

The future of the simplex method

Julian Hall

School of Mathematics

University of Edinburgh

29th June 2010



The future of the simplex method



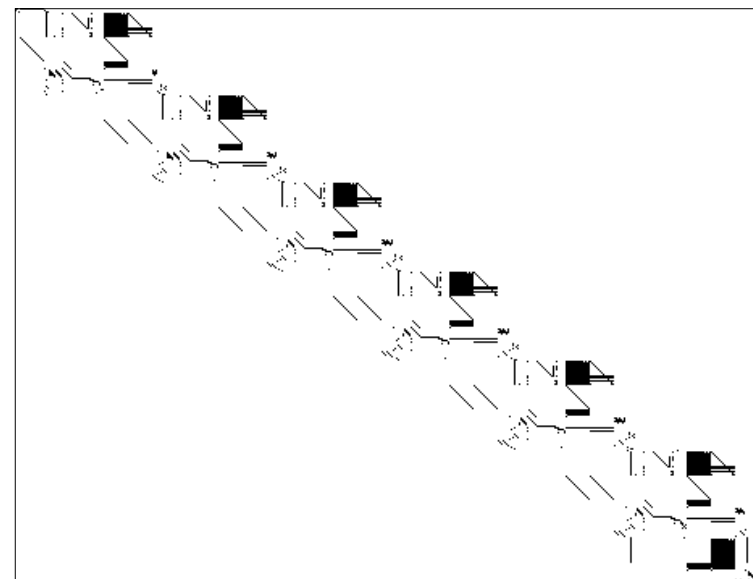
Overview

- Linear programming
 - Introduction
 - Mathematics
- Simplex method
 - Algorithm
 - Standard simplex method
 - Revised simplex method
- State of the art
 - Hyper-sparsity
 - Parallelism
 - Dual simplex method
 - Software
- Future
 - Serial performance
 - Multicore challenge
 - Data parallelism
 - Task parallelism
 - Structural parallelism
- Conclusions

Linear programming (LP)

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

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



STAIR: 356 rows, 467 columns and 3856 nonzeros

Mathematics of LP: Introduction

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

- **Geometry:** Feasible points form a convex polyhedron (feasible region)
- **Result:** At a vertex the variable set can be partitioned as

$$\mathcal{B} \cup \mathcal{N}$$

and the constraints written as

$$B\mathbf{x}_B + N\mathbf{x}_N = \mathbf{b}$$

such that

$$B \text{ is nonsingular and } \mathbf{x}_N = \mathbf{0}$$

Mathematics of LP: Reduced LP problem

At a vertex, for a partition $\mathcal{B} \cup \mathcal{N}$ with B nonsingular, the original problem is

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

Eliminate $\mathbf{x}_B$ from the objective to give the **reduced LP problem**

$$\begin{array}{ll} \text{minimize} & f = \mathbf{s}_N^T \mathbf{x}_N + \hat{f} \\ \text{subject to} & \hat{N} \mathbf{x}_N + I \mathbf{x}_B = \hat{\mathbf{b}} \\ & \mathbf{x}_N \geq \mathbf{0} \quad \mathbf{x}_B \geq \mathbf{0}, \end{array}$$

where $\hat{\mathbf{b}} = B^{-1}\mathbf{b}$, $\hat{N} = B^{-1}N$ and $\mathbf{s}_N$ is given by

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

Vertex is optimal $\iff \mathbf{s}_N \geq \mathbf{0}$

The (primal) simplex algorithm

Terminology:

- $x_B = B^{-1}b$ are the **basic variables**
- $x_N = 0$ are the **nonbasic variables**
- B is the **basis matrix**

In each iteration:

- Choose index $q \in \mathcal{N}$ of nonbasic variable x_q with $s_q < 0$
- Identify index $p \in \mathcal{B}$ of the first basic variable to be zeroed as x_q is increased from zero
- Exchange indices p and q between $\mathcal{B}$ and $\mathcal{N}$

Standard simplex method

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

In each iteration:

