

An open-source high performance dual simplex solver

Julian Hall¹ Qi Huangfu² Ivet Galabova¹

¹School of Mathematics, University of Edinburgh

²FICO

SIAM Conference on Optimization

22 May 2017



THE UNIVERSITY
of EDINBURGH



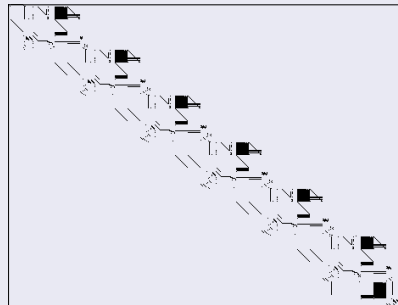
LP and high performance simplex solvers

$$\begin{array}{ll}\text{minimize} & f = \mathbf{c}^T \mathbf{x} \\ \text{subject to} & \mathbf{Ax} = \mathbf{b} \quad \mathbf{x} \geq \mathbf{0}\end{array}$$

Background

- Fundamental model in optimal decision-making
- Simplex method preferred when solving related problems
- High performance requires
 - Algorithmic tricks
 - Computational tricks

Example



STAIR: 356 rows, 467 columns and 3856 nonzeros

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 $\hat{c}_j/\hat{a}_{pj}$ 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
$\mathcal{B}$	$\hat{\mathbf{a}}_q$		$\hat{\mathbf{b}}$
	$\hat{a}_{pq}$	$\hat{\mathbf{a}}_p^T$	$\hat{b}_p$
	$\hat{c}_q$	$\hat{\mathbf{c}}_N^T$	

Data required

- Pivotal row $\hat{\mathbf{a}}_p^T = \mathbf{e}_p^T B^{-1} N$ via $B^T \pi_p = \mathbf{e}_p$ (BTRAN); $\hat{\mathbf{a}}_p^T = \pi_p^T N$ (PRICE)
- Pivotal column $\hat{\mathbf{a}}_q = B^{-1} \mathbf{a}_q$ via $B \hat{\mathbf{a}}_q = \mathbf{a}_q$ (FTRAN)

hsol: A high performance simplex solver

- Features
 - Model management: Add/delete/modify problem data
 - Functionality: Presolve + crash + advanced basis start
 - Algorithm
 - Dual simplex
 - Steepest edge pricing
 - Bound-flipping ratio test
 - Forrest-Tomlin update for $B := B + (\mathbf{a}_q - B\mathbf{e}_p)\mathbf{e}_p^T$
 - Efficiency: High performance serial and parallel computational components
- Fast
- Reliable
- Open-source C++

Presolve

- Presolve (and corresponding postsolve) has been implemented efficiently
Remove redundancies in the LP to reduce problem dimension Galabova (2016)

Crash

- Dual simplex “triangular basis” crash
Can be valuable, even with dual steepest edge weights
- Alternative crash techniques being studied H and Galabova (2016–date)

hsol: Exploiting hyper-sparsity

To solve $B\mathbf{x} = \mathbf{r}$

- Using decomposition of B given by $\{p_k, \mu_k, \boldsymbol{\eta}_k\}_{k=1}^K$
- Traditional technique transforms RHS into solution $\mathbf{x}$

do $k = 1, K$

$$r_{p_k} := r_{p_k} / \mu_k$$

$$\mathbf{r} := \mathbf{r} - r_{p_k} \boldsymbol{\eta}_k$$

end do

- When initial RHS is sparse: inefficient until $\mathbf{r}$ fills in

hsol: Exploiting hyper-sparsity

To solve $B\mathbf{x} = \mathbf{r}$

- Using decomposition of B given by $\{p_k, \mu_k, \boldsymbol{\eta}_k\}_{k=1}^K$
- When $\mathbf{r}$ is sparse skip $\boldsymbol{\eta}_k$ if r_{p_k} is zero

```
do  $k = 1, K$   
  if ( $r_{p_k} \neq 0$ ) then  
     $r_{p_k} := r_{p_k} / \mu_k$   
     $\mathbf{r} := \mathbf{r} - r_{p_k} \boldsymbol{\eta}_k$   
  end if  
end do
```

- When $\mathbf{x}$ is sparse, the dominant cost is the test for zero
- Requires efficient identification of vectors $\boldsymbol{\eta}_k$ to be applied

Gilbert and Peierls (1988)
H and McKinnon (1998–2005)
COAP best paper prize (2005)

hsol: 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 $\boldsymbol{\pi}_{\mathcal{P}} = B^{-1} \mathbf{e}_{\mathcal{P}}$
- Data-parallel PRICE to form $\hat{\mathbf{a}}_{\mathcal{P}}^T$ (as required)
- Task-parallel multiple FTRAN for primal, dual and weight updates

Huangfu and H (2011–2014)
COAP best paper prize (2015)

hsol: Performance and reliability

Extended testing using 159 test problems

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

Exclude 7 which are “hard”

