

# Integer Linear System Solving

**Pascal Giorgi**



*Symbolic Computation Group,  
University of Waterloo, Canada*

in collaboration with Arne Storjohann

*Challenges in Linear and Polynomial Algebra  
in Symbolic Computation Software  
October 1-6, 2005.*

# Motivations

Large linear systems are involved  
in many mathematical applications

## over a field :

- ▶ integers factorization [Odlyzko 1999],
- ▶ discrete logarithm [Odlyzko 1999 ; Thomé 2003],

## over the integers :

- ▶ number theory [Cohen 1993],
- ▶ group theory [Newman 1972],
- ▶ integer programming [Aardal, Hurkens, Lenstra 1999]

## Problem

Let  $A$  a non-singular matrix and  $b$  a vector defined over  $\mathbb{Z}$ .

Problem : Compute  $x = A^{-1}b$  over the rational numbers.

$$A = \begin{bmatrix} -289 & 236 & 79 & -268 \\ 108 & -33 & -211 & 309 \\ -489 & 104 & -24 & -25 \\ 308 & 99 & -108 & 66 \end{bmatrix}, \quad b = \begin{bmatrix} -131 \\ 321 \\ 147 \\ 43 \end{bmatrix}.$$

$$x = A^{-1}b = \begin{bmatrix} \frac{-9591197817}{95078} \\ 131244 \\ \frac{47539}{2909895} \\ \frac{665546}{2909895} \\ \frac{665546}{2909895} \end{bmatrix}$$

Main difficulty : **expression swell**

# Interest in linear algebra

Integer linear systems are central in recent linear algebra algorithms

- ▶ Determinant :

[Abbott, Bronstein, Mulders 1999 ; Storjohann 2005]

- ▶ Smith Form :

[Eberly, Giesbrecht, Villard 2000]

- ▶ Nullspace, Kernel :

[Chen, Storjohann 2005]

- ▶ Diophantine solutions :

[Giesbrecht 1997 ; Giesbrecht, Lobo, Saunders 1998 ; Mulders, Storjohann 2003 ; Mulders 2004]

# Algorithms for non-singular system solving

- ▶ Gaussian elimination and CRA  
 $O\sim(n^{\omega+1} \log \|A\|)$  bit operations
- ▶ Linear P-adic lifting [Monck, Carter 1979, Dixon 1982]  
 $O\sim(n^3 \log \|A\|)$  bit operations
- ▶ High order lifting [Storjohann 2005]  
 $O\sim(n^{\omega} \log \|A\|)$  bit operations

# P-adic algorithm for dense systems

Scheme to compute  $A^{-1}b$  :

1)

$$1.1) B := A^{-1} \bmod p$$

$$1.2) r := b$$

2) for  $i := 0$  to  $k$

$$2-1) x_i := Br \bmod p$$

$$2-2) r := (1/p)(r - A.x_i)$$

3)

$$3-1) x := \sum_{i=0}^k x_i \cdot p^i$$

3-2) *rational reconstruction on  $x$*

# P-adic algorithm for dense systems

Scheme to compute  $A^{-1}b$  :

1)

$$1.1) B := A^{-1} \bmod p$$

$$O^{\sim}(n^3 \log \|A\|)$$

$$1.2) r := b$$

2) for  $i := 0$  to  $k$

$$k = O^{\sim}(n)$$

$$2-1) x_i := Br \bmod p$$

$$O^{\sim}(n^2 \log \|A\|)$$

$$2-2) r := (1/p)(r - A.x_i)$$

$$O^{\sim}(n^2 \log \|A\|)$$

3)

$$3-1) x := \sum_{i=0}^k x_i \cdot p^i$$

3-2) *rational reconstruction on  $x$*

# Dense linear system in practice

Efficient implementations are available :

LinBox 1.0 [[www.linalg.org](http://www.linalg.org)]

IML library [[www.uwaterloo.ca/~z4chen/iml](http://www.uwaterloo.ca/~z4chen/iml)]

Details :

- ▶ level 3 BLAS-based matrix inversion over prime field
  - with LQUP factorization [[Dumas, Giorgi, Pernet 2004](#)]
  - with Echelon form [[Chen, Storjohann 2005](#)]
- ▶ level 2 BLAS-based matrix-vector product
  - use of CRT over the integers
- ▶ rational number reconstruction
  - half GCD [[Schönage 1971](#)]
  - heuristic using integer multiplication [[NTL library](#)]



# Timing of Dense linear system solving

use of LinBox library on Pentium 4 - 3.4Ghz, 2Go RAM.

