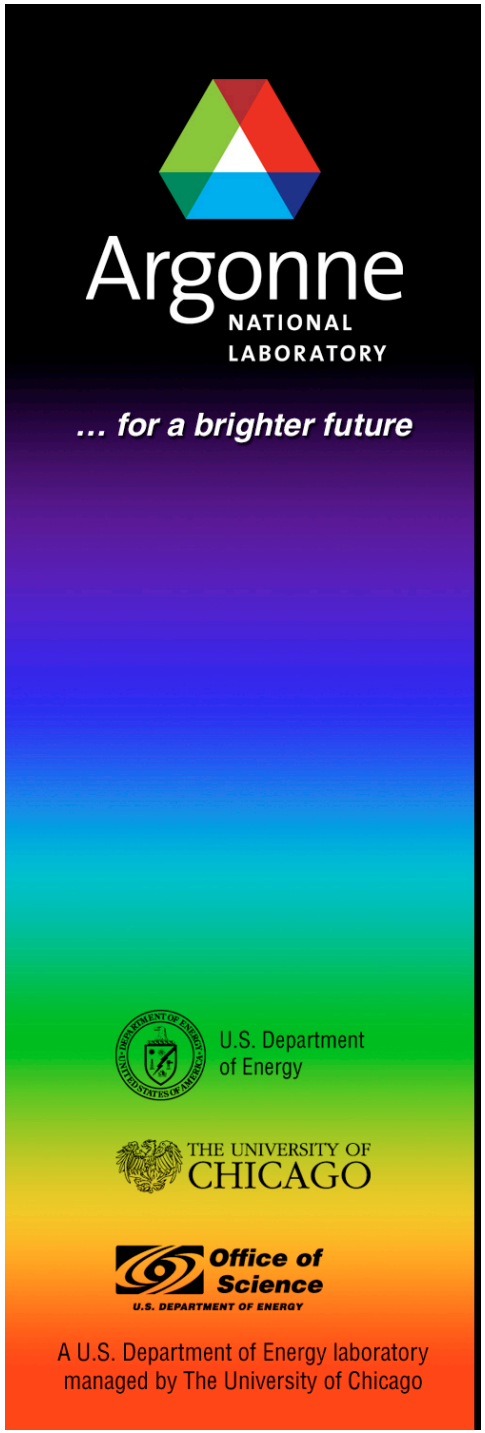


Differential Variational Inequalities

Mihai Anitescu,

David Shanno,

David Stewart

Ian Kopaczka

2008 SIAM Optimization Conference

Differential Variational Inequality –ODE case

- Formulation (Pang and Stewart, 08)

$$\dot{x}(t) = f(t, x(t), u(t))$$
$$u(t) \in \mathrm{SOL}(K, F(t, x(t), \cdot))$$
$$\Gamma(x(0), x(T)) = 0.$$

$$\widetilde{u} : [0, T] \to K,$$

- Here, the solution set SOL is defined as:

$$\int_0^T (\widetilde{u}(t) - u(t))^T F(t, x(t), u(t))\, \mathrm{d}t \geq 0.$$

- If K is a cone C. Most common cases, C is a direct sum of R^+ and R.

$$\dot{x}(t) = f(t, x(t), u(t)),$$
$$C \ni u(t) \perp F(t, x(t), u(t)) \in C^*.$$

Differential Variational Inequalities — Motivating Applications

- Contact Dynamics.
  - Rigid-Bodies: Differential Operator is ODE.
  - Deformable Bodies: Differential Operator is PDE.
  - Granular Flow, Masonry Stability, Rock Dynamics…
- Finance: Option Pricing-- American Options. PDE -based.
- Dynamics of multicristalline materials: evolution of the boundary between phases.
- Porous Media Flow.
- See Luo, Pang et al, and Kinderlehrer and Stampacchia Monographs..

Differential variational inequalities (DVI) —ODE case-- challenges

- Recently Categorized by Index (“how easy it is to solve for u infinitesimally”). similar to DAE
- Challenges
  - Do we have solution uniqueness for DVI? Since we do not have Peano (Stewart).
  - How to solve them? (Anitescu)
    - *Time stepping versus smoothing.*
    - *Nonpolyhedral C?*
  - Can they be used to compute hard MPECs appearing in stochastic games (Shanno)

Differential Variational Inequalities with PDE operator.

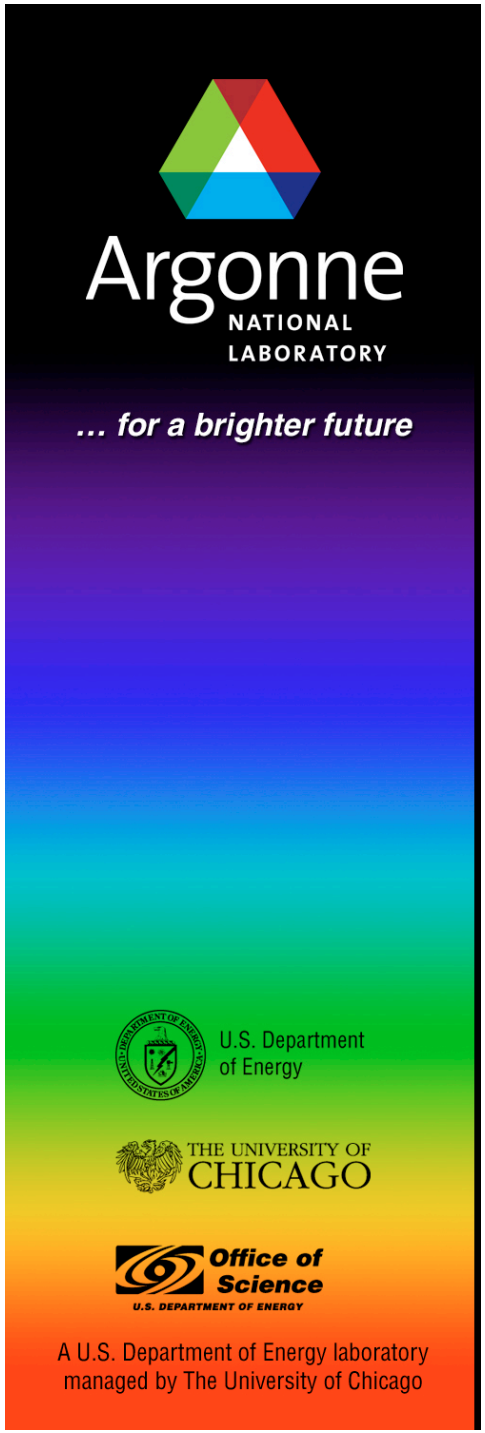
- Control of DVI (Kopaczka)—an MPEC

$$(\mathcal{P}) \qquad \begin{aligned} \min J(y,u) &:= \tfrac{1}{2}\|y-y_d\|^2_{L^2} + \tfrac{\nu}{2}\|u\|^2_{L^2} \\ \text{over} \quad & y \in H^1_\circ(\Omega);\, u \in L^2(\Omega), \\ \text{s.t.} \quad & y \in K, \end{aligned}$$

(D)VI → $a(y, v-y) \geq (u+f, v-y)_{L^2} \quad \forall\, v \in K,$

$a(\cdot,\cdot)$...bounded, coercive $\qquad K = \{v \in H^1_\circ(\Omega) : v \geq 0 \text{ a.e.}\}.$

Challenge– How to solve it scalably?-- Multigrid



An iterative solver for cone complementarity problems of nonsmooth multibody dynamics

Mihai ANITESCU [1],

[1] Argonne National Laboratories

Team

- Mihai Anitescu

- Alessandro Tasora
  - University of Parma,
  - Author of ChronoEngine

- Dan Negrut,
  - University of Wisconsin
  - Former ADAMS developer

Smoothing Versus Time-Stepping

- Recall, DVI (for C=R+)

$$\dot{x} = f\left(t, x(t), u(t)\right);$$
$$u \geq 0 \perp F\left(t, x(t), u(t)\right) \geq 0$$

- Smoothing

$$\dot{x} = f\left(t, x(t), u(t)\right);$$
$$u_i \; F_i\left(t, x(t), u(t)\right) = \varepsilon, \quad i = 1, 2, \ldots n_u$$

- Followed by forward Euler. Easy to implement!!

$$u_i^n \; F_i\left(t^{n-1}, x^{n-1}, u^{n-1}\right) = \varepsilon, \quad i = 1, 2, \ldots n_u$$
$$x^{n+1} = x^n + hf\left(t^n, x^n, u^n\right);$$

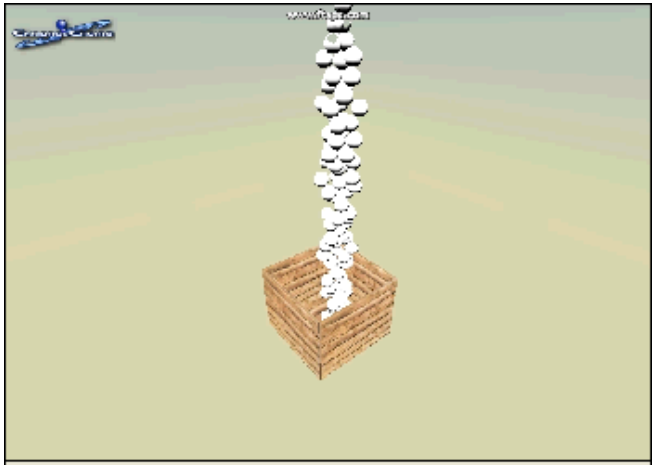
- Compare with the complexity of time-stepping

$$x^{n+1} = x^n + hf\left(t^{n+1}, x^{n+1}, u^{n+1}\right);$$
$$u^{n+1} \geq 0 \perp F\left(t^{n+1}, x^{n+1}, u^{n+1}\right) \geq 0$$

- But does it give good results?

