SYNPLEX
A task-parallel scheme for the revised simplex method

Julian Hall

School of Mathematics

University of Edinburgh

June 23rd 2005

Overview

- The (standard and revised) simplex method for linear programming

Overview

- The (standard and revised) simplex method for linear programming
- Approaches to parallelising the simplex method

Overview

- The (standard and revised) simplex method for linear programming
- Approaches to parallelising the simplex method
- SYNPLEX

Overview

- The (standard and revised) simplex method for linear programming
- Approaches to parallelising the simplex method
- SYNPLEX
- Results and conclusions

Solving LP problems

$$
\begin{array}{rl}
\text{minimize} & f = \boldsymbol{c}^T \boldsymbol{x} \\
\text{subject to} & A\boldsymbol{x} = \boldsymbol{b} \\
 & \boldsymbol{x} \geq \boldsymbol{0} \\
\text{where} & \boldsymbol{x} \in \mathbb{R}^n \quad \text{and} \quad \boldsymbol{b} \in \mathbb{R}^m
\end{array}
$$

Solving LP problems

$$\begin{array}{rl} \text{minimize} & f = \boldsymbol{c}^T\boldsymbol{x} \\ \text{subject to} & A\boldsymbol{x} = \boldsymbol{b} \\ & \boldsymbol{x} \geq \boldsymbol{0} \\ \text{where} & \boldsymbol{x} \in \mathbb{R}^n \quad \text{and} \quad \boldsymbol{b} \in \mathbb{R}^m \end{array}$$

- At any vertex the variables may be partitioned into index sets
  - $\mathcal{B}$ of m basic variables $\boldsymbol{x}_B \geq \boldsymbol{0}$
  - $\mathcal{N}$ of $n - m$ nonbasic variables $\boldsymbol{x}_N = \boldsymbol{0}$

Solving LP problems

$$\begin{array}{rl} \text{minimize} & f = \boldsymbol{c}^T\boldsymbol{x} \\ \text{subject to} & A\boldsymbol{x} = \boldsymbol{b} \\ & \boldsymbol{x} \geq \boldsymbol{0} \\ \text{where} & \boldsymbol{x} \in \mathbb{R}^n \quad \text{and} \quad \boldsymbol{b} \in \mathbb{R}^m \end{array}$$

- At any vertex the variables may be partitioned into index sets
  - $\mathcal{B}$ of m basic variables $\boldsymbol{x}_B \geq \boldsymbol{0}$
  - $\mathcal{N}$ of $n - m$ nonbasic variables $\boldsymbol{x}_N = \boldsymbol{0}$
- Components of $\boldsymbol{c}$ and columns of A are
  - the basic costs $\boldsymbol{c}_B$ and basis matrix B
  - the non-basic costs $\boldsymbol{c}_N$ and matrix N

Reduced LP problem

At any vertex the original problem is

$$\begin{array}{lccccccc} \text{minimize} & f & = & \boldsymbol{c}_N^T \boldsymbol{x}_N & + & \boldsymbol{c}_B^T \boldsymbol{x}_B & & \\ \text{subject to} & & & N\,\boldsymbol{x}_N & + & B\,\boldsymbol{x}_B & = & \boldsymbol{b} \\ & & & & & \boldsymbol{x}_N \geq \boldsymbol{0} \quad \boldsymbol{x}_B \geq \boldsymbol{0} & & \end{array}$$

Reduced LP problem

At any vertex the original problem is

$$\begin{array}{llcccccc} \text{minimize} & f & = & \boldsymbol{c}_N^T \boldsymbol{x}_N & + & \boldsymbol{c}_B^T \boldsymbol{x}_B & & \\ \text{subject to} & & & N\, \boldsymbol{x}_N & + & B\, \boldsymbol{x}_B & = & \boldsymbol{b} \\ & & & \multicolumn{3}{c}{\boldsymbol{x}_N \geq \boldsymbol{0} \quad \boldsymbol{x}_B \geq \boldsymbol{0}} & & \end{array}$$

