
**PLANNC: a projected Lagrangian based
implementation for constrained
nonlinear network flow problems**

F. Javier Heredia

Dept. of Statistics and Operations Research
Universitat Politècnica de Catalunya
Barcelona, Spain

20th IFIP TC7 conference on
System Modelling and Optimization
july 23-27 2001
Trier, Germany

FIRST PART :
General overview of the method.

- The Nonlinear Network Flow problem with Side Constraints (NNC).
- The projected Lagrangian methodology.

SECOND PART :
**Optimization of the linearized
subproblems (NNLC)^k :**
The NOXCB09 package.

- The primal partitioning active-set method.
- Computational results.

THIRD PART :
**The use of the NOXCB09 library into a
projected Lagrangian methodology**

- Structure of the prototype implementation.
- The PLANNC program.
- Preliminary computational results.

CONCLUSIONS

FIRST PART :
General overview of the method.

FORMULATION OF THE (NNC) PROBLEM

$$\begin{array}{l}
 \text{(NNC)} \left\{ \begin{array}{ll}
 \min & f(x) & (1a) \\
 \text{subj. a :} & & \\
 & Ax & = r & (1b) \\
 & c(x) + \mathbf{I}_n z_n & = b_n & (1c) \\
 & Tx + \mathbf{I}_l z_l & = b_l & (1d) \\
 & & & \\
 & 0 \leq x \leq u_x & (1e) \\
 & 0 \leq z_n \leq u_{zn} & (1f) \\
 & 0 \leq z_l \leq u_{zl} & (1g)
 \end{array} \right.
 \end{array}$$

(1a) : Objective function: $f: \mathbb{R}^n \rightarrow \mathbb{R}$, $f \in \mathcal{C}^2$

(1b) : Network equations: $A \in \mathbb{R}^{m \times \tilde{n}}$, $r \in \mathbb{R}^m$

(1c,1d) : Nonlinear and linear side constraints:

$$\begin{array}{l}
 c: \mathbb{R}^{t_n} \rightarrow \mathbb{R}, c \in \mathcal{C}^2, b_n \in \mathbb{R}^{t_n} \\
 T \in \mathbb{R}^{t_l \times \tilde{n}}, \text{rang}(T) = t_l, b \in \mathbb{R}^{t_l}
 \end{array}$$

(1e,1f,1g) : Arcs and slacks capacities:

$$\begin{array}{l}
 x, u_x \in \mathbb{R}^n \\
 z_n, u_{zn} \in \mathbb{R}_n^{\tilde{t}}, \mathbf{I}_n \in \mathbb{R}^{t_n \times \tilde{t}_n} \\
 z_l, u_{zl} \in \mathbb{R}_l^{\tilde{t}}, \mathbf{I}_l \in \mathbb{R}^{t_l \times \tilde{t}_l}
 \end{array}$$

THE PROJECTED LAGRANGIAN METHOD

Algorithm A1 : Projected Lagrangian Algorithm for problem (NNC)

- Select some starting values for x^0 , λ^0 and ρ^0 .

MAJOR ITERATION

Do While x^k not optimal for (NNC)

- Linearize the nonlinear side constraints at x^k :

$$c(x) \approx c^k(x) = c(x^k) + \nabla c(x^k)(x - x^k) \quad (2)$$

- **Minor iteration:** Solve the following *Nonlinear Network flow problem with Linear side Constraints (NNLC)*^k:

$$\begin{array}{l}
 \left. \begin{array}{l}
 \text{(NNLC)}^k \left\{ \begin{array}{l}
 \min \quad \Phi^k(x) \quad (3a) \\
 \text{subj. a :} \\
 Ax \quad \quad \quad = r \quad (3b) \\
 \nabla c(x^k)x + \mathbf{I}_n z_n = b_n^k \quad (3c) \\
 Tx \quad \quad + \mathbf{I}_l z_l = b_l \quad (3d) \\
 \\
 0 \leq x \leq u_x \quad (3e) \\
 0 \leq z_n \leq u_{z_l} \quad (3f) \\
 0 \leq z_l \leq u_{z_n} \quad (3g)
 \end{array}
 \right.
 \end{array}
 \right.
 \end{array}$$

where :

$$b_n^k = b_n - c(x^k) + \nabla c(x^k)x^k$$

$$\Phi^k(x) = \Phi^k(x; \lambda^k, \dots) \text{ (Merit function)}$$

- Update x^k , λ^k and $\Phi^k(x)$ from $[x^*]^k$ and $[\lambda^*]^k$.
- Set $k := k + 1$

End Do

RELEVANT FACTS.

