

Parallelising the dual revised simplex method

Julian Hall¹ Qi Huangfu² Miles Lubin³

¹School of Mathematics, University of Edinburgh

²FICO

³MIT

Convex Optimization and Beyond

Edinburgh

27 June 2014

Parallelising the dual revised simplex method: Overview

- Background
- Three approaches
 - Multiple iteration parallelism for general LP
 - Single iteration parallelism for general LP
 - Data parallelism for stochastic LP
- Conclusions

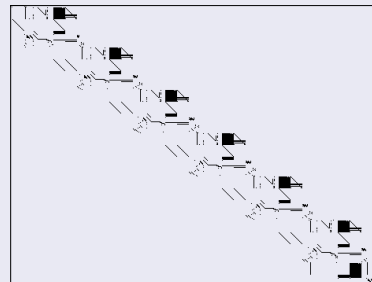
Linear programming (LP)

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

Background

- Fundamental model in optimal decision-making
- Solution techniques
 - Simplex method (1947)
 - Interior point methods (1984)
- Large problems have
 - 10^3 – 10^7 variables
 - 10^3 – 10^7 constraints
- Matrix A is (usually) sparse

Example



STAIR: 356 rows, 467 columns and 3856 nonzeros

Solving LP problems

$$\begin{array}{ll} \text{minimize} & f_P = \mathbf{c}^T \mathbf{x} \\ \text{subject to} & \mathbf{Ax} = \mathbf{b} \quad \mathbf{x} \geq \mathbf{0} \quad (P) \end{array} \quad \begin{array}{ll} \text{maximize} & f_D = \mathbf{b}^T \mathbf{y} \\ \text{subject to} & \mathbf{A}^T \mathbf{y} + \mathbf{s} = \mathbf{c} \quad \mathbf{s} \geq \mathbf{0} \quad (D) \end{array}$$

Optimality conditions

- For a partition $\mathcal{B} \cup \mathcal{N}$ of the variable set with nonsingular **basis matrix** B in

$$\mathbf{Bx}_B + \mathbf{Nx}_N = \mathbf{b} \text{ for (P)} \quad \text{and} \quad \begin{bmatrix} B^T \\ N^T \end{bmatrix} \mathbf{y} + \begin{bmatrix} \mathbf{s}_B \\ \mathbf{s}_N \end{bmatrix} = \begin{bmatrix} \mathbf{c}_B \\ \mathbf{c}_N \end{bmatrix} \text{ for (D)}$$

with $\mathbf{x}_N = \mathbf{0}$ and $\mathbf{s}_B = \mathbf{0}$

- **Primal** basic variables $\mathbf{x}_B$ given by $\hat{\mathbf{b}} = B^{-1}\mathbf{b}$
- **Dual** non-basic variables $\mathbf{s}_N$ given by $\hat{\mathbf{c}}_N^T = \mathbf{c}_N^T - \mathbf{c}_B^T B^{-1}N$
- Partition is optimal if there is
 - **Primal feasibility** $\hat{\mathbf{b}} \geq \mathbf{0}$
 - **Dual feasibility** $\hat{\mathbf{c}}_N \geq \mathbf{0}$

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

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

Scan $\hat{b}_i, i \in \mathcal{B}$, for a good candidate p to leave $\mathcal{B}$ **CHUZR**

Scan $\hat{c}_j/\hat{a}_{pj}, j \in \mathcal{N}$, for a good candidate q to leave $\mathcal{N}$ **CHUZY**

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

Update $\hat{\mathbf{b}} := \hat{\mathbf{b}} - \theta_p \hat{\mathbf{a}}_q$ $\theta_p = \hat{b}_p/\hat{a}_{pq}$ **UPDATE-PRIMAL**

Update $\hat{\mathbf{c}}_N^T := \hat{\mathbf{c}}_N^T - \theta_d \hat{\mathbf{a}}_p^T$ $\theta_d = \hat{c}_q/\hat{a}_{pq}$ **UPDATE-DUAL**

Major computational component

Update of tableau:

$$\hat{N} := \hat{N} - \frac{1}{\hat{a}_{pq}} \hat{\mathbf{a}}_q \hat{\mathbf{a}}_p^T$$

where $\hat{N} = B^{-1}N$

- Hopelessly inefficient for sparse LP problems
- Prohibitively expensive for large LP problems

