HiGHS: A High-Performance Linear Optimizer

Julian Hall Qi Huangfu Ivet Galabova Lukas Schork Michael Feldmeier

School of Mathematics, University of Edinburgh

60th OR Society Conference

12 September 2018

HiGHS: High performance linear optimization

- Linear optimization
  - Linear programming (LP)

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

  - Convex quadratic programming (QP)

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

    Q positive semi-definite
- High performance
  - Serial techniques exploiting sparsity in A
  - Parallel techniques exploiting multicore architectures

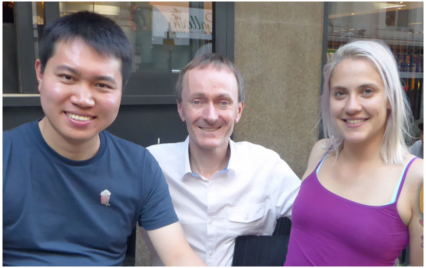
HiGHS: The team

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)
  - Google (2018)
- Qi Huangfu
  - PhD (2009-2013)
  - FICO Xpress (2013-2018)
  - MSc (2018–date)
- Lukas Schork: PhD (2015–2018)
- Michael Feldmeier: PhD (2018–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 $\widehat{\boldsymbol{c}}_N \geq \mathbf{0}$ Seek $\widehat{\boldsymbol{b}} \geq \mathbf{0}$

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

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

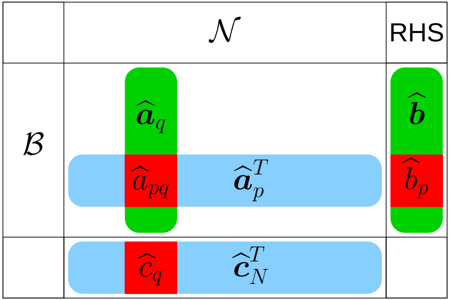
Update: Exchange p and q between $\mathcal{B}$ and $\mathcal{N}$

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

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

Data required

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

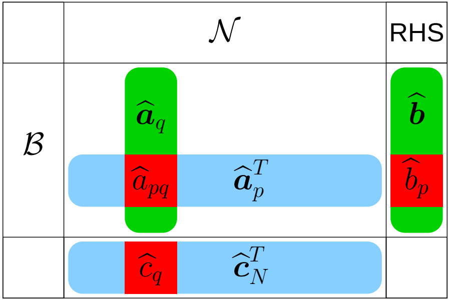


HiGHS: Dual simplex algorithm

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

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

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

Computation

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

Pivotal column via $B\,\widehat{\boldsymbol{a}}_q = \boldsymbol{a}_q$ FTRAN   Represent B^{-1} INVERT

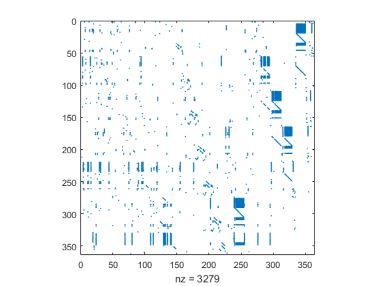
Update B^{-1} exploiting $\bar{B} = B + (\boldsymbol{a}_q - B\boldsymbol{e}_p)\boldsymbol{e}_p^T$ UPDATE

HiGHS: Solve $Bx = \boldsymbol{r}$ for sparse $\boldsymbol{r}$

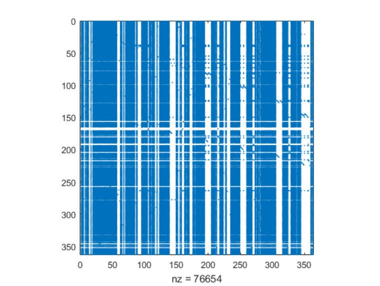
- Given $B = LU$, solve
$$L\boldsymbol{y} = \boldsymbol{r}; \quad U\boldsymbol{x} = \boldsymbol{y}$$
- In revised simplex method, $\boldsymbol{r}$ is sparse: consequences?
  - If B is irreducible then $\boldsymbol{x}$ is full
  - If B is highly reducible then $\boldsymbol{x}$ can be **sparse**
- Phenomenon of **hyper-sparsity**
  - Exploit it when *forming* $\boldsymbol{x}$
  - Exploit it when *using* $\boldsymbol{x}$

