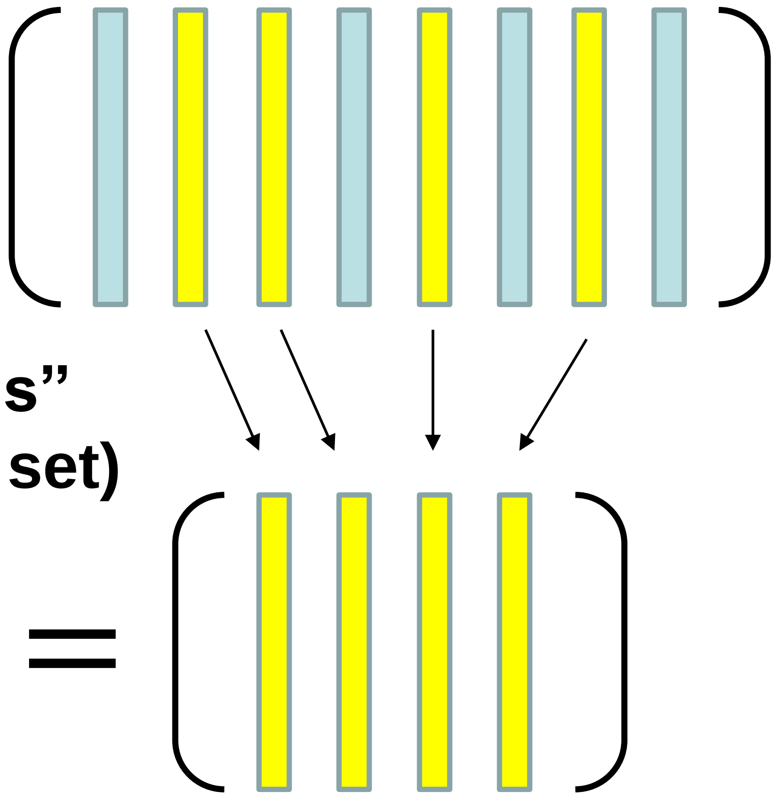


Interior Point



Solving LPs

Simplex



“Basis”
(active set)

$$A_B =$$

Interior Point (IPM)

“Augmented (Newton) System”

$$\begin{bmatrix} D & A^{\top} \\ A & \end{bmatrix}$$

“Normal Equations”

$$AD^{-1}A^{\top}$$

Solving LPs

IPM / Aug. System

$$\begin{bmatrix} D & A^{\mathsf{T}} \\ A & \end{bmatrix}$$

- $(m+n) \times (m+n)$, sparse
- Symmetric, indefinite
- Solution: Indefinite $\mathrm{LDL}^{\mathrm{T}}$ or MINRES method

IPM / Normal Equations

$$AD^{-1}A^{\mathsf{T}}$$

- $m \times m$, SPD, **might be dense**
- Squared condition number
- Solution: Cholesky-factorization or CG method

Solving LPs

IPM / Aug. System

$$\begin{bmatrix} D & A^{\top} \\ A & \end{bmatrix}$$

- $(m+n) \times (m+n)$, sparse
- Symmetric, indefinite
- Solution: Indefinite $\mathrm{LDL}^{\mathrm{T}}$ or MINRES method

IPM / Normal Equations

$$AD^{-1}A^{\top}$$

- $m \times m$, SPD, **might be dense**
- Squared condition number
- Solution: Cholesky-factorization or CG method

Introducing culip-lp…

An ongoing implementation of Mehrotra's Primal-Dual interior point algorithm [1], featuring…

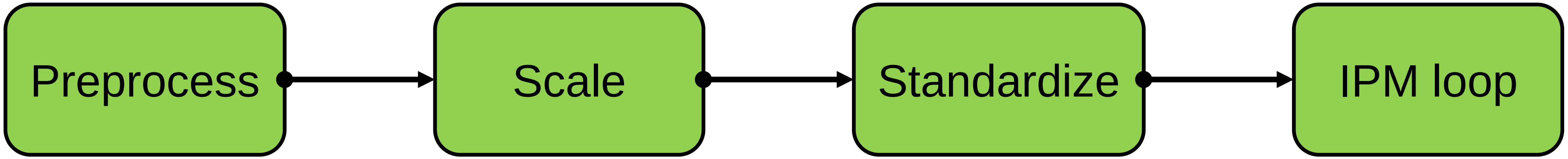
- ✓ (**Iterative**) Linear Algebra based on the "**Augmented Matrix**" approach,
- ✓ **Full-rank** guarantees,
- ✓ Comprehensive **preprocessing** & **scaling**.