- **Local convergence:**

- * Robinson (1972) proved the q-quadratically local convergence of the method, if the merit function and the iterates $\{x^{k+1}, \lambda^{k+1}\}$ are taken as:

$$\begin{aligned}\Phi^k(x) &= f(x) - \lambda^{k'}(c(x) - c^k(x)) \\ \{x^{k+1}, \lambda^{k+1}\} &:= \{[x^*]^k, [\lambda^*]^k\}\end{aligned}$$

- **Computational implementations :**

- * MINOS (Murtagh & Saunders 1982).
- * Their merit function is a *modified augmented Lagrangian function* $\Phi^k(x)$:

$$\Phi^k(x; x^k, \lambda^k, \rho^k) = f(x) - \lambda^{k'}(c(x) - c^k(x)) + \frac{\rho^k}{2} \|c(x) - c^k(x)\|_2$$

- **Efficient solution of the (NNLC)^k subproblems:**

- * NOXCB09 (Heredia and Nabona 1995).
 - * Active set method specialized through primal partitioning techniques.
-

SECOND PART :
Optimization of the linearized
subproblems $(\text{NNLC})^k$:
The NOXCB09 package.

FORMULATION OF THE PROBLEM (NNLC)^k

$$\begin{array}{l}
 \text{(NNLC)} \left\{ \begin{array}{ll}
 \min & f(x) & (4a) \\
 \text{subj. a :} & Ax = r & (4b) \\
 & Tx + \mathbf{I}_z z = b & (4c) \\
 & 0 \leq x \leq u_x & (4d) \\
 & 0 \leq z \leq u_z & (4e)
 \end{array} \right.
 \end{array}$$

(4a) : Objective function.

$$f: \mathbb{R}^n \rightarrow \mathbb{R}, f \in \mathcal{C}^2$$

(4b) : Network equations

$$A \in \mathbb{R}^{m \times \tilde{n}}, r \in \mathbb{R}^m$$

(4c) : Linear side constraints

$$T \in \mathbb{R}^{t \times \tilde{n}}, \text{rang}(T) = t, b \in \mathbb{R}^t$$

(4d) : Arcs capacities

$$x, u_x \in \mathbb{R}^n$$

(4e) : Bounds of the slacks

$$z, u_z \in \mathbb{R}^{\tilde{t}}, \mathbf{I}_z \in \mathbb{R}^{t \times \tilde{t}}$$

(\tilde{t} : number of inequality side constraints)

Solution procedure for the (NNLC) subproblem.

Algorithm A2 : Active set method

Find an initial feasible solution x^0 .

Define the *basic* \mathcal{B}^0 , *superbasic* \mathcal{S}^0 and *nonbasic* \mathcal{N}_0^0 , \mathcal{N}_u^0 variables associated to the current active set at x^0 .

Compute Z^0 , $g_z^0 = Z^{0'} \nabla f(x^0)$ and the Lagrange multipliers of the nonbasic variables σ_0^0 and σ_u^0 .

Set $k := 0$

Do

Relax some nonbasic variable from its active bound.

Update \mathcal{B}^k , \mathcal{S}^k , \mathcal{N}^k , Z^k and g_z^k .

Do While $\|g_z\| > \epsilon_G^k$

Compute a feasible descent direction :

$$H_z^k p_z^k = -g_z^k \quad ; \quad p^k := Z^k p_z^k$$

Find the maximal step length: $\bar{\alpha} = \min\{\bar{\alpha}_B, \bar{\alpha}_S\}$.

Linesearch: $\alpha^{*k} = \operatorname{argmin}_{0 < \alpha \leq \bar{\alpha}} \{f(x^k + \alpha p^k)\}$.

Update the current solution: $x^{k+1} := x^k + \alpha^{*k} p^k$.

If $\alpha^{*k} = \bar{\alpha}$ then

Update \mathcal{B}^k , \mathcal{S}^k , \mathcal{N}^k

End If

Update Z^k and g_z^k .

$k := k + 1$

End Do

Compute the Lagrange multipliers estimates σ_0^k , σ_u^k .