Inverse of a sparse matrix and solution of $B\boldsymbol{x} = \boldsymbol{r}$

Optimal B for LP problem `stair`

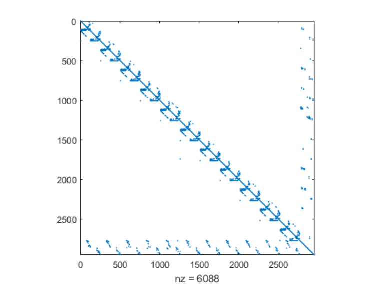


B^{-1} has density of 58%, so $B^{-1}\boldsymbol{r}$ is typically dense

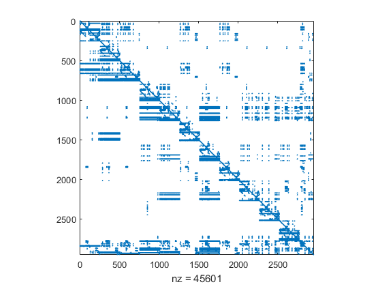


Inverse of a sparse matrix and solution of $B\mathbf{x} = \mathbf{r}$

Optimal B for LP problem pds-02



B^{-1} has density of 0.52%, so $B^{-1}\mathbf{r}$ is typically sparse—when $\mathbf{r}$ is sparse



HiGHS: Hyper-sparsity

- Use solution of $L\boldsymbol{x} = \boldsymbol{b}$
  - To illustrate the phenomenon of hyper-sparsity
  - To demonstrate how to exploit hyper-sparsity
- Apply principles to other triangular solves in the simplex method

Recall: Solve $L\boldsymbol{x} = \boldsymbol{b}$ using

function `ftranL`(L, $\boldsymbol{b}$, $\boldsymbol{x}$)
  $\boldsymbol{r} = \boldsymbol{b}$
  for all $j \in \{1, \ldots, m\}$ **do**
    for all $i : L_{ij} \neq 0$ **do**
      $r_i = r_i - L_{ij} r_j$
  $\boldsymbol{x} = \boldsymbol{r}$

When $\boldsymbol{b}$ is **sparse**

- Inefficient until $\boldsymbol{r}$ fills in

HiGHS: Hyper-sparsity

Better: Check r_j for zero

function `ftranL`(L, $\boldsymbol{b}$, $\boldsymbol{x}$)
- $\boldsymbol{r} = \boldsymbol{b}$
- **for all** $j \in \{1, \ldots, m\}$ **do**
  - **if** $r_j \neq 0$ **then**
    - **for all** $i : L_{ij} \neq 0$ **do**
      - $r_i = r_i - L_{ij} r_j$
- $\boldsymbol{x} = \boldsymbol{r}$

When $\boldsymbol{x}$ is **sparse**

- Few values of r_j are nonzero
- Check for zero dominates
- Requires more efficient identification of set $\mathcal{X}$ of indices j such that $r_j \neq 0$

Recall: major computational components

- FTRAN: Form $\widehat{\boldsymbol{a}}_q = B^{-1}\boldsymbol{a}_q$
- BTRAN: Form $\boldsymbol{\pi}_p = B^{-T}\boldsymbol{e}_p$
- PRICE: Form $\widehat{\boldsymbol{a}}_p^T = \boldsymbol{\pi}_p^T N$