➡ Towards solving large-scale LPs on the GPU as **open source** for everybody

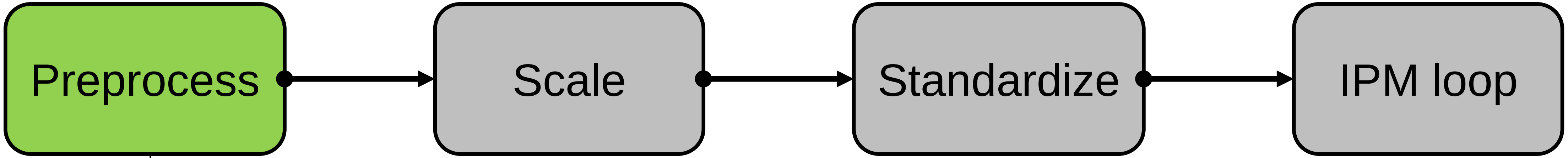
Progress report

IMPLEMENTING CULIP-LP

Solver architecture



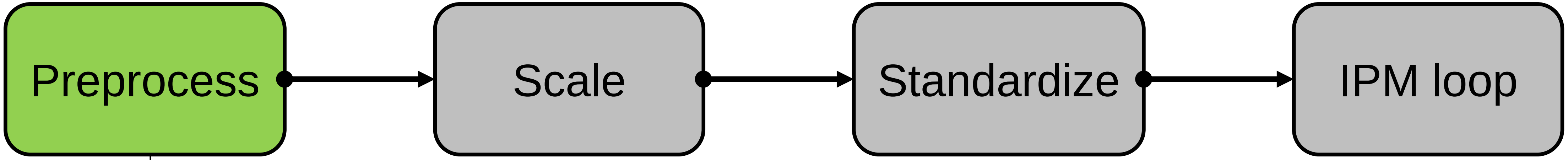
Solver architecture



Input data:

- Constraints $A_{eq}x = b_{eq}$
- Constraints $A_{le}x \leq b_{le}$
- Objective vector c
- Bounds (on some variables) l, u

Solver architecture



$A_{eq}x = b_{eq}$

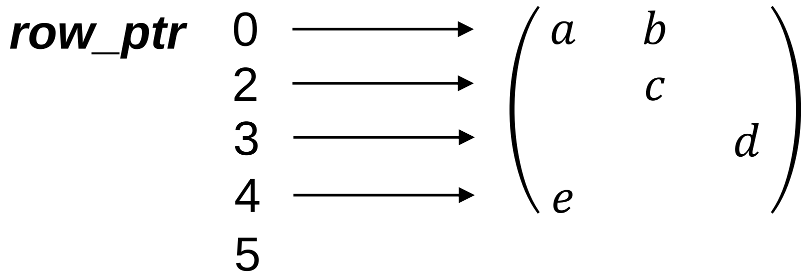
$A_{le}x \leq b_{le}$

c

l, u

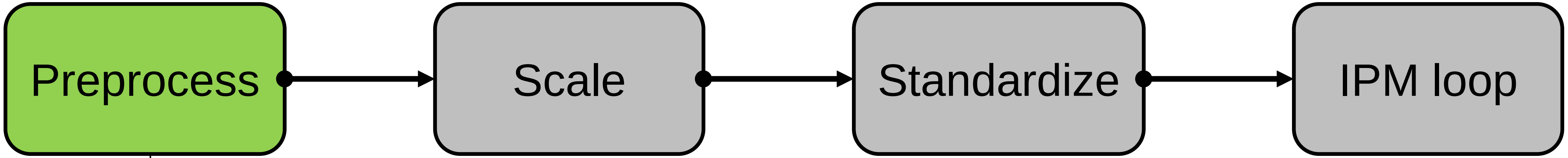
Storage format: CSR

- Compressed sparse row format
- Provides efficient row-based access by **3 arrays**:



0	1	1	2	0	***col_Ind***
a	b	c	d	e	***val***

Solver architecture



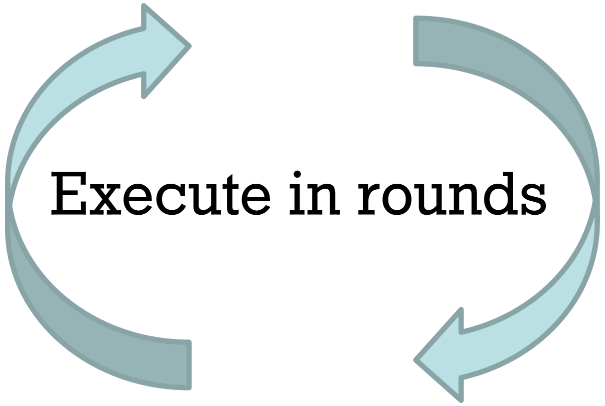
$$A_{eq}x = b_{eq}$$

$$A_{le}x \leq b_{le}$$

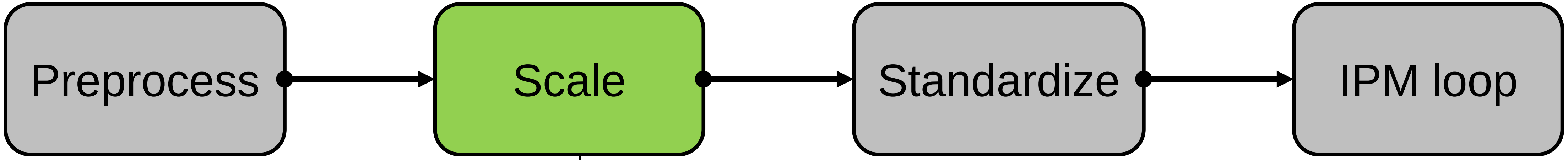
$$c$$

$$l, u$$

- **Example:** LP “pb-simp-nonunif” (see [2])
  - Input matrix: 1,4 Mio x 23k with 4,36 Mio nonzeros
  - Removed 1 singleton inequality
  - Removed 3629 low-forcing constraints
  - Removed 1 fixed variable
  - Removed 1,1 Mio (!) singleton inequalities
  - Result: approx. **3,6 Mio** nonzeros removed