Major computational components

$$\pi_p^T = \mathbf{e}_p^T B^{-1} \quad \text{BTRAN} \quad \hat{\mathbf{a}}_p^T = \pi_p^T N \quad \text{PRICE}$$

$$\hat{\mathbf{a}}_q = B^{-1} \mathbf{a}_q \quad \text{FTRAN} \quad \text{Invert } B \quad \text{INVERT}$$

Hyper-sparsity

- Vectors $\mathbf{e}_p$ and $\mathbf{a}_q$ are **always sparse**
 - B may be **highly reducible**; B^{-1} may be sparse
 - Vectors π_p , $\hat{\mathbf{a}}_p^T$ and $\hat{\mathbf{a}}_q$ **may be sparse**
 - Efficient implementations must exploit these features
- H and McKinnon (1998–2005), Bixby (1999)
Clp, Koberstein and Suhl (2005–2008)

Row selection: Dual steepest edge (DSE)

- Weight $\hat{b}_i$ by w_i : measure of $\|B^{-1} \mathbf{e}_i\|_2$
- Requires additional FTRAN but can reduce iteration count significantly

Column selection: Bound-flipping ratio test (BFRT)

- Minimizes the dual objective whilst remaining dual feasible
 - Dual variables may change sign if corresponding primal variables can flip bounds
- Requires additional FTRAN but can reduce iteration count significantly

Exploiting parallelism: Background

Data parallel standard simplex method

- Good parallel efficiency *was* achieved
- Only relevant for dense LP problems

Data parallel revised simplex method

- Only immediate parallelism is in forming $\pi_p^T N$
 - When $n \gg m$ significant speed-up *was* achieved
- Bixby and Martin (2000)

Task parallel revised simplex method

- Overlap computational components for different iterations
Wunderling (1996), H and McKinnon (1995-2005)
- Modest speed-up *was* achieved on general sparse LP problems

Parallelising the dual revised simplex method: Overview

Parallelising the dual revised simplex method: Overview

Single iteration parallelism for general LP

- Pure dual revised simplex
- **Data parallelism:** Form $\pi_p^T N$
- **Task parallelism:** Identify serial computation which can be overlapped

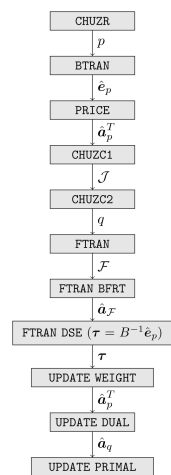
Multiple iteration parallelism for general LP

- Dual revised simplex with minor iterations of dual standard simplex
- **Data parallelism:** Form $\pi_p^T N$ and update (slice of) dual standard simplex tableau
- **Task parallelism:** Identify serial computation which can be overlapped

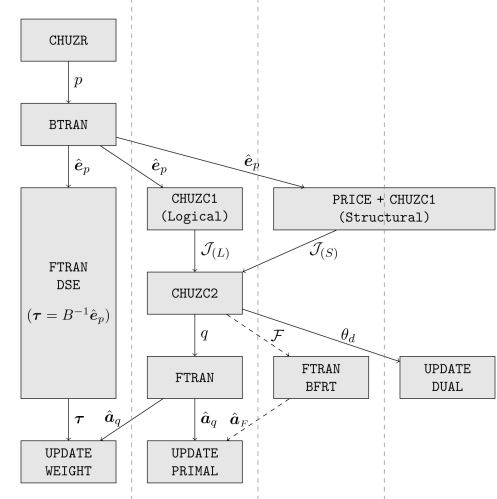
Data parallelism for stochastic LP

- Pure dual revised simplex for column-linked block angular LP problems
- **Data parallelism:** Solve $B^T \pi = e_p$, $B \hat{a}_q = a_q$ and form $\pi_p^T N$

Single iteration parallelism



- Computational components appear sequential
- Each has highly-tuned sparsity-exploiting serial implementation
- Exploit “slack” in data dependencies



- Parallel PRICE to form $\hat{\mathbf{a}}_p^T = \pi_p^T N$
- Other computational components serial
- Overlap any independent calculations
- Only four worthwhile threads unless $n \gg m$ so PRICE dominates
- More than Bixby and Martin (2000)
- Better than Forrest (2012)