Eliminate $\boldsymbol{x}_B$ from the objective to give

$$\begin{array}{llcccccc} \text{minimize} & f & = & \hat{\boldsymbol{c}}_N^T \boldsymbol{x}_N & + & \hat{f} & & \\ \text{subject to} & & & \hat{N}\, \boldsymbol{x}_N & + & I\, \boldsymbol{x}_B & = & \hat{\boldsymbol{b}} \\ & & & \multicolumn{3}{c}{\boldsymbol{x}_N \geq \boldsymbol{0} \quad \boldsymbol{x}_B \geq \boldsymbol{0}} & & \end{array}$$

Reduced LP problem

At any vertex the original problem is

$$\begin{array}{llcccccc} \text{minimize} & f & = & \boldsymbol{c}_N^T\boldsymbol{x}_N & + & \boldsymbol{c}_B^T\boldsymbol{x}_B & & \\ \text{subject to} & & & N\,\boldsymbol{x}_N & + & B\,\boldsymbol{x}_B & = & \boldsymbol{b} \\ & & & & \boldsymbol{x}_N \geq \boldsymbol{0} & \boldsymbol{x}_B \geq \boldsymbol{0} & & \end{array}$$

Eliminate $\boldsymbol{x}_B$ from the objective to give

$$\begin{array}{llcccccc} \text{minimize} & f & = & \hat{\boldsymbol{c}}_N^T\boldsymbol{x}_N & + & \hat{f} & & \\ \text{subject to} & & & \hat{N}\,\boldsymbol{x}_N & + & I\,\boldsymbol{x}_B & = & \hat{\boldsymbol{b}} \\ & & & & \boldsymbol{x}_N \geq \boldsymbol{0} & \boldsymbol{x}_B \geq \boldsymbol{0} & & \end{array}$$

where $\hat{\boldsymbol{b}} = B^{-1}\boldsymbol{b}$, $\hat{N} = B^{-1}N$, $\hat{f} = \boldsymbol{c}_B^T\hat{\boldsymbol{b}}$ and $\hat{\boldsymbol{c}}_N$ is the vector of **reduced costs**

$$\hat{\boldsymbol{c}}_N^T = \boldsymbol{c}_N^T - \boldsymbol{c}_B^T\hat{N}$$

The standard simplex method

	$\mathcal{N}$	$\mathcal{B}$	RHS
1 $\vdots$ m	$\hat{N}$	I	$\hat{\boldsymbol{b}}$
0	$\hat{\boldsymbol{c}}_N^T$	$\boldsymbol{0}^T$	$-\hat{f}$

The standard simplex method

	$\mathcal{N}$	$\mathcal{B}$	RHS
1 ⋮ m	$\hat{N}$	I	$\hat{\boldsymbol{b}}$
0	$\hat{\boldsymbol{c}}_N^T$	$\boldsymbol{0}^T$	$-\hat{f}$

In each iteration:

The standard simplex method

	$\mathcal{N}$	$\mathcal{B}$	RHS
1 $\vdots$ m	$\hat{N}$	$\boldsymbol{I}$	$\hat{\boldsymbol{b}}$
0	$\hat{\boldsymbol{c}}_N^T$	$\mathbf{0}^T$	$-\hat{f}$

In each iteration:

- Select the **pivotal column** q' of a nonbasic variable $q \in \mathcal{N}$ to be increased from zero

The standard simplex method

	$\mathcal{N}$	$\mathcal{B}$	RHS
1 $\vdots$ m	$\hat{N}$	I	$\hat{\boldsymbol{b}}$
0	$\hat{\boldsymbol{c}}_N^T$	$\boldsymbol{0}^T$	$-\hat{f}$

In each iteration:

- Select the **pivotal column** q' of a nonbasic variable $q \in \mathcal{N}$ to be increased from zero
- Find the **pivotal row** p of the first basic variable $p' \in \mathcal{B}$ to be zeroed

