HiGHS: a high-performance linear optimizer

Turning gradware into software

Julian Hall

School of Mathematics, University of Edinburgh

28th Biennial Conference on Numerical Analysis

University of Strathclyde, Glasgow

27 June 2019

THE UNIVERSITY *of* EDINBURGH

HiGHS

HiGHS: High performance linear optimization

- Linear optimization
  - Linear programming (LP)

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

  - Convex quadratic programming (QP)

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

    Q positive semi-definite

- High performance
  - Serial techniques exploiting sparsity in A
  - Parallel techniques exploiting multicore architectures

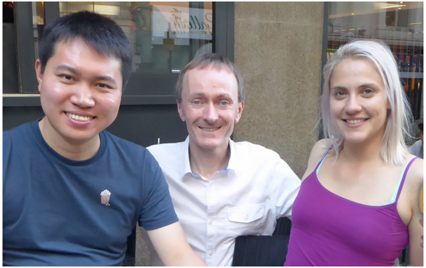
HiGHS: The team

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 (`hsol`)
- 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 $\widehat{\mathbf{c}}_N \geq \mathbf{0}$ Seek $\widehat{\mathbf{b}} \geq \mathbf{0}$

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

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

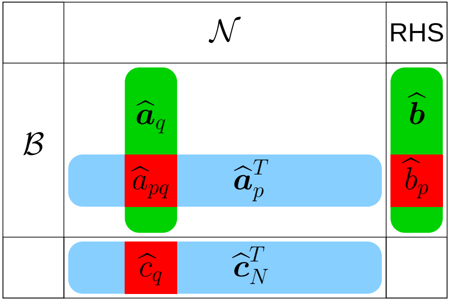
Update: Exchange p and q between $\mathcal{B}$ and $\mathcal{N}$

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

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

Data required

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

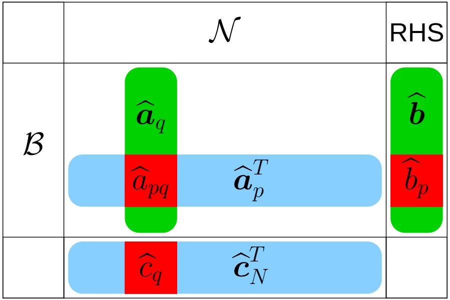


HiGHS: Dual simplex algorithm

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

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

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



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

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

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

Computation

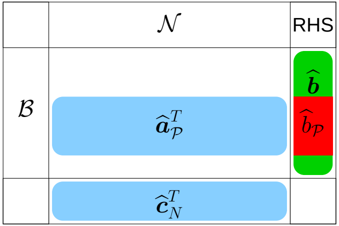
Pivotal row via $B^T \boldsymbol{\pi}_p = \mathbf{e}_p$ BTRAN and $\widehat{\mathbf{a}}_p^T = \boldsymbol{\pi}_p^T N$ PRICE

Pivotal column via $B \widehat{\mathbf{a}}_q = \mathbf{a}_q$ 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*



- Task-parallel multiple BTRAN to form $\boldsymbol{\pi}_\mathcal{P} = B^{-T}\mathbf{e}_\mathcal{P}$
- Data-parallel PRICE to form $\widehat{\mathbf{a}}_p^T$ (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 Mittelmann

Exclude 7 which are "hard"