Single iteration parallelism: 4-core sip vs 1-core hsol

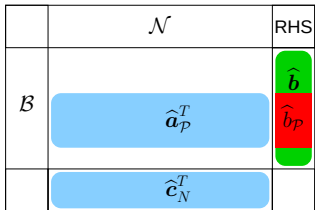
MODEL	Speedup	MODEL	Speedup	MODEL	Speedup
SGPF5Y6	0.67	MAROS-R7	1.12	WORLD	1.27
STORMG2-125	0.76	STP3D	1.15	DFL001	1.28
WATSON_2	0.78	NUG12	1.16	L30	1.28
KEN-18	0.79	PDS-40	1.16	LINF_520C	1.31
WATSON_1	0.80	DBIC1	1.21	PILOT87	1.31
QAP12	0.83	FOME12	1.22	SELF	1.36
STORMG2-1000	0.84	DCP2	1.23	LP22	1.45
PDS-80	1.05	NS1688926	1.23	DANO3MIP_LP	1.49
PDS-100	1.06	FOME13	1.24	TRUSS	1.58
CRE-B	1.08	MOD2	1.25	STAT96V4	2.05

- Geometric mean speedup is 1.13
- Performance is generally poor for problems with high hyper-sparsity
- Performance is generally good for problems with low hyper-sparsity

Multiple iteration parallelism

Multiple iteration parallelism

- sip has too little work to be performed in parallel to get good speedup
- 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 $\pi_{\mathcal{P}} = B^{-1}\mathbf{e}_{\mathcal{P}}$
- Data-parallel PRICE to form $\hat{\mathbf{a}}_{\mathcal{P}}^T$ (as required)
- Data-parallel tableau update
- Task-parallel multiple FTRAN for primal, dual and weight updates

Multiple iteration parallelism: 8-core pami vs 1-core pami

MODEL	Speedup	MODEL	Speedup	MODEL	Speedup
KEN-18	1.54	LINF_520C	2.00	WORLD	2.54
MAROS-R7	1.56	PDS-40	2.00	FOME12	2.58
CRE-B	1.62	NS1688926	2.10	TRUSS	2.67
STORMG2-125	1.70	FOME13	2.20	L30	2.74
WATSON_2	1.72	STORMG2-1000	2.25	DFL001	2.74
SELF	1.81	STP3D	2.33	LP22	2.75
WATSON_1	1.83	DBIC1	2.36	QAP12	2.75
PDS-100	1.88	SGPF5Y6	2.40	NUG12	2.81
DCP2	1.89	PILOT87	2.48	DANO3MIP_LP	3.10
PDS-80	1.92	MOD2	2.53	STAT96V4	3.50

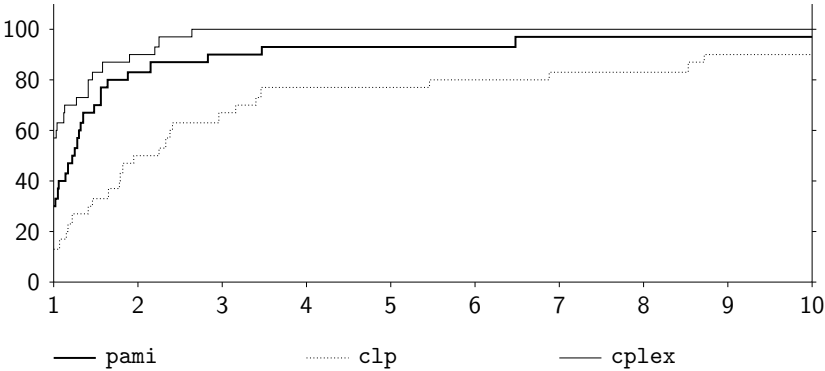
- Speed-up for all problems
- Geometric mean speedup is 2.23

Multiple iteration parallelism: 8-core pami vs 1-core hsol

MODEL	Speedup	MODEL	Speedup	MODEL	Speedup
MAROS-R7	0.47	PDS-40	1.35	LP22	1.67
LINF_520C	0.75	WORLD	1.37	NUG12	1.78
SELF	1.07	STORMG2-125	1.44	DFL001	1.81
PDS-80	1.16	PILOT87	1.50	SGPF5Y6	1.90
NS1688926	1.26	DCP2	1.52	TRUSS	1.94
PDS-100	1.29	FOME13	1.52	CRE-B	1.95
MOD2	1.29	WATSON_1	1.55	DANO3MIP_LP	2.12
L30	1.29	WATSON_2	1.61	STAT96V4	2.33
KEN-18	1.30	FOME12	1.61	STP3D	2.41
DBIC1	1.31	STORMG2-1000	1.66	QAP12	2.53

