

HiGHS: a high-performance linear optimizer

Turning gradware into software

Julian Hall

School of Mathematics, University of Edinburgh

ICCOPT, Berlin

6 August 2019



THE UNIVERSITY
of EDINBURGH



- Linear optimization

- Linear programming (LP)

$$\text{minimize } \mathbf{c}^T \mathbf{x} \quad \text{subject to } \mathbf{Ax} = \mathbf{b} \quad \mathbf{x} \geq \mathbf{0}$$

- Convex quadratic programming (QP)

$$\text{minimize } \frac{1}{2} \mathbf{x}^T \mathbf{Q} \mathbf{x} + \mathbf{c}^T \mathbf{x} \quad \text{subject to } \mathbf{Ax} = \mathbf{b} \quad \mathbf{x} \geq \mathbf{0}$$

$\mathbf{Q}$ positive semi-definite

- High performance

- Serial techniques exploiting sparsity in $\mathbf{A}$
 - Parallel techniques exploiting multicore architectures

HiGHS: The team

What's in a name?

HiGHS: **H**all, **i**vet **G**alabova, **H**uangfu and **S**chork

Team HiGHS

- Julian Hall: Reader (1990–date)
- Ivet Galabova: PhD (2016–date)
- Qi Huangfu
 - PhD (2009–2013)
 - FICO Xpress (2013–2018)
 - MSc (2018–date)
- Lukas Schork: PhD (2015–2018)
- Michael Feldmeier: PhD (2018–date)
- Joshua Fogg: PhD (2019–date)



HiGHS: Past (2011–2014)

Overview

- Written in C++ to study parallel simplex
- Dual simplex with standard algorithmic enhancements
- Efficient numerical linear algebra
- No interface or utilities

Concept

- High performance serial solver (`hso1`)
- Exploit limited task and data parallelism in standard dual RSM iterations (`sip`)
- Exploit greater task and data parallelism via minor iterations of dual SSM (`pami`)

Huangfu and H

HiGHS: Dual simplex algorithm

Assume $\hat{\mathbf{c}}_N \geq \mathbf{0}$ Seek $\hat{\mathbf{b}} \geq \mathbf{0}$

Scan $\hat{b}_i < 0$ for p to leave $\mathcal{B}$

Scan $\hat{c}_j / \hat{a}_{pj} < 0$ for q to leave $\mathcal{N}$

Update: Exchange p and q between $\mathcal{B}$ and $\mathcal{N}$

Update $\hat{\mathbf{b}} := \hat{\mathbf{b}} - \alpha_P \hat{\mathbf{a}}_q$ $\alpha_P = \hat{b}_p / \hat{a}_{pq}$

Update $\hat{\mathbf{c}}_N^T := \hat{\mathbf{c}}_N^T + \alpha_D \hat{\mathbf{a}}_p^T$ $\alpha_D = -\hat{c}_q / \hat{a}_{pq}$

Data required

- Pivotal row $\hat{\mathbf{a}}_p^T = \mathbf{e}_p^T B^{-1} N$
- Pivotal column $\hat{\mathbf{a}}_q = B^{-1} \mathbf{a}_q$

	$\mathcal{N}$		RHS
$\mathcal{B}$	$\hat{\mathbf{a}}_q$		$\hat{\mathbf{b}}$
	$\hat{a}_{pq}$	$\hat{\mathbf{a}}_p^T$	$\hat{b}_p$
	$\hat{c}_q$	$\hat{\mathbf{c}}_N^T$	

HiGHS: Dual simplex algorithm

Assume $\hat{\mathbf{c}}_N \geq \mathbf{0}$ Seek $\hat{\mathbf{b}} \geq \mathbf{0}$