Applying ADAMS to granular flow

- ADAMS is the workhorse of engineering dynamics.
- ADAMS/View Procedure for simulating.
- Spheres: diameter of 60 mm and a weight of 0.882 kg.
- Forces:smoothing with stiffness of 1E5, force exponent of 2.2, damping coefficient of 10.0, and a penetration depth of 0.1



ChronoIrrlicht collision avicompressed

balls_dropping

ADAMS versus ChronoEngine

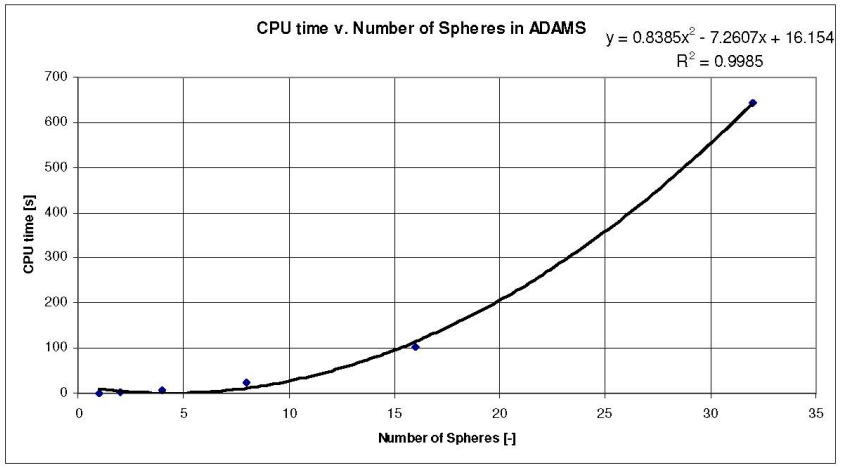
Table 1: Number of rigid bodies v. CPU time in ADAMS

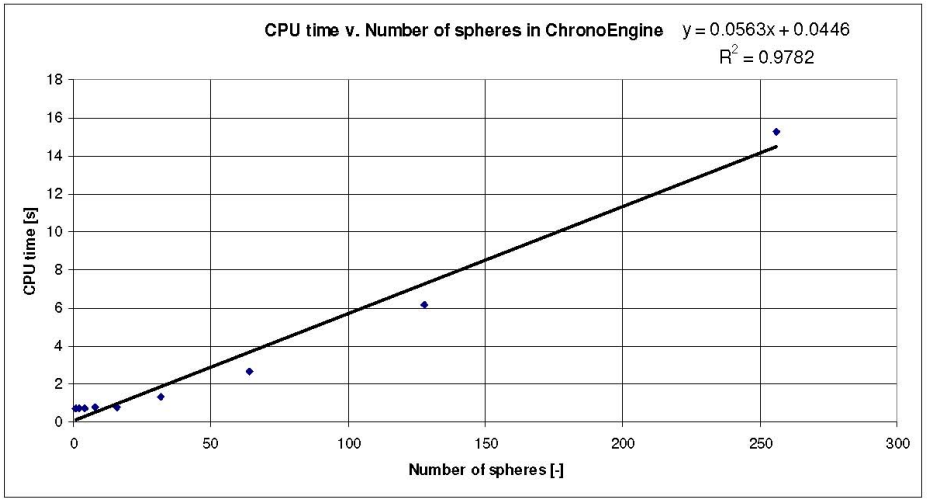
Number of Spheres	Max Number of Mutual Contacts [-]	CPU time (seconds)
1	1	0.41
2	3	3.3
4	14	7.75
8	44	25.36
16	152	102.78
32	560	644.4

The following graph shows the nonlinear increase in the CPU time as the number of colliding bodies increases.

Table 2: Number of rigid bodies v. CPU time in ChronoEngine

Number of Spheres	Max Number of Mutual Contacts [-]	CPU time (seconds)
1	1	0.70
2	3	0.73
4	14	0.73
8	44	0.76
16	152	0.82
32	560	1.32
64	2144	2.65
128	8384	6.17
256	33152	15.30





Time stepping is the way to go !!

Nonsmooth contact dynamics

- Differential problem with equilibrium constraints – DPEC.

$$
\begin{aligned}
M\frac{dv}{dt} &= \sum_{j=1,2,\ldots,p}\left(c_n^{(j)}n^{(j)} + \beta_1^{(j)}t_1^{(j)} + \beta_2^{(j)}t_2^{(j)}\right) + f_c(q,v) + k(t,q,v) \\
\frac{dq}{dt} &= v \\
c_n^{(j)} &\geq 0 \quad \perp \quad \Phi^{(j)}(q) \geq 0, \qquad j = 1,2,\ldots,p \\
\left(\beta_1^{(j)},\beta_2^{(j)}\right) &= \operatorname{argmin}_{\mu^{(j)}c_n^{(j)} \geq \sqrt{\left(\beta_1^{(j)}+\beta_2^{(j)}\right)^2}} \left[\left(v^T t_1^{(j)}\right)\beta_1 + \left(v^T t_2^{(j)}\right)\beta_2\right]
\end{aligned}
$$

Friction Model

Conic Complementarity IS NATURAL in Coulomb Models.

- Coulomb model.

$$\left(\beta_1^{(j)}, \beta_2^{(j)}\right) = \operatorname{argmin}_{\mu^{(j)} c_n^{(j)} \geq \sqrt{\left(\beta_1^{(j)} + \beta_2^{(j)}\right)^2}} \left[\left(v^T t_1^{(j)}\right)\beta_1 + \left(v^T t_2^{(j)}\right)\beta_2\right]$$

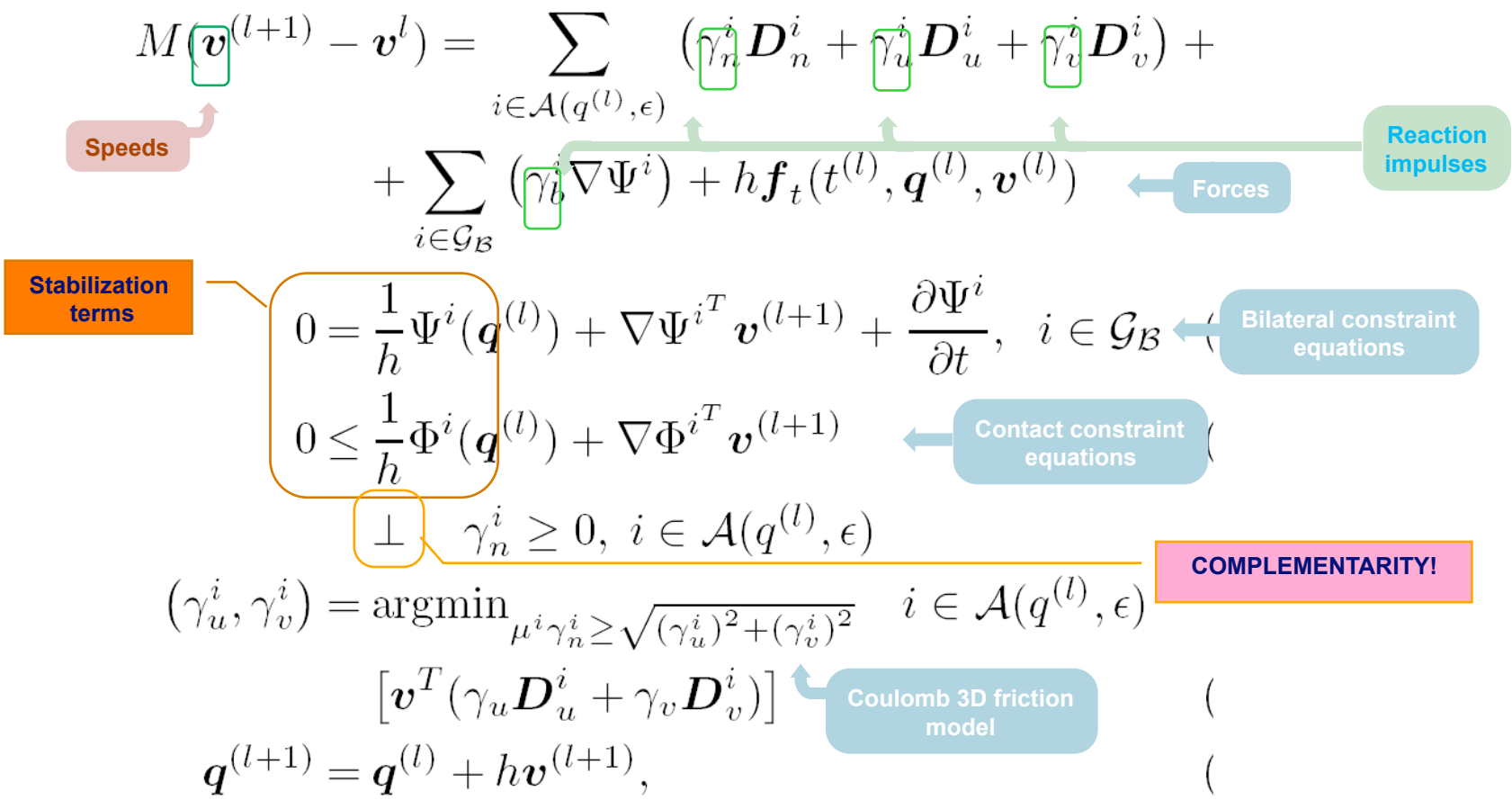
$$K = \left\{(x, y, z) \middle| \mu^{(j)} z \geq \sqrt{y^2 + x^2}\right\} \quad K^* = \left\{(x, y, z) \middle| z \geq \mu^{(j)} \sqrt{y^2 + x^2}\right\}$$

$$\begin{pmatrix} c_n^{(j)} \\ \beta_1^{(j)} \\ \beta_2^{(j)} \end{pmatrix} \in K \quad \perp \quad \begin{pmatrix} \mu^{(j)} \sqrt{\left(v^T t_1^{(j)}\right)^2 + \left(v^T t_2^{(j)}\right)^2} \\ v^T t_1^{(j)} \\ v^T t_2^{(j)} \end{pmatrix} \in K^*$$

- Most previous approaches discretize friction cone to use LCP…
- Can we still get convergence but not do that?