- Increase x_q from zero
- Use the **pivotal column** q of $\hat{N}$ and $\hat{b}$ to find the **pivotal row** p
- Exchange indices p and q between $\mathcal{B}$ and $\mathcal{N}$
- Update tableau corresponding to this **basis change**
 - Scale row p by $\hat{N}_{pq}$
 - Eliminate $\hat{N}_{iq}$, $i = 0, \dots, m, i \neq p$

Revised simplex method

- Maintains a representation of B^{-1} (rather than $\hat{N}$)
- Pivotal column is

$$\hat{\mathbf{a}}_q = B^{-1} \mathbf{a}_q, \text{ where } \mathbf{a}_q \text{ is column } q \text{ of } A$$

- Pivotal row is

$$\hat{\mathbf{a}}_p^T = \boldsymbol{\pi}_p^T N, \text{ where } \boldsymbol{\pi}_p^T = \mathbf{e}_p^T B^{-1}$$

- Representation of B^{-1} is updated by exploiting

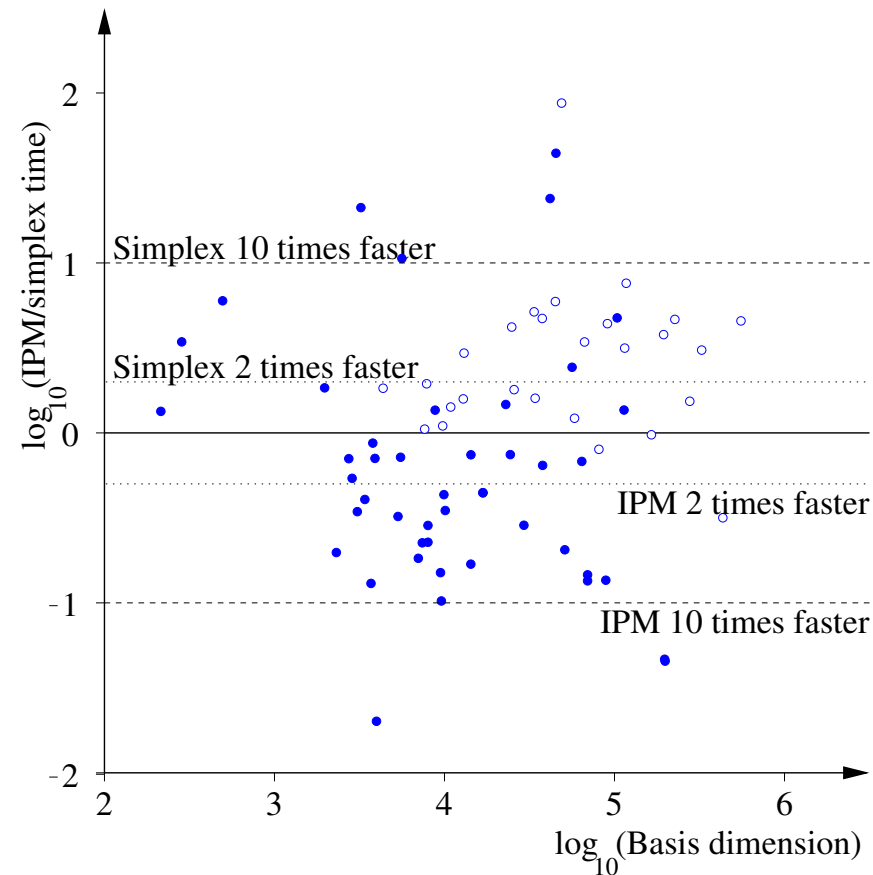
$$B^{-1} := \left(I - \frac{(\hat{\mathbf{a}}_q - \mathbf{e}_p) \mathbf{e}_p^T}{\hat{a}_{pq}} \right) B^{-1}$$

- Periodically B^{-1} is formed from scratch
- Efficient solution of large sparse LP problems requires the revised simplex method