Random dense linear system with coefficients over 3 bits :

n	500	1000	2000	3000	4000	5000
time	0.6s	4.3s	31.1s	99.6s	236.8s	449.2s

Random dense linear system with coefficients over 20 bits :

n	500	1000	2000	3000	4000	5000
time	1.8s	12.9s	91.5s	299.7s	706.4s	MT

performances improvement by a factor 10  
compare to NTL's tuned implementation

what does happen when matrices are sparse?

We consider sparse matrices with  $O(n)$  non zero elements  
↪ matrix-vector product needs only  $O(n)$  operations.

Scheme to compute  $A^{-1}b$  :

1)

$$1.1) B := A^{-1} \bmod p$$

$$1.2) r := b$$

2) for  $i := 0$  to  $k$

$$2-1) x_i := Br \bmod p$$

$$2-2) r := (1/p)(r - A.x_i)$$

3)

$$3-1) x := \sum_{i=0}^k x_i \cdot p^i$$

3-2) *rational reconstruction on  $x$*

# Sparse linear system and P-adic lifting

P-adic lifting doesn't improve complexity as in dense case.

↪ computing the modular inverse is proscribed due to fill-in

Solution [Wiedemann 1986 ; Kaltofen, Saunders 1991] :

modular inverse is replaced by modular minimal polynomial

Let  $A \in \mathbb{Z}_p^{n \times n}$  of full rank and  $b \in \mathbb{Z}_p^n$ . Then  $x = A^{-1}b$  can be expressed as a linear combination of the Krylov subspace  $\{b, Ab, \dots, A^n b\}$

# Sparse linear system and P-adic lifting

P-adic lifting doesn't improve complexity as in dense case.

↪ computing the modular inverse is proscribed due to fill-in

Solution [Wiedemann 1986 ; Kaltofen, Saunders 1991] :

modular inverse is replaced by modular minimal polynomial

Let  $A \in \mathbb{Z}_p^{n \times n}$  of full rank and  $b \in \mathbb{Z}_p^n$ . Then  $x = A^{-1}b$  can be expressed as a linear combination of the Krylov subspace  $\{b, Ab, \dots, A^n b\}$

Let  $\Pi^A(\lambda) = c_0 + c_1\lambda + \dots + \lambda^d \in \mathbb{Z}_p[\lambda]$  be the minimal polynomial of  $A$

# Sparse linear system and P-adic lifting

P-adic lifting doesn't improve complexity as in dense case.

↪ computing the modular inverse is proscribed due to fill-in

Solution [Wiedemann 1986 ; Kaltofen, Saunders 1991] :

modular inverse is replaced by modular minimal polynomial

Let  $A \in \mathbb{Z}_p^{n \times n}$  of full rank and  $b \in \mathbb{Z}_p^n$ . Then  $x = A^{-1}b$  can be expressed as a linear combination of the Krylov subspace  $\{b, Ab, \dots, A^{n-1}b\}$

Let  $\Pi^A(\lambda) = c_0 + c_1\lambda + \dots + \lambda^d \in \mathbb{Z}_p[\lambda]$  be the minimal polynomial of  $A$

$$A^{-1}b = \frac{-1}{c_0}(c_1b + c_2Ab + \dots + A^{d-1}b)$$

# Sparse linear system and P-adic lifting

P-adic lifting doesn't improve complexity as in dense case.

↪ computing the modular inverse is proscribed due to fill-in

Solution [Wiedemann 1986 ; Kaltofen, Saunders 1991] :

modular inverse is replaced by modular minimal polynomial

Let  $A \in \mathbb{Z}_p^{n \times n}$  of full rank and  $b \in \mathbb{Z}_p^n$ . Then  $x = A^{-1}b$  can be expressed as a linear combination of the Krylov subspace  $\{b, Ab, \dots, A^{n-1}b\}$

Let  $\Pi^A(\lambda) = c_0 + c_1\lambda + \dots + \lambda^d \in \mathbb{Z}_p[\lambda]$  be the minimal polynomial of  $A$

$$A^{-1}b = \underbrace{\frac{-1}{c_0}(c_1b + c_2Ab + \dots + A^{d-1}b)}_x$$

# P-adic algorithm for sparse systems

Scheme to compute  $A^{-1}b$  :

