Linear algebra considerations for the multi-threaded simulation of mechanical systems

Dan Negrut (dan.negrut@mscsoftware.com)*

Abstract. A solution method suitable for the multi-threaded simulation of mechanical systems represented in Cartesian coordinates is proposed and analyzed. In a state-space framework for the solution of the Differential Algebraic Equations (DAE) of Multibody Dynamics, the position/velocity stabilization and the acceleration computation are based on iterative solvers applied to equivalent reduced problems. The most in-depth computational aspect analyzed is the preconditioning, i.e., the direct solution of the reduced systems. Provided a topology index reduction is first applied to the model, the effort for the direct solution of the reduced systems is shown to be of order $O(N_J)$, where N_J is the number of joints in the model. The recurring theme of the paper is the central role that the topology of the mechanical system plays in the overall performance of the numerical simulation. Based on the topology of the model, parallel computational threads can be established to start in the equation formulation and continue through the iterative numerical algorithms employed for the numerical solution. Task scheduling these parallel threads is expected to redeem real-time performance for certain classes of complex applications.

Keywords: mechanical system simulation, coordinate partitioning, order $O(N_J)$ solution method, preconditioned conjugate gradient, multi-threaded solution

1. Introduction

The generalized coordinates used in this paper are Cartesian coordinates for position, and Euler parameters for orientation of body centroidal reference frames; i.e., for body i, $\mathbf{r}_i = [x_i, y_i, z_i]^T$, and $\mathbf{p}_i = [e_{i0}, e_{i1}, e_{i2}, e_{i3}]^T$, respectively. The array of generalized coordinates for a model containing n_b rigid bodies is defined as

$$\mathbf{q} = \left[\begin{array}{cccccc} \mathbf{r}_1^T & \dots & \mathbf{r}_{n_b}^T & \mathbf{p}_1^T & \dots & \mathbf{p}_{n_b}^T \end{array} \right]^T \in I\!R^n,\ n = 7n_b \tag{1}$$

where the Euler parameters $\mathbf{p}_i$ satisfy the normalization constraint [13]

$$\mathbf{p}_i^T \mathbf{p}_i = 1,\ 1 \leq i \leq n_b \tag{2}$$

The velocity state of the system is described by the array of generalized velocities $\dot{\mathbf{q}} = \left[\dot{\mathbf{r}}_1^T, \dots, \dot{\mathbf{p}}_{n_b}^T\right]^T$, where the over-dot denotes differentiation with respect to time.

* MSC.Software, 2300 Traverwood Drive, Ann Arbor, MI 48105, USA

The joints of the model induce kinematic constraints on the position and velocity generalized coordinates. Without loss of generality, these constraints are considered to be holonomic.

$$\mathbf{\Phi}(\mathbf{q},t) = \left[\ \Phi_1(\mathbf{q},t)\ \ \ldots\ \ \Phi_m(\mathbf{q},t)\ \right]^T = \mathbf{0} \tag{3}$$

If m_k represents the number of constraints induced by the joint k, then $\mathbf{\Phi}(\mathbf{q},t) \in I\!R^m$, where $m = \sum_{k=1}^{N_J} m_k$, with N_J denoting the number of joints present in the mechanical system. Differentiating Eq.3 with respect to time yields the velocity kinematic constraint equation

$$\eta\left(\mathbf{\Phi},\mathbf{q}\right)\,\dot{\mathbf{r}} + \rho\left(\mathbf{\Phi},\mathbf{q}\right)\,\bar{\omega} + \mathbf{\Phi}_t(\mathbf{q},t) = \mathbf{0} \tag{4}$$

In Eq.4, $\bar{\omega} = [\bar{\omega}_1^T, \ldots, \bar{\omega}_{n_b}^T]^T$ is the array of local angular velocities, and

$$\eta\left(\mathbf{\Phi},\mathbf{q}\right) = \left[\eta_1\left(\mathbf{\Phi},\mathbf{q}\right), \ldots, \eta_{n_b}\left(\mathbf{\Phi},\mathbf{q}\right)\right] \tag{5a}$$

$$\rho\left(\mathbf{\Phi},\mathbf{q}\right) = \left[\rho_1\left(\mathbf{\Phi},\mathbf{q}\right), \ldots, \rho_{n_b}\left(\mathbf{\Phi},\mathbf{q}\right)\right] \tag{5b}$$

where $\eta_i\left(\mathbf{\Phi},\mathbf{q}\right) \in I\!R^{m\times 3}$ and $\rho_i\left(\mathbf{\Phi},\mathbf{q}\right) \in I\!R^{m\times 3}$ are time-independent linearization operators associated with body i that act on the constraints $\mathbf{\Phi}$ in the configuration $\mathbf{q}$. Finally, note that $\dot{\mathbf{p}}_i$ is obtained from $\bar{\omega}_i$ by means of the velocity transformation matrix $\mathbf{G}_i\left(\mathbf{p}\right)$, [13]

$$\dot{\mathbf{p}}_i = \frac{1}{2}\mathbf{G}_i^T\left(\mathbf{p}\right)\,\bar{\omega} \tag{6}$$

The acceleration kinematic constraint equation is obtained by differentiating Eq.4 with respect to time,

$$\eta\left(\mathbf{\Phi},\mathbf{q}\right)\,\ddot{\mathbf{r}} + \rho\left(\mathbf{\Phi},\mathbf{q}\right)\,\dot{\bar{\omega}} = \tau\left(\mathbf{q},\dot{\mathbf{q}},t\right) \tag{7}$$

where τ is a function that as indicated only depends on position, velocity, and time. The second time derivative of the Euler parameters $\ddot{\mathbf{p}}_i$ is obtained from $\dot{\bar{\omega}}_i$ as [13]

$$\ddot{\mathbf{p}}_i = \frac{1}{2}\mathbf{G}_i^T\left(\mathbf{p}\right)\,\dot{\bar{\omega}}_i - \frac{1}{4}\left(\bar{\omega}_i^T\bar{\omega}_i\right)\mathbf{p}_i \tag{8}$$

The time evolution of the mechanical system is the solution of the Newton-Euler equations of motion, which for a system of n_b rigid bodies with centroidal body-fixed reference frame assumes the form [13, 24],

$$\begin{bmatrix} \mathbf{M} & \mathbf{0} \\ \mathbf{0} & \bar{\mathbf{J}} \end{bmatrix}\begin{bmatrix} \ddot{\mathbf{r}} \\ \dot{\bar{\omega}} \end{bmatrix} + \begin{bmatrix} \eta^T\left(\mathbf{\Phi},\mathbf{q}\right) \\ \rho^T\left(\mathbf{\Phi},\mathbf{q}\right) \end{bmatrix}\lambda = \begin{bmatrix} \mathbf{f} \\ \hat{\mathbf{n}} \end{bmatrix} \tag{9}$$

where $\mathbf{M} = diag(\mathbf{M}_1, \ldots, \mathbf{M}_{n_b})$, $\bar{\mathbf{J}} = diag\left(\bar{\mathbf{J}}_1, \ldots, \bar{\mathbf{J}}_{n_b}\right)$, $\ddot{\mathbf{r}} = [\ddot{\mathbf{r}}_1^T, \ldots, \ddot{\mathbf{r}}_{n_b}^T]^T$, $\mathbf{f} = [\mathbf{f}_1^T, \ldots, \mathbf{f}_{n_b}^T]^T$, and $\hat{\mathbf{n}} = [\hat{\mathbf{n}}_1^T, \ldots, \hat{\mathbf{n}}_{n_b}^T]^T$. Here for body i, $\mathbf{M}_i$

is the constant diagonal mass matrix, $\mathbf{f}_i$ is the vector of applied forces, $\bar{\mathbf{J}}_i$ is the constant diagonal inertia tensor, $\bar{\mathbf{n}}_i$ is the applied torque expressed in the local reference frame, and $\hat{\mathbf{n}}_i = \bar{\mathbf{n}}_i - \tilde{\bar{\omega}}_i \bar{\mathbf{J}}_i \bar{\omega}_i$. The notation convention that a vector quantity with an over-bar is represented in a local reference frame is observed in what follows. Likewise, the tilde operator ($\sim$) applied to a vector $\mathbf{x} \in I\!R^3$ produces the skew symmetric matrix associated with $\mathbf{x}$.