Solver architecture



$$A_{eq}x = b_{eq}$$

$$A_{le}x \leq b_{le}$$

$$c$$

$$l, u$$

Goal: Reduce element variance in matrices

➢ Scaling [3] makes a difference

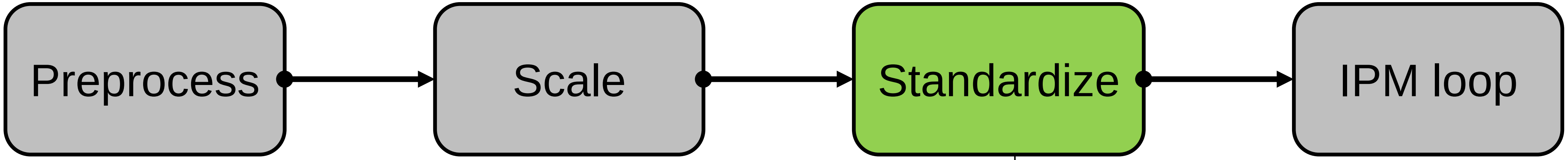
1. Geometric scaling (1x – 4x)

$$A_{i,\cdot} = \frac{A_{i,\cdot}}{\max\left(|A_{i,\cdot}|\right)\min\left(|A_{i,\cdot}|\right)}$$

2. Equilibration (1x)

$$A_{i,\cdot} = \frac{A_{i,\cdot}}{\left\|A_{i,\cdot}\right\|_2}$$

Solver architecture



Goal: Format LP in standard form

- Shift variables:

$$l \leq x \leq u \;\rightarrow 0 \leq x' \leq u + l$$

- Split (free) variables

$$x \rightarrow x = x^{+} - x^{-} \qquad x^{+}, x^{-} \geq 0$$

- Build std' matrix: $\begin{pmatrix} A_{le} & I \\ A_{eq} & \end{pmatrix} = \begin{pmatrix} b_{Le} \\ b_{eq} \end{pmatrix}$

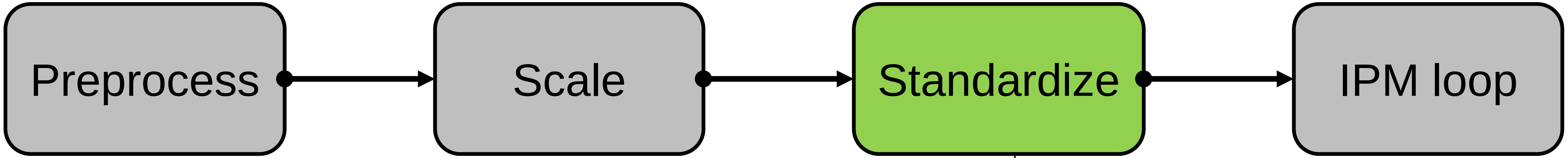
$$A_{eq}x = b_{eq}$$

$$A_{le}x \leq b_{le}$$

$$c$$

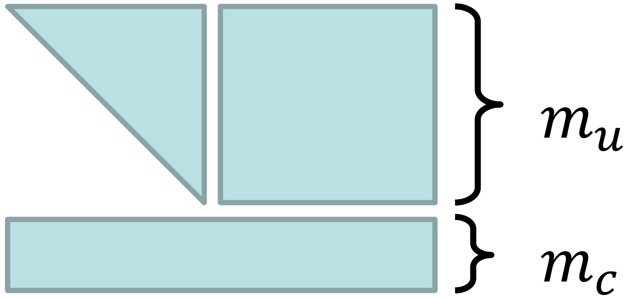
$$l, u$$

Solver architecture



Ensure A has full rank (symbolically)

$$PAQ =$$



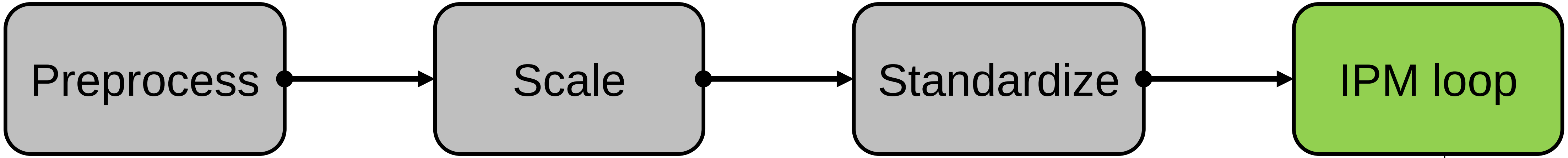
$$m_u \leq rank(A) \leq structural\ rank(A)$$