Until $\sigma_0^k \geq -\epsilon_o$ and $\sigma_u^k \leq \epsilon_o$

STRUCTURE OF THE PROBLEM (NNLC)

- Equality constraint matrix.

$$M = \begin{matrix} m \\ t \end{matrix} \left\{ \begin{matrix} \tilde{n} & \tilde{t} \\ A & \mathbf{0} \\ T & \mathbf{I}_z \end{matrix} \right\} = \begin{matrix} m & s & n - m - s \\ B & S & N \end{matrix}$$

- Superbasic S , non-basic N and basic B matrices.

$$S = \begin{matrix} s_x & s_z \\ A_s & \mathbf{0} \\ T_s & \mathbf{I}_s \end{matrix} \begin{matrix} m \\ t \end{matrix} ; \quad N = \begin{matrix} |\mathcal{N}_x| & |\mathcal{N}_z| \\ A_N & \mathbf{0} \\ T_N & \mathbf{I}_N \end{matrix} \begin{matrix} m \\ t \end{matrix}$$

$$B = \begin{matrix} m & c_x & c_z \\ A_A & A_c & \mathbf{0} \\ T_A & T_c & \mathbf{I}_c \end{matrix} \begin{matrix} m \\ t \end{matrix}$$

PRIMAL PARTITIONING

- Every basis of the (NNLC) problem may be placed in the following form (Kennington and Helgason 1980):

$$B = \begin{array}{c} \begin{array}{|c|c|c|} \hline & m & c_x & c_z \\ \hline & A_{\mathcal{A}} & A_c & \mathbf{0} \\ \hline & T_{\mathcal{A}} & T_c & \mathbf{I}_c \\ \hline & \text{Key} & \text{Non Key} & \\ \hline \end{array} & \begin{array}{l} m \\ t \end{array} \\ \end{array} \quad (5)$$

where $A_{\mathcal{A}}$ is a basis (spanning tree) of the network matrix A

- The systems of equations $Bv = w$ and $v'B = w$ can be computed solving:

- * Systems of equations with the matrix $A_{\mathcal{A}}$, which can be computed efficiently through network flow techniques.

- * Systems of equations with the general (dense) matrix Q .

$$Q = [T_c \mid \mathbf{I}_c] - T_{\mathcal{A}} A_{\mathcal{A}}^{-1} [A_c \mid \mathbf{0}] \in \mathbb{R}^{t \times t}$$

(working basis)

COMPUTATIONS WITH THE BASIC MATRIX B

- Resolution of the linear system $Bw = v$:

$$w = \begin{bmatrix} w_A \\ - \\ w_C \end{bmatrix} = B^{-1} \begin{bmatrix} v_A \\ - \\ v_T \end{bmatrix} = \begin{bmatrix} A_A^{-1}(v_A - [A_c | \mathbf{0}]Q^{-1}(v_T - T_A A_A^{-1}v_A)) \\ \hline Q^{-1}(v_T - T_A A_A^{-1}v_A) \end{bmatrix} \quad (6a)$$

$$= \begin{bmatrix} A_A^{-1}v_A - [\Theta_c | \mathbf{0}]Q^{-1}(v_T - T_A A_A^{-1}v_A) \\ \hline Q^{-1}(v_T - T_A A_A^{-1}v_A) \end{bmatrix} \quad (6b)$$

Where $A_A^{-1}A_c = \Theta_c$, is the matrix of the basic cycles of the non-key arcs.

- Algorithm to compute $Bw = v$:

Algorithm A3 :

$$w = B^{-1}v, \text{ without } \overset{\circ}{C}.$$

1	$\gamma_1 \xleftarrow{F.X.} A_A^{-1}v_A$
2	$\gamma_2 \longleftarrow v_T - T_A \gamma_1$
3	$w_C \longleftarrow Q^{-1}\gamma_2$
4	$\gamma_2 \longleftarrow v_A - [A_c \mathbf{0}]w_C$
5	$w_A \xleftarrow{F.X.} A_A^{-1}\gamma_2$

Algorithm A4 :

$$w = B^{-1}v, \text{ with } \overset{\circ}{C}.$$

1	$\gamma_1 \xleftarrow{F.X.} A_A^{-1}v_A$
2	$\gamma_2 \longleftarrow v_T - T_A \gamma_1$
3	$w_C \longleftarrow Q^{-1}\gamma_2$
4	$w_A \xleftarrow{A4.1} \gamma_1 - \Theta_c w_C$

$$\text{Product } w = Zv = \begin{bmatrix} -B^{-1}S \\ \hline \mathbf{I} \\ \hline \mathbf{0} \end{bmatrix} v$$