The time evolution of a mechanical system simultaneously satisfies the ODE in Eq.9 and the constraints in Eqs.2 and 3. This indicates that the numerical solution is obtained by solving a differential algebraic problem of index 3 [3]. There is a multitude of methods dedicated to the solution of the constrained equations of motion such as: stabilization methods [5], [2], projection methods [11], [10], [17], and state-space methods [25], [21]. The approach used in this paper is state-space based, and uses a subset of the generalized coordinates to express the time evolution of the whole mechanical system [25]. This *coordinate partitioning* (CP) method is robust and, used in the context of explicit integration, has proven to be very well suited for real-time simulation [14].

The starting point for the CP method is a partitioning of $\mathbf{q}$ in Eq.1 in dependent and independent coordinates $\mathbf{u} \in \mathbf{R}^m$, and $\mathbf{v} \in \mathbf{R}^{ndof}$, $ndof = n - m$. The partitioning is represented by two mappings $\nu : \{1, 2, \ldots, ndof\} \rightarrow S_{indep}$, and $\mu : \{1, 2, \ldots, m\} \rightarrow S_{dep}$, with $S_{indep} \cup S_{dep} = \{1, 2, \ldots, n\}$ and $S_{indep} \cap S_{dep} = \emptyset$. With this notation

$$\mathbf{v}[i] = \mathbf{q}[\nu(i)] \ , \ 1 \leq i \leq ndof, \text{ and } \mathbf{u}[j] = \mathbf{q}[\mu(j)] \ , \ 1 \leq j \leq m \quad (10)$$

The cornerstone of the CP method is the requirement that $\mathbf{u}$ is chosen such that the sub-Jacobian of the constraints is nonsingular

$$\det\left(\mathbf{\Psi}_{\mathbf{u}}(\mathbf{q}, t)\right) \neq 0 \quad (11)$$

where $\mathbf{\Psi}(\mathbf{q}, t) = \left[\mathbf{\Phi}_{\mathbf{u}}^T(\mathbf{q}, t), \mathbf{p}_1^T \mathbf{p}_1 - 1, \ldots, \mathbf{p}_{nb}^T \mathbf{p}_{nb} - 1\right]^T$.

To advance the numerical solution to time t_{s+1}, consider that at time t_s the generalized position $\mathbf{q}_s$, and velocity $\dot{\mathbf{q}}_s$, are available. The angular velocity $\bar{\omega}_s$ is obtained based on Eq.6, while the accelerations $\ddot{\mathbf{r}}_s$ and $\dot{\bar{\omega}}_s$, along with the Lagrange multipliers λ_s are computed as the solution of the linear system of Eqs.(7) and (9)

$$\begin{bmatrix} \mathbf{M} & \mathbf{0} & \eta^T(\mathbf{\Phi}, \mathbf{q}) \\ \mathbf{0} & \bar{\mathbf{J}} & \rho^T(\mathbf{\Phi}, \mathbf{q}) \\ \eta(\mathbf{\Phi}, \mathbf{q}) & \rho(\mathbf{\Phi}, \mathbf{q}) & \mathbf{0} \end{bmatrix} \begin{bmatrix} \ddot{\mathbf{r}} \\ \dot{\bar{\omega}} \\ \lambda \end{bmatrix} = \begin{bmatrix} \mathbf{f} \\ \hat{\mathbf{n}} \\ \tau(\mathbf{q}, \dot{\mathbf{q}}, t) \end{bmatrix} \quad (12)$$

where the subscript s was dropped for convenience. The generalized acceleration $\ddot{\mathbf{p}}_s$ is recovered as indicated in Eq.8. If h is the integration

step-size, the configuration at $t_{s+1} = t_s + h$ is found using an explicit multi-step formula [3] to integrate the independent accelerations $\ddot{\mathbf{v}}_s$; i.e., $\ddot{\mathbf{q}}_s\left(\nu(i)\right),\ 1 \leq i \leq ndof$, two consecutive times to obtain first the independent velocities $\dot{\mathbf{v}}_{s+1}$, and then the independent positions $\mathbf{v}_{s+1}$, respectively. Finally, the dependent generalized coordinates $\mathbf{u}_{s+1}$ are computed such that the condition $\mathbf{\Psi}(\mathbf{q}, t) = \mathbf{0}$ is satisfied, while the dependent generalized velocities $\dot{\mathbf{u}}_{s+1}$ are computed to satisfy the kinematic velocity constraint equations of Eq.4, and the time derivative of the constraints in Eq.2.

The outline of the paper is as follows. Section 2 introduces in a constrained optimization framework an algorithm for the dependent position and velocity recovery. Section 3 discusses for a proposed solution sequence the role of the linear algebra support. The solution to the acceleration computation, and to the stabilization problem cast in the constrained optimization framework are shown to require the solution of a set of associated reduced problems (positive definite linear systems of lower dimension). Section 4 presents a direct solution method for the reduced problems introduced in Section 3. Section 5 discusses a multi-threaded iterative solver for the reduced problems. Concluding remarks and future directions of research are outlined in Section 6.

2. Optimization-based position and velocity stabilization

2.1. Position Stabilization

Consider a configuration of the array of generalized coordinates $\mathbf{q}_{s+1}^{(0)} = \left[\left(\mathbf{r}_{s+1}^{(0)}\right)^T \ \left(\mathbf{p}_{s+1}^{(0)}\right)^T\right]^T$ of Eq.1 obtained after explicit integration of the generalized velocity $\dot{\mathbf{q}}_s$. Typically, the configuration $\mathbf{q}_{s+1}^{(0)}$ is not consistent; i.e., $\mathbf{\Psi}(\mathbf{q}_{s+1}^{(0)}, t_{s+1}) \neq \mathbf{0}$ and some entries in this vector must be modified to satisfy the kinematic constraints. In the framework of the CP method, only the dependent coordinates, i.e., the components of $\mathbf{u}_{s+1}$, are adjusted to ensure that Eqs.2 and 3 are satisfied.