BTRAN: Form $\boldsymbol{\pi}_p = B^{-T}\boldsymbol{e}_p$

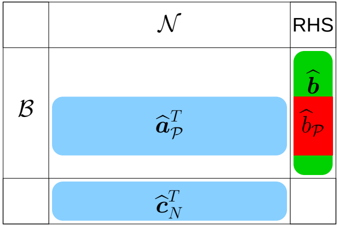
- Transposed triangular solves
- $L^T\boldsymbol{x} = \boldsymbol{b}$ has $x_i = b_i - \boldsymbol{l}_i^T\boldsymbol{x}$
  - Hyper-sparsity: $\boldsymbol{l}_i^T\boldsymbol{x}$ typically zero
  - Also store L (and U) row-wise and use FTRAN code

PRICE: Form $\widehat{\boldsymbol{a}}_p^T = \boldsymbol{\pi}_p^T N$

- Hyper-sparsity: $\boldsymbol{\pi}_p^T$ is sparse
- Store N row-wise
- Form $\widehat{\boldsymbol{a}}_p^T$ as a combination of rows of N for nonzeros in $\boldsymbol{\pi}_p^T$

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

Huangfu and H (2011–2014)

COAP best paper prize (2015)

HiGHS: Performance and reliability

Extended testing using 159 test problems

- 98 Netlib
- 16 Kennington
- 4 Industrial
- 41 Mittelmann

Exclude 7 which are "hard"