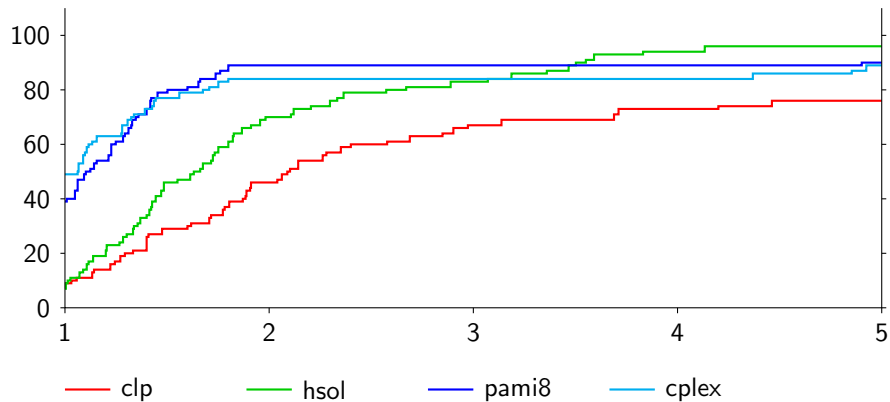
Performance

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

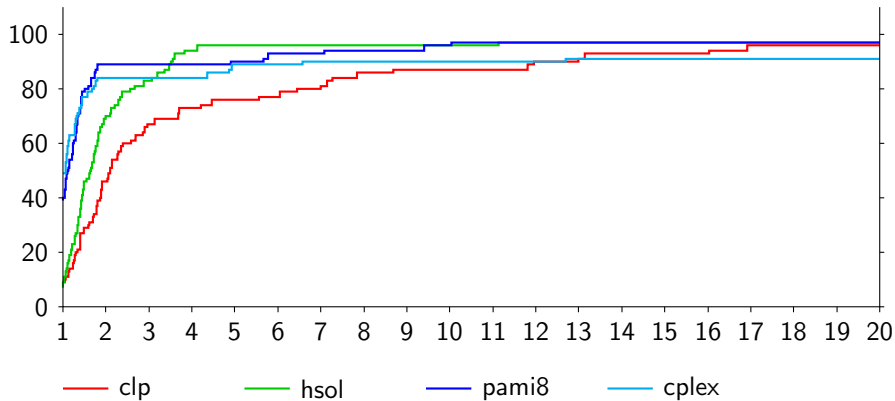
- Dual simplex
- No presolve
- No crash

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

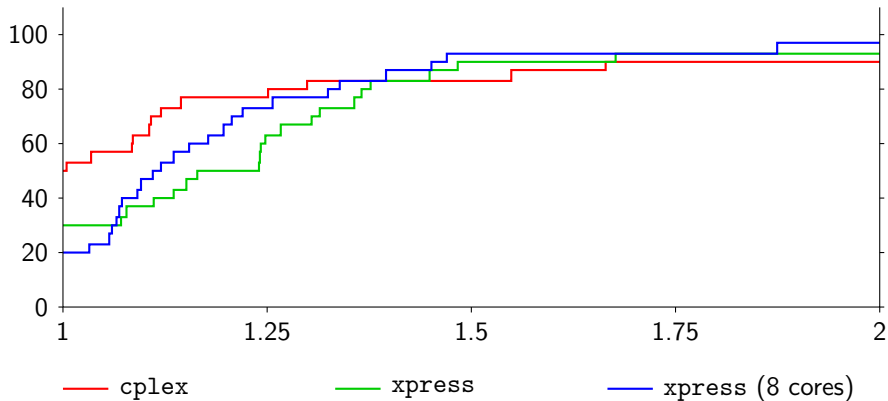
hsol: Performance



hsol: Reliability



hsol: Impact



- pami ideas incorporated in [FICO Xpress](#) (Huangfu 2014)
- Xpress simplex solver now fastest commercial simplex solver

For 2017

- SCIP interface
- QP solver
- MIP solver

For 2018

- MIP presolve
- MIQP solver

Long term

Replacement for c1p?

Commercial involvement

- Cargill (feed formulation)
- Google (techniques in glqp)
- Financial services

Summary

- `hsol` is a fast and reliable simplex solver
 - Full functionality for practical use
 - Built on award-winning computational techniques
 - Parallel computing capability (pami)
 - open-source C++
- Full suite of mixed-integer linear and quadratic solvers under development



J. A. J. Hall and K. I. M. McKinnon.

Hyper-sparsity in the revised simplex method and how to exploit it.
Computational Optimization and Applications, 32(3):259–283, December 2005.



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

Parallelizing the dual revised simplex method.
Technical Report ERGO-14-011, School of Mathematics, University of Edinburgh, 2014.
Accepted for publication in *Mathematical Programming Computation*.



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

Novel update techniques for the revised simplex method.
Computational Optimization and Applications, 60(4):587–608, 2015.

Slides: <http://www.maths.ed.ac.uk/hall/SIOPT17>