The standard simplex method

	$\mathcal{N}$	$\mathcal{B}$	RHS
1 $\vdots$ m	$\hat{N}$	I	$\hat{\boldsymbol{b}}$
0	$\hat{\boldsymbol{c}}_N^T$	$\boldsymbol{0}^T$	$-\hat{f}$

In each iteration:

- Select the **pivotal column** q' of a nonbasic variable $q \in \mathcal{N}$ to be increased from zero
- Find the **pivotal row** p of the first basic variable $p' \in \mathcal{B}$ to be zeroed
- Exchange indices p' and q between sets $\mathcal{B}$ and $\mathcal{N}$
- Update tableau corresponding to this **basis change**

The standard simplex method (cont.)

Advantages:

- Easy to understand
- Simple to implement

The standard simplex method (cont.)

Advantages:

- Easy to understand
- Simple to implement

Disadvantages:

- Expensive: the matrix $\hat{N}$ 'usually' treated as full
  - Storage requirement: $O(mn)$ memory locations
  - Computation requirement: $O(mn)$ floating point operations per iteration
- Numerically unstable

Revised simplex method

Given $\hat{\boldsymbol{c}}_N$, $\hat{\boldsymbol{b}}$ and a representation of B^{-1}, repeat

`CHUZC`: Scan the reduced costs $\hat{\boldsymbol{c}}_N$ for a good candidate q to enter the basis

Revised simplex method

Given $\hat{\boldsymbol{c}}_N$, $\hat{\boldsymbol{b}}$ and a representation of B^{-1}, repeat

`CHUZC:` Scan the reduced costs $\hat{\boldsymbol{c}}_N$ for a good candidate q to enter the basis
`FTRAN:` Form the pivotal column $\hat{\boldsymbol{a}}_q = B^{-1}\boldsymbol{a}_q$, where $\boldsymbol{a}_q$ is column q of A

Revised simplex method

Given $\hat{\boldsymbol{c}}_N$, $\hat{\boldsymbol{b}}$ and a representation of B^{-1}, repeat

`CHUZC:` Scan the reduced costs $\hat{\boldsymbol{c}}_N$ for a good candidate q to enter the basis
`FTRAN:` Form the pivotal column $\hat{\boldsymbol{a}}_q = B^{-1}\boldsymbol{a}_q$, where $\boldsymbol{a}_q$ is column q of A
`CHUZR:` Scan the ratios $\hat{b}_i/\hat{a}_{iq}$ for the row p of a good candidate to leave the basis

Revised simplex method

Given $\hat{\boldsymbol{c}}_N$, $\hat{\boldsymbol{b}}$ and a representation of B^{-1}, repeat

`CHUZC:` Scan the reduced costs $\hat{\boldsymbol{c}}_N$ for a good candidate q to enter the basis
`FTRAN:` Form the pivotal column $\hat{\boldsymbol{a}}_q = B^{-1}\boldsymbol{a}_q$, where $\boldsymbol{a}_q$ is column q of A
`CHUZR:` Scan the ratios $\hat{b}_i/\hat{a}_{iq}$ for the row p of a good candidate to leave the basis
Let $\alpha = \hat{b}_p/\hat{a}_{pq}$
Update $\hat{\boldsymbol{b}} := \hat{\boldsymbol{b}} - \alpha\hat{\boldsymbol{a}}_q$

Revised simplex method

Given $\hat{\boldsymbol{c}}_N$, $\hat{\boldsymbol{b}}$ and a representation of B^{-1}, repeat

`CHUZC:` Scan the reduced costs $\hat{\boldsymbol{c}}_N$ for a good candidate q to enter the basis
`FTRAN:` Form the pivotal column $\hat{\boldsymbol{a}}_q = B^{-1}\boldsymbol{a}_q$, where $\boldsymbol{a}_q$ is column q of A
`CHUZR:` Scan the ratios $\hat{b}_i/\hat{a}_{iq}$ for the row p of a good candidate to leave the basis
Let $\alpha = \hat{b}_p/\hat{a}_{pq}$
Update $\hat{\boldsymbol{b}} := \hat{\boldsymbol{b}} - \alpha\hat{\boldsymbol{a}}_q$
`BTRAN:` Form $\boldsymbol{\pi}^T = \boldsymbol{e}_p^T B^{-1}$