- Geometric mean speedup is 1.49
- Lower than speedup relative to 1-core pami
 - Geometric mean speed of 1-core pami relative to 1-core hsol is 0.67

Multiple iteration parallelism: Performance profile benchmarking



- pami is plainly better than clp
- pami is comparable with cplex
- pami ideas have been incorporated in [FICO Xpress](#) (Huangfu 2014)

Data parallelism for stochastic LPs

Stochastic MIP problems: General

Two-stage stochastic LPs have column-linked block angular structure

$$\begin{array}{llllllllll}
 \text{minimize} & \mathbf{c}_0^T \mathbf{x}_0 & + & \mathbf{c}_1^T \mathbf{x}_1 & + & \mathbf{c}_2^T \mathbf{x}_2 & + & \dots & + & \mathbf{c}_N^T \mathbf{x}_N & & \\
 \text{subject to} & A\mathbf{x}_0 & & & & & & & & & = & \mathbf{b}_0 \\
 & T_1\mathbf{x}_0 & + & W_1\mathbf{x}_1 & & & & & & & = & \mathbf{b}_1 \\
 & T_2\mathbf{x}_0 & & & + & W_2\mathbf{x}_2 & & & & & = & \mathbf{b}_2 \\
 & \vdots & & & & & & & & & \vdots \\
 & T_N\mathbf{x}_0 & & & & & & & + & W_N\mathbf{x}_N & = & \mathbf{b}_N \\
 & \mathbf{x}_0 \geq \mathbf{0} & & \mathbf{x}_1 \geq \mathbf{0} & & \mathbf{x}_2 \geq \mathbf{0} & & \dots & & \mathbf{x}_N \geq \mathbf{0} & &
 \end{array}$$

- Variables $\mathbf{x}_0 \in \mathbb{R}^{n_0}$ are **first stage** decisions
- Variables $\mathbf{x}_i \in \mathbb{R}^{n_i}$ for $i = 1, \dots, N$ are **second stage** decisions
Each corresponds to a **scenario** which occurs with modelled probability
- The objective is the expected cost of the decisions
- In stochastic MIP problems, some/all decisions are discrete

Stochastic MIP problems: For Argonne

- Power systems optimization project at Argonne
- Integer second-stage decisions
- Stochasticity comes from availability of wind-generated electricity
- Initial experiments carried out using model problem
- Number of scenarios increases with refinement of probability distribution sampling
- Solution via branch-and-bound
 - Solve root node using parallel IPM solver PIPS Lubin, Petra *et al.* (2011)
 - Solve subsequent nodes using parallel dual simplex solver PIPS-S Lubin, H *et al.* (2013)

Stochastic MIP problems: General

Convenient to permute the LP thus:

$$\begin{array}{llllllllll}
 \text{minimize} & \mathbf{c}_1^T \mathbf{x}_1 & + & \mathbf{c}_2^T \mathbf{x}_2 & + & \dots & + & \mathbf{c}_N^T \mathbf{x}_N & + & \mathbf{c}_0^T \mathbf{x}_0 & & \\
 \text{subject to} & W_1\mathbf{x}_1 & & & & & & & + & T_1\mathbf{x}_0 & = & \mathbf{b}_1 \\
 & & & W_2\mathbf{x}_2 & & & & & + & T_2\mathbf{x}_0 & = & \mathbf{b}_2 \\
 & & & & & & & & & \vdots & \vdots \\
 & & & & & & & & + & T_N\mathbf{x}_0 & = & \mathbf{b}_N \\
 & & & & & & & & & A\mathbf{x}_0 & = & \mathbf{b}_0 \\
 & \mathbf{x}_1 \geq \mathbf{0} & & \mathbf{x}_2 \geq \mathbf{0} & & \dots & & \mathbf{x}_N \geq \mathbf{0} & & \mathbf{x}_0 \geq \mathbf{0} & &
 \end{array}$$

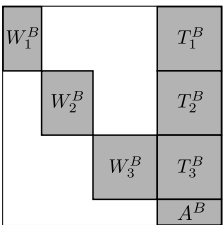
- Inversion of the basis matrix B is key to revised simplex efficiency
- For column-linked BALP problems

$$B = \begin{bmatrix} W_1^B & & T_1^B \\ & \ddots & \vdots \\ & & W_N^B & T_N^B \\ & & & A^B \end{bmatrix}$$