Time stepping scheme -- original

- A measure differential inclusion solution can be obtained by time-stepping (Stewart, 1998, Anitescu 2006)



Time Stepping -- Convex Relaxation

- A modification (relaxation, to get convex QP with conic constraints):

$$
\begin{aligned}
M(\boldsymbol{v}^{(l+1)} - \boldsymbol{v}^{l}) = & \sum_{i \in \mathcal{A}(q^{(l)}, \epsilon)} \left(\gamma_n^i \boldsymbol{D}_n^i + \gamma_u^i \boldsymbol{D}_u^i + \gamma_v^i \boldsymbol{D}_v^i\right) + \\
& + \sum_{i \in \mathcal{G}_{\mathcal{B}}} \left(\gamma_b^i \nabla \Psi^i\right) + h \boldsymbol{f}_t(t^{(l)}, \boldsymbol{q}^{(l)}, \boldsymbol{v}^{(l)}) \\
0 = & \frac{1}{h} \Psi^i(\boldsymbol{q}^{(l)}) + \nabla \Psi^{i^T} \boldsymbol{v}^{(l+1)} + \frac{\partial \Psi^i}{\partial t}, \quad i \in \mathcal{G}_{\mathcal{B}} \\
0 \leq & \frac{1}{h} \Phi^i(\boldsymbol{q}^{(l)}) + \nabla \Phi^{i^T} \boldsymbol{v}^{(l+1)} \boxed{- \mu^i \sqrt{(\boldsymbol{D}_u^{i,T} \boldsymbol{v})^2 + (\boldsymbol{D}_v^{i,T} \boldsymbol{v})^2}} \\
\perp & \quad \gamma_n^i \geq 0, \; i \in \mathcal{A}(q^{(l)}, \epsilon) \\
(\gamma_u^i, \gamma_v^i) = & \operatorname{argmin}_{\mu^i \gamma_n^i \geq \sqrt{(\gamma_u^i)^2 + (\gamma_v^i)^2}} \quad i \in \mathcal{A}(q^{(l)}, \epsilon) \\
& \left[\boldsymbol{v}^T (\gamma_u \boldsymbol{D}_u^i + \gamma_v \boldsymbol{D}_v^i)\right] \\
\boldsymbol{q}^{(l+1)} = & \boldsymbol{q}^{(l)} + h \boldsymbol{v}^{(l+1)},
\end{aligned}
$$

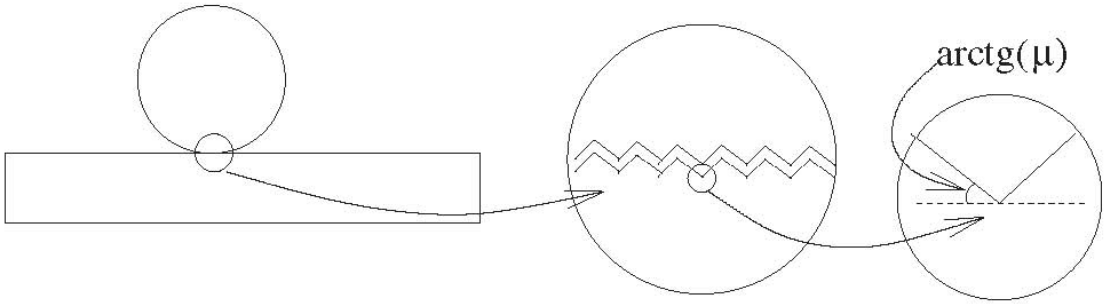
(For small μ and/or small speeds, almost no one-step differences from the Coulomb theory)

But In any case, converges to same MDI as unrelaxed scheme.

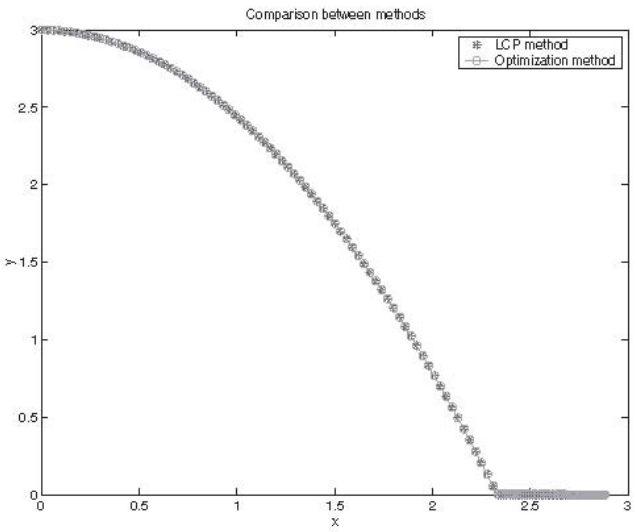
[see M.Anitescu, "Optimization Based Simulation of Nonsmooth Rigid Body Dynamics"]

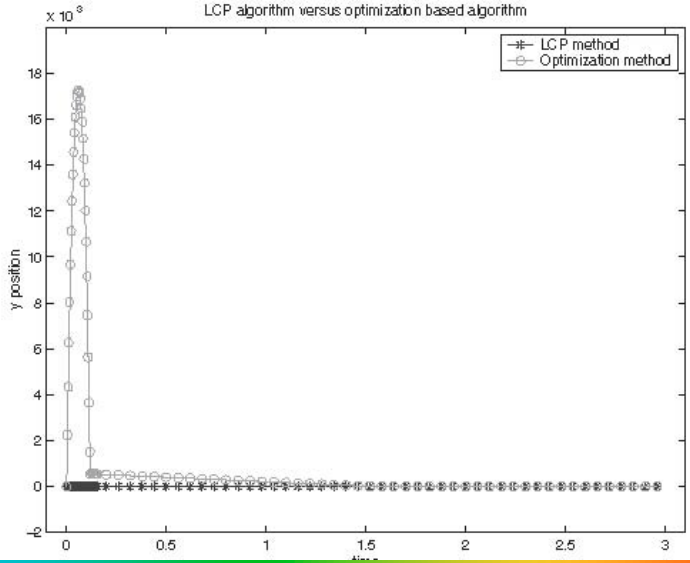
What is physical meaning of the relaxation?

- Origin



- Behavior





Cone complementarity

- Aiming at a more compact formulation:

$$\boldsymbol{b}_{\mathcal{A}} = \left\{ \frac{1}{h}\Phi^{i_1}, 0, 0, \frac{1}{h}\Phi^{i_2}, 0, 0, \ldots, \frac{1}{h}\Phi^{i_{n_{\mathcal{A}}}}, 0, 0 \right\}$$

$$\boldsymbol{\gamma}_{\mathcal{A}} = \left\{ \gamma_n^{i_1}, \gamma_u^{i_1}, \gamma_v^{i_1}, \gamma_n^{i_2}, \gamma_u^{i_2}, \gamma_v^{i_2}, \ldots, \gamma_n^{i_{n_{\mathcal{A}}}}, \gamma_u^{i_{n_{\mathcal{A}}}}, \gamma_v^{i_{n_{\mathcal{A}}}} \right\}$$

$$\boldsymbol{b}_{\mathcal{B}} = \left\{ \frac{1}{h}\Psi^1 + \frac{\partial \Psi^1}{\partial t}, \frac{1}{h}\Psi^2 + \frac{\partial \Psi^2}{\partial t}, \ldots, \frac{1}{h}\Psi^{n_{\mathcal{B}}} + \frac{\partial \Psi^{n_{\mathcal{B}}}}{\partial t} \right\}$$

$$\boldsymbol{\gamma}_{\mathcal{B}} = \left\{ \gamma_b^1, \gamma_b^2, \ldots, \gamma_b^{n_{\mathcal{B}}} \right\}$$

$$D_{\mathcal{A}} = \left[D^{i_1} | D^{i_2} | \ldots | D^{i_{n_{\mathcal{A}}}} \right], \quad i \in \mathcal{A}(\boldsymbol{q}^l, \epsilon) \qquad D^i = \left[\boldsymbol{D}_n^i | \boldsymbol{D}_u^i | \boldsymbol{D}_v^i \right]$$