Revised simplex method

Given $\hat{\boldsymbol{c}}_N$, $\hat{\boldsymbol{b}}$ and a representation of B^{-1}, repeat

`CHUZC:` Scan the reduced costs $\hat{\boldsymbol{c}}_N$ for a good candidate q to enter the basis
`FTRAN:` Form the pivotal column $\hat{\boldsymbol{a}}_q = B^{-1}\boldsymbol{a}_q$, where $\boldsymbol{a}_q$ is column q of A
`CHUZR:` Scan the ratios $\hat{b}_i/\hat{a}_{iq}$ for the row p of a good candidate to leave the basis
Let $\alpha = \hat{b}_p/\hat{a}_{pq}$
Update $\hat{\boldsymbol{b}} := \hat{\boldsymbol{b}} - \alpha\hat{\boldsymbol{a}}_q$
`BTRAN:` Form $\boldsymbol{\pi}^T = \boldsymbol{e}_p^T B^{-1}$
`PRICE:` Form the pivotal row $\hat{\boldsymbol{a}}_p^T = \boldsymbol{\pi}^T N$

Revised simplex method

Given $\hat{\boldsymbol{c}}_N$, $\hat{\boldsymbol{b}}$ and a representation of B^{-1}, repeat

`CHUZC:` Scan the reduced costs $\hat{\boldsymbol{c}}_N$ for a good candidate q to enter the basis

`FTRAN:` Form the pivotal column $\hat{\boldsymbol{a}}_q = B^{-1}\boldsymbol{a}_q$, where $\boldsymbol{a}_q$ is column q of A

`CHUZR:` Scan the ratios $\hat{b}_i/\hat{a}_{iq}$ for the row p of a good candidate to leave the basis

Let $\alpha = \hat{b}_p/\hat{a}_{pq}$

Update $\hat{\boldsymbol{b}} := \hat{\boldsymbol{b}} - \alpha\hat{\boldsymbol{a}}_q$

`BTRAN:` Form $\boldsymbol{\pi}^T = \boldsymbol{e}_p^T B^{-1}$

`PRICE:` Form the pivotal row $\hat{\boldsymbol{a}}_p^T = \boldsymbol{\pi}^T N$

Update reduced costs $\hat{\boldsymbol{c}}_N^T := \hat{\boldsymbol{c}}_N^T - \hat{c}_q\hat{\boldsymbol{a}}_p^T$

Revised simplex method

Given $\hat{\boldsymbol{c}}_N$, $\hat{\boldsymbol{b}}$ and a representation of B^{-1}, repeat

`CHUZC:` Scan the reduced costs $\hat{\boldsymbol{c}}_N$ for a good candidate q to enter the basis
`FTRAN:` Form the pivotal column $\hat{\boldsymbol{a}}_q = B^{-1}\boldsymbol{a}_q$, where $\boldsymbol{a}_q$ is column q of A
`CHUZR:` Scan the ratios $\hat{b}_i/\hat{a}_{iq}$ for the row p of a good candidate to leave the basis
Let $\alpha = \hat{b}_p/\hat{a}_{pq}$
Update $\hat{\boldsymbol{b}} := \hat{\boldsymbol{b}} - \alpha\hat{\boldsymbol{a}}_q$
`BTRAN:` Form $\boldsymbol{\pi}^T = \boldsymbol{e}_p^T B^{-1}$
`PRICE:` Form the pivotal row $\hat{\boldsymbol{a}}_p^T = \boldsymbol{\pi}^T N$
Update reduced costs $\hat{\boldsymbol{c}}_N^T := \hat{\boldsymbol{c}}_N^T - \hat{c}_q\hat{\boldsymbol{a}}_p^T$
`If` (growth in factors) `then`
`INVERT:` Form a representation of B^{-1}
`else`
`UPDATE:` Update the representation of B^{-1} corresponding to the basis change
`end if`