- W_i^B are columns corresponding to n_i^B basic variables in scenario i
- $\begin{bmatrix} T_1^B \\ \vdots \\ T_N^B \\ A^B \end{bmatrix}$ are columns corresponding to n_0^B basic first stage decisions

- Inversion of the basis matrix B is key to revised simplex efficiency
- For column-linked BALP problems

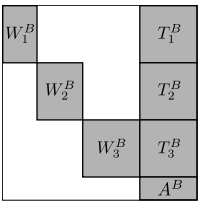
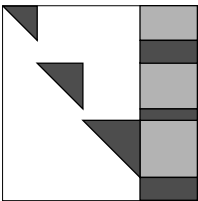
$$B = \begin{bmatrix} W_1^B & & T_1^B \\ & \ddots & \vdots \\ & & W_N^B & T_N^B \\ & & & A^B \end{bmatrix}$$



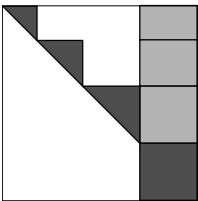
- B is nonsingular so
 - W_i^B are “tall”: full column rank
 - $[W_i^B \ T_i^B]$ are “wide”: full row rank
 - A^B is “wide”: full row rank
- Scope for parallel inversion is immediate and well known

Duff and Scott (2004)

- Eliminate sub-diagonal entries in each W_i^B (independently)



- Apply elimination operations to each T_i^B (independently)



- Accumulate non-pivoted rows from the W_i^B with A^B and complete elimination

- After Gaussian elimination, have invertible representation of

$$B = \left[\begin{array}{ccc|c} S_1 & & & C_1 \\ & \ddots & & \vdots \\ & & S_N & C_N \\ \hline R_1 & \dots & R_N & V \end{array} \right] = \left[\begin{array}{c|c} S & C \\ \hline R & V \end{array} \right]$$

- Specifically
 - $L_i U_i = S_i$ of dimension n_i^B
 - $\hat{C}_i = L_i^{-1} C_i$
 - $\hat{R}_i = R_i U_i^{-1}$
 - LU factors of the Schur complement $M = V - RS^{-1}C$ of dimension n_0^B
- Scope for parallelism since each GE applied to $[W_i^B \ T_i^B]$ is independent

FTRAN for $B\mathbf{x} = \mathbf{b}$

Solve $\begin{bmatrix} S & C \\ R & V \end{bmatrix} \begin{bmatrix} \mathbf{x}_\bullet \\ \mathbf{x}_0 \end{bmatrix} = \begin{bmatrix} \mathbf{b}_\bullet \\ \mathbf{b}_0 \end{bmatrix}$ as

- 1 $L_i \mathbf{y}_i = \mathbf{b}_i, i = 1, \dots, N$
- 2 $\mathbf{z}_i = \hat{R}_i \mathbf{y}_i, i = 1, \dots, N$
- 3 $\mathbf{z} = \mathbf{b}_0 - \sum_{i=1}^N \mathbf{z}_i$
- 4 $M \mathbf{x}_0 = \mathbf{z}$
- 5 $U_i \mathbf{x}_i = \mathbf{y}_i - \hat{C}_i \mathbf{x}_0, i = 1, \dots, N$

- Appears to be dominated by parallelizable
 - Solves $L_i \mathbf{y}_i = \mathbf{b}_i$ and $U_i \mathbf{x}_i = \mathbf{y}_i - \hat{C}_i \mathbf{x}_0$
 - Products $\hat{R}_i \mathbf{y}_i$ and $\hat{C}_i \mathbf{x}_0$
- Curse of exploiting hyper-sparsity
 - In simplex, $\mathbf{b}_\bullet$ is from constraint column
 - Either $\begin{bmatrix} \mathbf{t}_{1q} \\ \vdots \\ \mathbf{t}_{Nq} \end{bmatrix}$ or, more likely, $\begin{bmatrix} \mathbf{0} \\ \mathbf{w}_{iq} \\ \mathbf{0} \end{bmatrix}$
 - In latter case, the $\mathbf{y}_i$ inherit structure
 - Only one $L_i \mathbf{y}_i = \mathbf{w}_{iq}$
 - Only one $\hat{R}_i \mathbf{y}_i$
- Less scope for parallelism than anticipated