$Ax = b$

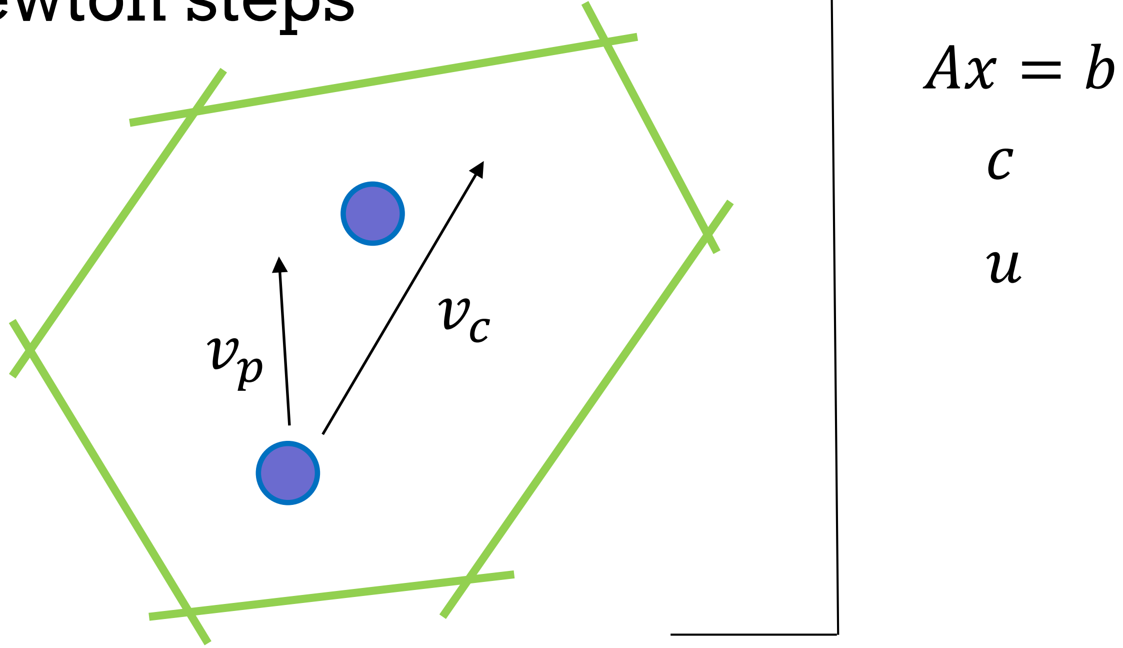
c

u

Solver architecture



- **Goal**: Solve KKT conditions by Newton steps
- Steps:
  - Augmented matrix assembly
  - **Solving the (indefinite) augmented matrix**
  - **Solve twice**: predictor and corrector
  - Stepsize along $v = v_p + v_c$



Solving the augmented system

$$\begin{bmatrix} D & A^{\top} \\ A & \end{bmatrix}$$

Iterative strategy:

- Symmetric, indefinite: use MINRES [4] (in parts)
- Equilibrate system implicitly
- Preconditioner: Experiments ongoing

Direct strategy: ~ 95% of computation

- Symmetric, indefinite: use SPRAL SSIDS [5]
- Reordering by METIS [6]
- Scaling for large pivots

Intermediate findings

PERFORMANCE EVALUATION

Benchmark problems

Problem name [7]	M	N	NNZ
ex9	40,962	10,404	517,112
ex10	696,608	17,680	1,162,000
neos-631710	169,576	167,056	834,166
bley_xl1	175,620	5831	869,391
map06	328,818	164,547	549,920
map10	328,818	164,547	549,920
nb10tb	150,495	73340	1,172,289
neos-142912	58,726	416,040	1,855,220
in	1,526,202	1,449,074	6,811,639

Performance

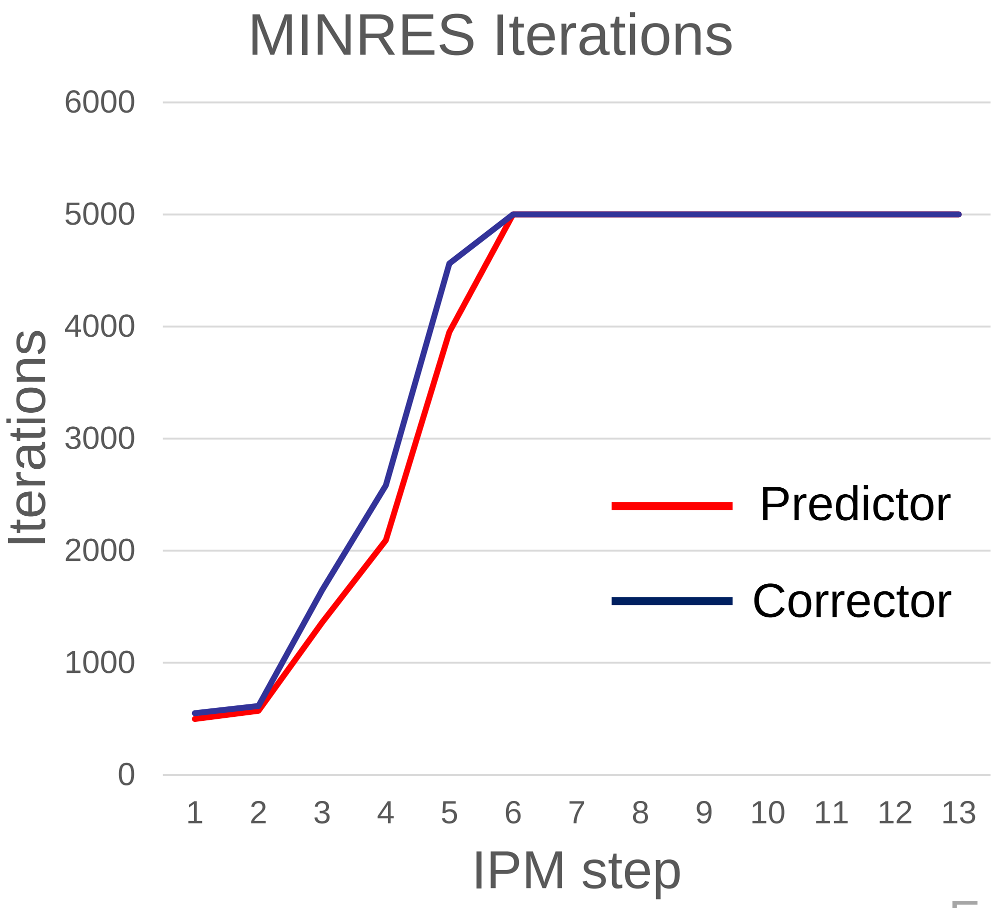
Problem name [7]	NNZ	CLP barr [sec]	culip-lp [sec]
ex9	517,112	X (NC)	**81**
ex10	1,162,000	X (NS)	**141**
neos-631710	834,166	**172**	478
bley_xl1	869,391	X (NS)	**1,492**
map06	549,920	X (NC)	**466**
map10	549,920	X (NC)	**615**
nb10tb	1,172,289	X (NC)	**2,461**
neos-142912	1,855,220	**356**	447
in	6,811,639	X (NS)	NC