- $D_{\mathcal{B}} = \left[\nabla\Psi^{i_1} | \nabla\Psi^{i_2} | \ldots | \nabla\Psi^{i_{n_{\mathcal{B}}}} \right], \quad i \in \mathcal{G}_{\mathcal{B}}$

$$\boldsymbol{b}_{\mathcal{E}} \in \mathbb{R}^{n_{\mathcal{E}}} = \{ \boldsymbol{b}_{\mathcal{A}}, \boldsymbol{b}_{\mathcal{B}} \}$$

$$\boldsymbol{\gamma}_{\mathcal{E}} \in \mathbb{R}^{n_{\mathcal{E}}} = \{ \boldsymbol{\gamma}_{\mathcal{A}}, \boldsymbol{\gamma}_{\mathcal{B}} \}$$

$$D_{\mathcal{E}} = [D_{\mathcal{A}} | D_{\mathcal{B}}]$$

Cone complementarity

- Also define:

$$\tilde{\boldsymbol{k}}^{(l)} = M\boldsymbol{v}^{(l)} + h\boldsymbol{f}_t(t^{(l)}, \boldsymbol{q}^{(l)}, \boldsymbol{v}^{(l)})$$

$$N = D_{\mathcal{E}}^T M^{-1} D_{\mathcal{E}}$$

$$\boldsymbol{r} = D_{\mathcal{E}}^T M^{-1} \tilde{\boldsymbol{k}} + \boldsymbol{b}_{\mathcal{E}}$$

- Then:

$$\begin{aligned}
M(\boldsymbol{v}^{(l+1)} - \boldsymbol{v}^l) = & \sum_{i \in \mathcal{A}(q^{(l)}, \epsilon)} (\gamma_n^i \boldsymbol{D}_n^i + \gamma_u^i \boldsymbol{D}_u^i + \gamma_v^i \boldsymbol{D}_v^i) + \\
& + \sum_{i \in \mathcal{G}_{\mathcal{B}}} (\gamma_b^i \nabla \Psi^i) + h\boldsymbol{f}_t(t^{(l)}, \boldsymbol{q}^{(l)}, \boldsymbol{v}^{(l)}) \\
0 = & \frac{1}{h}\Psi^i(\boldsymbol{q}^{(l)}) + \nabla {\Psi^i}^T \boldsymbol{v}^{(l+1)} + \frac{\partial \Psi^i}{\partial t}, \quad i \in \mathcal{G}_{\mathcal{B}} \\
0 \leq & \frac{1}{h}\Phi^i(\boldsymbol{q}^{(l)}) + \nabla {\Phi^i}^T \boldsymbol{v}^{(l+1)} \\
& \perp \quad \gamma_n^i \geq 0, \; i \in \mathcal{A}(q^{(l)}, \epsilon) \\
(\gamma_u^i, \gamma_v^i) = & \operatorname{argmin}_{\mu^i \gamma_n^i \geq \sqrt{(\gamma_u^i)^2 + (\gamma_v^i)^2}} \quad i \in \mathcal{A}(q^{(l)}, \epsilon) \\
& [\boldsymbol{v}^T(\gamma_u \boldsymbol{D}_u^i + \gamma_v \boldsymbol{D}_v^i)]
\end{aligned}$$

becomes..

$$(N\boldsymbol{\gamma}_{\mathcal{E}} + \boldsymbol{r}) \in -\Upsilon^{\circ} \quad \perp \quad \boldsymbol{\gamma}_{\mathcal{E}} \in \Upsilon$$

This is a CCP,
CONE COMPLEMENTARITY PROBLEM

Cone complementarity—Decomposable cones.

- Here we introduced the convex cone

$$\Upsilon = \left(\bigoplus_{i \in \mathcal{A}(\boldsymbol{q}^l, \epsilon)} \mathcal{FC}^i \right) \bigoplus \left(\bigoplus_{i \in \mathcal{G}_\mathcal{B}} \mathcal{BC}^i \right)$$

$\mathcal{FC}^i$ In R^3 is i-th friction cone

$\mathcal{BC}^i$ is R

- ..and its polar cone:

$$\Upsilon^\circ = \left(\bigoplus_{i \in \mathcal{A}(\boldsymbol{q}^l, \epsilon)} \mathcal{FC}^{i\circ} \right) \bigoplus \left(\bigoplus_{i \in \mathcal{G}_\mathcal{B}} \mathcal{BC}^{i\circ} \right)$$

CCP: $(N\boldsymbol{\gamma}_\mathcal{E} + \boldsymbol{r}) \in -\Upsilon^\circ \quad \perp \quad \boldsymbol{\gamma}_\mathcal{E} \in \Upsilon$

General: The iterative method

- How to practically solve the Cone Complementarity Problem?

$$(N\boldsymbol{\gamma}_{\mathcal{E}} + \boldsymbol{r}) \in -\Upsilon^{\circ} \quad \perp \quad \boldsymbol{\gamma}_{\mathcal{E}} \in \Upsilon$$

- Our method: use a fixed-point iteration

$$\boldsymbol{\gamma}^{r+1} = \lambda \Pi_{\Upsilon} \left(\boldsymbol{\gamma}^{r} - \omega B^{r} \left(N\boldsymbol{\gamma}^{r} + \boldsymbol{r} + K^{r} \left(\boldsymbol{\gamma}^{r+1} - \boldsymbol{\gamma}^{r} \right) \right) \right) + (1 - \lambda)\, \boldsymbol{\gamma}^{r}$$

- with matrices:
- ..and a non-extensive orthogonal pı operator onto feasible set

$$B^{r} = \begin{bmatrix} \eta_1 I_{n_1} & 0 & \cdots & 0 \\ 0 & \eta_2 I_{n_2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \eta_{n_k} I_{n_{n_k}} \end{bmatrix}$$

$$N^{T} = \begin{bmatrix} 0 & K_{12} & K_{13} & \cdots & K_{1n_k} \\ 0 & 0 & K_{23} & \cdots & K_{2n_k} \\ 0 & 0 & 0 & \cdots & K_{3n_k} \\ \vdots & \vdots & \cdots & \ddots & \vdots \\ 0 & 0 & 0 & & 0 \end{bmatrix}$$

$$\Pi_{\Upsilon} : \mathbb{R}^{n\varepsilon} \rightarrow \mathbb{R}^{n\varepsilon}$$

General: The iterative method

- ASSUMPTIONS

A1 The matrix N of the problem (CCP) is symmetric and positive semi-definite.

A2 There exists a positive number, $\alpha > 0$ such that, at any iteration r, $r = 0, 1, 2, \ldots$, we have that $B^r \succ \alpha I$

A3 There exists a positive number, $\beta > 0$ such that, at any iteration r, $r = 0, 1, 2, \ldots$, we have that $(x^{r+1} - x^r)^T \left((\lambda \omega B^r)^{-1} + K^r - \frac{N}{2} \right) (x^{r+1} - x^r) \geq \beta \left\| x^{r+1} - x^r \right\|^2$.

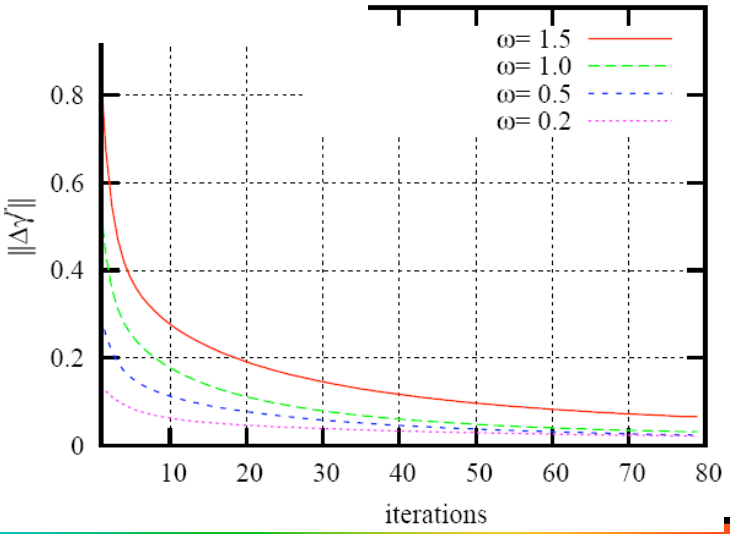
Always satisfied in multibody systems

Essentially free choice, we use identity blocks

Use ω overrelaxation factor to adjust this

- Under the above assumptions, we can prove THEOREMS about convergence.

- The method produces a bounded sequence with an unique accumulation point.



General: Theory

$$(OC)\quad \begin{array}{ll} \min & f(x) = \frac{1}{2}x^T N x + r^T x \\ s.t. & x_i \in \Upsilon^i, \qquad\qquad i = 1, 2, \ldots, n_k. \end{array}$$

Theorem Assume that $x^0 \in \Upsilon$ and that the sequences of matrices B^r and K^r are bounded. Then we have that

$$f(x^{r+1}) - f(x^r) \leq -\beta \left\| x^{r+1} - x^r \right\|^2$$

for any iteration index r, and any accumulation point of the sequence x^r is a solution of (CCP).

