Methods for the fast and approximate solution of mixed LCPs

José Luis Morales [1] Jorge Nocedal [2] Mikhail Smelyanskiy [3]

[1]Departamento de Matemáticas, ITAM
[2]ECSD, Northwestern University.
[3]INTEL Corporation.

Eighth US-Mexico Workshop on Optimization and its Applications, Huatulco, Oax., 2007.

Aims of the talk

In this talk we explore the solution of mixed linear complementarity problems (mLCP).
The focus is on the fast and approximate solution of medium to large scale instances arising in the computer game industry.
Methods used in our study:

1. iterative methods based on splittings and projections PSOR. [1].
2. interior point methods IPM. [2]
3. 1) + subspace minimization [3]

[1]Game industry

[2]M.C. Ferris, A.J. Wathen and P. Armand. Limited Memory Solution of Bound Constrained ... 2006

[3]M. Kocvara et al. A new method ... *Numerische Mathematik*, 1996

Aims of the talk

In this talk we explore the solution of mixed linear complementarity problems (mLCP).
The focus is on the fast and approximate solution of medium to large scale instances arising in the computer game industry.
Methods used in our study:

1. iterative methods based on splittings and projections PSOR. [1].
2. interior point methods IPM. [2]
3. 1) + subspace minimization [3]

[1] Game industry

[2] M.C. Ferris, A.J. Wathen and P. Armand. Limited Memory Solution of Bound Constrained ... 2006

[3] M. Kocvara et al. A new method ... *Numerische Mathematik*, 1996

Aims of the talk

In this talk we explore the solution of mixed linear complementarity problems (mLCP).
The focus is on the fast and approximate solution of medium to large scale instances arising in the computer game industry.
Methods used in our study:

1. iterative methods based on splittings and projections PSOR. [1].
2. interior point methods IPM. [2]
3. 1) + subspace minimization [3]

[1] Game industry

[2] M.C. Ferris, A.J. Wathen and P. Armand. Limited Memory Solution of Bound Constrained ... 2006

[3] M. Kocvara et al. A new method ... *Numerische Mathematik* 1996

Aims of the talk

In this talk we explore the solution of mixed linear complementarity problems (mLCP).
The focus is on the fast and approximate solution of medium to large scale instances arising in the computer game industry.
Methods used in our study:

1. iterative methods based on splittings and projections PSOR. [1].
2. interior point methods IPM. [2]
3. 1) + subspace minimization [3]

[1] Game industry

[2] M.C. Ferris, A.J. Wathen and P. Armand. Limited Memory Solution of Bound Constrained ... 2006

[3] M. Kocvara et al. A new method ... *Numerische Mathematik, 1996.*

mLCP :

$$\begin{aligned} Mz + q &= w \\ z_i w_i &= 0, \\ z_i,\ w_i &\geq 0, \quad i \in \mathcal{I} \\ w_j &= 0, \quad j \notin \mathcal{I} \end{aligned}$$

where

- M is symmetric positive definite $n \times n$ matrix;
- q, z, w are n-dimensional vectors.
- $\mathcal{I} \subset \{1, 2, \ldots, n\}$. $\#\mathcal{I} = 2n/3$