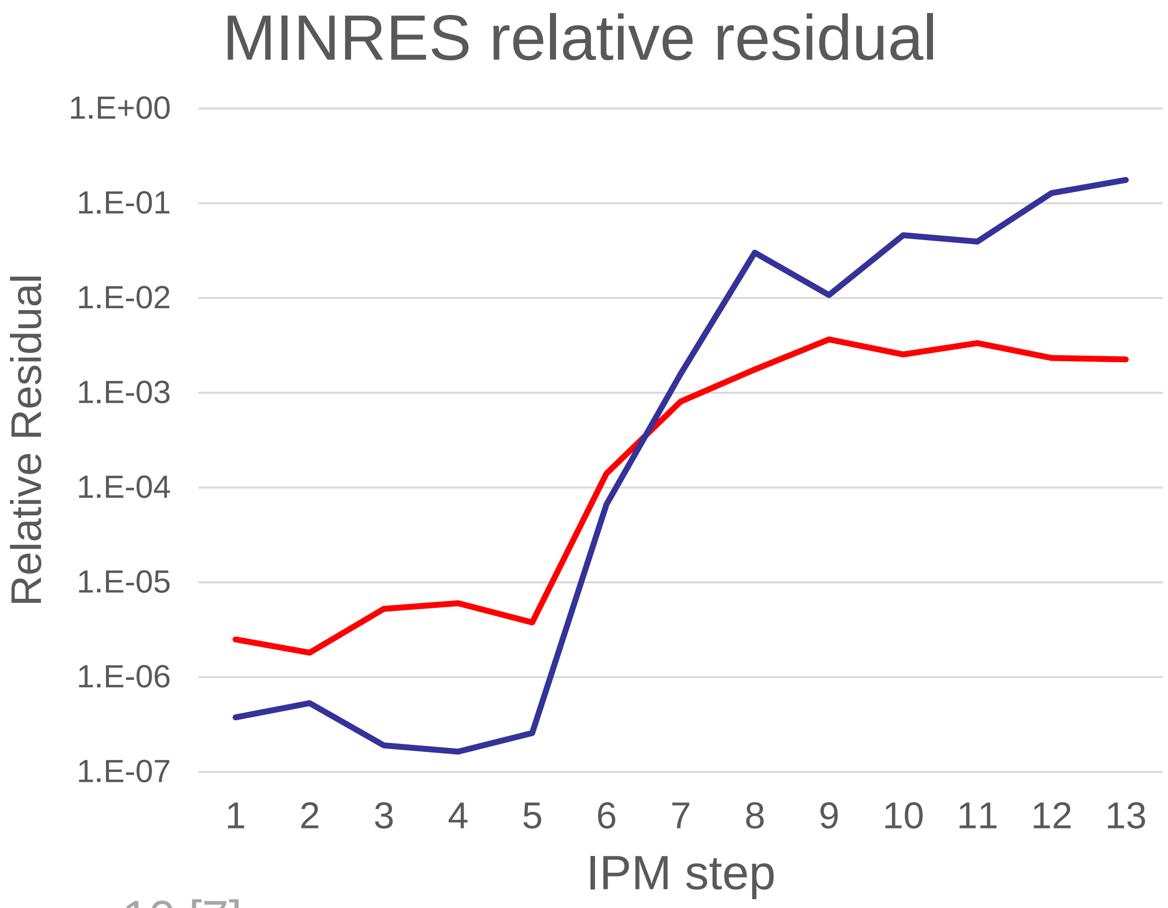
X – failed, **NS** – did not start 1st iteration, **NC** – did not converged within 1 hour

Runtime breakdown

Problem: map10 [7]

Iterative vs. direct methods





Example: map10 [7]