1)

$$1.1) \Pi := \text{minpoly}(A) \bmod p$$

$$1.2) r := b$$

2) for  $i := 0$  to  $k$

$$2-1) x_i := (-1/\Pi[0]) \sum_{j=1}^{\deg \Pi} \Pi[j] \cdot A^{j-1} r \bmod p$$

$$2-2) r := (1/p)(r - A \cdot x_i)$$

3)

$$3-1) x := \sum_{i=0}^k x_i \cdot p^i$$

3-2) *rational reconstruction on  $x$*



# P-adic algorithm for sparse systems

Scheme to compute  $A^{-1}b$  :

1)

$$1.1) \Pi := \text{minpoly}(A) \bmod p \quad O^{\sim}(n^2 \log \|A\|)$$

$$1.2) r := b$$

2) for  $i := 0$  to  $k$

$$k = O^{\sim}(n)$$

$$2-1) x_i := (-1/\Pi[0]) \sum_{j=1}^{\deg \Pi} \Pi[j] \cdot A^{j-1} r \bmod p \quad O^{\sim}(n^2 \log \|A\|)$$

$$2-2) r := (1/p)(r - A \cdot x_i) \quad O^{\sim}(n \log \|A\|)$$

3)

$$3-1) x := \sum_{i=0}^k x_i \cdot p^i$$

3-2) *rational reconstruction on  $x$*

Issue : computation of Krylov space  $\{r, Ar, \dots, A^{\deg \Pi} r\} \bmod p$

# Integer sparse linear system in practice

use of LinBox library on Pentium 4 - 3.4Ghz, 2Go RAM.

non-singular sparse linear system with coefficients over 3 bits  
and 10 non zero elements per row.

n	400	900	1600	2500
CRT + Wiedemann	7.3s	79.2s	464s	1769s
P-adic + Wiedemann	3.1s	32.7s	185s	709s
dense solver				

# Integer sparse linear system in practice

use of LinBox library on Pentium 4 - 3.4Ghz, 2Go RAM.

non-singular sparse linear system with coefficients over 3 bits  
and 10 non zero elements per row.

n	400	900	1600	2500
CRT + Wiedemann	7.3s	79.2s	464s	1769s
P-adic + Wiedemann	3.1s	32.7s	185s	709s
dense solver	0.5s	3.5s	13s	41s

# Integer sparse linear system in practice

use of LinBox library on Pentium 4 - 3.4Ghz, 2Go RAM.

non-singular sparse linear system with coefficients over 3 bits and 10 non zero elements per row.

n	400	900	1600	2500
CRT + Wiedemann	7.3s	79.2s	464s	1769s
P-adic + Wiedemann	3.1s	32.7s	185s	709s
dense solver	0.5s	3.5s	13s	41s

2-1)  $x_i = Br \bmod p$  (dense case)

2-1)  $x_i = (-1/\Pi[0]) \sum_{j=1}^{\deg \Pi} \Pi[j].A^{i-1}r \bmod p$  (sparse case)

Remark :

$n$  sparse matrix applications is far from **level 2 BLAS** in practice.

# Our objectives

In practice :

Integrate level 2 and 3 BLAS in integer sparse solver

In theory :

Improve bit complexity of sparse integer linear system solving  
 $\implies \tilde{O}(n^\delta)$  bits operations with  $\delta < 3$ ?

# Integration of BLAS in sparse solver

Goal :

- Minimize the number of sparse matrix-vector products.
- Maximize the calls of level 2 and 3 BLAS.

Block Wiedemann algorithm is well designed to incorporate BLAS.

Let  $k$  be the blocking factor of Block Wiedemann algorithm.  
then

- ▶ the number of sparse matrix-vector product is divided by roughly  $k$ .
- ▶ order  $k$  level 3 BLAS are integrated.

# Block Wiedemann and P-adic

Replace vector projections by block of vectors projections

$$k \{ [ U ] \} \begin{bmatrix} A^i \end{bmatrix} \begin{bmatrix} V \end{bmatrix} \quad k$$

Let  $N = n/k$

need right minimal block generator  $P \in \mathbb{Z}_p[X]$  of

$$\{ UV, UAV, UA^2V, \dots, UA^{2N}V \}$$

the cost to compute  $P$  is :

- ▶  $O(k^3 N^2)$  field operations [Coppersmith 1994],
- ▶  $\tilde{O}(k^3 N \log N)$  field operations [Beckermann, Labahn 1994; Kaltofen 1995; Thomé 2002],
- ▶  $\tilde{O}(k^\omega N \log N)$  field operations [Giorgi, Jeannerod, Villard 2003].