The proposed approach handles the recovery of the dependent position $\mathbf{u}_{s+1}$ in a constrained optimization framework, where the constraints are precisely the position kinematic constraints $\mathbf{\Psi}(\mathbf{q}, t) = \mathbf{0}$, and the cost function $f(\mathbf{q})$ through large weights penalizes any attempt to change the value of an independent variable. Thus,

$$f(\mathbf{q}) = \frac{1}{2}\left[\begin{array}{cc}\left(\mathbf{r}-\mathbf{r}^{(0)}\right)^T & \left(\mathbf{p}-\mathbf{p}^{(0)}\right)^T\end{array}\right]\left[\begin{array}{cc}\mathbf{W_r} & \mathbf{0}\\ \mathbf{0} & \mathbf{W_p}\end{array}\right]\left[\begin{array}{c}\mathbf{r}-\mathbf{r}^{(0)}\\ \mathbf{p}-\mathbf{p}^{(0)}\end{array}\right] \quad (13)$$

where the subscript $s+1$ was dropped for convenience. With $\nu(k)$ defined in Eq.10, the weight matrix $\mathbf{W} = diag\,(\mathbf{W_r},\, \mathbf{W_p}) \in I\!R^{n\times n}$ is defined as follows: $\mathbf{W}(i,j) = 0$ if $i \neq j$, $\mathbf{W}(i,i) = 10^5$ if $(\exists)\ 1 \leq k \leq ndof$ such that $i = \nu(k)$, or $\mathbf{W}(i,i) = 1$ otherwise. The value 10^5 is meant to stress that a large weight penalizes any attempt to modify the value of an independent coordinate, and for that purpose, other large values can be considered as well.

The optimization problem is solved via a modified Newton method as a sequence of convex optimization problems. The constraints are first linearized to produce

$$\begin{aligned} \mathbf{\Phi}\left(\mathbf{r},\mathbf{p},t\right) &= \mathbf{\Phi}\left(\mathbf{r}^{(0)},\mathbf{p}^{(0)},t\right) + \mathbf{\Phi_r}\left(\mathbf{r}^{(0)},\mathbf{p}^{(0)},t\right)\left(\mathbf{r}-\mathbf{r}^{(0)}\right) \\ &+ \mathbf{\Phi_p}\left(\mathbf{r}^{(0)},\mathbf{p}^{(0)},t\right)\left(\mathbf{p}-\mathbf{p}^{(0)}\right) \\ \mathbf{p}_i^T\,\mathbf{p}_i - 1 &= \mathbf{p}_i^{(0)T}\mathbf{p}_i^{(0)} - 1 + 2\mathbf{p}_i^{(0)T}\left(\mathbf{p}_i-\mathbf{p}_i^{(0)}\right)\ , \quad 1 \leq i \leq n_b \end{aligned} \tag{14}$$

Since $\mathbf{\Phi_r} = \eta\left(\mathbf{\Phi},\mathbf{q}\right)$ and $\mathbf{\Phi_p} = \rho\left(\mathbf{\Phi},\mathbf{q}\right)\cdot 2\mathbf{G}$, introducing the variables $\mathbf{x} = \left[\mathbf{x}_1^T,\ldots,\mathbf{x}_{n_b}^T\right]^T$, $\mathbf{z} = \left[\mathbf{z}_1^T,\ldots,\mathbf{z}_{n_b}^T\right]^T$, α_i

$$\begin{aligned} \mathbf{x}_i &= \mathbf{r}-\mathbf{r}^{(0)} \in I\!R^3 \\ \mathbf{z}_i &= 2\mathbf{G}\left(\mathbf{p}_i-\mathbf{p}_i^{(0)}\right) \in I\!R^3 \\ \alpha_i &= 2\mathbf{p}_i^{(0)T}\left(\mathbf{p}_i-\mathbf{p}_i^{(0)}\right) \in I\!R \end{aligned} \tag{15}$$

the linearized constraint equations assume the equivalent form

$$\mathbf{\Phi}\left(\mathbf{r}^{(0)},\mathbf{p}^{(0)},t\right) + \eta\left(\mathbf{\Phi},\mathbf{q}\right)\mathbf{x} + \rho\left(\mathbf{\Phi},\mathbf{q}\right)\mathbf{z} = \mathbf{0} \tag{16a}$$

$$\mathbf{p}_i^{(0)T}\mathbf{p}_i^{(0)} - 1 + \alpha_i = 0, \quad 1 \leq i \leq n_b \tag{16b}$$

If $\mathbf{p}_i^{(0)}$, $1 \leq i \leq n_b$ are initially normalized, i.e., $\alpha_i = 0$, since $\mathbf{G}_i\mathbf{G}_i^T = \mathbf{I}_3$, and $\mathbf{G}_i\mathbf{p}_i^{(0)} = \mathbf{0}$ [13], from Eq.15 for a given value of $\mathbf{z}_i$,

$$\mathbf{p}_i - \mathbf{p}_i^{(0)} = \frac{1}{2}\mathbf{G}_i^T\mathbf{z}_i + \frac{1}{2}\alpha_i\mathbf{p}_i^{(0)} = \frac{1}{2}\mathbf{G}_i^T\mathbf{z}_i \tag{17}$$

Defining the block diagonal matrix $\mathbf{G} = diag\,(\mathbf{G}_1,\ldots,\mathbf{G}_{n_b})$, the optimization problem can be equivalently expressed in terms of the new variables $\mathbf{x}$ and $\mathbf{z}$ as the minimization of the cost function

$$g(\mathbf{x},\mathbf{z}) = \frac{1}{2}\begin{bmatrix}\mathbf{x}^T & \frac{1}{2}\mathbf{z}^T\mathbf{G}\end{bmatrix}\begin{bmatrix}\mathbf{W_r} & \mathbf{0}\\ \mathbf{0} & \mathbf{W_p}\end{bmatrix}\begin{bmatrix}\mathbf{x}\\ \frac{1}{2}\mathbf{G}^T\mathbf{z}\end{bmatrix} \tag{18}$$

subject to the linear constraint equation of Eq.16a. This is a convex optimization problem [6], whose optimum is reached at the solution of the linear system

$$\begin{bmatrix} \mathbf{W_r} & \mathbf{0} & \eta^T \\ \mathbf{0} & \frac{1}{4}\mathbf{G}\mathbf{W_p}\mathbf{G}^T & \rho^T \\ \eta & \rho & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{z} \\ \lambda_{\mathrm{P}} \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ -\mathbf{\Phi}\left(\mathbf{r}^{(0)}, \mathbf{p}^{(0)}, t\right) \end{bmatrix} \tag{19}$$

where λ_{P} is the Lagrange multiplier associated with the position optimization problem, and the dependency of η and ρ on $\mathbf{\Phi}$ and $\mathbf{q}^{(0)}$ was dropped for convenience. Since the weight matrices $\mathbf{W_r}$ and $\mathbf{W_p}$ are positive definite and $\mathbf{G}_i\mathbf{G}_i^T = \mathbf{I}_3$, the block diagonal matrix $diag(\mathbf{W_r}, \frac{1}{4}\mathbf{G}\mathbf{W_p}\mathbf{G}^T)$ is positive definite. Therefore, the linear system of Eq.19 is guaranteed to have a solution provided the constraints are linearly independent.

The algorithm for finding a consistent set of dependent coordinates $\mathbf{q}[\mu(j)]$, $1 \leq j \leq m$ at time t_{s+1} starts by changing the values obtained for these variables after explicit integration, such that the Euler parameterization constraints of Eq.2 hold. One iteration is concluded by finding the solution of the augmented linear system of Eq.19, a process that is discussed in Section 3. Once $\mathbf{x}$ and $\mathbf{z}$ are computed, a new corrected configuration is obtained as

$$\mathbf{r}_i = \mathbf{r}_i^{(0)} + \mathbf{x}_i \ , \ 1 \leq i \leq n_b \tag{20a}$$

$$\mathbf{p}_i = \mathbf{p}_i^{(0)} + \frac{1}{2}\mathbf{G}^T\mathbf{z}_i \ , \ 1 \leq i \leq n_b \tag{20b}$$

with the caveat that according to the coordinate partitioning algorithm employed here, only the dependent generalized coordinates $\mathbf{q}[\mu(j)]$, $1 \leq j \leq m$ are actually being updated.
The iterative process continues by loading the new values $\mathbf{r}$ and $\mathbf{p}$ into $\mathbf{r}^{(0)}$ and $\mathbf{p}^{(0)}$, respectively; i.e., $\mathbf{r} \mapsto \mathbf{r}^{(0)}$ and $\mathbf{p} \mapsto \mathbf{p}^{(0)}$. Then another iteration is carried out by following this two-stage process; i.e., normalization followed by correction with solution of optimization problem in Eq.19. The iterative process is stopped when the norm of the correction and that of the residual $\mathbf{\Phi}\left(\mathbf{r}^{(0)}, \mathbf{p}^{(0)}, t\right)$ are acceptably small, and the Euler parameterization constraints of Eq.2 are satisfied.