Corollary Assume that the friction cone of the configuration is pointed The algorithm produces a bounded sequence, and any **accumulation point results in the same velocity solution**

- Simple, but first result of this nature for conic constraints—and HIGHLY EFFICIENT

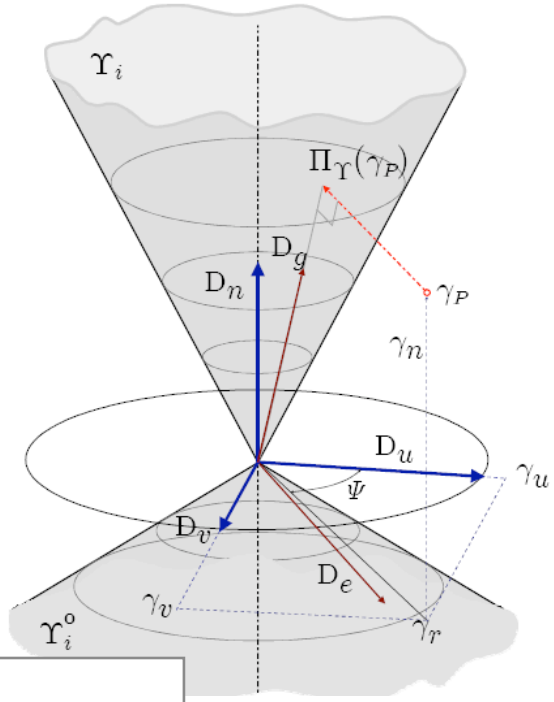
General: Interesting Extensions

- N non symmetric, but positive semidefinite.
- Parallelizing the algorithms: block Jacobi with Gauss Seidel blocks.
- Asynchronous version of the algorithm, particularly for use with GPU.

The projection operator

- *For each frictional contact constraint:*

$$\Pi_\Upsilon = \left\{ \Pi_{\Upsilon_1}(\boldsymbol{\gamma}_1)^T, \ldots \Pi_{\Upsilon_{n_\mathcal{A}}}(\boldsymbol{\gamma}^{n_\mathcal{A}})^T, \Pi_b^1(\gamma_b^1), \ldots, \Pi_b^{n_\mathcal{B}}(\gamma_b^{n_\mathcal{B}}) \right\}^T$$



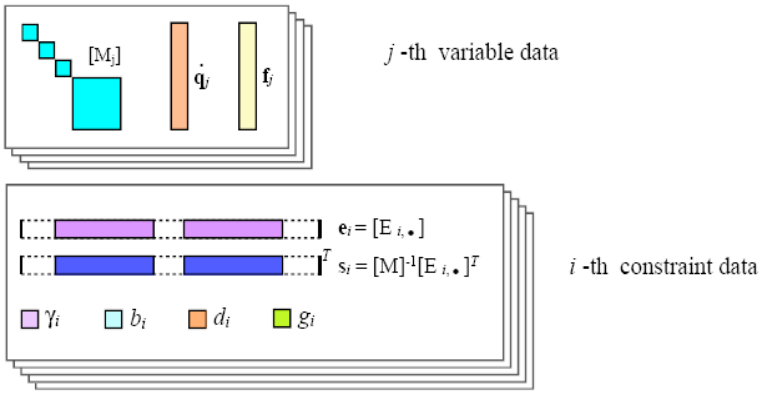
- *For each bilateral constraint, simply do nothing.*
- The complete operator:

$$\forall i \in \mathcal{A}(\boldsymbol{q}^{(l)}, \epsilon)$$

$$\begin{aligned}
\gamma_r < \mu_i \gamma_n \quad & \Pi_i = \boldsymbol{\gamma}_i \\
\gamma_r < -\frac{1}{\mu_i}\gamma_n \quad & \Pi_i = \{0,0,0\} \\
\gamma_r > \mu_i \gamma_n \wedge \gamma_r > -\frac{1}{\mu_i}\gamma_n \quad & \Pi_{i,n} = \frac{\gamma_r \mu_i + \gamma_n}{\mu_i^2 + 1} \\
& \Pi_{i,u} = \gamma_u \frac{\mu_i \Pi_{i,n}}{\gamma_r} \\
& \Pi_{i,v} = \gamma_v \frac{\mu_i \Pi_{i,n}}{\gamma_r}
\end{aligned}$$

The algorithm

- Development of an efficient algorithm for fixed point iteration:



- *avoid temporary data, exploit sparsity. Never compute explicitly the N matrix!*
- *implemented in incremental form. Compute only deltas of multipliers.*
- *O(n) space requirements and supports premature termination*
- *for real-time purposes: O(n) time*

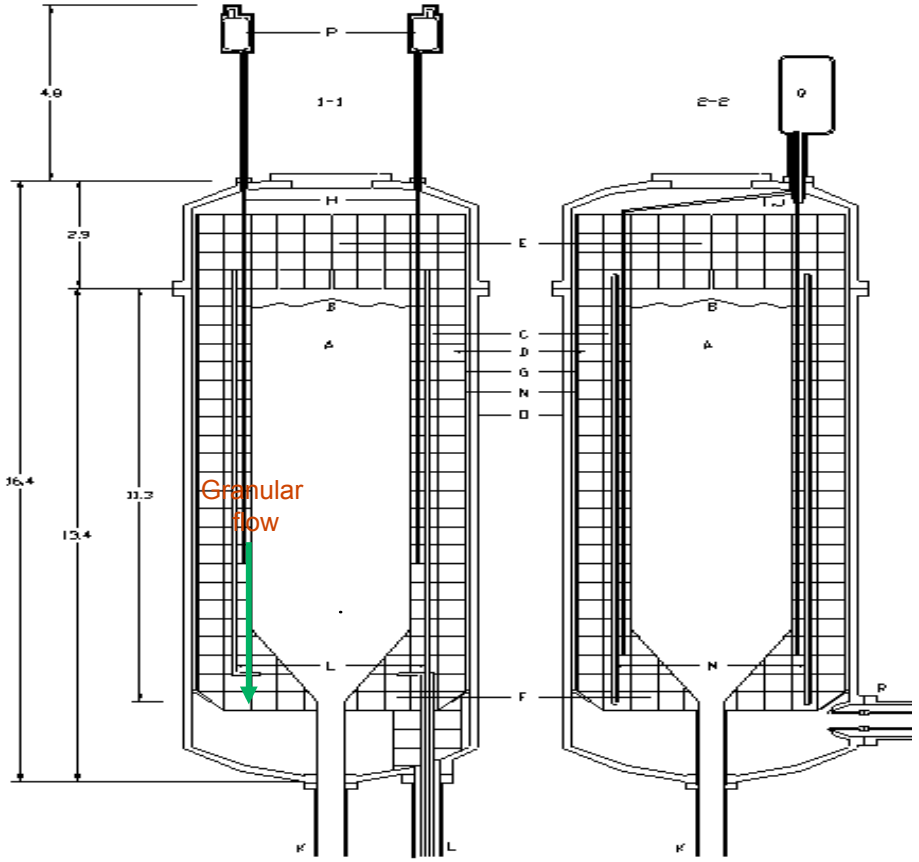
The algorithm is specialized, for minimum memory use!

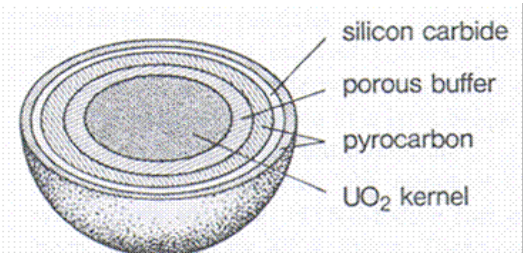
(1) *// Pre-compute some data for friction constraints*
(2) **for** $i := 1$ **to** $n_{\mathcal{A}}$
(3) $\quad \boldsymbol{s}_a^i = M^{-1} D^i$
(4) $\quad g_a^i = D^{i,T} \boldsymbol{s}_a^i$
(5) $\quad \eta_a^i = \frac{3}{\mathrm{Trace}(g_a^i)}$
(6) *// Pre-compute some data for bilateral constraints*
(7) **for** $i := 1$ **to** $n_{\mathcal{B}}$
(8) $\quad s_b^i = M^{-1} \boldsymbol{\nabla} \boldsymbol{\Psi}^i$
(9) $\quad g_b^i = \boldsymbol{\nabla} \boldsymbol{\Psi}^{i,T} s_b^i$
(10) $\quad \eta_b^i = \frac{1}{g_b^i}$
(11)
(12) *// Initialize impulses*
(13) **if** warm start with initial guess $\boldsymbol{\gamma}_{\mathcal{E}}^*$
(14) $\quad \boldsymbol{\gamma}_{\mathcal{E}}^0 = \boldsymbol{\gamma}_{\mathcal{E}}^*$
(15) **else**
(16) $\quad \boldsymbol{\gamma}_{\mathcal{E}}^0 = 0$
(17)
(18) *// Initialize speeds*
(19) $\boldsymbol{v} = \sum_{i=1}^{n_{\mathcal{A}}} \boldsymbol{s}_a^i \boldsymbol{\gamma}_a^{i,0} + \sum_{i=1}^{n_{\mathcal{B}}} s_b^i \gamma_b^{i,0} + M^{-1} \tilde{\boldsymbol{k}}$