## Block Wiedemann and P-adic

Scheme to compute  $A^{-1}b$  :

1) for  $i := 0$  to  $k$

1-1)  $\Pi := \text{block minpoly } \{UA^i V_i\}_{1..2N} \text{ mod } p$

1-2)  $x_i := LC(A^i \cdot V_i, \Pi[i]_{*,1}) \text{ mod } p$

1-3)  $r := (1/p)(r - A \cdot x_i)$

2)

2-1)  $x := \sum_{i=0}^k x_i \cdot p^i$

2-2) *rational reconstruction on x*



# Block Wiedemann and P-adic

Scheme to compute  $A^{-1}b$  :

1) for  $i := 0$  to  $k$

$$k = \tilde{O}(n)$$

$$1-1) \Pi := \text{block minpoly} \{UA^i V_i\}_{1..2N} \bmod p \quad \tilde{O}(k^2 n \log \|A\|)$$

$$1-2) x_i := LC(A^i \cdot V_i, \Pi[i]_{*,1}) \bmod p \quad \tilde{O}(n^2 \log \|A\|)$$

$$1-3) r := (1/p)(r - A \cdot x_i) \quad \tilde{O}(n \log \|A\|)$$

2)

$$2-1) x := \sum_{i=0}^k x_i \cdot p^i$$

2-2) rational reconstruction on  $x$

Not satisfying : computation of block minpoly. at each steps

How to avoid the computation of the block minimal polynomial?

## Alternative to Block Wiedemann

Express the inverse of the sparse matrix through structured forms.

↔ **block Hankel structure**

We consider

$$K_U = \begin{bmatrix} U \\ UA \\ \dots \\ UA^{N-1} \end{bmatrix}, \quad K_V = [V | AV | \dots | A^{N-1}V]$$

## Alternative to Block Wiedemann

Express the inverse of the sparse matrix through structured forms.

↪ block Hankel structure

We consider

$$K_U = \begin{bmatrix} U \\ UA \\ \dots \\ UA^{N-1} \end{bmatrix}, \quad K_V = [V | AV | \dots | A^{N-1}V]$$

than we have

$$K_U A K_V = H \text{ with } H \text{ a block Hankel matrix.}$$

Now, we can express  $A^{-1}$  with

$$A^{-1} = K_V H^{-1} K_U$$

# Alternative to Block Wiedemann

Nice property on block Hankel matrix inverse [Gohberg, Krupnik 1972, Labahn, Koo Choi, Cabay 1990]

$$H^{-1} = \underbrace{\begin{bmatrix} * & \dots & * \\ \vdots & \ddots & \vdots \\ * & & \end{bmatrix}}_{H_1} \underbrace{\begin{bmatrix} * & \dots & * \\ & \ddots & \vdots \\ & & * \end{bmatrix}}_{T_1} - \underbrace{\begin{bmatrix} * & \dots & * \\ \vdots & \ddots & \vdots \\ * & & \end{bmatrix}}_{H_2} \underbrace{\begin{bmatrix} * & \dots & * \\ & \ddots & \vdots \\ & & * \end{bmatrix}}_{T_2}$$

where  $H_1, H_2$  are block Hankel matrices and  $T_1, T_2$  are block Toeplitz matrices

# Alternative to Block Wiedemann

Nice property on block Hankel matrix inverse [Gohberg, Krupnik 1972, Labahn, Koo Choi, Cabay 1990]

$$H^{-1} = \underbrace{\begin{bmatrix} * & \dots & * \\ \vdots & \ddots & \vdots \\ * & & \end{bmatrix}}_{H_1} \underbrace{\begin{bmatrix} * & \dots & * \\ & \ddots & \vdots \\ & & * \end{bmatrix}}_{T_1} - \underbrace{\begin{bmatrix} * & \dots & * \\ \vdots & \ddots & \vdots \\ * & & \end{bmatrix}}_{H_2} \underbrace{\begin{bmatrix} * & \dots & * \\ & \ddots & \vdots \\ & & * \end{bmatrix}}_{T_2}$$

where  $H_1, H_2$  are block Hankel matrices and  $T_1, T_2$  are block Toeplitz matrices

Block coefficients in  $H_1, H_2, T_1, T_2$  come from Hermite Pade approximant on Block coefficients of  $H$  [Labahn, Koo Choi, Cabay 1990].

Complexity of  $H^{-1}$  reduces to polynomial matrix multiplication [Giorgi, Jeannerod, Villard 2003].