Outline

Outline

Outline

Outline

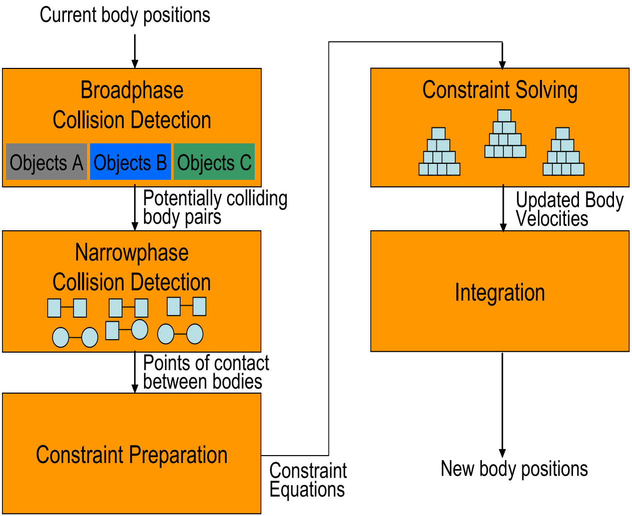
Outline

Outline

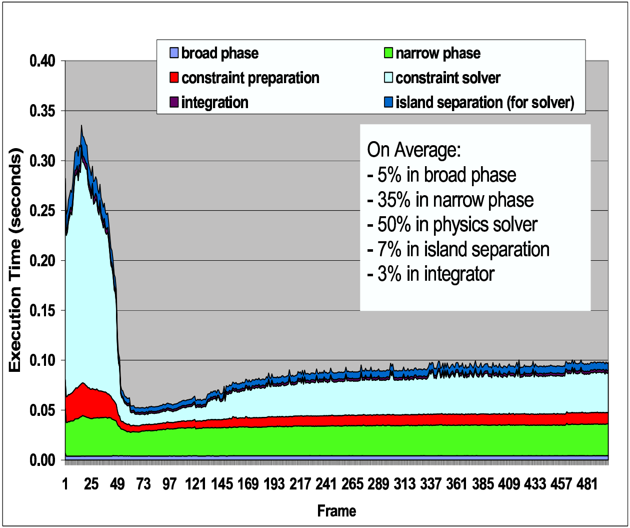
Challenges

- mLCP's in computer game industry: modeling contact forces in physical simulation.
- Limited amount of resources: a) CPU time (real time); b) memory; c) stability; d) low accuracy.
- Modern systems consist of a physical model that takes into account interactions between pairs of bodies:
  - 1 inequality that amounts for contact.
  - 2 equalities that model friction between the bodies.
- More realism in games $\Longrightarrow$ more complex models in terms of physics $\Longrightarrow$ Faster numerical algorithms.

Physical Simulation Pipeline



Time breakdown of Physical Simulation.



Press Esc to start the game.

Structure of an mLCP.

- $M = JDJ^T + C$. J is rectangular.
  Rows correspond to interactions. Columns correspond to bodies.
- D is block-diagonal.
  Incorporates inertia into the model.
- C is diagonal with positive entries. C has physical meaning.

$$(JDJ^T + C)\lambda = c.$$

- λ is the vector of contact forces. Some of its components are bounded below.
- Test problems.
  Open Dynamics Engine (ODE). Real games simulation (castle destruction demo).

Structure of an mLCP.

- $M = JDJ^T + C$. J is rectangular.
  Rows correspond to interactions. Columns correspond to bodies.
- D is block-diagonal.
  Incorporates inertia into the model.
- C is diagonal with positive entries. C has physical meaning.

$$(JDJ^T + C)\lambda = c.$$

- λ is the vector of contact forces. Some of its components are bounded below.
- Test problems.
  Open Dynamics Engine (ODE). Real games simulation (castle destruction demo).

Structure of an mLCP.

- $M = JDJ^T + C$. J is rectangular.
  Rows correspond to interactions. Columns correspond to bodies.
- D is block-diagonal.
  Incorporates inertia into the model.
- C is diagonal with positive entries. C has physical meaning.

$$(JDJ^T + C)\lambda = c.$$

- λ is the vector of contact forces. Some of its components are bounded below.
- Test problems.
  Open Dynamics Engine (ODE). Real games simulation (castle destruction demo).

Structure of an mLCP.

- $M = JDJ^T + C$. J is rectangular.
  Rows correspond to interactions. Columns correspond to bodies.
- D is block-diagonal.
  Incorporates inertia into the model.
- C is diagonal with positive entries. C has physical meaning.

$$(JDJ^T + C)\lambda = c.$$

- λ is the vector of contact forces. Some of its components are bounded below.
- Test problems.
  Open Dynamics Engine (ODE). Real games simulation (castle destruction demo).

Structure of an mLCP.

- $M = JDJ^T + C$. J is rectangular.
  Rows correspond to interactions. Columns correspond to bodies.
- D is block-diagonal.
  Incorporates inertia into the model.
- C is diagonal with positive entries. C has physical meaning.

$$(JDJ^T + C)\lambda = c.$$

- λ is the vector of contact forces. Some of its components are bounded below.
- Test problems.
  Open Dynamics Engine (ODE). Real games simulation (castle destruction demo).

Structure of an mLCP.