Scan $\hat{b}_i < 0$ for p to leave $\mathcal{B}$

Scan $\hat{c}_j / \hat{a}_{pj} < 0$ for q to leave $\mathcal{N}$

Update: Exchange p and q between $\mathcal{B}$ and $\mathcal{N}$

Update $\hat{\mathbf{b}} := \hat{\mathbf{b}} - \alpha_P \hat{\mathbf{a}}_q$ $\alpha_P = \hat{b}_p / \hat{a}_{pq}$

Update $\hat{\mathbf{c}}_N^T := \hat{\mathbf{c}}_N^T + \alpha_D \hat{\mathbf{a}}_p^T$ $\alpha_D = -\hat{c}_q / \hat{a}_{pq}$

	$\mathcal{N}$		RHS
$\mathcal{B}$	$\hat{\mathbf{a}}_q$		$\hat{\mathbf{b}}$
	$\hat{a}_{pq}$	$\hat{\mathbf{a}}_p^T$	$\hat{b}_p$
	$\hat{c}_q$	$\hat{\mathbf{c}}_N^T$	

Computation

Pivotal row via $B^T \pi_p = \mathbf{e}_p$ **BTRAN** and $\hat{\mathbf{a}}_p^T = \pi_p^T N$ **PRICE**

Pivotal column via $B \hat{\mathbf{a}}_q = \mathbf{a}_q$ **FTRAN** Represent B^{-1} **INVERT**

Update B^{-1} exploiting $\bar{B} = B + (\mathbf{a}_q - B\mathbf{e}_p)\mathbf{e}_p^T$ **UPDATE**

HiGHS: Multiple iteration parallelism with pami option

- Perform standard dual simplex minor iterations for rows in set $\mathcal{P}$ ($|\mathcal{P}| \ll m$)
- Suggested by Rosander (1975) but never implemented efficiently *in serial*

	$\mathcal{N}$	RHS
$\mathcal{B}$	$\hat{\mathbf{a}}_{\mathcal{P}}^T$	$\hat{\mathbf{b}}$
		$\hat{b}_{\mathcal{P}}$
	$\hat{\mathbf{c}}_N^T$	

- Task-parallel multiple BTRAN to form $\boldsymbol{\pi}_{\mathcal{P}} = \mathbf{B}^{-T} \mathbf{e}_{\mathcal{P}}$
- Data-parallel PRICE to form $\hat{\mathbf{a}}_p^T$ (as required)
- Task-parallel multiple FTRAN for primal, dual and weight updates

Huangfu and H (2011–2014)
MPC best paper prize (2018)

HiGHS: Performance and reliability

Extended testing using 159 test problems

- 98 Netlib
- 16 Kennington
- 4 Industrial
- 41 [Mittelman](#)

Exclude 7 which are “hard”

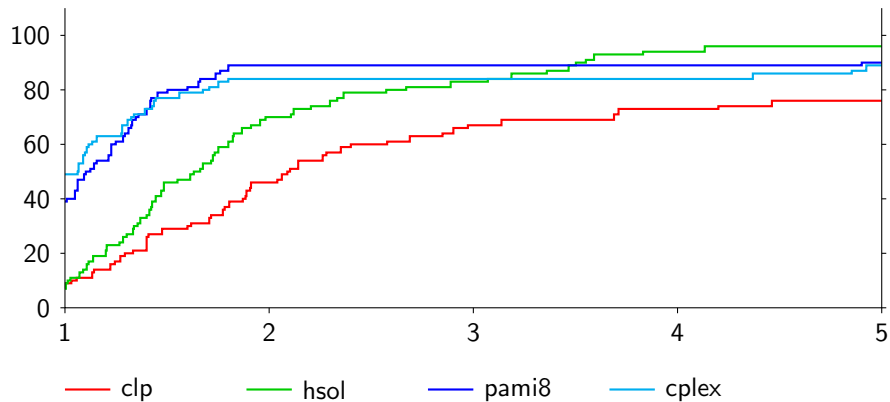
Performance

Benchmark against c1p (v1.16) and cplex (v12.5)