(21) *// Main iteration loop*
(22) **for** $r := 0$ **to** r_{max}
(23) $\quad$ *// Loop on frictional constraints*
(24) $\quad$ **for** $i := 1$ **to** $n_{\mathcal{A}}$
(25) $\quad\quad \boldsymbol{\delta}_a^{i,r} = \left(\boldsymbol{\gamma}_a^{i,r} - \omega \eta_a^i \left(D^{i,T} \boldsymbol{v}^r + \boldsymbol{b}_a^i \right) \right);$
(26) $\quad\quad \boldsymbol{\gamma}_a^{i,r+1} = \lambda \Pi_{\Upsilon} \left(\boldsymbol{\delta}_a^{i,r} \right) + (1-\lambda) \boldsymbol{\gamma}_a^{i,r}$;
(27) $\quad\quad \Delta \boldsymbol{\gamma}_a^{i,r+1} = \boldsymbol{\gamma}_a^{i,r+1} - \boldsymbol{\gamma}_a^{i,r}$;
(28) $\quad\quad \boldsymbol{v} := \boldsymbol{v} + \boldsymbol{s}_a^{i^T} \Delta \boldsymbol{\gamma}_a^{i,r+1}.$
(29) $\quad$ *// Loop on bilateral constraints*
(30) $\quad$ **for** $i := 1$ **to** $n_{\mathcal{B}}$
(31) $\quad\quad \delta_b^{i,r} = \left(\gamma_b^{i,r} - \omega \eta_b^i \left(\nabla \boldsymbol{\Psi}^{i,T} \boldsymbol{v}^r + b_b^i \right) \right);$
(32) $\quad\quad \gamma_b^{i,r+1} = \lambda \Pi_{\Upsilon} \left(\delta_b^{i,r} \right) + (1-\lambda) \gamma_b^{i,r}$;
(33) $\quad\quad \Delta \gamma_b^{i,r+1} = \gamma_b^{i,r+1} - \gamma_b^{i,r}$;
(34) $\quad\quad \boldsymbol{v} := \boldsymbol{v} + s_b^{i^T} \Delta \gamma_b^{i,r+1}.$
(35)
(36) **return** $\boldsymbol{\gamma}_{\mathcal{E}}$, $\boldsymbol{v}$

Simulating the PBR nuclear reactor

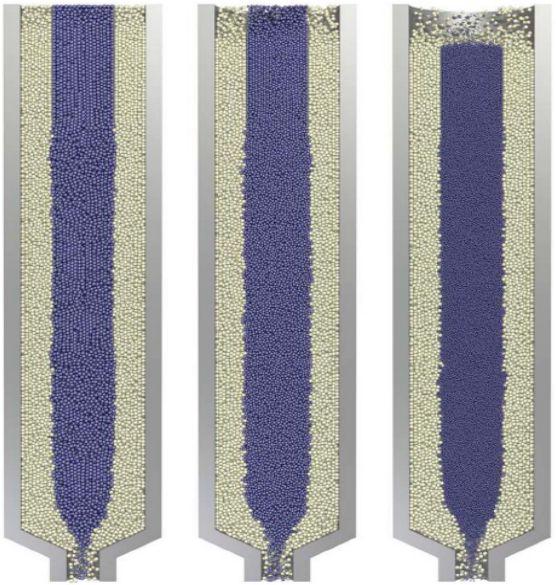
- The PBR nuclear reactor:
  - Fourth generation design
  - Inherently safe, by Doppler broadening of fission cross section
  - Helium cooled > 1000 °C
  - Can crack water (mass production of hydrogen)
  - Continuous cycling of 360’000 graphite spheres in a pebble bed





Simulating the PBR nuclear reactor

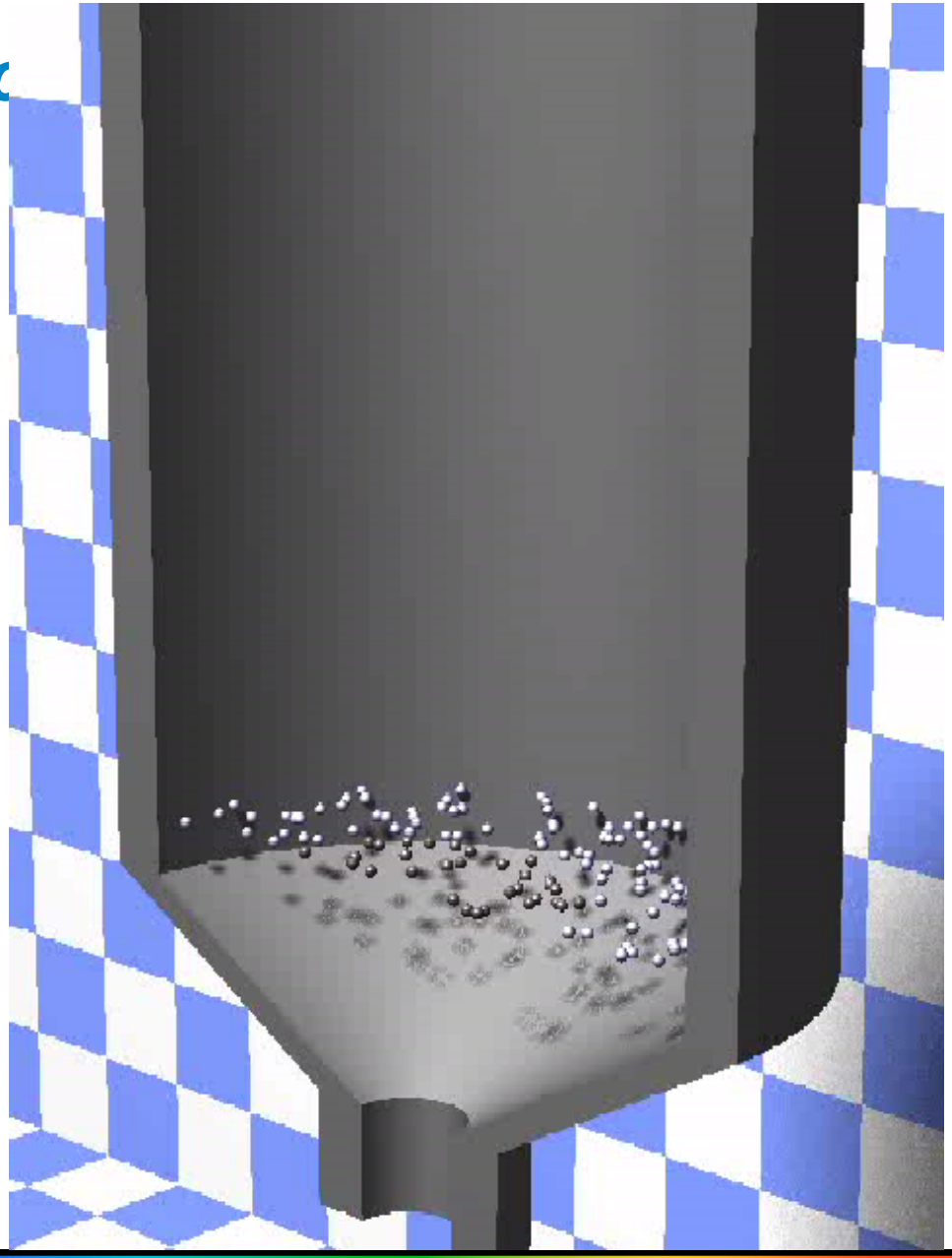
- Problem of bidisperse granular flow with dense packing.
- Previous attempts: DEM methods on supercomputers at Sandia Labs regularization)
- 40 seconds of simulation for 440,000 pebbles needs 1 week on 64 processors dedicated cluster (Rycroft et al.)



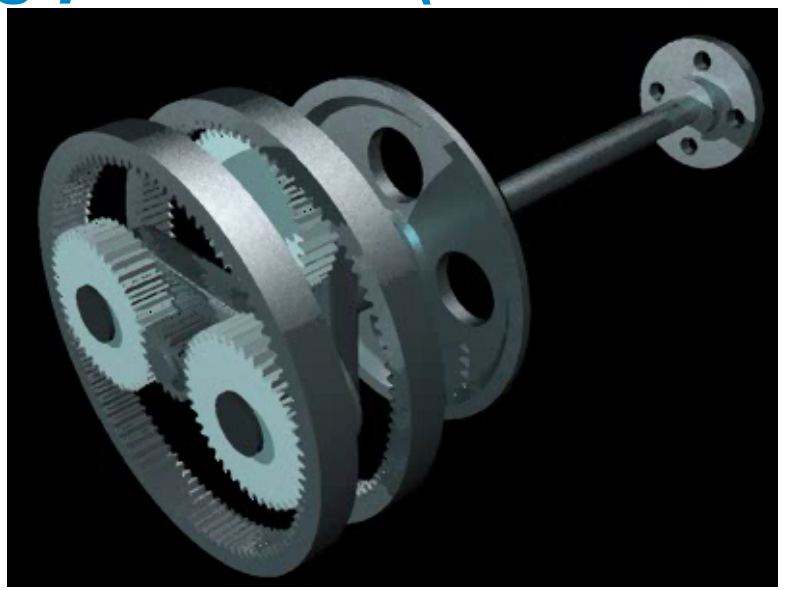
model a frictionless wall, $\mu_w=0.0$. For the current simulations we set $k_t=\frac{2}{7}k_n$ and choose $k_n=2\times10^5\ gm/d$. While this is significantly less than would be realistic for graphite pebbles, where we expect $k_n>10^{10}\ gm/d$, such a spring constant would be prohibitively computationally expensive, as the time step scales as $\delta t\propto k_n^{-1/2}$ for collisions to be modeled effectively. Previous simulations have shown that