- $M = JDJ^T + C$. J is rectangular.
  Rows correspond to interactions. Columns correspond to bodies.
- D is block-diagonal.
  Incorporates inertia into the model.
- C is diagonal with positive entries. C has physical meaning.

$$(JDJ^T + C)\lambda = c.$$

- λ is the vector of contact forces. Some of its components are bounded below.
- Test problems.
  Open Dynamics Engine (ODE). Real games simulation (castle destruction demo).

Reformulating the mLCP as a QP

M is symmetric positive definite, thus from the formulation

$$\begin{aligned} M_{11}z_1 + M_{12}z_2 + q_1 &= 0, \\ M_{12}^T z_1 + M_{22}z_2 + q_2 &= w_2, \\ z_2^T w_2 &= 0, \\ (z_2, w_2) &\geq 0, \end{aligned} \qquad M = \begin{bmatrix} M_{11} & M_{12} \\ M_{12}^T & M_{22} \end{bmatrix}$$

we have that

$$z_1 = -M_{11}^{-1}(q_1 + M_{12}z_2),$$

and by substituting z_1 into the second block we get

$$(M_{22} - M_{12}^T M_{11}^{-1} M_{12})z_2 + q_2 - M_{12}^T M_{11}^{-1} q_1 = w_2.$$

Reformulating the mLCP as a QP

M is symmetric positive definite, thus from the formulation

$$\begin{aligned} M_{11}z_1 + M_{12}z_2 + q_1 &= 0, \\ M_{12}^T z_1 + M_{22}z_2 + q_2 &= w_2, \\ z_2^T w_2 &= 0, \\ (z_2, w_2) &\geq 0, \end{aligned} \qquad M = \begin{bmatrix} M_{11} & M_{12} \\ M_{12}^T & M_{22} \end{bmatrix}$$

we have that

$$z_1 = -M_{11}^{-1}(q_1 + M_{12}z_2),$$

and by substituting z_1 into the second block we get

$$(M_{22} - M_{12}^T M_{11}^{-1} M_{12})z_2 + q_2 - M_{12}^T M_{11}^{-1} q_1 = w_2.$$

Now with the definitions

$$\bar{M} = M_{22} - M_{12}^T M_{11}^{-1} M_{12}, \qquad \bar{q} = q_2 - M_{12}^T M_{11}^{-1} q_1,$$

and the conditions

$$z_2^T w_2 = 0, \qquad z_2 \geq 0, \quad w_2) \geq 0,$$

we obtain the KKT optimality conditions of the strictly convex quadratic programming problem

$$\begin{array}{ll} \text{minimize} & q(z_2) = \frac{1}{2} z_2 \bar{M} z_2 + \bar{q}^T z_2 \\ \text{subject to} & z_2 \geq 0. \end{array}$$

Now with the definitions

$$\bar{M} = M_{22} - M_{12}^T M_{11}^{-1} M_{12}, \qquad \bar{q} = q_2 - M_{12}^T M_{11}^{-1} q_1,$$

and the conditions

$$z_2^T w_2 = 0, \qquad z_2 \geq 0, \quad w_2) \geq 0,$$

we obtain the KKT optimality conditions of the strictly convex quadratic programming problem

$$\begin{array}{ll} \text{minimize} & q(z_2) = \dfrac{1}{2} z_2 \bar{M} z_2 + \bar{q}^T z_2 \\ \text{subject to} & z_2 \geq 0. \end{array}$$

Outline

Characteristics of the test problems

prob	n_b	$n_c = n$	nz(J)	nz(DJ^T)	nz(M)	cond(M)
1	7	18	199	204	162	5.83e+01
2	8	45	416	434	779	2.92e+03
3	8	48	423	441	868	2.38e+03
4	235	1 044	9 283	9 718	14 211	4.58e+04
5	449	1 821	18 447	18 932	28 010	4.22e+04
6	907	5 832	65 084	65 565	176 735	5.11e+07
7	948	7 344	82 038	82 646	269 765	9.02e+07
8	966	8 220	92 155	93 076	368 604	9.19e+07
9	976	8 745	97 757	98 488	373 848	6.45e+07
10	977	9 534	106 861	107 977	494 118	1.03e+08

Outline

PSOR, [4]

The PSOR method is associated with the splitting

$$M = B + E, \quad B = L + \omega^{-1} D_M.$$

where L is the strict lower triangular part of M, D_M is the diagonal part of M, and $\omega \in (0, 2)$ is the overrelaxation parameter.
Iterations of PSOR can be written as follows

$$z_i^{k+1} = \max(0, z_i^k - \omega M_{ii}^{-1}(q_i + \sum_{j<i} M_{ij} z_j^{k+1} + \sum_{j \geq i} M_{ij} z_j^k)), \quad i = 1, 2, \ldots$$

where k denotes iteration and i denotes components

[4] R.W. Cottle, J-S Pang, and R.E. Stone. The Linear Complementarity Problem. Academic Press, 1992.