2.2. Velocity Stabilization

In an optimization framework similar to the one employed for position stabilization, a consistency argument leads to the requirement that the weight matrix for the velocity stabilization must be identical to the one used in the position stabilization cost function. Thus, the velocity optimization problem minimizes

$$f(\dot{\mathbf{q}}) = \frac{1}{2}\left[\begin{array}{cc}\left(\dot{\mathbf{r}}-\dot{\mathbf{r}}^{(0)}\right)^T & \left(\dot{\mathbf{p}}-\dot{\mathbf{p}}^{(0)}\right)^T\end{array}\right]\left[\begin{array}{cc}\mathbf{W_r} & \mathbf{0}\\ \mathbf{0} & \mathbf{W_p}\end{array}\right]\left[\begin{array}{c}\dot{\mathbf{r}}-\dot{\mathbf{r}}^{(0)}\\ \dot{\mathbf{p}}-\dot{\mathbf{p}}^{(0)}\end{array}\right] \quad (21)$$

subject to the velocity kinematic constraints

$$\mathbf{\Phi_r}(\mathbf{q},t)\,\dot{\mathbf{r}} + \mathbf{\Phi_p}(\mathbf{q},t)\,\dot{\mathbf{p}} + \mathbf{\Phi}_t(\mathbf{q},t) = \mathbf{0} \quad (22a)$$

and velocity parameterization constraints obtained by taking a time derivative of Eq.2

$$\dot{\mathbf{p}}_i^T\mathbf{p}_i = 0\ ,\ 1 \le i \le n_b \quad (22b)$$

Defining $\mathbf{x} = \dot{\mathbf{r}} - \dot{\mathbf{r}}^{(0)}$ and $\mathbf{z} = \bar{\omega}_i - \bar{\omega}_i^{(0)}$, based on Eq.6 the optimization problem is equivalently reformulated as the minimization of

$$f(\mathbf{x},\mathbf{z}) = \frac{1}{2}\left[\begin{array}{cc}\mathbf{x}^T & \frac{1}{2}\mathbf{z}^T\mathbf{G}\end{array}\right]\left[\begin{array}{cc}\mathbf{W_r} & \mathbf{0}\\ \mathbf{0} & \mathbf{W_p}\end{array}\right]\left[\begin{array}{c}\mathbf{x}\\ \frac{1}{2}\mathbf{G}^T\mathbf{z}\end{array}\right] \quad (23a)$$

subject to the constraint

$$\eta(\mathbf{\Phi},\mathbf{q})\,\mathbf{x} + \rho(\mathbf{\Phi},\mathbf{q})\,\mathbf{z} + \mathbf{\Phi}_t(\mathbf{q},t) = \mathbf{0} \quad (23b)$$

The optimum configuration for this problem is the solution of the augmented linear system

$$\left[\begin{array}{ccc}\mathbf{W_r} & \mathbf{0} & \eta^T(\mathbf{\Phi},\mathbf{q})\\ \mathbf{0} & \frac{1}{4}\mathbf{G}\mathbf{W_p}\mathbf{G}^T & \rho^T(\mathbf{\Phi},\mathbf{q})\\ \eta(\mathbf{\Phi},\mathbf{q}) & \rho(\mathbf{\Phi},\mathbf{q}) & \mathbf{0}\end{array}\right]\left[\begin{array}{c}\mathbf{x}\\ \mathbf{z}\\ \lambda_{\mathtt{V}}\end{array}\right] = \left[\begin{array}{c}\mathbf{0}\\ \mathbf{0}\\ -\mathbf{\Phi}_t(\mathbf{q},t)\end{array}\right] \quad (24)$$

where $\lambda_{\mathtt{V}}$ is the Lagrange multiplier associated with the velocity problem. For $1 \le i \le n_b$, the new velocities are computed as

$$\dot{\mathbf{r}}_i = \dot{\mathbf{r}}_i^{(0)} + \mathbf{x}_i \quad (25a)$$

$$\dot{\mathbf{p}}_i = \dot{\mathbf{p}}_i^{(0)} + \frac{1}{2}\mathbf{G}_i^T\mathbf{z}_i \quad (25b)$$

with the caveat that corrections are only applied to the dependent generalized velocities $\dot{\mathbf{q}}[\mu(j)]$, $1 \le j \le m$. The iterative algorithm continues by loading the new values $\dot{\mathbf{r}}$ and $\dot{\mathbf{p}}$ into $\dot{\mathbf{r}}^{(0)}$ and $\dot{\mathbf{p}}^{(0)}$, respectively, and proceeds by finding a new set of corrections. The process stops when the error in satisfying the conditions of Eqs.22a and 22b is small enough.

3. The solution sequence. The reduced problems.

In the explicit integration based CP algorithm, the time evolution of the mechanical system is determined as a sequence of solution cycles that advance in time the state of the model. Each solution cycle has four steps. Step 1 computes the system accelerations, $\ddot{\mathbf{r}}$ and $\dot{\bar{\omega}}$. In Step 2, the Euler parameter acceleration $\ddot{\mathbf{p}}_i$ is computed on a per body basis as indicated in Eq.8. Using an explicit multi-step formula, the accelerations $\ddot{\mathbf{r}}_i$ and $\ddot{\mathbf{p}}_i$ are integrated twice to obtain the velocity and position at time t_{s+1}. In Step 3 the dependent positions are modified to satisfy the position kinematic constraint equations. In Step 4 the dependent velocities obtained during Step 2 are adjusted for the new velocity configuration to satisfy the velocity kinematic constraint equations. At the end of Step 4, the simulation time is advanced and a new solution cycle is started.

The coefficient matrices that appear in Steps 1, 3, and 4; i.e., in the acceleration problem in Eq.12, and in the position and velocity stabilization problems in Eqs.19 and 24, share a similar structure. They are obtained by symmetrically augmenting a positive definite matrix with additional rows and columns. With $\eta = \eta(\mathbf{\Phi}, \mathbf{q})$ and $\rho = \rho(\mathbf{\Phi}, \mathbf{q})$, the solution of the linear system in Eq.12 is obtained by first computing the Lagrange multiplier $\lambda = \left[\lambda_1^T, \ldots, \lambda_{N_J}^T\right]^T$, as the solution of the linear system

$$\left(\eta \mathbf{M}^{-1} \eta^T + \rho \bar{\mathbf{J}}^{-1} \rho^T\right) \lambda = \eta \mathbf{M}^{-1} \mathbf{f} + \rho \bar{\mathbf{J}}^{-1} \hat{\mathbf{n}} - \tau \tag{26a}$$

and then readily obtaining the acceleration as the solution of the diagonal linear system

$$\begin{bmatrix} \mathbf{M} & \mathbf{0} \\ \mathbf{0} & \bar{\mathbf{J}} \end{bmatrix} \begin{bmatrix} \ddot{\mathbf{r}} \\ \dot{\bar{\omega}} \end{bmatrix} = \begin{bmatrix} \mathbf{f} \\ \hat{\mathbf{n}} \end{bmatrix} - \begin{bmatrix} \eta^T(\mathbf{\Phi}) \\ \rho^T(\mathbf{\Phi}) \end{bmatrix} \lambda \tag{26b}$$

The linear system in Eq.26a is called the *acceleration reduced system*, and its coefficient matrix, denoted by $\mathbf{E}_1 \in I\!R^{m \times m}$, is called the *acceleration reduced matrix*. Since the mass and inertia tensor $\mathbf{M}$ and $\bar{\mathbf{J}}$, respectively, are positive definite, the reduced matrix is positive definite as long as the constraints are independent.