Numerical difficulty

TECHNISCHE UNIVERSITÄT DARMSTADT

Condition of matrix

$$\begin{bmatrix} D & A^\top \\ A & \end{bmatrix}$$

- depends mainly on

  $D = diag(x) \cdot diag(s)$

- with strong duality towards the end often yielding

  $$\frac{\max(x_i s_i)}{\min(x_i s_i)} \approx 10^{10}$$

Remedies

- 2x2 pivoting in factorizations (e.g. $LDL^\top$ in SPRAL)
- Preconditioning for MINRES or GMRES

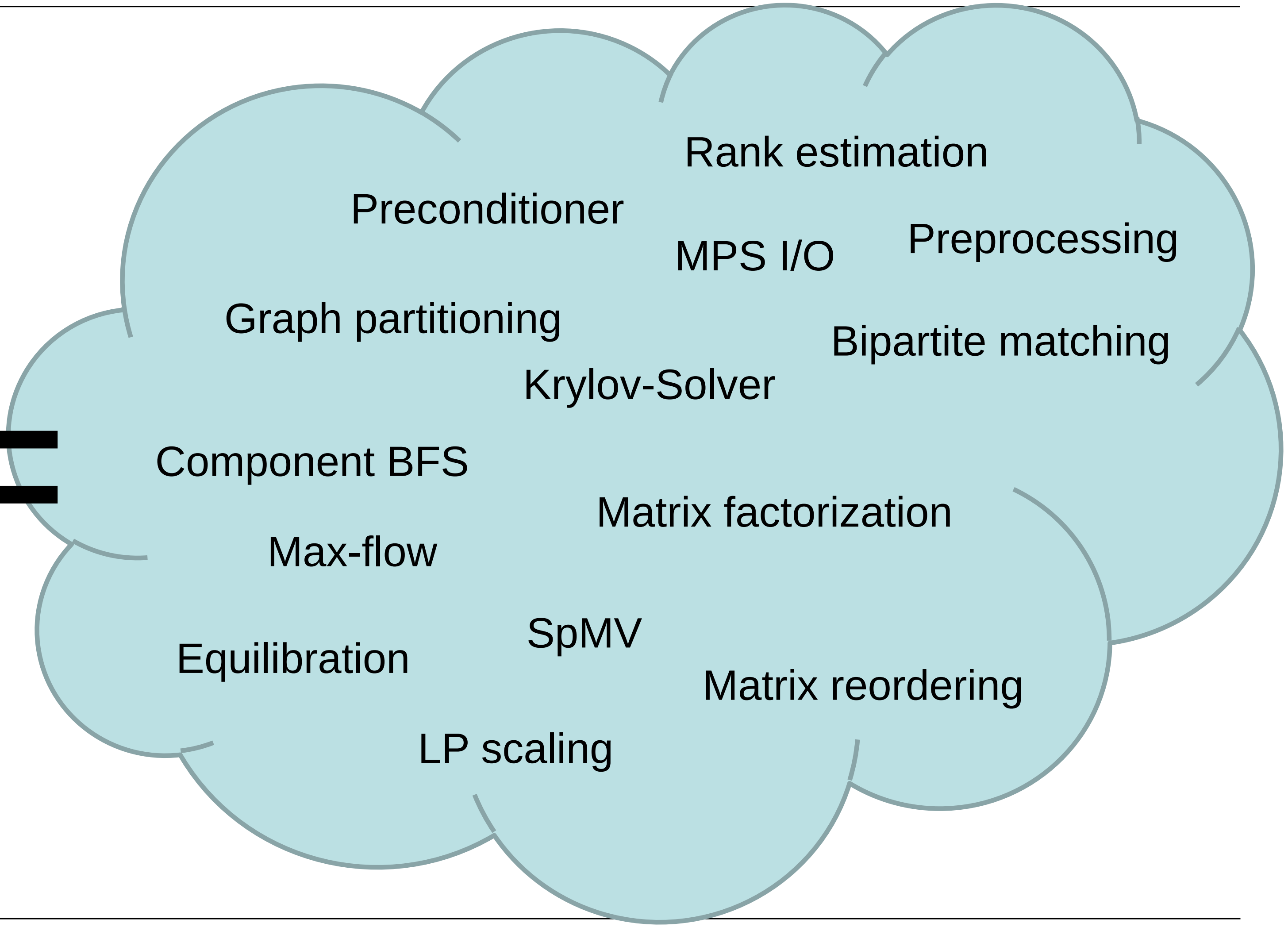
expect speed-up here

where
$x^T=[x_1,\ldots,x_n]$ are solution and
$s^T=[s_1,\ldots,s_n]$ are slack variables

What’s keeping you from optimizing your runtime?