State of the art: Hyper-sparsity

- Major computational cost each iteration is forming $B^{-1}\mathbf{r}_F$, $\mathbf{r}_B^T B^{-1}$ and $\mathbf{r}_\pi^T N$
- Vectors $\mathbf{r}_F$, $\mathbf{r}_B$ are always sparse
- Vector $\mathbf{r}_\pi$ may be sparse
- If the results of these operations are (usually) sparse then LP is said to be **hyper-sparse**
- LP structure means B^{-1} **is sparse**
- Exploiting hyper-sparsity has been the major contribution to the revised simplex method over the past decade
H and McKinnon (1998–2005)

State of the art: Exploiting hyper-sparsity



Simplex method typically better than interior point methods for hyper-sparse LP problems (○)
but frequently not for other LPs (●)

State of the art: Parallelism

Data parallel standard simplex method

- Good parallel efficiency achieved
- Totally uncompetitive with **serial** revised simplex method without prohibitive resources

Data parallel revised simplex method

- Only immediate parallelism is in forming $r_{\pi}^T N$
- When $n \gg m$, forming $r_{\pi}^T N$ dominates the cost of iterations: significant speed-up achieved
Bixby and Martin (2000)
- One “fully” parallel implementation: no speed-up achieved
Shu (1995)

State of the art: Task parallelism

Overlap computational components for different iterations

- **Wunderling (1996)**
“Fully” parallel for only two processors: good results only when $n \gg m$
- **ASYNPLEX: H and McKinnon (1995)**
Fully parallel (inefficient) revised simplex variant: speed-up of 5 but numerically unstable
- **PARSMI: H and McKinnon (1996)**
Fully parallel revised simplex variant: speed-up of 2 but numerically unstable
- **SYNPLEX: H (2005)**
“Fully” parallel revised simplex variant: speed-up of 3 and numerically stable

All data and task parallel implementations compromised by serial inversion of B

State of the art: Dual simplex method

- Dual simplex algorithm: in each iteration
 - Choose index $p \in \mathcal{B}$ of dual variable s_p with $x_p < 0$
 - Identify index $q \in \mathcal{N}$ of **the best** dual variable to be zeroed as s_p is increased from zero
 - Exchange indices p and q between $\mathcal{B}$ and $\mathcal{N}$
- Dual revised simplex method
 - Compute pivotal row
 - Perform dual ratio test to identify pivotal column
 - Compute pivotal column
- Dual revised simplex method preferred to primal
 - Easier to find dual feasible point
 - Dual ratio test makes better progress
 - Dual simplex is default in commercial codes
 - Feasibility retained when branching/cutting in MIP

State of the art: Software

- Several highly-developed commercial solvers
 - GuRoBi: **Gu**, **R**othberg and **B**ixby
 - CPLEX: Was ILOG, now IBM
 - Xpress
 - Mosek
- Many good public domain solvers: best are
 - Clp: H is now Project Manager
 - GLPK
 - Soplex
- No parallel solvers

Future: Serial performance

- Serial performance is spectacularly good
 - Flop count per iteration is near optimal
 - Number of iterations is near optimal
- Possibly some scope for improving cache efficiency
 - Identify/promote nonzeros in short dense vectors
- Don't look for much more improvement!

Future: Multicore challenge

- Moore's law drives core counts per processor, but clock speeds will stabilise
- Simplex method must exploit parallelism for further performance improvement
- Heterogenous desk-top architectures
 - Small number of CPUs with small number of fast cores
 - One GPU with large number of slower cores
- Multicore simplex challenge
 - Solve problems faster than serial codes
 - Solve larger problems

Future: Data parallelism

- Standard simplex method on a GPU
 - Should parallelise well
 - No good implementations (yet)
 - Space for (eg) $m = 2 \times 10^4$, $n = 3 \times 10^4$
 - Numerically unstable
 - May be no faster than a good serial solver
- To be implemented by H, Huangfu and Smith

Future: Task parallelism

- Parallel **dual** revised simplex
- Generate small section of standard simplex tableau

$$\hat{\mathbf{a}}_p^T = \boldsymbol{\pi}_p^T N, \text{ where } \boldsymbol{\pi}_p^T = \mathbf{e}_p^T B^{-1} \quad \forall p \in \mathcal{P}$$