PSOR, [4]

The PSOR method is associated with the splitting

$$M = B + E, \quad B = L + \omega^{-1} D_M.$$

where L is the strict lower triangular part of M, D_M is the diagonal part of M, and $\omega \in (0, 2)$ is the overrelaxation parameter.
Iterations of PSOR can be written as follows

$$z_i^{k+1} = \max(0, z_i^k - \omega M_{ii}^{-1}(q_i + \sum_{j<i} M_{ij} z_j^{k+1} + \sum_{j \geq i} M_{ij} z_j^k)), \quad i = 1, 2, \ldots$$

where k denotes iteration and i denotes components

[4] R.W. Cottle, J-S Pang, and R.E. Stone. The Linear Complementarity Problem. Academic Press, 1992.

IPM, [5]

At every iteration of the interior point method a system of linear equations

$$M^k p^k = b^k;$$

is solved to provide the search direction p^k.

$$M^k = \begin{bmatrix} M_{11} & M_{12} \\ M_{12}^T & M_{22} + W_2^k [Z_2^k]^{-1} \end{bmatrix},$$

Since M^k is spd, the systems are solved by means of the sparse Cholesky factorization.

[5] S.J. Wright. Primal-Dual Interior-Point Methods. SIAM, Philadelphia, Pennsylvania, 1997.

IPM, [5]

At every iteration of the interior point method a system of linear equations

$$M^k p^k = b^k;$$

is solved to provide the search direction p^k.

$$M^k = \begin{bmatrix} M_{11} & M_{12} \\ M_{12}^T & M_{22} + W_2^k [Z_2^k]^{-1} \end{bmatrix},$$

Since M^k is spd, the systems are solved by means of the sparse Cholesky factorization.

[5] S.J. Wright. Primal-Dual Interior-Point Methods. SIAM, Philadelphia, Pennsylvania, 1997.

PSOR + Subspace minimization

Algorithm

1. Perform some iterations of PSOR $\Longrightarrow \mathcal{A}$.
2. Check convergence.
3. Eliminate active variables and their corresponding rows from original matrix: $\Longrightarrow \bar{M}$.
4. Solve $\bar{M}\bar{z} + \bar{q} = 0$ for the remaining (free) variables.
5. Project the corresponding part of the solution onto the positive orthant.
6. Update active set: $\Longrightarrow \mathcal{A}$.
7. Repeat 3-6 until consistency
8. Go to step 1.

PSOR + Subspace minimization

Algorithm

1. Perform some iterations of PSOR $\Longrightarrow \mathcal{A}$.
2. Check convergence.
3. Eliminate active variables and their corresponding rows from original matrix: $\Longrightarrow \bar{M}$.
4. Solve $\bar{M}\bar{z} + \bar{q} = 0$ for the remaining (free) variables.
5. Project the corresponding part of the solution onto the positive orthant.
6. Update active set: $\Longrightarrow \mathcal{A}$.
7. Repeat 3-6 until consistency
8. Go to step 1.

PSOR + Subspace minimization

Algorithm

1. Perform some iterations of PSOR $\Longrightarrow \mathcal{A}$.
2. Check convergence.
3. Eliminate active variables and their corresponding rows from original matrix: $\Longrightarrow \bar{M}$.
4. Solve $\bar{M}\bar{z} + \bar{q} = 0$ for the remaining (free) variables.
5. Project the corresponding part of the solution onto the positive orthant.
6. Update active set: $\Longrightarrow \mathcal{A}$.
7. Repeat 3-6 until consistency
8. Go to step 1.