According to Section 2, recovering the dependent positions and velocities during Steps 3 and 4 requires the solution of two different linear systems that share an identical augmented coefficient matrix. The corresponding reduced matrix is denoted by $\mathbf{E}_2$, and it is called the *stabilization reduced matrix*. Drawing a parallel to the acceleration computation, $\mathbf{E}_2$ is used to compute two sets of Lagrange multipliers $\lambda_{\mathtt{P}}$ and $\lambda_{\mathtt{V}}$, that correspond to the dependent position recovery, and

dependent velocity recovery, respectively. Following the nomenclature introduced for the acceleration computation, for the stabilization reduced matrix the role of the "mass" matrix is played by the weight matrix $\mathbf{W_r}$ associated with the translational coordinates, while the "inertia tensor" matrix is replaced by the matrix $\frac{1}{4}\mathbf{G}\mathbf{W_p}\mathbf{G}^T$, which is block diagonal and positive definite.
The strategy exercised during Step 3 for position recovery is similar to the one used for acceleration computation. Referring to Eqs.20a and 20b, the position corrections $\mathbf{x}$ and $\mathbf{z}$ are found as the solution of the block diagonal linear system

$$\begin{bmatrix} \mathbf{W_r} & \mathbf{0} \\ \mathbf{0} & \frac{1}{4}\mathbf{G}\mathbf{W_p}\mathbf{G}^T \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{z} \end{bmatrix} = - \begin{bmatrix} \eta^T(\mathbf{\Phi}) \\ \rho^T(\mathbf{\Phi}) \end{bmatrix} \lambda_{\mathtt{P}} \tag{27}$$

Finally, for the velocity stabilization problem during Step 4, the corrections in $\dot{\mathbf{r}}$ and $\dot{\mathbf{p}}$ are found as the solution of a system identical to the one in Eq.27, with the caveat that $\lambda_{\mathtt{V}}$ replaces $\lambda_{\mathtt{P}}$ in the right-hand side of the linear system.

Section 4 presents a direct solver for the solution of the reduced acceleration problem. For the same acceleration reduced problem, Section 5 introduces an iterative solver, which in turn uses the direct solver of Section 4 for preconditioning purposes. Although not discussed in detail, the same methodology; i.e., preconditioning followed by iterative solution, is used to solve the stabilization reduced problem to recover the dependent generalized positions and velocities.

4. Topology-based sparse direct solver

If two bodies, b_i and b_k, are connected through joint j, they are called *j-adjacent* bodies. Since there is an ordering relationship among the bodies of a system, of the two adjacent bodies, one has a lower index, and is called the left-body, or `l`-body, while the higher-index body is called the `r`-body. They are denoted by $\mathtt{l}(j)$, and $\mathtt{r}(j)$, respectively.
The b_i*-connectivity set* of a body b_i is defined as the union of all joints that link body b_i to other bodies in the system, and is denoted by $\zeta(b_i)$. The *joint index* $\mathcal{J}(b_i)$ of a body b_i is defined as the number of elements in $\zeta(b_i)$. The topology index $\mathcal{J}$ of a model is defined as the largest joint index of any body in the model.

For each joint j, there is a corresponding equation in the reduced system in which the coefficient matrix of λ_j is positive definite. This is called the defining equation for joint j, and assumes the expression

$$\eta^j_{\mathtt{l}(j)}\mathbf{M}^{-1}_{\mathtt{l}(j)}\sum_{l\in\zeta(\mathtt{l}(j))}\eta^{l^T}_{\mathtt{l}(j)}\lambda_l+\rho^j_{\mathtt{l}(j)}\bar{\mathbf{J}}^{-1}_{\mathtt{l}(j)}\sum_{l\in\zeta(\mathtt{l}(j))}\rho^{l^T}_{\mathtt{l}(j)}\lambda_l \tag{28}$$
$$+\quad \eta^j_{\mathtt{r}(j)}\mathbf{M}^{-1}_{\mathtt{r}(j)}\sum_{l\in\zeta(\mathtt{r}(j))}\eta^{l^T}_{\mathtt{r}(j)}\lambda_l+\rho^j_{\mathtt{r}(j)}\bar{\mathbf{J}}^{-1}_{\mathtt{r}(j)}\sum_{l\in\zeta(\mathtt{r}(j))}\rho^{l^T}_{\mathtt{r}(j)}\lambda_l \;=$$
$$\eta^j_{\mathtt{l}(j)}\mathbf{M}^{-1}_{\mathtt{l}(j)}\mathbf{f}_{\mathtt{l}(j)}+\rho^j_{\mathtt{l}(j)}\bar{\mathbf{J}}^{-1}_{\mathtt{l}(j)}\hat{\mathbf{n}}_{\mathtt{l}(j)}+\eta^j_{\mathtt{r}(j)}\mathbf{M}^{-1}_{\mathtt{r}(j)}\mathbf{f}_{\mathtt{r}(j)}+\rho^j_{\mathtt{r}(j)}\bar{\mathbf{J}}^{-1}_{\mathtt{r}(j)}\hat{\mathbf{n}}_{\mathtt{r}(j)}-\tau^j$$

Eliminating the subscript 1 for notational convenience, the positive definite coefficient matrix $\mathbf{E}_{jj}$ associated with λ_j in its defining equation assumes the form

$$\mathbf{E}_{jj}=\sum_{b\in\{\mathtt{l}(j),\mathtt{r}(j)\}}\left(\eta^j_b\mathbf{M}^{-1}_b{\eta^j_b}^T+\rho^j_b\bar{\mathbf{J}}^{-1}_b{\rho^j_b}^T\right) \tag{29}$$

Based on Eq.28, a Lagrange multiplier λ_l corresponding to joint l appears in the defining equation of joint j provided either $l\in\zeta(\mathtt{l}(j))$ (in which case let $b=\mathtt{l}(j)$), or $l\in\zeta(\mathtt{r}(j))$ (in which case let $b=\mathtt{r}(j)$). The coefficient matrix associated with λ_l in Eq.28 is

$$\mathbf{E}_{jl}=\eta^j_b\mathbf{M}^{-1}_b{\eta^l_b}^T+\rho^j_b\bar{\mathbf{J}}^{-1}_b{\rho^l_b}^T \tag{30}$$

The direct solution of the reduced system is obtained by repeatedly applying a two-stage process in an approach similar to the one proposed for open-loop topologies by [16]. First, a Lagrange multiplier λ_j is solved for in its defining equation, and then eliminated from the defining equation of any joint $l\in\zeta(\mathtt{l}(j))$. The first stage is called *isolation (I)*, and the second one is called *elimination (E)*. This process has a graphical interpretation presented for a simple model in Fig.1. In this figure, the topology of the model is represented as a graph in which bodies map into vertices, while the graph's edges, labeled using squares, correspond to the connecting joints. A broken arrow indicates the Lagrange multiplier isolated and eliminated during a particular step. After each *IE*-step, one edge of the graph vanishes, leading to a new topology that is subject to a new *IE*-step. At the end of IE_4, λ_4 becomes available, and then backward substitution computes λ_3, etc.