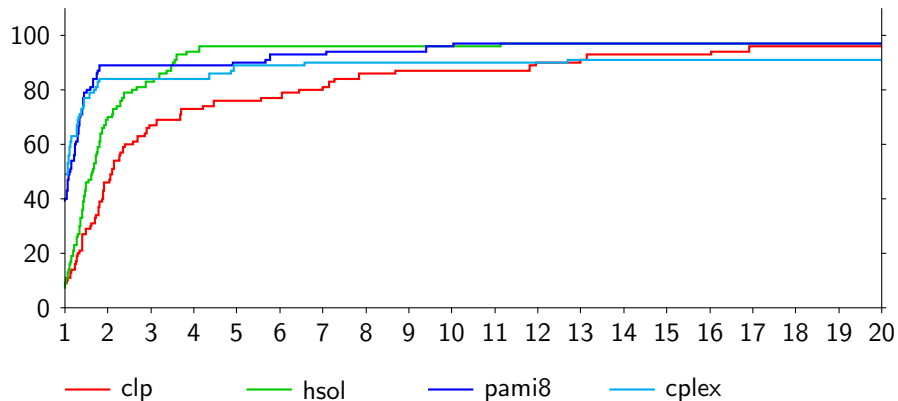
- Dual simplex
- No presolve
- No crash

Ignore results for 82 LPs with minimum solution time below 0.1s

HiGS: Performance



HiGHS: Reliability



HiGHS: **Intermission (2014-2016)**

Developments

- Model management: Load/add/delete/modify problem data
Feldmeier, Galabova, H
- Interfaces
Feldmeier, Galabova and Vigerske
- Presolve
Galabova
- Crash
Galabova and H
- Interior point method
Schork

Source

- Open source (MIT license)
- No third party code
- GitHub: [ERGO-Code/HiGHS](#)
- COIN-OR: Replacement for Clp?

Interfaces

- | • Existing | • Prototypes | • Planned |
|---|---|---|
| <ul style="list-style-type: none">• C++ HiGHS class• Load from .mps• Load from .lp• C• C#• Julia• FORTRAN• OSI (almost!) | <ul style="list-style-type: none">• GAMS• SCIP | <ul style="list-style-type: none">• AMPL• MATLAB• Mosel• PuLp• Python• R |
- Suggestions?

HiGHS: Benchmarking

- No more excuses!
- Use the 40 Mittelmann test LP problems
 - Some familiar - some not
 - Some easy - some not
 - Some new! (28/05/19)

	Rows	Cols	Nonzeros	$\frac{\text{Rows}}{\text{Cols}}$	$\frac{\text{Nonzeros}}{\text{Rows} \times \text{Cols}}$	$\frac{\text{Nonzeros}}{\max(\text{Rows}, \text{Cols})}$
Min	960	1560	38304	1/255	0.0005%	2.2
Geomean	54256	72442	910993	0.75	0.02%	6.5
Max	986069	1259121	11279748	85	16%	218.0

- Compete with other solvers in “vanilla” state

Aim: eliminate rows, columns and nonzeros

Wide range of techniques

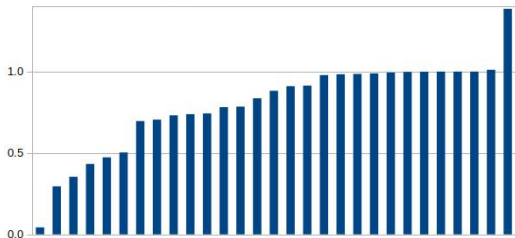
- Simple: interpret singleton rows as bounds on variables
- Complex: LP folding

Presolve measure

Product of

- Relative number of rows
- Relative number of columns
- Relative number of nonzeros

HiGHS presolve relative to C1p



Presolve measure relative to C1p

- Better than C1p for 2/29 LPs!
- Within a factor 0.9 for 14/29 LPs
- Within a factor 0.7 for 23/29 LPs
- Within a factor 0.3 for 28/29 LPs
- Poor for one LP!