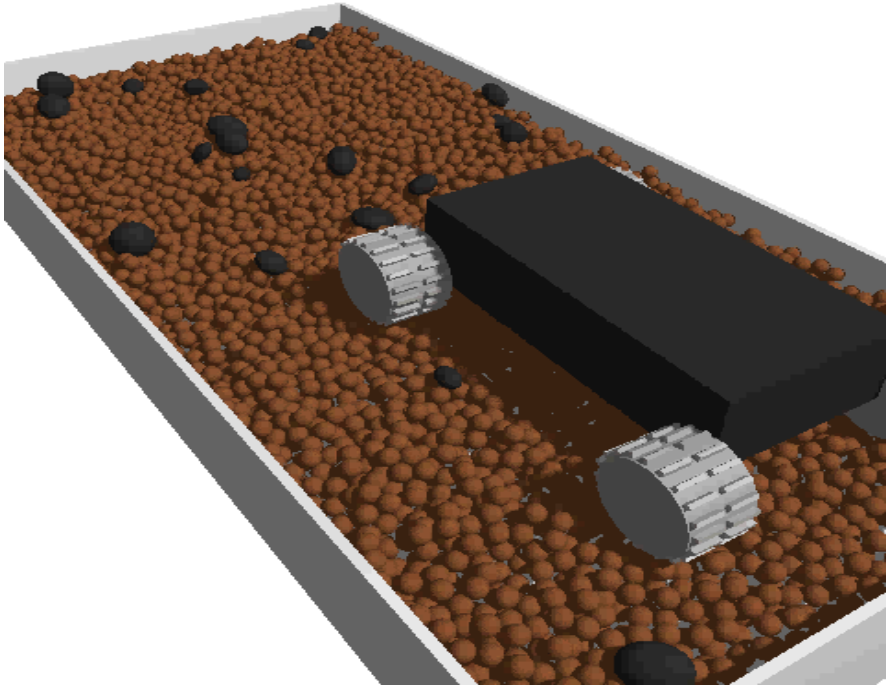
Simulating the PBR nuc

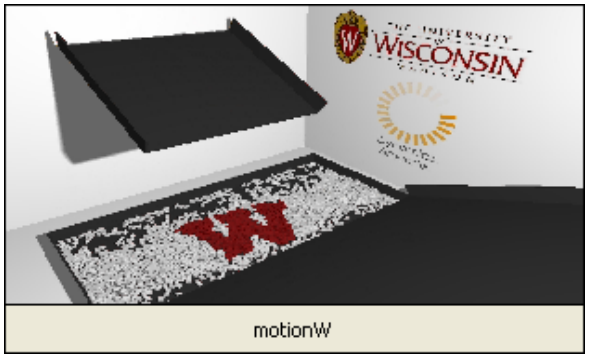
- 160’000 Uranium-Graphite spheres, 600’000 contacts on average
- Two millions of primal variables, six millions of dual variables
- *1 day on a Windows station…*
- But we are limited by the 2GB user mode limit, 64 bit port in progress—but linear scaling..
- We estimate 3CPU days, compare with 450 CPU days for an incomplete solution in 2006 !!!



In addition, we can approach efficiently approach many engineering problems (see website for papers)

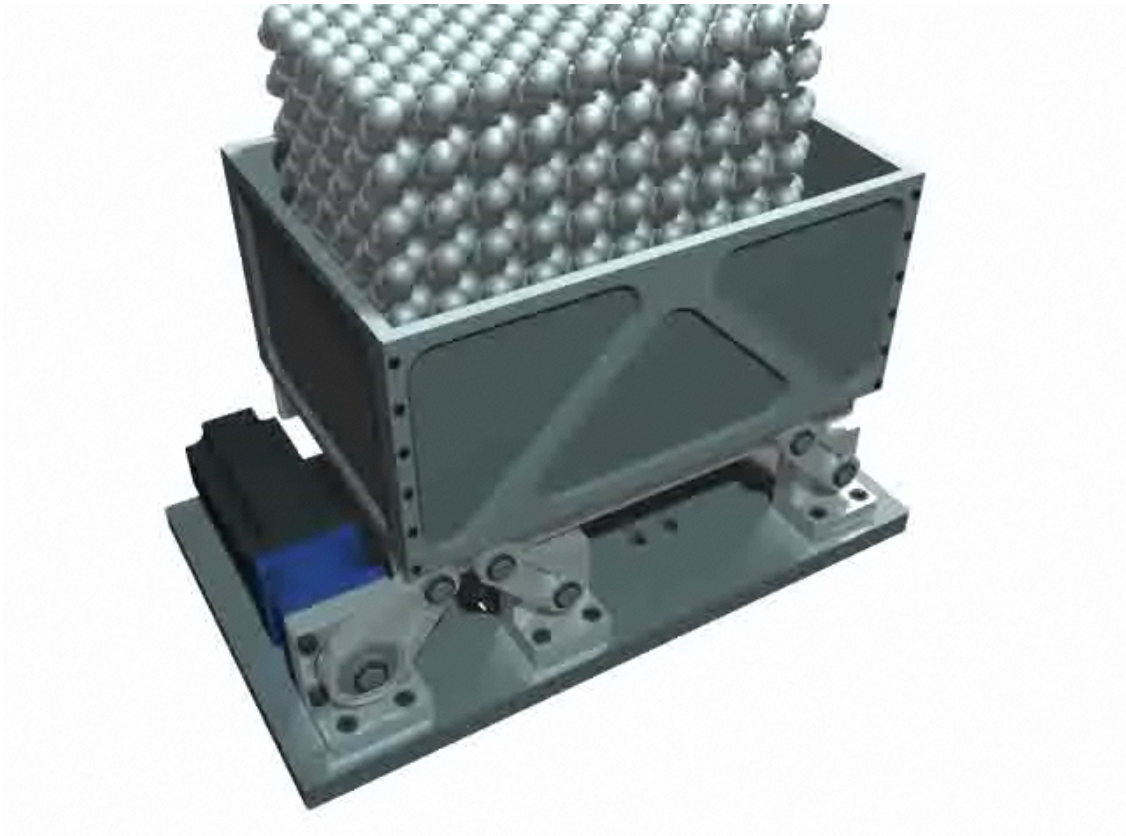






Examples

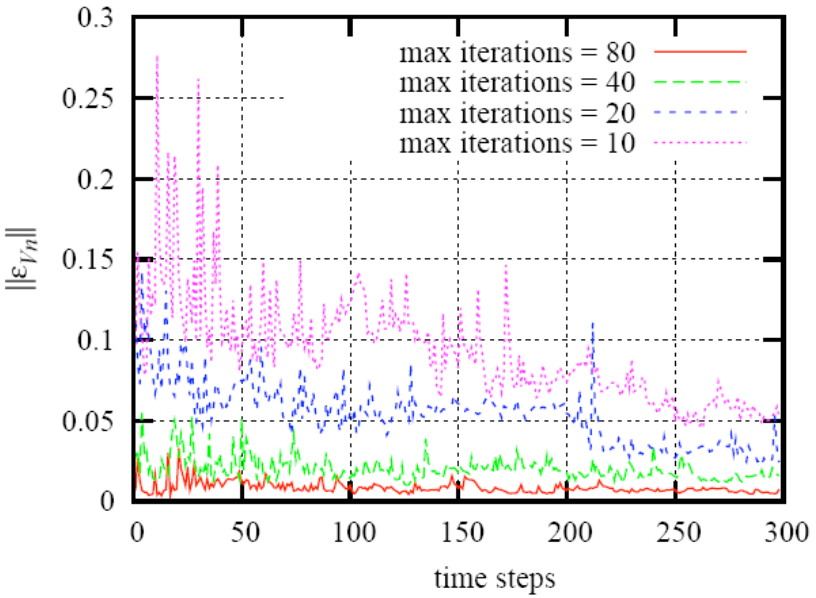
- Example: size-segregation in shaker, with thousands of steel spheres



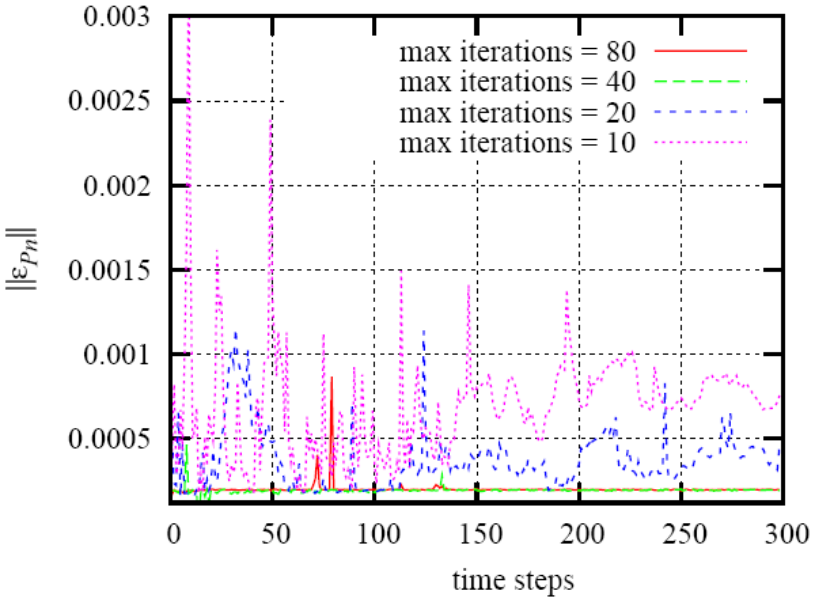
Note: solution beyond reach of Lemke-type LCP solvers!