4.1. Performance Analysis

The performance analysis starts with an account of the total number of operations to compute the matrix $\mathbf{E}$. First note that since the reduced matrix is symmetric, only the block diagonal and the upper half entries of the matrix are computed. For rigid bodies with centroidal and principal reference frames, since $\mathbf{M}^{-1}_{b_k}$ and $\bar{\mathbf{J}}^{-1}_{b_k}$ are constant and diagonal

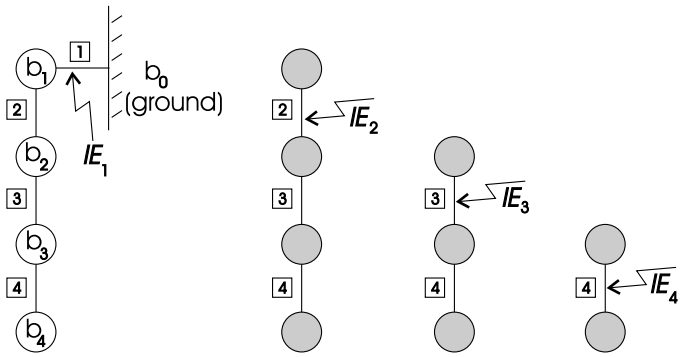


Figure 1. IE steps for multiple pendulum.

for any body b_k, the number of additions A_{jl} and multiplications M_{jl} required to compute an arbitrary term such as $\mathbf{E}_{jl}$ in Eq.30 is

$$\begin{aligned} A_{jl} &= 4m_l m_j \\ M_{jl} &= 6m_l m_j + 6\min(m_j, m_l) \end{aligned} \tag{31}$$

where for joint j, m_j represents the number of constraints induced by this joint. In order to simplify the operation count, assume that all joints have the same number of constraints denoted in what follows by c, $c \leq 6$. Then the contribution of the joints connected to a body b_k to the upper part of the reduced matrix $\mathbf{E}$ results in A_{b_k} additions and M_{b_k} multiplications

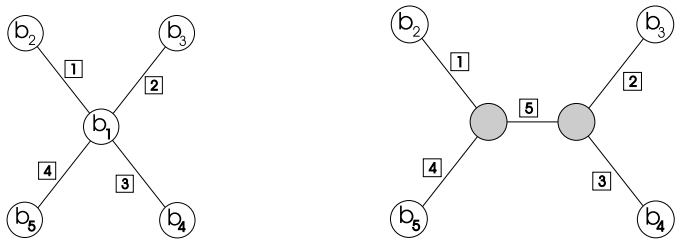
$$\begin{aligned} A_{b_k} &= 2c^2\mathcal{J}(b_k)\left(\mathcal{J}(b_k)+1\right) \\ M_{b_k} &= 3c(c+1)\mathcal{J}(b_k)\left(\mathcal{J}(b_k)+1\right) \end{aligned} \tag{32}$$

Overall, the number of additions A and multiplications M required to compute $\mathbf{E}$, is the sum over all bodies b_k in the model of A_{b_k} and M_{b_k}, respectively. In the worst case scenario, all joints would have $c = 6$ constraints, and the index of all bodies would be equal with the index of the mechanism $\mathcal{J}$. Therefore, an upper bound on the number of additions and multiplications is

$$\begin{aligned} A &= 72\mathcal{J}(\mathcal{J}+1)\, n_b \\ M &= 126\mathcal{J}(\mathcal{J}+1)\, n_b \end{aligned} \tag{33}$$

which indicates that the number of operations to compute $\mathbf{E}$ grows linearly with the number of bodies in the model. Note though that the topology index $\mathcal{J}$ can be anywhere from two for a chain of pendulums or the track of a vehicle, to $n_b - 1$ for star topologies.

As far as the factorization of the reduced matrix is concerned, for any open loop topology of index $\mathcal{J}$, an upper bound on the number of operations can be obtained by conservatively assuming that each



a. Actual b. Virtual

Figure 2. Index 4 Star. Before and after index reduction

IE-stage handles a joint that belongs to the connectivity set of a body of joint index $\mathcal{J}$. The *isolation* step requires one inversion and $\mathcal{J}$ multiplications, while the *elimination* step takes $\mathcal{J}^2$ multiplications and $\mathcal{J}(\mathcal{J}-1)$ additions. For full factorization, the algorithm will take N_J inversions and no more than $\mathcal{J}^2 \cdot N_J$ multiplications and $\mathcal{J}(\mathcal{J}-1) \cdot N_J$ additions. Note that for low-index topologies the algorithm is efficient, as the computational effort only increasing linearly and with a small constant slope with the dimension of the problem.

4.2. Topology Index Reduction

The performance of the algorithm in the proposed form is poor when dealing with star topologies such as in Fig.2.a. According to Eq.28, the reduced matrix for such a mechanical system is dense and solving the reduced system requires $O(N_J^3)$ operations. Qualitatively, it is clear that for star-like, open-loop mechanisms, the proposed algorithm does not fare well against the algorithm proposed in a Cartesian framework by [4], or for that matter with any algorithm that solves for generalized accelerations in a recursive formulation [9, 1]. The performance of the proposed algorithm can be improved by dividing the high-index body in an appropriate number of smaller virtual bodies connected by fixed joints; i.e., joints that remove all relative degrees of freedom between the two bodies. This operation reduces the topology index of the mechanism, while increasing the number of unknowns; i.e., Lagrange multipliers in the reduced system. In Fig.2.b., the shaded vertices correspond to the two virtual bodies created by dividing body b_1 in Fig.2.a. Likewise, the virtual joint 5 is a fixed joint holding the new parts together to ensure that the dynamics of the system remain unchanged. These new bodies and joints are called virtual because they do not have a physical counterpart. Their effect is a topology change for the sole purpose of leading to an equivalent, but simpler, reduced linear system.

Table I. Star topology operation count.

	Star Structure				*Index Reduction*			
N_J	A	M	I	NNZ	A	M	I	NNZ
3	10	10	6	6	10	10	6	6
4	22	22	10	10	20	20	11	11
5	40	40	15	15	30	30	16	16
6	65	65	21	21	40	40	21	21
7	98	98	28	28	50	50	26	26
8	140	140	36	36	60	60	31	31
16	920	920	136	136	140	140	71	71

The idea behind topology index reduction is that a star-like topology should be regarded as the result of a bad elimination sequence applied to a virtual mechanism. The effort for computing the Lagrange multipliers is expected to decrease by going back to this virtual mechanism via topology index reduction, and applying a better elimination sequence on its reduced matrix. Thus, the virtual mechanism whose topology graph is presented in Fig.2.b. can lead to a topology graph like in Fig.2.a., if the Lagrange multiplier associated with joint 5 was the first one subjected to an *IE*-step. If, on the other hand, the elimination order is $\{\lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5\}$, the number of additions and multiplications is reduced from 22 to 20. The results in Table I, where A stands for the number of block matrix additions, M stands for the number of block matrix multiplications, I stands for the number of block matrix inversions, and NNZ represents the number of non-zeros in the upper part of the symmetric reduced matrix $\mathbf{E}$, suggest that the larger the index of the mechanism, the more effective the topology index reduction idea becomes. What is important to note is that reducing the index by dividing the higher index bodies does not result in an increased computational effort to obtain the entries in the new reduced matrix. This is because the effort to obtain an entry in any reduced matrix is constant, and as results in Table I indicate, the number of nonzero entries actually becomes smaller after index reduction.