Aim: Identify basis more likely to be feasible

- Start with “all-slack” basis so $B = I$
- Perform basis changes
 - Replace fixed slack with free/bounded/boxed structural
 - Maintain near-triangular B

Bixby (1992)

- More aggressive crash also aims to
 - Replace boxed slack with free/bounded structural
 - Replace bounded slack with free structural
 - Maintain triangular B

Maros and Mitra (1998)

- Designed for primal simplex method: can it be valuable for dual simplex method?

Aim: Identify basis more likely to be optimal

- “Idiot” crash

Forrest

HiGHS: Benchmarks (1 August 2019)

Commercial

- Xpress
- Gurobi
- Cplex
- Mosek
- COPT
- QSOPT
- Matlab

Open-source

- Clp (COIN-OR)
- Glop (Google)
- Soplex (ZIB)
- Glpk (GNU)
- Lpsolve

Solver	COPT	Clp	Mosek	HiGHS	Matlab	Glop	Soplex
Time	1	1.5	3.6	5.4	6.9	7.2	10
Solved	40	40	37	37	32	35	36

Solver	QSOPT	Glpk	Lpsolve
Time	26	28	108
Solved	34	31	23

HiGHS: Improvements

Solver	COPT	Clp	Mosek	HiGHS	Matlab	Glop	Soplex
Time	1	1.5	3.6	5.4	6.9	7.2	10
Solved	40	40	37	37	32	35	36

Why is the HiGHS score so bad?

- HiGHS triangular crash not used
- HiGHS parallel code not used
- Clp has a better presolve
- Clp has the Idiot crash
- Clp has a primal simplex solver

HiGHS: Tuning for the test problems

Everybody does it...

QAP problems

- Matrix entries $+1$ and -1 , but $B = LU$ fills in
- Dual steepest edge weights become very inaccurate
- Monitor errors and switch to Devex
- Improves solution time for QAP15/NUG15 by a factor of 15!
- Contributes to reliability on general LPs?

HiGHS score now improved from 5.4 to c.4.4

Solve all the problems!

- CONT11: Recognise when Markowitz tolerance ($u = 0.1$) needs to be increased
- IRISH-E: Add numerical pivot test to postsolve
- STP3D: Improve primal simplex “clean-up”

Immediate enhancements

- Add triangular crash
- Resurrect pami

HiGHS: The future

LP

- Improve presolve (Galabova)
- Add Idiot crash (Galabova)
- Add crossover (Hall)
- Add primal simplex solver (Huangfu)
- Improved Idiot crash (Galabova)
- Direct linear system solver for IPM (?)

MIP

- Publicise existing solver
- Improve existing solver

QP

- Active set QP solver (Feldmeier)
- IPM QP solver

Further interfaces

- | | | |
|----------|---------|----------|
| • AMPL | • Mosel | • Python |
| • MATLAB | • PuLp | • R |



- High performance LP solver: simplex and interior point
- Reads: .mps and .lp
- Interfaces: C++ (native) C, C#, Julia, FORTRAN
- Research and consultancy

Slides:

<http://www.maths.ed.ac.uk/hall/ICCOPT19>

HiGHS: <http://www.highs.dev/>



I. L. Galabova and J. A. J. Hall.

The "idiot" crash quadratic penalty algorithm for linear programming and its application to linearizations of quadratic assignment problems.

Optimization Methods and Software, April 2019.

Published online.



Q. Huangfu and J. A. J. Hall.

Novel update techniques for the revised simplex method.

Computational Optimization and Applications,

60(4):587–608, 2015.



Q. Huangfu and J. A. J. Hall.

Parallelizing the dual revised simplex method.

Mathematical Programming Computation, 10(1):119–142,

2018.



L. Schork and J. Gondzio.

Implementation of an interior point method with basis preconditioning.

Technical Report ERGO-18-014, School of Mathematics, University of Edinburgh, 2018.