Factored representation of B^{-1}

- Each iteration, $\boldsymbol{a}_q$ replaces column p of B

Factored representation of B^{-1}

- Each iteration, $\boldsymbol{a}_q$ replaces column p of B

$$B := B + (\boldsymbol{a}_q - \boldsymbol{a}_p)\boldsymbol{e}_p^T$$

Factored representation of B^{-1}

- Each iteration, $\boldsymbol{a}_q$ replaces column p of B

$$B := B + (\boldsymbol{a}_q - \boldsymbol{a}_p)\boldsymbol{e}_p^T \quad \Rightarrow \quad B^{-1} := \left(I - \frac{(\hat{\boldsymbol{a}}_q - \boldsymbol{e}_p)\boldsymbol{e}_p^T}{\hat{a}_{pq}} \right) B^{-1}$$

Factored representation of B^{-1}

- Each iteration, $\boldsymbol{a}_q$ replaces column p of B

$$B := B + (\boldsymbol{a}_q - \boldsymbol{a}_p)\boldsymbol{e}_p^T \quad \Rightarrow \quad B^{-1} := \left(I - \frac{(\hat{\boldsymbol{a}}_q - \boldsymbol{e}_p)\boldsymbol{e}_p^T}{\hat{a}_{pq}} \right) B^{-1}$$

- When using the product form update $B^{-1} = E_U^{-1} B_0^{-1}$

Factored representation of B^{-1}

- Each iteration, $\boldsymbol{a}_q$ replaces column p of B

$$B := B + (\boldsymbol{a}_q - \boldsymbol{a}_p)\boldsymbol{e}_p^T \quad \Rightarrow \quad B^{-1} := \left(I - \frac{(\hat{\boldsymbol{a}}_q - \boldsymbol{e}_p)\boldsymbol{e}_p^T}{\hat{a}_{pq}} \right) B^{-1}$$

- When using the product form update $B^{-1} = E_U^{-1} B_0^{-1}$
  - B_0^{-1} is represented by the INVERT etas
  - E_U^{-1} is represented by the UPDATE etas

Factored representation of B^{-1}

- Each iteration, $\boldsymbol{a}_q$ replaces column p of B

$$B := B + (\boldsymbol{a}_q - \boldsymbol{a}_p)\boldsymbol{e}_p^T \quad \Rightarrow \quad B^{-1} := \left(I - \frac{(\hat{\boldsymbol{a}}_q - \boldsymbol{e}_p)\boldsymbol{e}_p^T}{\hat{a}_{pq}} \right) B^{-1}$$

- When using the product form update $B^{-1} = E_U^{-1} B_0^{-1}$
  - B_0^{-1} is represented by the INVERT etas
  - E_U^{-1} is represented by the UPDATE etas
  - FTRAN ($\hat{\boldsymbol{a}}_q = B^{-1}\boldsymbol{a}_q$) is performed as

$$\tilde{\boldsymbol{a}}_q = B_0^{-1}\boldsymbol{a}_q \quad \text{and} \quad \hat{\boldsymbol{a}}_q = E_U^{-1}\tilde{\boldsymbol{a}}_q$$

Factored representation of B^{-1}

- Each iteration, $\boldsymbol{a}_q$ replaces column p of B

$$B := B + (\boldsymbol{a}_q - \boldsymbol{a}_p)\boldsymbol{e}_p^T \quad \Rightarrow \quad B^{-1} := \left(I - \frac{(\hat{\boldsymbol{a}}_q - \boldsymbol{e}_p)\boldsymbol{e}_p^T}{\hat{a}_{pq}} \right) B^{-1}$$