The operation counts in Table I indicate that when the proposed algorithm is applied to a star-topology, as anticipated, the number of operations for factorization is of the order $O(N_J^3)$. A second conclusion is that if index reduction is applied to a star-topology, the number of operations required for the factorization of the virtual reduced matrix increases only linearly with the dimension of the original model. Moreover, index reduction increases the degree in which a multi-threaded

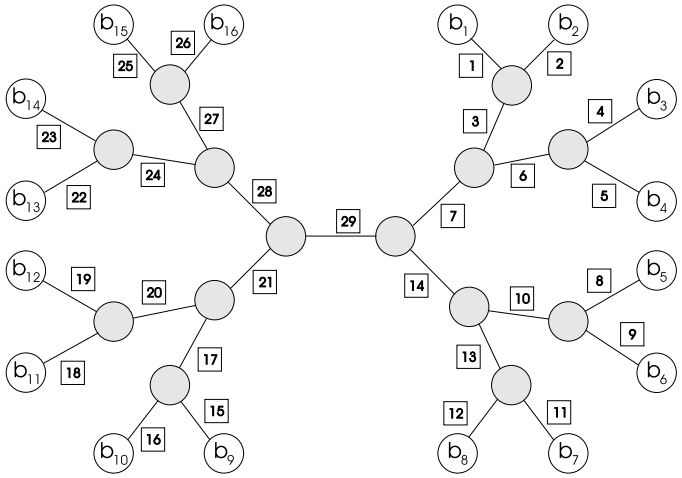


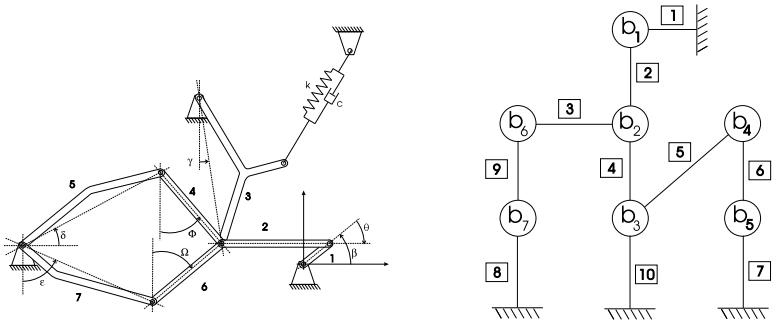
Figure 3. Index reduction for star-16 topology.

approach can be leveraged for solving the reduced system. For a star-like topology with $N_J = 16$ joints such as in Fig.3, index reduction allows the *IE*-process to be carried out simultaneously on λ_1, λ_3, λ_5, λ_7, λ_9, λ_{11}, λ_{13}, λ_{15}. Then the *IE*-process can be carried out in parallel for λ_2, λ_4, λ_6, λ_8, λ_{10}, λ_{12}, λ_{14}, λ_{16}. This argument can be continued, and the rule is that for a star topology with $N_J = 2^n$ constraints, after index reduction the first parallel *IE* step can be carried out through 2^{n-1} parallel threads in two stages, then the second *IE* step can be carried out through 2^{n-2} threads in two stages, and so on. The $n-1$ *IE* step is carried out on two threads in two stages, while the last *IE* step is carried out by one thread. Asymptotically, for a 2^n star topology, a multi-threaded elimination sequence can be carried out after index reduction in less time than required by the solution on a single thread of the reduced system corresponding to a model with $2n$ pendulums.

Concluding on the performance analysis, when applied for the solution of the reduced system associated with an open-loop mechanism, the direct algorithm requires $O(N_J)$ operations and results in no fill-in. If high-index bodies are present in the model, obtaining $O(N_J)$ performance calls for a preliminary index reduction step to bring the index of the mechanism to 3 or 4. For no fill-in, the IE step is recursively applied to any edge connected to a leaf vertex in the associated topology graph.

4.3. The relevance of the isolation–elimination sequence

It is important that a preliminary symmetric permutation $\mathbf{PEP}^T$ is applied to the reduced matrix $\mathbf{E}$ to optimize the fill-in and the compu-



a. Model b. Topology graph

Figure 4. Seven body mechanism

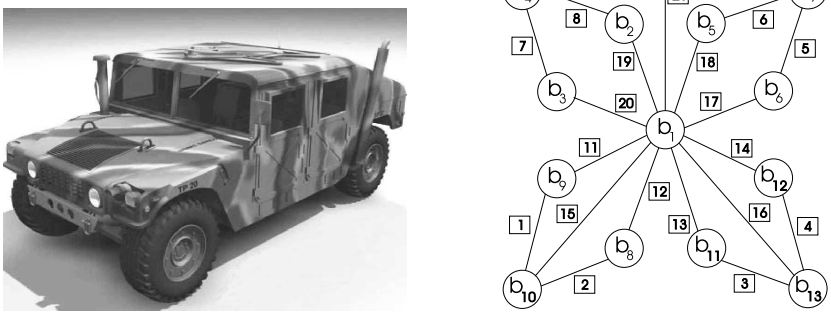
Table II. Seven-body mechanism computational effort.

Elimination Sequence	A	M	I	F
1 2 7 6 5 8 9 3 4 10	35	35	21	0
4 2 3 5 6 9 1 7 8 10	111	130	40	19

tational effort. From a physical perspective, this permutation translates into a renumbering of the joints present in the mechanical system, a task that is done only once before the beginning of the simulation. Reordering the unknowns to minimize the amount of computation during factorization is an NP complete problem, and the reordering is therefore based on some heuristic [18, 15, 22].

In Fig.4, a seven-body mechanism [23] is represented along with one possible topology graph associated with this mechanism. For this model, two different elimination sequences result in different levels of computational effort for the solution of the reduced system, as indicated in Table. II. The operation count indicates that the slowdown caused by picking a bad elimination sequence is about 200%.

To further underline the importance of the elimination sequence in the overall performance of the algorithm, consider the example of a High Mobility Multi–Wheeled Vehicle (HMMWV) presented in Fig.5.a. In Fig.5.b., the vehicle is modeled using 14 bodies. Vertex number 1 is the chassis, 2 and 5 are the right and left front upper control arms, 3 and 6 are the right and left front lower control arms, 9 and 12 are the right and left rear lower control arms, and 8 and 11 are the right and left rear upper control arms. Bodies b_4, b_7, b_{10}, and b_{13} are the wheel spindles, and body b_{14} is the steering rack. Joints 1 through 8, 18, and



a. Picture b. Topology graph

Figure 5. HMMWV example

Table III. HMMWV reduced system solution effort.

Elimination Sequence	*A*	*M*	*I*	*F*	*NNZ*
Bad	1240	1336	195	96	99
Good	459	469	109	10	99
Index reduction	220	233	90	13	77

19 are spherical, 9, 10, 15, 16 are distance constraints, 11 through 14, 17, and 20 are revolute, and 21 is translational. The topology index of this model is $\mathcal{J} = 11$.