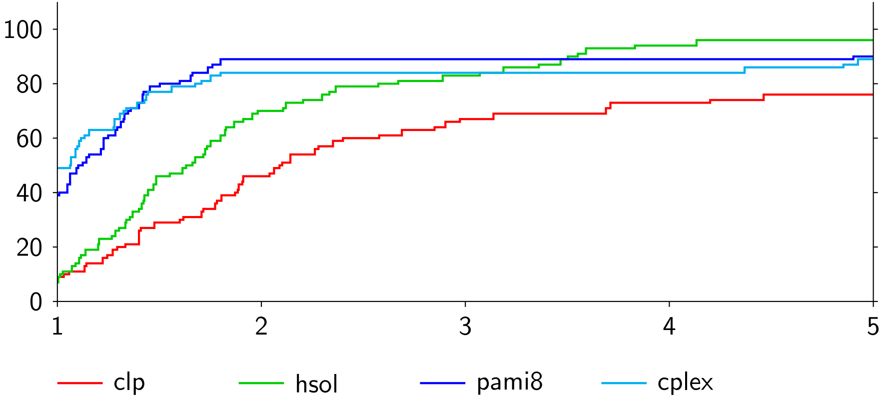
Performance

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

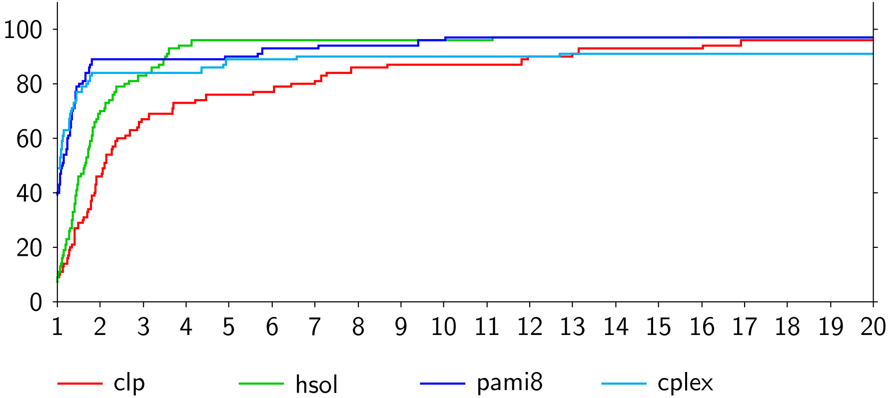
- Dual simplex
- No presolve
- No crash

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

HiGHS: Performance



HiGHS: Reliability



HiGHS: Present (2016–date)

Developments

- Model management: Load/add/delete/modify problem data

  Feldmeier, Galabova, H
- Interfaces

  Feldmeier, Galabova, Vigerske
- Presolve

  Galabova
- Crash

  H and Galabova
- Interior point method

  Schork

HiGHS: Access

Source

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

Interfaces

- Existing
  - C++ HiGHS class
  - Load from .mps
  - Load from .lp
  - C
  - C#
  - Julia
  - FORTRAN
  - OSI (almost!)
- Prototypes
  - GAMS
  - SCIP
- Planned
  - 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

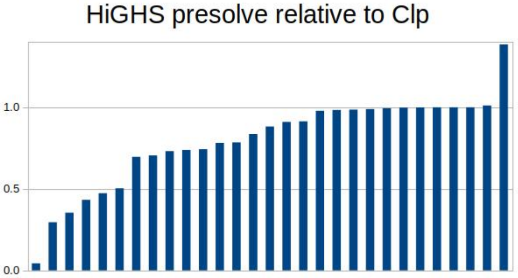
- Compete with other solvers in "vanilla" state

HiGHS: Presolve

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

Presolve measure relative to `Clp`

- Better than `Clp` 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!

HiGHS: Crash

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 (4 June 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	Matlab	Glop	Soplex	QSopt	Glpk	Lpsolve
Time	1	1.3	3.1	5.9	6.1	8.5	22.2	24.0	92.2

Where's HiGHS?

HiGHS: Benchmarks (17 Mar 2019)

Solver	Clp	Mosek	SAS	HiGHS	Glop	Matlab	Soplex	Glpk	Lpsolve
Time	1	2.8	3.2	5.3	6.4	6.6	10.1	26	112

Why is the HiGHS score so bad?

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

HiGHS: Selective results

Test set	Clp	HiGHS
Mittelmann (17 March 2019)	1	5.3
All 40 LPs (23 April 2019)	1	3.1
All 40 LPs (23 June 2019)	1	4.0
Less 14 LPs where Idiot crash aids Clp significantly	1	3.6
Less 8 LPs where Clp uses primal simplex	1	3.1
Remaining 14 LPs that HiGHS can solve	1	1.5

What's still to come with HiGHS?

- pami
- Triangular crash
- Study 29 new test problems
- Improve presolve

HiGHS: The future

- LP
  - Add Idiot crash (Galabova)
  - Add crossover (Hall)
  - Add primal simplex solver (Huangfu)
  - Improved Idiot crash (Galabova)
  - Direct solver for IPM (?)
- QP
  - Active set QP solver (Feldmeier)
  - IPM QP solver
- Interfaces
  - AMPL
  - MATLAB
  - Mosel
  - PuLp
  - Python
  - 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/EURO19

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.