- When using the product form update $B^{-1} = E_U^{-1} B_0^{-1}$
  - B_0^{-1} is represented by the INVERT etas
  - E_U^{-1} is represented by the UPDATE etas
  - FTRAN ($\hat{\boldsymbol{a}}_q = B^{-1}\boldsymbol{a}_q$) is performed as

$$\tilde{\boldsymbol{a}}_q = B_0^{-1}\boldsymbol{a}_q \quad \text{and} \quad \hat{\boldsymbol{a}}_q = E_U^{-1}\tilde{\boldsymbol{a}}_q$$

  - BTRAN ($\boldsymbol{\pi}^T = \boldsymbol{e}_p^T B^{-1}$) is performed as

$$\tilde{\boldsymbol{\pi}}^T = \boldsymbol{e}_p^T E_U^{-1} \quad \text{and} \quad \boldsymbol{\pi}^T = \tilde{\boldsymbol{\pi}}^T B_0^{-1}$$

Revised simplex method with multiple pricing

`CHUZC`: Scan $\hat{\boldsymbol{c}}_N$ for a set $\mathcal{Q}$ of good candidates to enter the basis

Revised simplex method with multiple pricing

`CHUZC`: Scan $\hat{\boldsymbol{c}}_N$ for a set $\mathcal{Q}$ of good candidates to enter the basis
`FTRAN`: Form $\hat{\boldsymbol{a}}_j = B^{-1}\boldsymbol{a}_j$, $\forall\, j \in \mathcal{Q}$, where $\boldsymbol{a}_j$ is column j of A

Revised simplex method with multiple pricing

`CHUZC`: Scan $\hat{\boldsymbol{c}}_N$ for a set $\mathcal{Q}$ of good candidates to enter the basis

`FTRAN`: Form $\hat{\boldsymbol{a}}_j = B^{-1}\boldsymbol{a}_j$, $\forall\, j \in \mathcal{Q}$, where $\boldsymbol{a}_j$ is column j of A

`Loop` {minor iterations}

  `CHUZC_MI`: Scan $\hat{\boldsymbol{c}}_Q$ for a good candidate q to enter the basis

  `CHUZR`: Scan the ratios $\hat{b}_i/\hat{a}_{iq}$ for the row p of a good candidate to leave the basis

  `UPDATE_MI`: Update $\mathcal{Q} := \mathcal{Q}\backslash\{q\}$; $\hat{\boldsymbol{b}} := \hat{\boldsymbol{b}} - \alpha\hat{\boldsymbol{a}}_q$; $\hat{\boldsymbol{a}}_j$ and $\hat{\boldsymbol{c}}_j$, $\forall\, j \in \mathcal{Q}$

`End loop` {minor iterations}

Revised simplex method with multiple pricing

```
CHUZC:  Scan ĉ_N for a set Q of good candidates to enter the basis
```

`CHUZC:` Scan $\hat{\boldsymbol{c}}_N$ for a set $\mathcal{Q}$ of good candidates to enter the basis

`FTRAN:` Form $\hat{\boldsymbol{a}}_j = B^{-1}\boldsymbol{a}_j$, $\forall\, j \in \mathcal{Q}$, where $\boldsymbol{a}_j$ is column j of A

`Loop` {minor iterations}

  `CHUZC_MI:` Scan $\hat{\boldsymbol{c}}_Q$ for a good candidate q to enter the basis

  `CHUZR:` Scan the ratios $\hat{b}_i/\hat{a}_{iq}$ for the row p of a good candidate to leave the basis

  `UPDATE_MI:` Update $\mathcal{Q} := \mathcal{Q}\backslash\{q\}$; $\hat{\boldsymbol{b}} := \hat{\boldsymbol{b}} - \alpha\hat{\boldsymbol{a}}_q$; $\hat{\boldsymbol{a}}_j$ and $\hat{\boldsymbol{c}}_j$, $\forall\, j \in \mathcal{Q}$