Overlap serial solves and data parallel matrix-vector products

- Perform minor iterations of standard simplex method
- Compute pivotal columns

$$\hat{\mathbf{a}}_q = B^{-1} \mathbf{a}_q \quad \forall \quad q \in \mathcal{Q}$$

Overlap serial solves

- Data parallel reinversion is required
- Subject of current research by H and Huangfu with Format International and NAIS

Future: Structural parallelism

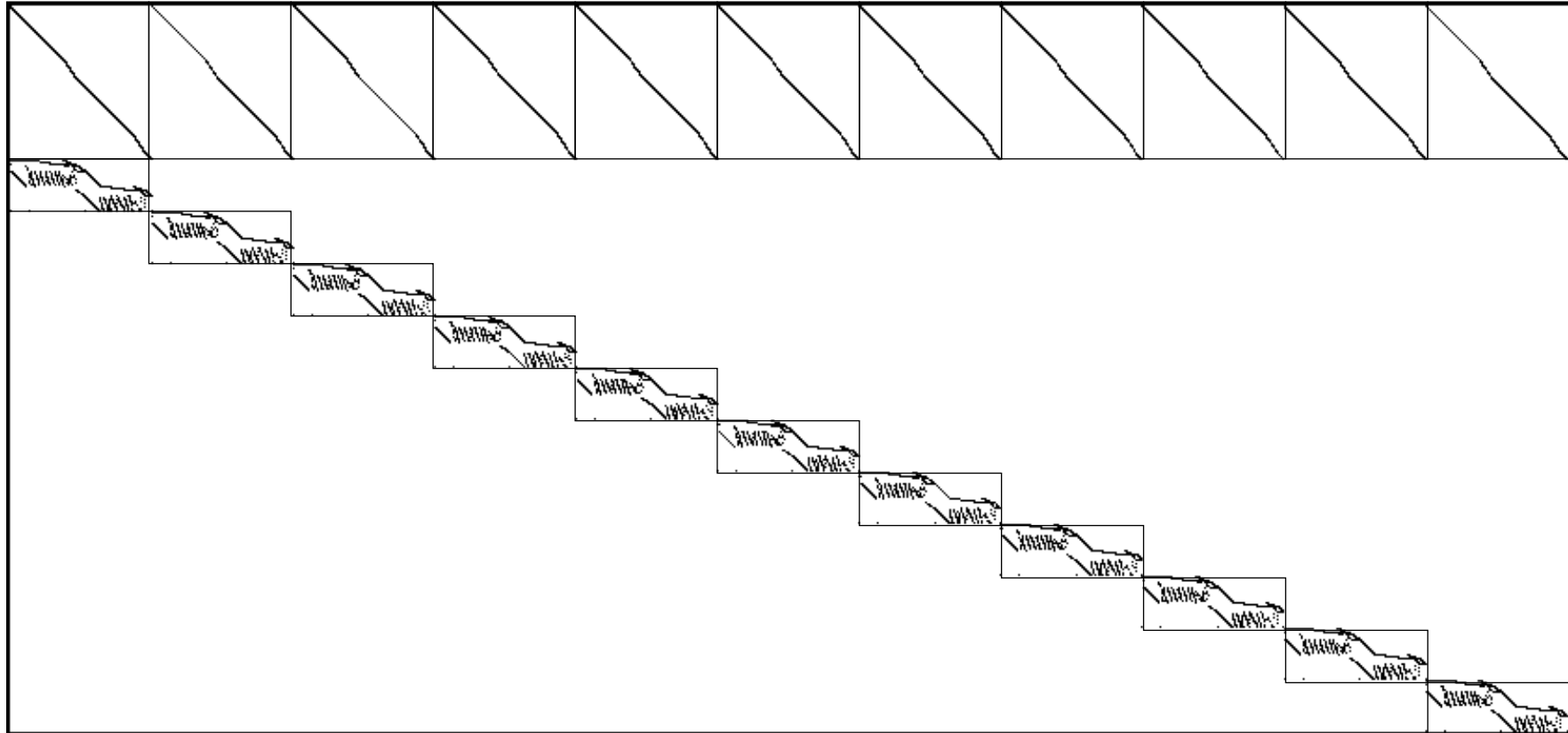
$$\begin{array}{ll} \text{minimize} & f = \mathbf{c}^T \mathbf{x} \\ \text{subject to} & \mathbf{A}\mathbf{x} = \mathbf{b} \quad \mathbf{x} \geq \mathbf{0} \end{array}$$