BTRAN for $B^T \mathbf{x} = \mathbf{b}$

Solve $\begin{bmatrix} S^T & R^T \\ C^T & V^T \end{bmatrix} \begin{bmatrix} \mathbf{x}_\bullet \\ \mathbf{x}_0 \end{bmatrix} = \begin{bmatrix} \mathbf{b}_\bullet \\ \mathbf{b}_0 \end{bmatrix}$ as

- 1 $U_i^T \mathbf{y}_i = \mathbf{b}_i, i = 1, \dots, N$
- 2 $\mathbf{z}_i = \hat{C}_i^T \mathbf{y}_i, i = 1, \dots, N$
- 3 $\mathbf{z} = \mathbf{b}_0 - \sum_{i=1}^N \mathbf{z}_i$
- 4 $M^T \mathbf{x}_0 = \mathbf{z}$
- 5 $L_i^T \mathbf{x}_i = \mathbf{y}_i - \hat{R}_i^T \mathbf{x}_0, i = 1, \dots, N$

- Appears to be dominated by parallelizable
 - Solves $U_i^T \mathbf{y}_i = \mathbf{b}_i$ and $L_i^T \mathbf{x}_i = \mathbf{y}_i - \hat{R}_i^T \mathbf{x}_0$
 - Products $\hat{C}_i^T \mathbf{y}_i$ and $\hat{R}_i^T \mathbf{x}_0$
- Curse of exploiting hyper-sparsity
 - In simplex, $\mathbf{b} = \mathbf{e}_p$
 - At most one solve $U_i^T \mathbf{y}_i = \mathbf{b}_i$
 - At most one $\hat{C}_i^T \mathbf{y}_i$
- Less scope for parallelism than anticipated

- PRICE forms

$$\begin{bmatrix} \pi_1^T & \pi_2^T & \dots & \pi_N^T & \pi_0^T \end{bmatrix} \begin{bmatrix} W_1^N & & & T_1^N \\ & W_2^N & & T_2^N \\ & & \ddots & \vdots \\ & & & W_N^N & T_N^N \\ & & & & A^N \end{bmatrix}$$
$$= \begin{bmatrix} \pi_1^T W_1^N & \pi_2^T W_2^N & \dots & \pi_N^T W_N^N & \pi_0^T A^N + \sum_{i=1}^N \pi_i^T T_i^N \end{bmatrix}$$

- Dominated by parallelizable products $\pi_i^T W_i^N$ and $\pi_i^T T_i^N$

- Update of the invertible representation of B is second major factor in revised simplex efficiency
- Each iteration column $\mathbf{a}_q$ of the constraint matrix replaces column $B\mathbf{e}_p$ of B

$$B' = B[I + (\hat{\mathbf{a}}_q - \mathbf{e}_p)\mathbf{e}_p^T]$$

- Unfortunately, the structure of B is not generally maintained
- PIPS-S uses standard **product form** update

$$B'^{-1} = [I + (\hat{\mathbf{a}}_q - \mathbf{e}_p)\mathbf{e}_p^T]^{-1} B^{-1} = E^{-1} B^{-1} \quad \text{where} \quad E^{-1} = I - \frac{1}{\hat{a}_{pq}} (\hat{\mathbf{a}}_q - \mathbf{e}_p)\mathbf{e}_p^T$$

- Form $\mathbf{x} = E^{-1}\mathbf{b}$ as $x_p = -\frac{b_p}{\hat{a}_{pq}}$ then $\mathbf{x}_{p'} = \mathbf{b}_{p'} + \hat{\mathbf{a}}_q x_p$
- Updates $\{E_k\}_{k=1}^K$ of B_0 to B_K require $\{\hat{\mathbf{a}}_{q_k}\}_{k=1}^K$ and $\mathcal{P} = \{p_k\}_{k=1}^K, |\mathcal{P}| \ll m$
- Exploit parallelism when forming $\mathbf{x} = E_K^{-1} \dots E_1^{-1} \mathbf{b}$ thus
 - Compute $\mathbf{x}_{\mathcal{P}}$ serially
 - Compute $\mathbf{x}_{\mathcal{P}'}$ as a parallel matrix-vector product

$$\mathbf{x}_{\mathcal{P}'} = \mathbf{b}_{\mathcal{P}'} + \begin{bmatrix} \hat{\mathbf{a}}_{q_1} & \dots & \hat{\mathbf{a}}_{q_K} \end{bmatrix} \mathbf{x}_{\mathcal{P}}$$