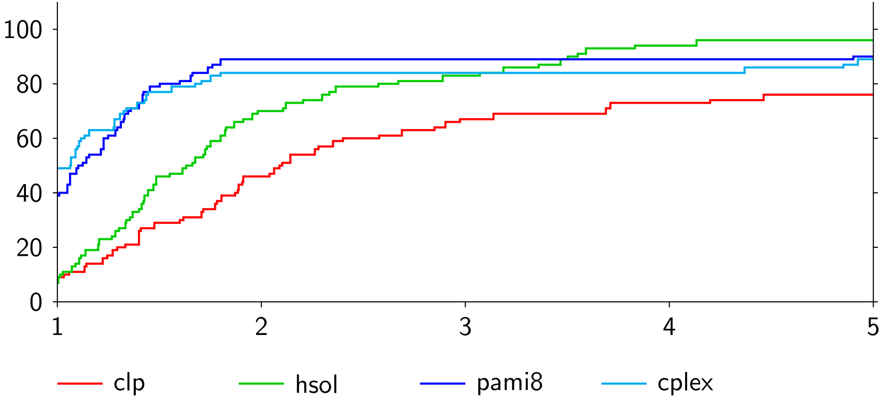
Performance

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

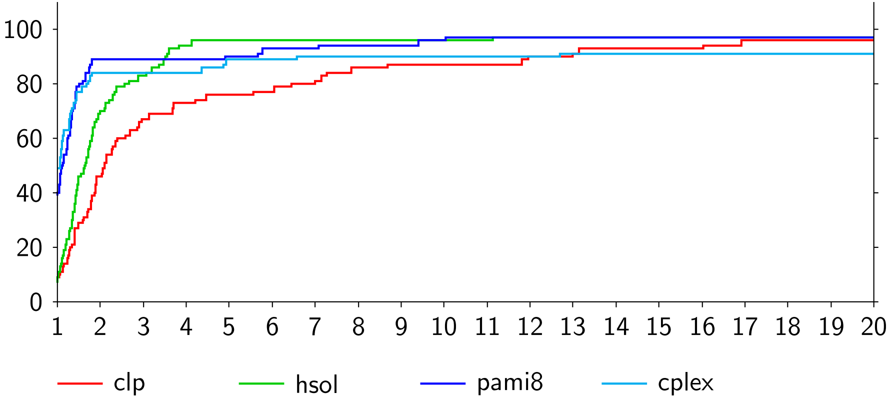
- Dual simplex
- No presolve
- No crash

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

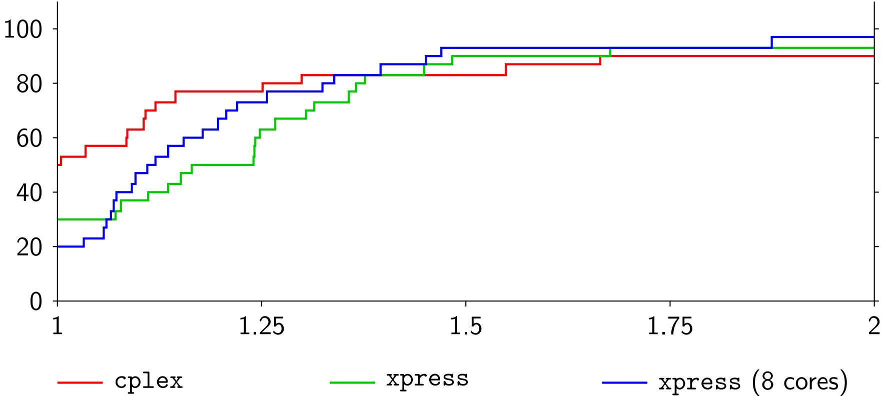
HiGHS: Performance



HiGHS: Reliability



HiGHS: Impact



- pami ideas incorporated in FICO Xpress (Huangfu 2014)
- Xpress has been the fastest simplex solver for most of the past five years

HiGHS: Present (2016–2018)

Features

- Model management: Add/delete/modify problem data
- SCIP interface

Presolve

- Presolve (and corresponding postsolve) has been implemented efficiently

  Remove redundancies in the LP to reduce problem dimension

Galabova

Crash

- Dual simplex "triangular basis" crash
- Alternative crash techniques being studied

H and Galabova

Interior point method

- Reliable "Matrix-free" implementation: Solve normal equations iteratively

Schork

HiGHS: From C++

- `HModel`: the model
  - `load_fromMPS(const char *filename)`: Load from an MPS file
  - `scaleModel()`: Scale the model
  - `clearModel()`: Clear the model
- `HDual`: the dual simplex solver
  - `solve(HModel *model)`: Solve the model
- Many other utilities
  - Modify model
  - Extract optimal primal and dual solution values
  - Extract basis
- Class corresponding to IPM solver

HiGHS: Interfaces

- Existing
  - `HModel::load_fromArrays(Many parameters!);`
    Pass model and return solution and basis by reference
  - SCIP (almost!)
- Planned
  - AMPL
  - MATLAB
  - R
  - Python
  - FORTRAN
  - Load from `.lp` file

HiGHS: The future is quadratic!

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

In particular

- $Q = A^T A$ and no equations

  Idiot crash for fast approximate solution of (some) LP problems (Galabova and H)
- $Q = I$

  Regularization terms in novel techniques for fast approximate solution of LP problems
- Q as Hessian of the Lagrangian

  SQP methods for NLP

Solve efficiently (direct and/or iterative methods) for objective

$$f(\boldsymbol{x}) = \frac{\rho}{2}\boldsymbol{x}^T Q \boldsymbol{x} + \frac{\mu}{2}||A\boldsymbol{x} - \boldsymbol{b}||_2^2 + \frac{\nu}{2}||\boldsymbol{x} - \boldsymbol{x}_0||_2^2 + \boldsymbol{c}^T \boldsymbol{x}$$

Feldmeier (2018–date)

HiGHS: Availability

Open source for academic use

- License: MIT
- Code: https://github.com/ERGO-Code/HiGHS
- Replacement for `clp`?

Commercial use

- Inspection copy (£1000)
- Support (POA)

To close

HiGHS

- Reliable high performance LP solver: simplex and interior point
- Integration with SCIP
- QP solver coming: for novel LP techniques and NLP

Slides:

http://www.maths.ed.ac.uk/hall/OR18

Code:

https://github.com/ERGO-Code/HiGHS

I. L. Galabova and J. A. J. Hall.
A quadratic penalty algorithm for linear programming and its application to linearizations of quadratic assignment problems.
Technical Report ERGO-18-009, School of Mathematics, University of Edinburgh, 2018.

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.