- **Option A:** first $\gamma_A = -Sv$ and then $w = B^{-1}\gamma$

$$Sv = \begin{bmatrix} A_S & \mathbf{0} \\ T_S & \mathbf{I}_S \end{bmatrix} \begin{bmatrix} v_x \\ v_z \end{bmatrix} = \begin{bmatrix} A_S v_x \\ \hline T_S v_x + \mathbf{I}_S v_z \end{bmatrix} \quad (7)$$

Algorithm A5 : $-B^{-1}(Sv)$

$$\begin{array}{l} \boxed{1} \quad \gamma_A \longleftarrow -A_S v_x \\ \boxed{2} \quad \gamma_T \longleftarrow -T_S v_x - \mathbf{I}_S v_z \\ \boxed{3} \quad w_B \xleftarrow{A5} B^{-1}\gamma \end{array}$$

- **Option B:** to develop $B^{-1}S$.

$$w_B = \begin{bmatrix} w_A \\ \hline w_C \end{bmatrix} = -B^{-1}Sv = \begin{bmatrix} -\Theta_S v_x - [\Theta_C | \mathbf{0}] Q^{-1} (T_A \Theta_S v_x - T_S v_x - \mathbf{I}_S v_z) \\ \hline Q^{-1} (T_A \Theta_S v_x - T_S v_x - \mathbf{I}_S v_z) \end{bmatrix} \quad (8)$$

Algorithm A6 : $w_B = Zv$.

$$\begin{array}{l} \boxed{1} \quad \gamma_1 \xleftarrow{A4.2} \Theta_S v_x \\ \boxed{2} \quad \gamma_2 \longleftarrow T_A \gamma_1 - T_S v_x - \mathbf{I}_S v_z \\ \boxed{3} \quad [w'_{c_x} \mid w'_{c_z}] \longleftarrow -Q^{-1} \gamma_2 \\ \boxed{4} \quad w_A \xleftarrow{A4.1} \gamma_1 - \Theta_C w_{c_x} \end{array}$$

OTHER IMPLEMENTATION ISSUES

- **Linesearch :**

- * Bertsekas method (Bertsekas (1982), Toint & Tuyttens (1991)).

- * Backtracking linesearch with quadratic and cubic fit.

- **Descent direction computation:** $\tilde{H}_z p_z = -g_z(x)$

- * (TNM) Truncated Newton Method (Dembo & Steihaug (1983), Toint & Tuyttens (1991)).

- * (QNM) Quasi-Newton method (Murtagh & Saunders (1978)).

- **Working basis update $[Q^{-1}]^k$:** cada cert nombre d'iteracions

- * **Product form of the inverse :**

$$[Q^{-1}]^k = \prod_{t \geq i \geq 1} E_r^i$$

- ▷ P^3 algorithm of Hellerman & Rarick : $[Q^{-1}]^k = R \left(\prod_{t \geq i \geq 1} \tilde{E}_r^i \right) P$

- ▷ Partial pivoting.

- * **LU factorization.**

- ▷ Subroutines F01BRF and F04AXF (NAG library).

EIO/UPC problem tests

• **110 problemes (NNLC).**

* **DIMACS problems:**

▷ Random generators of linear network flow problems (FL) :

Rmfgen (Goldfarb, Grigoriadis).

Grid-on-Torus (Goldberg).

Gridgen (Lee & Orlin).

▷ Addition of side constraints to the DIMACS problems: *Di2no*.

▷ Objective functions :

Namur (Toint & Tuyttens).

EIO1.

* **Real Short term Hydro-Thermal Coordination problems.**

• **Characteristics of the EIO/UPC test collection:**

* 1524 arcs, 360 nodes → 23832 arcs and 6589 nodes.