$$\mathbf{A} = \begin{bmatrix} A_{00} & A_{01} & A_{02} & \dots & A_{0r} \\ & A_{11} & & & \\ & & A_{22} & & \\ & & & \ddots & \\ & & & & A_{rr} \end{bmatrix}$$

Block angular LP (BALP) problems

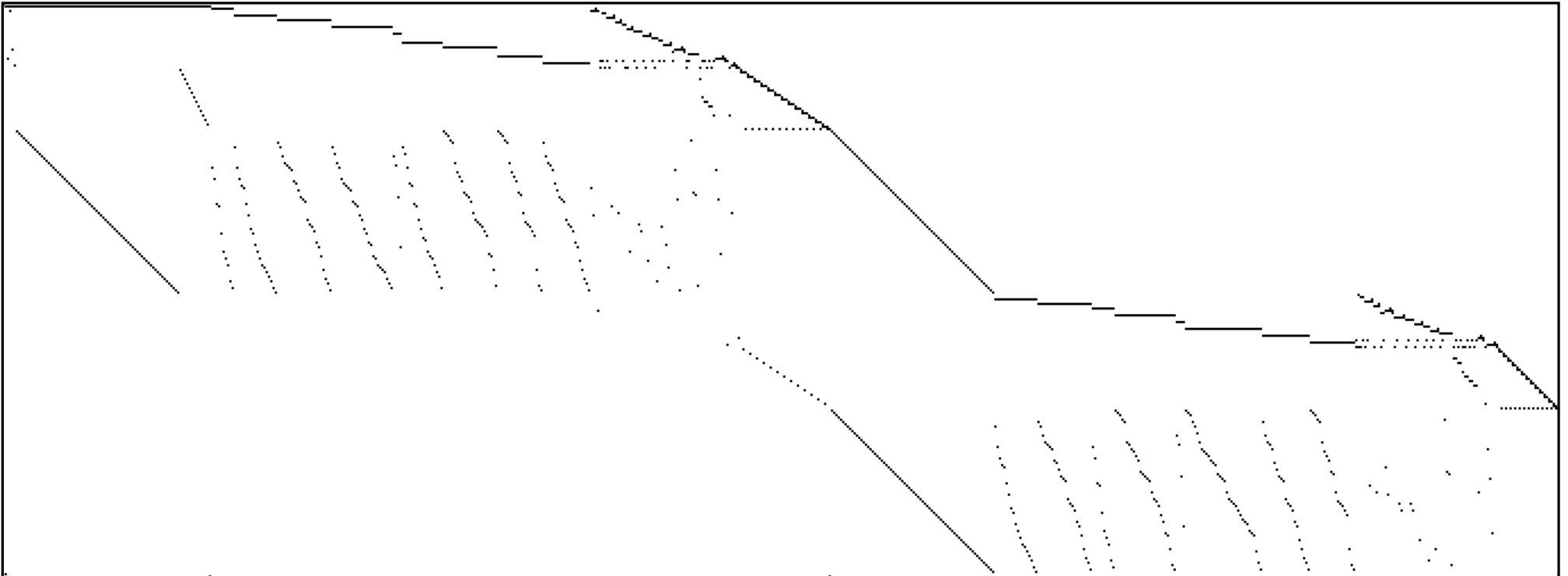
- Occur naturally in (eg) decentralised planning and multicommodity flow
- BALP structure can be identified in general sparse LPs