PSOR + Subspace minimization

Algorithm

1. Perform some iterations of PSOR $\Longrightarrow \mathcal{A}$.
2. Check convergence.
3. Eliminate active variables and their corresponding rows from original matrix: $\Longrightarrow \bar{M}$.
4. Solve $\bar{M}\bar{z} + \bar{q} = 0$ for the remaining (free) variables.
5. Project the corresponding part of the solution onto the positive orthant.
6. Update active set: $\Longrightarrow \mathcal{A}$.
7. Repeat 3-6 until consistency
8. Go to step 1.

PSOR + Subspace minimization

Algorithm

1. Perform some iterations of PSOR $\Longrightarrow \mathcal{A}$.
2. Check convergence.
3. Eliminate active variables and their corresponding rows from original matrix: $\Longrightarrow \bar{M}$.
4. Solve $\bar{M}\bar{z} + \bar{q} = 0$ for the remaining (free) variables.
5. Project the corresponding part of the solution onto the positive orthant.
6. Update active set: $\Longrightarrow \mathcal{A}$.
7. Repeat 3-6 until consistency
8. Go to step 1.

PSOR + Subspace minimization

Algorithm

1. Perform some iterations of PSOR $\Longrightarrow \mathcal{A}$.
2. Check convergence.
3. Eliminate active variables and their corresponding rows from original matrix: $\Longrightarrow \bar{M}$.
4. Solve $\bar{M}\bar{z} + \bar{q} = 0$ for the remaining (free) variables.
5. Project the corresponding part of the solution onto the positive orthant.
6. Update active set: $\Longrightarrow \mathcal{A}$.
7. Repeat 3-6 until consistency
8. Go to step 1.

PSOR + Subspace minimization

Algorithm

1. Perform some iterations of PSOR $\Longrightarrow \mathcal{A}$.
2. Check convergence.
3. Eliminate active variables and their corresponding rows from original matrix: $\Longrightarrow \bar{M}$.
4. Solve $\bar{M}\bar{z} + \bar{q} = 0$ for the remaining (free) variables.
5. Project the corresponding part of the solution onto the positive orthant.
6. Update active set: $\Longrightarrow \mathcal{A}$.
7. Repeat 3-6 until consistency
8. Go to step 1.

PSOR + Subspace minimization

Algorithm

1. Perform some iterations of PSOR $\Longrightarrow \mathcal{A}$.
2. Check convergence.
3. Eliminate active variables and their corresponding rows from original matrix: $\Longrightarrow \bar{M}$.
4. Solve $\bar{M}\bar{z} + \bar{q} = 0$ for the remaining (free) variables.
5. Project the corresponding part of the solution onto the positive orthant.
6. Update active set: $\Longrightarrow \mathcal{A}$.
7. Repeat 3-6 until consistency
8. Go to step 1.

PSOR + Subspace minimization

Algorithm

1. Perform some iterations of PSOR $\Longrightarrow \mathcal{A}$.
2. Check convergence.
3. Eliminate active variables and their corresponding rows from original matrix: $\Longrightarrow \bar{M}$.
4. Solve $\bar{M}\bar{z} + \bar{q} = 0$ for the remaining (free) variables.
5. Project the corresponding part of the solution onto the positive orthant.
6. Update active set: $\Longrightarrow \mathcal{A}$.
7. Repeat 3-6 until consistency
8. Go to step 1.

Outline

Preliminary numerical results

- Dell Inspiron 630. 1 Processor Pentium 4. 1 GB in RAM. LRHE.
- Linear algebra: PARDISO. Both codes written in Fortran.
- $||r_1|| \leq 10^{-8}$; $z_2^T w_2 \leq 10^{-8}$

prob	CPU(s)	nz(L)	#fact
5	.08/.13	17905/24194	5/19
6	.44/.97	114864/216080	7/19
7	.46/1.6	218522/406152	8/19
8	.74/3.2	398821/797889	7/16
9	.91/2.4	341911/646929	9/19
10	1.2/4.8	604892/1222209	6/16

Outline

Remarks and future work

- PSOR is quite succesful in predicting the active set at the solution.
- The proposed algorithm avoids the computation of M_{11}^{-1}. It also seems superior to IPM in terms of: a) CPU time; b) use of memory; c) accuracy (very low $\longrightarrow$ very high).
- Next steps: a) extensive testing; b) convergence proof.

Game is over!