# Alternative to Block Wiedemann

Scheme to compute  $A^{-1}b$  :

1)

1-1) compute  $S := \{UA^{i+1}V\}_{i=0..N-1} \bmod p$

1-2) compute  $H^{-1} \bmod p$  from  $S$

1-3)  $r = b$

2) for  $i := 0$  to  $k$

2-1)  $x_i := K_V \cdot H^{-1} \cdot K_U \cdot r \bmod p$

2-2)  $r := (1/p)(r - A \cdot x_i)$

3)

3-1)  $x := \sum_{i=0}^k x_i \cdot p^i$

3-2) rational reconstruction on  $x$

# Alternative to Block Wiedemann

Scheme to compute  $A^{-1}b$  :

1)

1-1) compute  $S := \{UA^{i+1}V\}_{i=0..N-1} \bmod p$   $O((n^2 k \log \|A\|))$

1-2) compute  $H^{-1} \bmod p$  from  $S$   $O^{\sim}(nk^2 \log \|A\|)$

1-3)  $r = b$

2) for  $i := 0$  to  $k$

$k = O^{\sim}(n)$

2-1)  $x_i := K_V \cdot H^{-1} \cdot K_U \cdot r \bmod p$   $O^{\sim}((n^2 + nk) \log \|A\|)$

2-2)  $r := (1/p)(r - A \cdot x_i)$   $O^{\sim}(n \log \|A\|)$

3)

3-1)  $x := \sum_{i=0}^k x_i \cdot p^i$

3-2) rational reconstruction on  $x$

# Alternative to Block Wiedemann

Scheme to compute  $A^{-1}b$  :

1)

1-1) compute  $S := \{UA^{i+1}V\}_{i=0..N-1} \bmod p$   $O((n^2 k \log \|A\|))$

1-2) compute  $H^{-1} \bmod p$  from  $S$   $O^{\sim}(nk^2 \log \|A\|)$

1-3)  $r = b$

2) for  $i := 0$  to  $k$

$k = O^{\sim}(n)$

2-1)  $x_i := K_V \cdot H^{-1} \cdot K_U \cdot r \bmod p$   $O^{\sim}((n^2 + nk) \log \|A\|)$

2-2)  $r := (1/p)(r - A \cdot x_i)$   $O^{\sim}(n \log \|A\|)$

3)

3-1)  $x := \sum_{i=0}^k x_i \cdot p^i$

3-2) rational reconstruction on  $x$



## Applying block Krylov space

$$K_V = [V | AV | \dots | A^{N-1}V]$$

↓

$$K_V = [V | \quad ] + A [ \quad | V | \quad ] + \dots + A^{N-1} [ \quad \quad | V ]$$

Applying  $K_V$  to a vector corresponds to :

- $N - 1$  linear combinations of columns of  $V$
- $N - 1$  applications of  $A$

## Applying block Krylov space

$$K_V = [V | AV | \dots | A^{N-1}V]$$

↓

$$K_V = [V | \quad ] + A [ \quad | V | \quad ] + \dots + A^{N-1} [ \quad \quad | V ]$$

Applying  $K_V$  to a vector corresponds to :

- $N - 1$  linear combinations of columns of  $V$
- $N - 1$  applications of  $A$

$$O(Nkn \log \|A\|)$$

$$O(nN \log \|A\|)$$

## Applying block Krylov space

$$K_V = [V | AV | \dots | A^{N-1}V]$$

↓

$$K_V = [V | \quad ] + A [ \quad | V | \quad ] + \dots + A^{N-1} [ \quad \quad | V ]$$

Applying  $K_V$  to a vector corresponds to :

- $N - 1$  linear combinations of columns of  $V$
- $N - 1$  applications of  $A$

$O(Nkn \log \|A\|)$

How to improve the complexity ?

## Applying block Krylov space

$$K_V = [V | AV | \dots | A^{N-1}V]$$

↓

$$K_V = [V | \quad ] + A [ \quad | V | \quad ] + \dots + A^{N-1} [ \quad \quad | V ]$$

Applying  $K_V$  to a vector corresponds to :

- $N - 1$  linear combinations of columns of  $V$
- $N - 1$  applications of  $A$

$O(Nkn \log \|A\|)$

How to improve the complexity ?