Tests

- Feasibility accuracy increases with number of iterations:



Speed violation in constraints

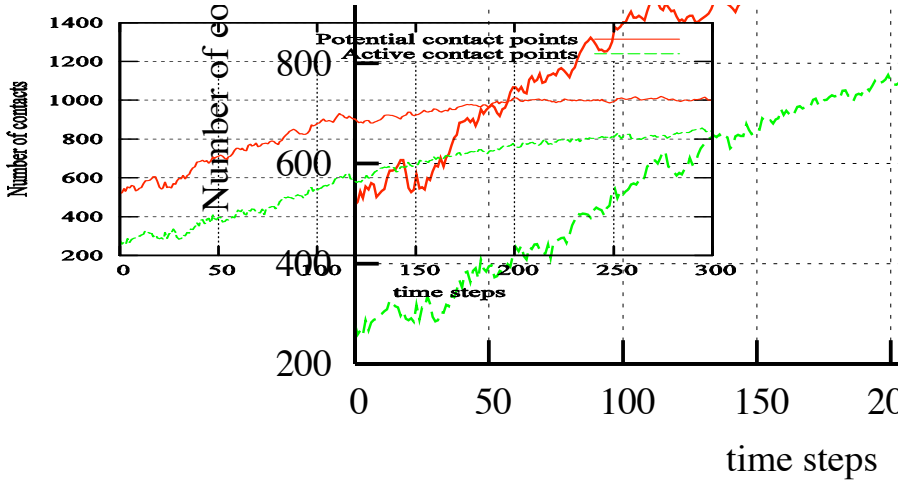


Position error in constraints (penetration)

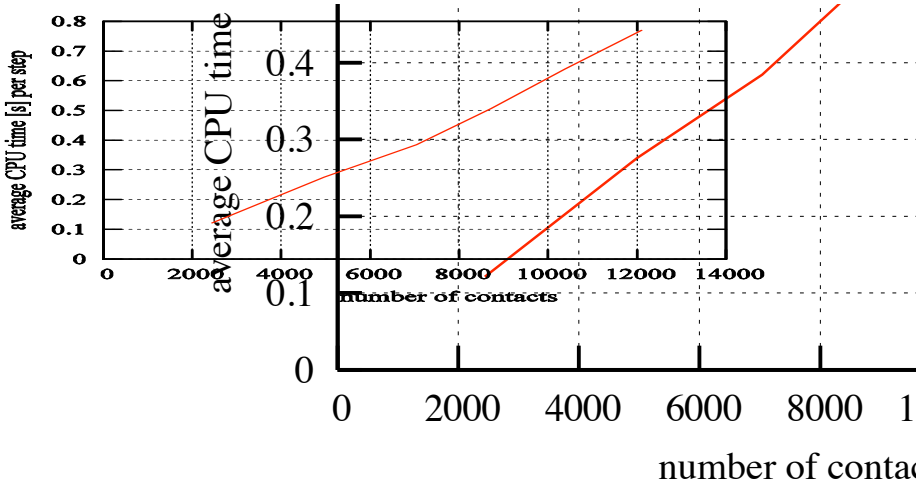
(with example of 300 spheres in shaker)

Tests: Scalability

- CPU effort per contact, since our contacts are the problem variables.
- Penetration error was uniformly no larger than 0.2% of diameter.

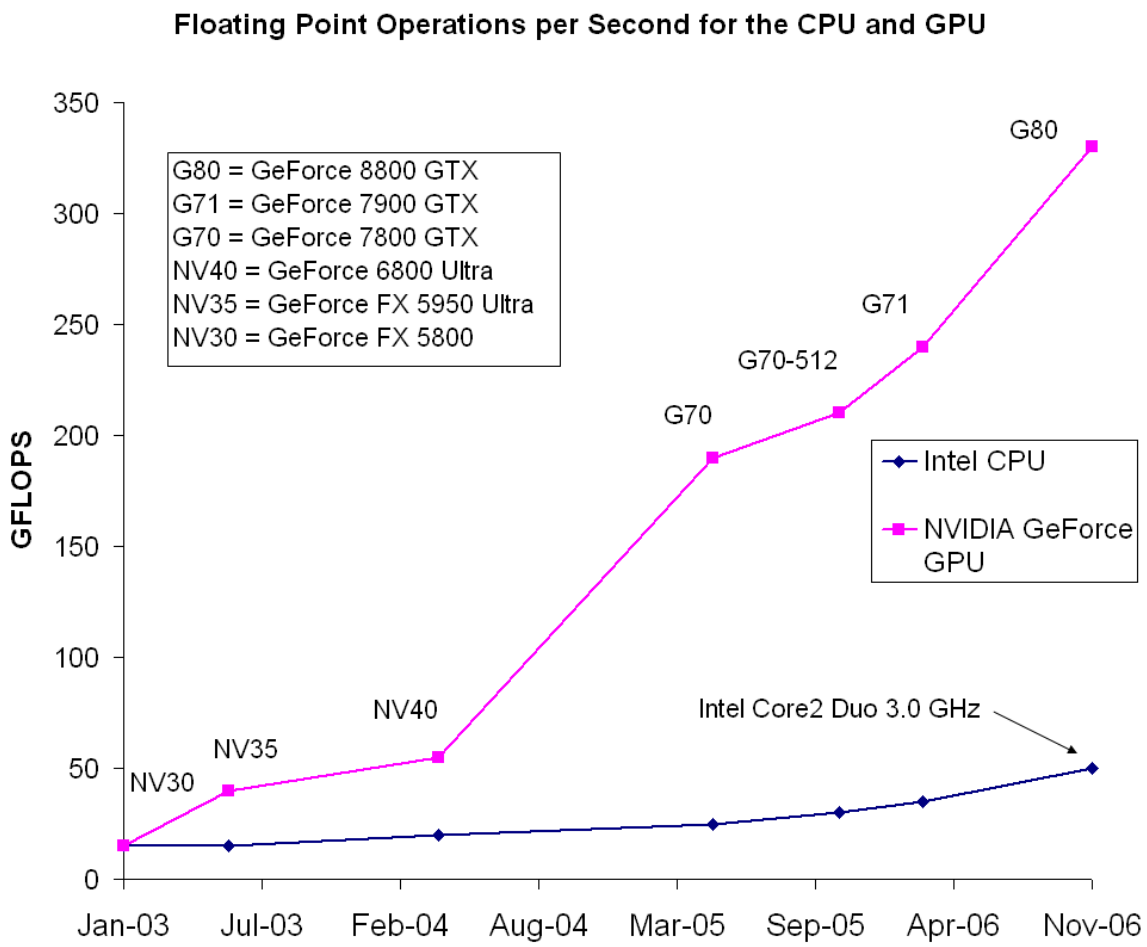


Number of contacts in time, 300 spheres



CPU time per step for 300-1500 spheres

New large scale computational opportunity Graphical Processing Unit

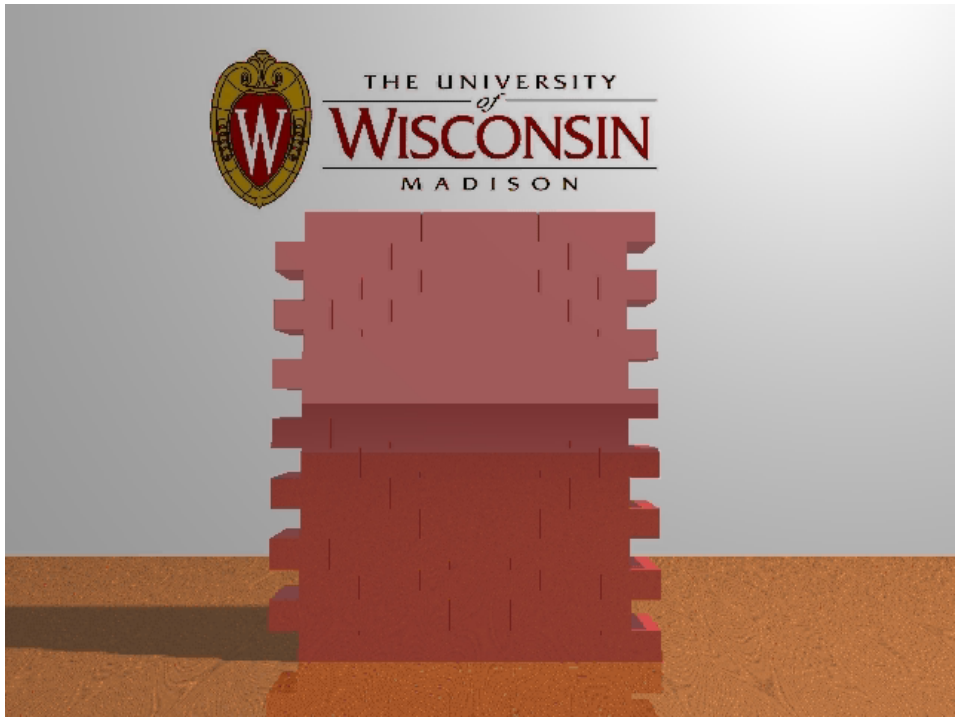


IBM BlueGene/L—GPU comparison

- Entry model: 1024 dual core nodes
- 5.7 Tflop (compare to 0.5 Tflop for NVIDIA Tesla GPU)
- Dedicated OS
- Dedicated power management solution
- Require dedicated IT support
- Price (2007): $1.4 million
- Same GPU power (2008): 7K!!!

Brick Wall Example…

- Times reported are in seconds for one second long simulation
- GPU: NVIDIA GeForce 8800 GTX



Bricks	Sequential Version	GPU Co-processing Version
1000	43	6
2000	87	10
8000	319	42

Conclusions

- We have defined a new algorithm for complementarity problems with conic constraints.
- We have shown that it can solve very large problems in granular flow far faster than DEM.
- It is the first iterative algorithm that provably converges for nonsmooth rigid body dynamics.
- It scalability is decent.
- We have created a multithreaded implementation and GPU port increases computational speed by a factor of 7-8.