`End loop` {minor iterations}

`For` {each basis change} `do`

  `BTRAN:` Form $\boldsymbol{\pi}^T = \boldsymbol{e}_p^T B^{-1}$

  `PRICE:` Form pivotal row $\hat{\boldsymbol{a}}_p^T = \boldsymbol{\pi}^T N$ and update $\hat{\boldsymbol{c}}_N := \hat{\boldsymbol{c}}_N - \hat{c}_q\hat{\boldsymbol{a}}_p^T$

  `If` {growth in factors} `then`

    `INVERT:` Form a new representation of B^{-1}

  `else`

    `UPDATE:` Update the representation of B^{-1} corresponding to the basis change

  `end if`

`End do`

Revised simplex method with multiple pricing

Disadvantages:

- Column selected in second and subsequent minor iteration is not the best
  Number of iterations required to solve the LP may increase

Revised simplex method with multiple pricing

Disadvantages:

- Column selected in second and subsequent minor iteration is not the best
  Number of iterations required to solve the LP may increase
- Some columns in $\mathcal{Q}$ may become unattractive during minor iterations
  Work of some FTRANs may be wasted

Revised simplex method with multiple pricing

Disadvantages:

- Column selected in second and subsequent minor iteration is not the best
  Number of iterations required to solve the LP may increase
- Some columns in $\mathcal{Q}$ may become unattractive during minor iterations
  Work of some FTRANs may be wasted

Advantages:

- Offers scope for task parallelism

Parallelising the simplex method

Why?

Parallelising the simplex method

Why?

- Never been done

Parallelising the simplex method

Why?

- Never been done
- Simplex method (still) very widely used

Parallelising the simplex method

Why?

- Never been done
- Simplex method (still) very widely used
- Enables significantly larger problems to be solved

Parallelising the simplex method

Why?

- Never been done
- Simplex method (still) very widely used
- Enables significantly larger problems to be solved

How?

Parallelising the simplex method

Why?

- Never been done
- Simplex method (still) very widely used
- Enables significantly larger problems to be solved

How?

- Exploit **data parallelism**
  Use several processors simultaneously to perform a single operation

Parallelising the simplex method

Why?

- Never been done
- Simplex method (still) very widely used
- Enables significantly larger problems to be solved

How?

- Exploit **data parallelism**
  Use several processors simultaneously to perform a single operation
- Exploit **task parallelism**
  Perform more than one operation simultaneously using several processors

Parallelising simplex computational components

Component	Properties	Scope for data parallelism

Parallelising simplex computational components

Component	Properties	Scope for data parallelism
CHUZC	Pass through a vector	Immediate

Parallelising simplex computational components

Component	Properties	Scope for data parallelism
CHUZC	Pass through a vector	Immediate
FTRAN	INVERT etas are short: some may be applied independently	Little
	UPDATE etas are long(er): may be applied as a matrix vector product	Immediate

Parallelising simplex computational components

Component	Properties	Scope for data parallelism
CHUZC	Pass through a vector	Immediate
FTRAN	INVERT etas are short: some may be applied independently	Little
	UPDATE etas are long(er): may be applied as a matrix vector product	Immediate
UPDATE_MI	Dense Gauss-Jordan elimination	Immediate

Parallelising simplex computational components

Component	Properties	Scope for data parallelism
CHUZC	Pass through a vector	Immediate
FTRAN	INVERT etas are short: some may be applied independently	Little
	UPDATE etas are long(er): may be applied as a matrix vector product	Immediate
UPDATE_MI	Dense Gauss-Jordan elimination	Immediate
CHUZR	Pass through a vector	Immediate

Parallelising simplex computational components

Component	Properties	Scope for data parallelism
CHUZC	Pass through a vector	Immediate
FTRAN	INVERT etas are short: some may be applied independently	Little
	UPDATE etas are long(er): may be applied as a matrix vector product	Immediate
UPDATE_MI	Dense Gauss-Jordan elimination	Immediate
CHUZR	Pass through a vector	Immediate
BTRAN	UPDATE etas: negligible computation	
	INVERT etas: (as **FTRAN**)	Little