- Similar trick for parallelising $\mathbf{x}^T = \mathbf{b}^T E_K^{-1} \dots E_1^{-1}$

Lubin, H et al. (2013)

Results

Results: Stochastic LP test problems

Test Problem	1st Stage		2nd-Stage Scenario		Nonzero Elements		
	n_0	m_0	n_i	m_i	A	W_i	T_i
Storm	121	185	1,259	528	696	3,220	121
SSN	89	1	706	175	89	2,284	89
UC12	3,132	0	56,532	59,436	0	163,839	3,132
UC24	6,264	0	113,064	118,872	0	327,939	6,264

- Storm and SSN are publicly available
- UC12 and UC24 are stochastic unit commitment problems developed at Argonne
 - Aim to choose optimal on/off schedules for generators on the power grid of the state of Illinois over a 12-hour and 24-hour horizon
 - In practice each scenario corresponds to a weather simulation
Model problem generates scenarios by normal perturbations

Zavala (2011)

Results: Baseline serial performance for large instances

Serial performance of PIPS-S and c1p					
Problem	Dimensions	Solver	Iterations	Time (s)	Iter/sec
Storm	$n = 10,313,849$	PIPS-S	6,353,593	385,825	16.5
8,192 scen.	$m = 4,325,561$	c1p	6,706,401	133,047	50.4
SSN	$n = 5,783,651$	PIPS-S	1,025,279	58,425	17.5
8,192 scen.	$m = 1,433,601$	c1p	1,175,282	12,619	93.1
UC12	$n = 1,812,156$	PIPS-S	1,968,400	236,219	8.3
32 scen.	$m = 1,901,952$	c1p	2,474,175	39,722	62.3
UC24	$n = 1,815,288$	PIPS-S	2,142,962	543,272	3.9
16 scen.	$m = 1,901,952$	c1p	2,441,374	41,708	58.5

Speed-up of PIPS-S relative to 1-core PIPS-S and 1-core clp

Cores	Storm	SSN	UC12	UC24
1	1.0	1.0	1.0	1.0
4	3.6	3.5	2.7	3.0
8	7.3	7.5	6.1	5.3
16	13.6	15.1	8.5	8.9
32	24.6	30.3	14.5	
clp	8.5	6.5	2.4	0.7

	Storm	SSN	UC12	UC24
Scenarios	32,768	32,768	512	256
Variables	41,255,033	23,134,297	28,947,516	28,950,648
Constraints	17,301,689	5,734,401	30,431,232	30,431,232

Speed-up of PIPS-S relative to 1-core PIPS-S and 1-core clp

Cores	Storm	SSN	UC12	UC24
1	1	1	1	1
8	15	19	7	6
16	52	45	14	12
32	117	103	26	22
64	152	181	44	41
128	202	289	60	64
256	285	383	70	80
clp	299	45	67	68

- Instance of UC12
 - 8,192 scenarios
 - 463,113,276 variables
 - 486,899,712 constraints
- Requires 1 TB of RAM ($\geq$ 1024 Blue Gene cores)
- Runs from an advanced basis

Cores	Iterations	Time (h)	Iter/sec
1024	Exceeded execution time limit		
2048	82,638	6.14	3.74
4096	75,732	5.03	4.18
8192	86,439	4.67	5.14

Parallelising the dual revised simplex method: Conclusions

Parallelising the dual revised simplex method: Conclusions

- Two schemes for general LP problems
 - Meaningful performance improvement
 - Have led to publicised advances in a leading commercial solver
- One scheme for stochastic LP problems
 - Demonstrated scalable parallel performance... for highly specialised problems... on highly specialised machines
 - Solved problems which would be intractable using commercial serial solvers
- Helped develop two really talented young researchers: Qi Huangfu and Miles Lubin

Slides: <http://www.maths.ed.ac.uk/hall/COB14/>

Paper: M. Lubin, J. A. J. Hall, C. G. Petra, and M. Anitescu
Parallel distributed-memory simplex for large-scale stochastic LP problems

Computational Optimization and Applications, 55(3):571–596, 2013



Cup winners: 2013