⇒ using special block projections  $U$  and  $V$



# Experimental algorithm

Scheme to compute  $A^{-1}b$  :

1)

*1-1) choose at random special  $U$  and  $V$*

*1-1) compute  $S := \{UA^{i+1}V\}_{i=0..N-1} \bmod p$*

*1-2) compute  $H^{-1} \bmod p$  from  $S$*

*1-3)  $r = b$*

2) for  $i := 0$  to  $k$

*2-1)  $x_i := K_V \cdot H^{-1} \cdot K_U \cdot r \bmod p$*

*2-2)  $r := (1/p)(r - A \cdot x_i)$*

3)

*3-1)  $x := \sum_{i=0}^k x_i \cdot p^i$*

*3-2) rational reconstruction on  $x$*

# Experimental algorithm

Scheme to compute  $A^{-1}b$  :

1)

1-1) choose at random special  $U$  and  $V$

1-1) compute  $S := \{UA^{i+1}V\}_{i=0..N-1} \bmod p$   $O(n^2 \log \|A\|)$

1-2) compute  $H^{-1} \bmod p$  from  $S$   $O(\tilde{nk}^2 \log \|A\|)$

1-3)  $r = b$

2) for  $i := 0$  to  $k$

$k = O(\tilde{n})$

2-1)  $x_i := K_V \cdot H^{-1} \cdot K_U \cdot r \bmod p$   $O(\tilde{((nN + nk) \log \|A\|)})$

2-2)  $r := (1/p)(r - A \cdot x_i)$   $O(\tilde{(n \log \|A\|)})$

3)

3-1)  $x := \sum_{i=0}^k x_i \cdot p^i$

3-2) rational reconstruction on  $x$

# Experimental algorithm

Scheme to compute  $A^{-1}b$  :

1)

1-1) choose at random special  $U$  and  $V$

1-1) compute  $S := \{UA^{i+1}V\}_{i=0..N-1} \bmod p$   $O(n^2 \log \|A\|)$

1-2) compute  $H^{-1} \bmod p$  from  $S$   $O^{\sim}(nk^2 \log \|A\|)$

1-3)  $r = b$

2) for  $i := 0$  to  $k$

$k = O^{\sim}(n)$

2-1)  $x_i := K_V \cdot H^{-1} \cdot K_U \cdot r \bmod p$   $O^{\sim}((nN + nk) \log \|A\|)$

2-2)  $r := (1/p)(r - A \cdot x_i)$   $O^{\sim}(n \log \|A\|)$

3)

3-1)  $x := \sum_{i=0}^k x_i \cdot p^i$

3-2) rational reconstruction on  $x$

taking  $N = k = \sqrt{n}$  gives a complexity of  $O^{\sim}(n^{2.5} \log \|A\|)$



# Prototype implementation

LinBox project (Canada-France-USA) : [www.linalg.org](http://www.linalg.org)

Our tools :

- ▶ BLAS-based matrix multiplication and matrix-vector product
- ▶ polynomial matrix arithmetic  
*Karatsuba algorithm, middle product*
- ▶ vector polynomial Evaluation/Interpolation using matrix multiplication

# Performances

use of LinBox library on Pentium 4 - 3.4Ghz, 2Go RAM.

non-singular sparse linear system with coefficients over 3 bits  
and 10 non zero elements per row.

n	400	900	1600	2500
CRT + Wiedemann	7.3s	79.2s	464s	1769s
P-adic + Wiedemann	3.1s	32.7s	185s	709s
P-adic + block Hankel	1.7s	8.9s	30s	81s
dense solver	0.5s	3.5s	13s	41s

## Performances under progress

use of LinBox library on SMP machine 64 Itanium 2 - 1.6Ghz, 128Go RAM.

non-singular sparse linear system with coefficients over 3 bits and 30 non zero elements per row.

matrix dimension 10.000.

▶  $block = 500 \hookrightarrow 10.323s \approx 2h50mn$

▶  $block = 400 \hookrightarrow 9.655s \approx 2h40mn$

▶  $block = 200 \hookrightarrow 22.966s \approx 6h20mn$

⇒ using dense solving  $4.347s \approx 1h12mn$

matrix dimension 20.000.

▶  $block = 400 \hookrightarrow 47.545s \approx 13h15mn$

# Conclusions

We provide an efficient algorithm for solving sparse integer linear system :

- ▶ improve the complexity by a factor  $\sqrt{n}$  (**heuristic**).
- ▶ allow efficiency by minimizing sparse matrix operation and maximizing BLAS use.

## **On going improvement :**

- ▶ optimize the code (use of FFT, minimize the constant)
- ▶ provide an automatic choice of block dimension
- ▶ prove conjecture on special block projection