Parallelising simplex computational components

Component	Properties	Scope for data parallelism
CHUZC	Pass through a vector	Immediate
FTRAN	INVERT etas are short: some may be applied independently	Little
	UPDATE etas are long(er): may be applied as a matrix vector product	Immediate
UPDATE_MI	Dense Gauss-Jordan elimination	Immediate
CHUZR	Pass through a vector	Immediate
BTRAN	UPDATE etas: negligible computation	
	INVERT etas: (as **FTRAN**)	Little
PRICE	Matrix vector product	Immediate

Parallelising simplex computational components

Component	Properties	Scope for data parallelism
CHUZC	Pass through a vector	Immediate
FTRAN	INVERT etas are short: some may be applied independently	Little
	UPDATE etas are long(er): may be applied as a matrix vector product	Immediate
UPDATE_MI	Dense Gauss-Jordan elimination	Immediate
CHUZR	Pass through a vector	Immediate
BTRAN	UPDATE etas: negligible computation	
	INVERT etas: (as **FTRAN**)	Little
PRICE	Matrix vector product	Immediate
INVERT	Searches through B_0 and (half-)FTRANs	Little (traditionally)

Past approaches

Standard simplex method

- Good parallel efficiency achieved

Past approaches

Standard simplex method

- Good parallel efficiency achieved... many times!

Past approaches

Standard simplex method

- Good parallel efficiency achieved... many times!
- Totally uncompetitive with serial RSM without a prohibitively large number of processors

Past approaches

Standard simplex method

- Good parallel efficiency achieved... many times!
- Totally uncompetitive with serial RSM without a prohibitively large number of processors

Data parallel revised simplex method

- Only the immediate parallelism in PRICE has been exploited

Past approaches

Standard simplex method

- Good parallel efficiency achieved... many times!
- Totally uncompetitive with serial RSM without a prohibitively large number of processors

Data parallel revised simplex method

- Only the immediate parallelism in PRICE has been exploited
- Significant speed-up only obtained when $n \gg m$ so PRICE dominates
  For such problems an efficient serial solver uses partial pricing so PRICE no longer dominates

Data/task parallel revised simplex method (with multiple pricing)

Wunderling (1996)

- Parallel (except for INVERT) for only two processors
- Good results only for problems when $n \gg m$

Data/task parallel revised simplex method (with multiple pricing)

Wunderling (1996)

- Parallel (except for INVERT) for only two processors
- Good results only for problems when $n \gg m$

ASYNPLEX: Hall and McKinnon (1995)

- Fully task parallel (inefficient) variant of the revised simplex method
- Speed-up (on Cray T3D) of up to 5 on modest Netlib problems

Data/task parallel revised simplex method (with multiple pricing)

Wunderling (1996)

- Parallel (except for INVERT) for only two processors
- Good results only for problems when $n \gg m$

ASYNPLEX: Hall and McKinnon (1995)

- Fully task parallel (inefficient) variant of the revised simplex method
- Speed-up (on Cray T3D) of up to 5 on modest Netlib problems

PARSMI: Hall and McKinnon (1996)

- Fully task parallel revised simplex method with multiple pricing
- Speed-up (on Cray T3D) of between 1.7 and 1.9 on modest Netlib problems

(Some) ASYNPLEX and PARSMI limitations

- Asynchronous—so *very* hard to implement

(Some) ASYNPLEX and PARSMI limitations

- Asynchronous—so *very* hard to implement
- Numerically unstable—due to overlapping INVERT with basis changes

(Some) ASYNPLEX and PARSMI limitations

- Asynchronous—so *very* hard to implement
- Numerically unstable—due to overlapping INVERT with basis changes
- Reduced costs always out-of-date—more iterations and wasted FTRANs

(Some) ASYNPLEX and PARSMI limitations

- Asynchronous—so *very* hard to implement
- Numerically unstable—due to overlapping INVERT with basis changes
- Reduced costs always out-of-date—more iterations and wasted FTRANs
- Significant communication overhead