LP Solver
(a.k.a “the black box”)

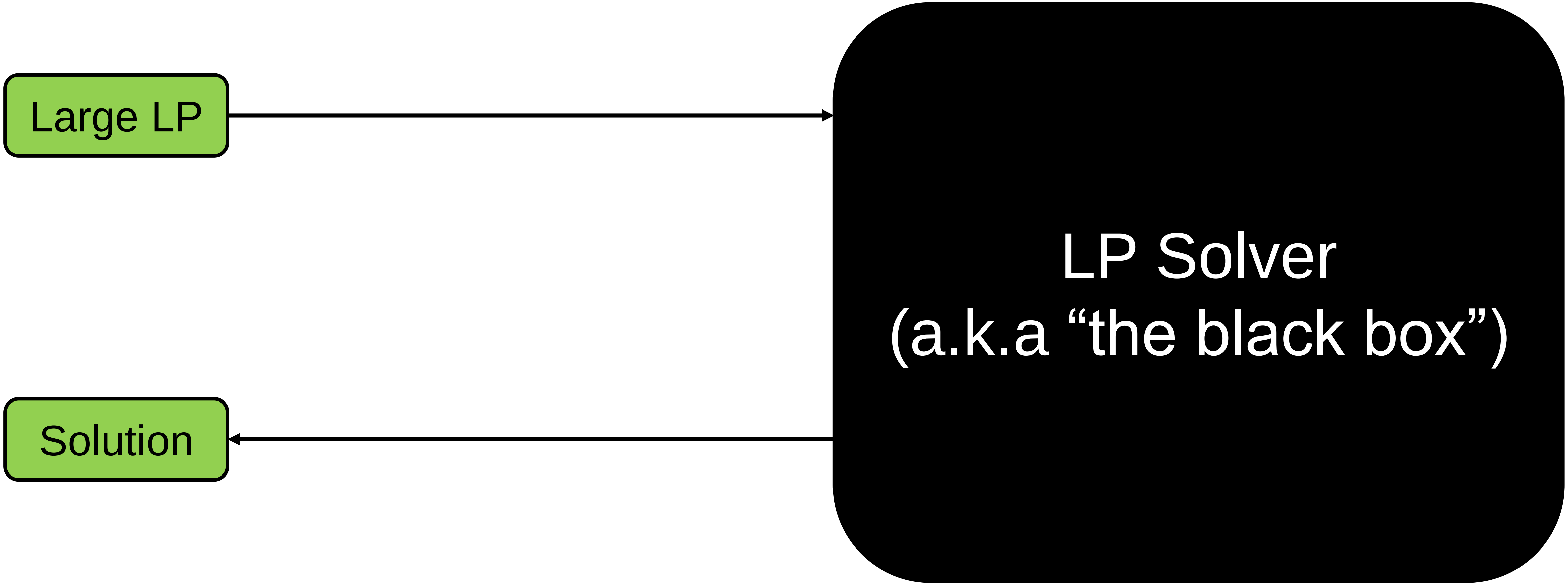
=



Higher-Level Parallelism in LP

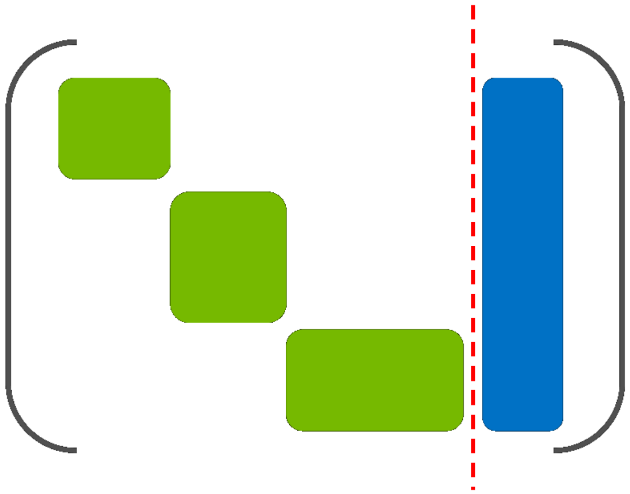
FEASIBILITY STUDY: LP DECOMPOSITIONS

Solving an LP: The usual setup

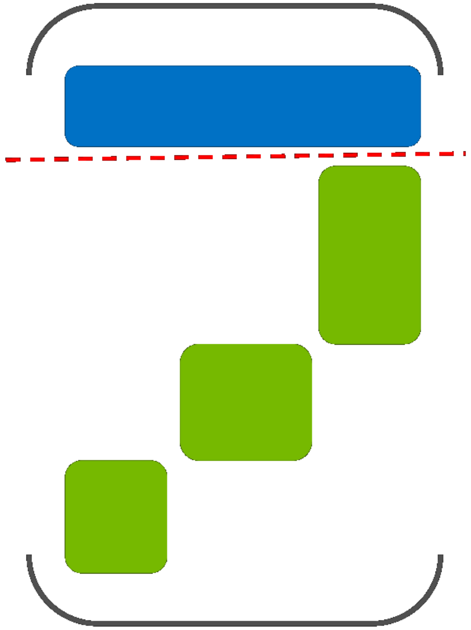


LP-decompositions: feasibility

Decomposition works on structure of the constraint matrix A:

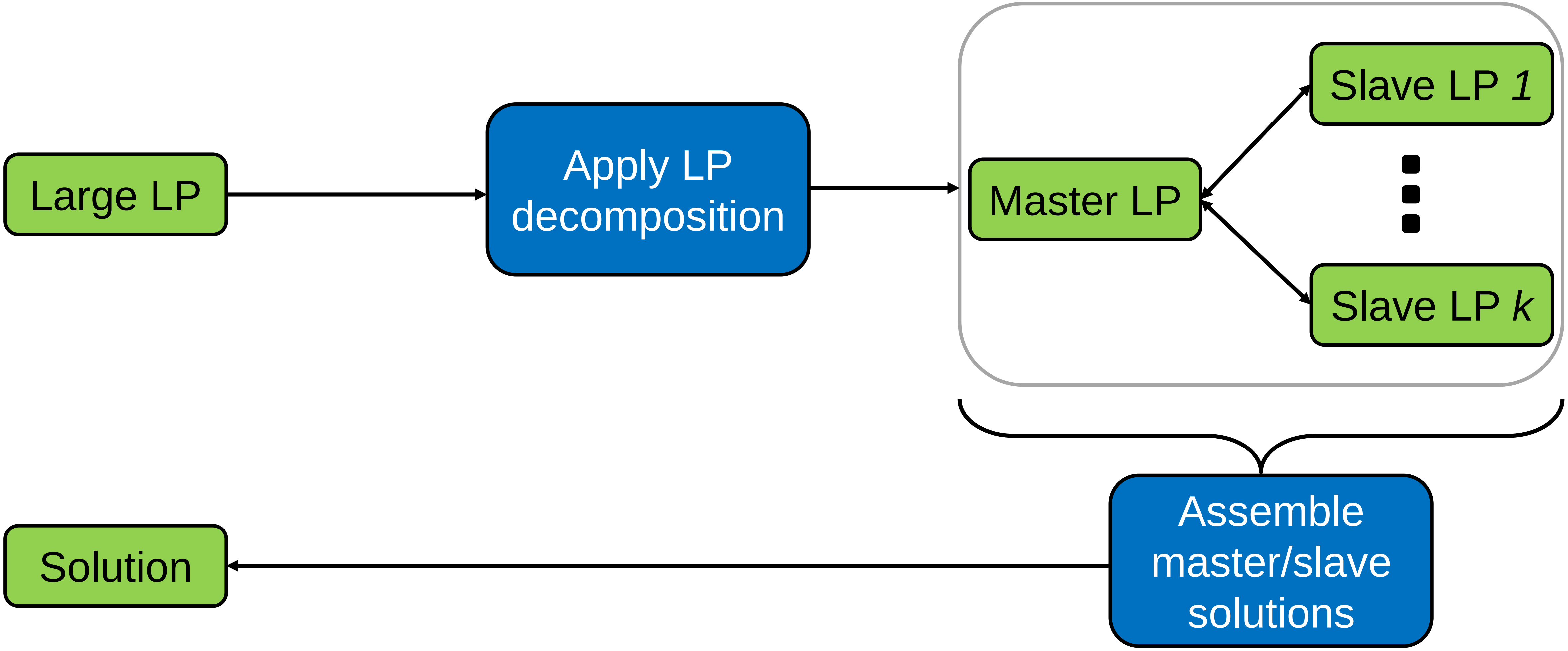


Benders [9]



Dantzig-Wolfe [10]

Higher-level parallelism by LP decomposition



LP-decompositions: prototype

Implemented a Benders' decomposition using hypergraph partitioning:

Name	M	N	K	# iterations	# statics	# coupling
pigeon-10	1331	430	2	93	58	21
	1331	430	4	102	71	27
	1331	430	6	102	71	28
glass4	715	322	2	15	4	4
	715	322	4	14	5	5
	715	322	6	13	6	6
aflow40b (dual)	4170	2728	4	1048	0	3
	4170	2728	6	1017	0	3

Acknowledgements

The work of Daniel Thuerck is supported by the

'Excellence Initiative' of the German Federal and State Governments

and the

Graduate School of Computational Engineering

at Technische Universität Darmstadt.

References

[1] Mehrotra, Sanjay. "On the implementation of a primal-dual interior point method." *SIAM Journal on optimization* 2.4 (1992): 575-601.

[2] Gondzio, Jacek. "Presolve analysis of linear programs prior to applying an interior point method." *INFORMS Journal on Computing* 9.1 (1997): 73-91.

[3] Gondzio, Jacek. "Presolve analysis of linear programs prior to applying an interior point method." *INFORMS Journal on Computing* 9.1 (1997): 73-91.

[4] Paige, Christopher C., and Michael A. Saunders. "Solution of sparse indefinite systems of linear equations." *SIAM journal on numerical analysis* 12.4 (1975): 617-629.

[5] Paige, Christopher C., and Michael A. Saunders. "Solution of sparse indefinite systems of linear equations." *SIAM journal on numerical analysis* 12.4 (1975): 617-629.

References

[6] Karypis, George, and Vipin Kumar. "A fast and high quality multilevel scheme for partitioning irregular graphs." *SIAM Journal on scientific Computing* 20.1 (1998): 359-392.

[7] Koch, Thorsten, et al. "MIPLIB 2010." *Mathematical Programming Computation* 3.2 (2011): 103-163.

[8] Forrest, John, David de la Nuez, and Robin Lougee-Heimer. "CLP user guide." *IBM Research* (2004).

[9] Benders, Jacques F. "Partitioning procedures for solving mixed-variables programming problems." *Numerische mathematik* 4.1 (1962): 238-252.

[10] Dantzig, George B., and Philip Wolfe. "Decomposition principle for linear programs." *Operations research* 8.1 (1960): 101-111.