pds-02: 3518 rows, 7535 columns, 11 diagonal blocks

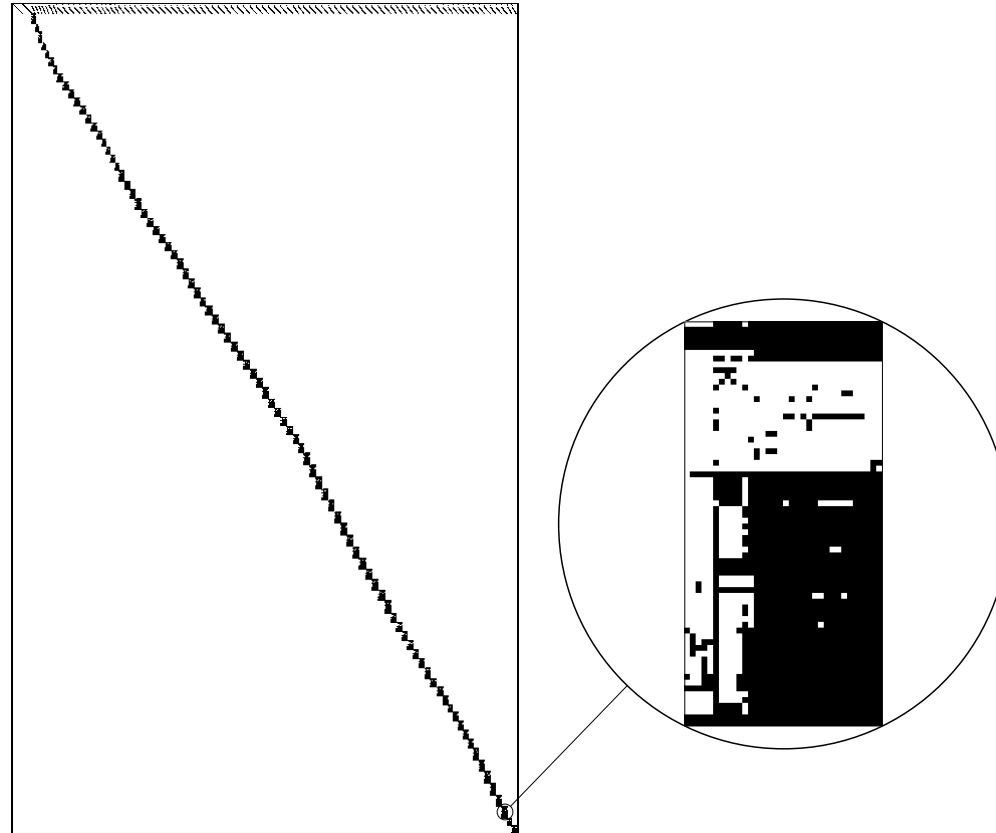


Source: Patient Distribution System, Carolan *et al.* (1990)

Diagonal block of pds-02



dcpl: 4950 rows, 3007 columns, 87 diagonal blocks



Source: Industrial, Hall (1997)

Future: Task parallel dual revised simplex for BALP

- For BALP problems the basis matrix is $B = \begin{bmatrix} D & E \\ R & T \end{bmatrix}$
 - D is small and square
 - E is unstructured and rectangular
 - R is block-rectangular
 - T is block-diagonal, with each diagonal block being non-singular
- Let $\hat{R} = T^{-1}R$: it has the same structure as R
- The **Schur complement** of T is $W = D - ET^{-1}R = D - E\hat{R}$
- To solve systems involving B requires operations with
 - matrices E and $\hat{R}$
 - an invertible representation of T and W
- Need to update representation as basis changes

Future: Prototype parallel BALP solver

- Current research by H and Smith
- In collaboration with SAS
- BALP structure identifies cheaply the parallelism in hyper-sparse operations with B^{-1}
- Target platform
 - Dual quad-core AMD Opteron 2378 system—to solve systems involving B
 - Nvidia GPU for matrix-vector products and standard simplex minor iterations
- High performance **serial** BALP solver has been developed
- Just have to parallelise the algebra!

Conclusions

- Simplex method still crucial to optimal decision-making
- Must attempt to exploit parallelism
- Developing better parallel implementations than current serial solvers will be very hard!
- Any success will attract attention!