* Ratio (# of side const./ # network equations) : 1% → 100%

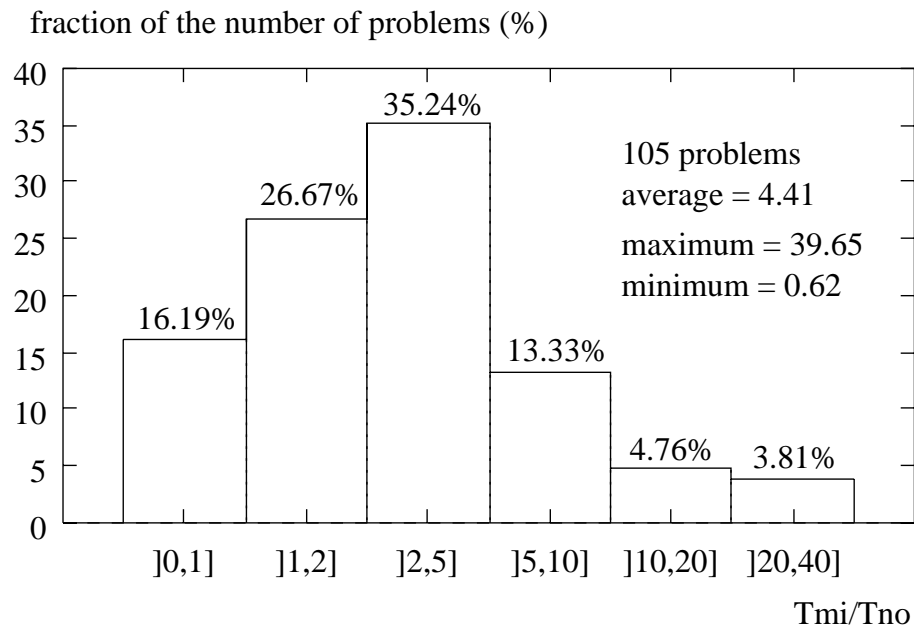
* # of nonzero elements of the S.C. : 0.02% → 10%

* # of superbasic variables at the optimal solution: 0 → 3352

COMPUTATIONAL RESULTS

(summary)

- **Efficiency of NOXCB09 vs. MINOS 5.3:**



- **Dependency of the efficiency on the size of the problem:**

Scale	Maximal dimensions			$t_{\text{MI}}/t_{\text{NO}}$		
	arcs	nodes	# S.C.	Average	Maximum	Minimum
Large	18000	3000	750	5.10	17.96	0.84
Medium	8064	2479	840	2.87	10.00	0.92
Small	2256	697	240	1.34	2.47	0.62

* *The efficiency of NOXCB09 seems to increase with the dimension of the (NNLC) problem.*

THIRD PART :
**The use of the NOXCB09 library into a
projected Lagrangian methodology**

STRUCTURE OF THE IMPLEMENTATION

Algorithm A7 : a NOXCB09 based Projected Lagrangian method

Select some starting values for x^0 , λ^0 and ρ^0 .

Set $k := 0$

Do

Linearize the nonlinear side constraints at x^k .

Do

Solve (NNLC)^k with NOXCB09 and the merit function $\Phi^k(x; x^k, \lambda^k, \rho^k)$. This optimization provides $[x^*]^k$ and $[\lambda^*]^k$.

If (NNLC)^k infeasible then

Relax the RHS of the linearized n.s.c. b_n^k .

End If

Until (NNLC)^k feasible

Find new values of x^k and λ^k from $[x^*]^k$ and $[\lambda^*]^k$.

Update de penalty parameter ρ^k , if necessary.

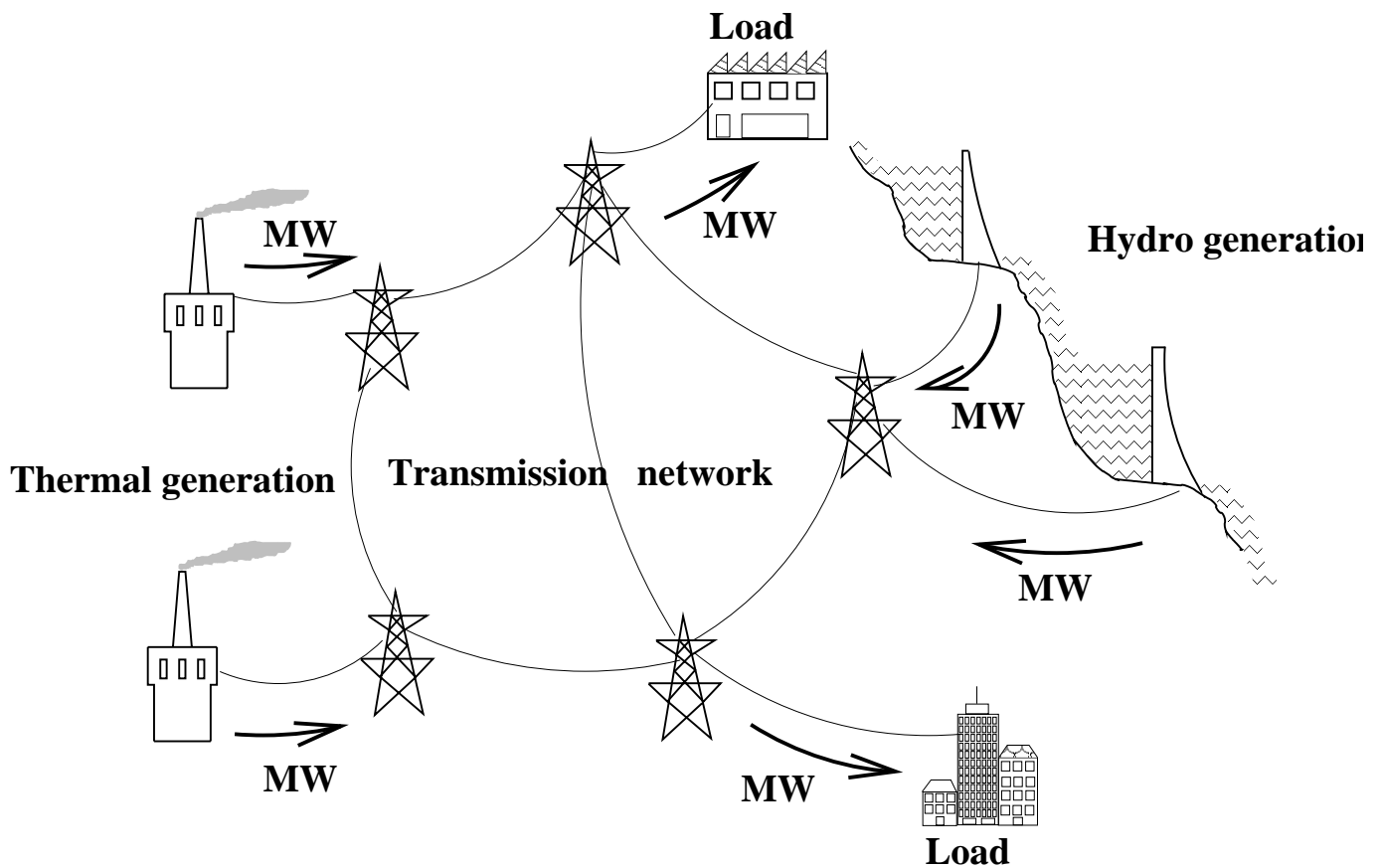
Set $k := k + 1$

Until x^k optimal for (NNC)

- **Updates rules** the update rules follows those suggested by Murtagh & Saunders (1982) and implemented in MINOS.

PROTOTYPE IMPLEMENTATION

- **PLANNC** , a projected Lagrangian NOXCB09 based package to solve the (NNC) problem.
- Applied to solve the **Short-Term Hydrothermal Scheduling** problem:



PRELIMINARY COMPUTATIONAL RESULTS

Table I : characteristics of the (NNC) problems.

Problem ident.	Power system size					(NNC) problem size			
	Nr	Nu	Nm	Nb	Ni	arcs	nodes	n.s.c.	l.s.c.
A24x	3	4	6	5	24	888	290	24	96
B24x	6	4	6	5	24	1152	355	24	96
B48x	6	4	6	5	48	2256	697	48	192

Table II : Computational results.

Problem ident.	Major/Minor iter		CPU (s) ¹			millisec/iter ¹		
	PLANNC	MINOS	PLANNC	MINOS	ratio	PLANNC	MINOS	ratio
A24x	3/2384	53/2030	2.3	4.8	2.08	0.96	2.3	2.4
B24x	3/3947	99/3855	3.1	15.3	4.93	0.78	3.9	5.0
B48x	3/7988	119/4656	6.4	36.2	5.65	0.80	7.7	9.6

¹ SUN UltraSparc 2

CONCLUSIONS

- A projected Lagrangian methodology for the **(NNC)** problem has been proposed.
 - This methodology exploits the network structure of the problem applying the **NOXCB09** package to solve the linearized subproblems **(NNLC)^k**. The efficiency of this code has been proved through extensive computational tests.
 - A prototype implementation, **(PLANNC)**, of the proposed specialized projected Lagrangian method, has been implemented and used to solve a limited set of **(NNC)** problems.
 - The preliminary computational results seems to corroborate the efficiency of the method, although more computational evidence is needed. These results also reveals the possibility of increasing the convergence rate with a more careful implementation.
-