Since the number of joints used to model the vehicle is 21, an exhaustive search to determine the best and worst elimination sequences is impractical. For comparing two alternatives, common sense was used to define two elimination sequences. A "Bad" sequence eliminates the Lagrange multipliers in the order $\{\lambda_{21}, \lambda_{20}, \lambda_{19}, \lambda_{18}, \lambda_{17}, \lambda_{11}, \lambda_{12}, \lambda_{13}, \lambda_{14}, \lambda_{15}, \lambda_{16}, \lambda_9, \lambda_{10}, \lambda_7, \lambda_8, \lambda_5, \lambda_6, \lambda_1, \lambda_2, \lambda_3, \lambda_4\}$, while a "Good" sequence eliminates the Lagrange multipliers in the order $\{\lambda_1, \lambda_2, \ldots, \lambda_{21}\}$. The number of additions A, multiplications M, inversions I, resulting fill-in F, and initial nonzero entries *NNZ* for these alternatives is provided in Table III. The last row of the table contains results obtained after topology index reduction is applied to the HMMWV model, a process that leads to the model in Fig.6. The topology index of the virtual model is $\mathcal{J} = 4$. In this example, the topology index reduction is equivalent to a partitioning of the chassis body in several smaller virtual bodies interconnected by fixed joints.

The results in Tables II and III indicate that the topology of the mechanism and the order in which the *IE*-sequence is carried out are

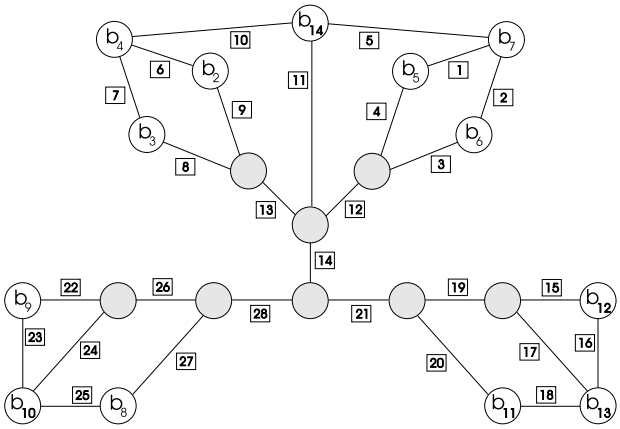


Figure 6. Index reduction applied to HMMWV.

the most important factors in determining the computational effort associated with the direct solution of the reduced system. The connectivity of the bodies in the system, as measured by the topology index, directly influences the reduced matrix computation and factorization effort. This effort is affected to a lesser extent by the presence of closed loops in the topology, which only guarantees that a certain amount of fill-in is inevitable. This is unlike the case when a recursive formalism is employed for the dynamic analysis of a mechanical system, [8] where for good performance, the lack of closed loops is by far more important than the topology index.

5. Topology based multi-threaded iterative solver

With the reduced matrices $\mathbf{E}_1$ and $\mathbf{E}_2$ positive definite, the algorithm of choice for the solution of the acceleration and position-velocity reduced problems is the preconditioned conjugate gradient [12]. The algorithm scales well on Single Instruction Multiple Data (SIMD) architectures, and it is guaranteed to converge within m iterations, where m is the number of constraints in the mechanism. Fast convergence; i.e., in significantly less than m iterations, depends on the quality of the preconditioning. During iteration k, the preconditioning amounts to finding the solution of a linear system $\mathbf{W}\mathbf{c}_k = \mathbf{e}_k$, where $\mathbf{W}$ is an old reduced matrix $\mathbf{E}_1$ ($\mathbf{E}_2$ for the position-velocity problem), evaluated at a previous time step. Thus, preconditioning amounts to a direct solution of the reduced problem, an issue discussed in Section 4.

For the acceleration problem, the focus below is on evaluating terms of the form $\hat{\mathbf{e}}_k = \mathbf{E}_1\mathbf{d}_k \in I\!R^m$ and $\hat{e}_k = \mathbf{d}_k^T\mathbf{E}_1\mathbf{d}_k \in I\!R$, in the context of

parallelizing the algorithm on a per body basis using a shared memory framework provided by the OpenMP standard [7]. Note that $\hat{\mathbf{e}}_k$ and $\hat{e}_k$ are the only quantities required by the preconditioned conjugate gradient algorithm to compute the solution of the reduced problem [19].

For the vector $\mathbf{d}_k \in I\!R^m$, the components that are multiplying the columns corresponding to joint j in the reduced matrix $\mathbf{E}$ are denoted by $\mathbf{d}_k^{(j)}$. Similarly, the contribution corresponding to joint j in the matrix-vector product $\mathbf{E}\mathbf{d}_k$ and the scalar $\hat{e}_k$ are denoted by $\hat{\mathbf{e}}_k^{(j)}$ and $\hat{e}_k^{(j)}$, respectively. Defining for body $\mathtt{l}(j)$ the quantities

$$\begin{aligned}
\mathbf{f}^{\mathtt{C},\mathtt{l}(j)} &= \textstyle\sum_{l\in\zeta(\mathtt{l}(j))} \eta_{\mathtt{l}(j)}^{l^T}\mathbf{d}_k^{(l)} & \mathbf{n}^{\mathtt{C},\mathtt{l}(j)} &= \textstyle\sum_{l\in\zeta(\mathtt{l}(j))} \rho_{\mathtt{l}(j)}^{l^T}\mathbf{d}_k^{(l)} \\
\mathbf{a}_{\mathtt{T}}^{\mathtt{C},\mathtt{l}(j)} &= \mathbf{M}_{\mathtt{l}(j)}^{-1}\mathbf{f}^{\mathtt{C},\mathtt{l}(j)} & \mathbf{a}_{\mathtt{R}}^{\mathtt{C},\mathtt{l}(j)} &= \bar{\mathbf{J}}_{\mathtt{l}(j)}^{-1}\mathbf{n}^{\mathtt{C},\mathtt{l}(j)} \\
\mathbf{\Pi}_{\mathtt{T}}^{j,\mathtt{l}(j)} &= \eta_{\mathtt{l}(j)}^{j}\mathbf{a}_{\mathtt{T}}^{\mathtt{C},\mathtt{l}(j)} & \mathbf{\Pi}_{\mathtt{R}}^{j,\mathtt{l}(j)} &= \rho_{\mathtt{l}(j)}^{j}\mathbf{a}_{\mathtt{R}}^{\mathtt{C},\mathtt{l}(j)}
\end{aligned}$$

according to Eq.28, for a joint j

$$\hat{\mathbf{e}}_k^{(j)} = \mathbf{\Pi}_{\mathtt{T}}^{j,\mathtt{l}(j)} + \mathbf{\Pi}_{\mathtt{R}}^{j,\mathtt{l}(j)} + \mathbf{\Pi}_{\mathtt{T}}^{j,\mathtt{r}(j)} + \mathbf{\Pi}_{\mathtt{R}}^{j,\mathtt{r}(j)} \tag{34}$$

where for body $\mathtt{r}(j)$ the terms $\mathbf{\Pi}_{\mathtt{T}}^{j,\mathtt{r}(j)}$ and $\mathbf{\Pi}_{\mathtt{R}}^{j,\mathtt{r}(j)}$ are defined as for body $\mathtt{l}(j)$. The quantities $\mathbf{f}^{\mathtt{C},\mathtt{l}(j)}$ and $\mathbf{n}^{\mathtt{C},\mathtt{l}(j)}$ are interpreted as the resultant reaction force and torque, respectively, with which the joints adjacent to body $\mathtt{l}(j)$ act on this body. Likewise, the quantities $\mathbf{a}_{\mathtt{T}}^{\mathtt{C},\mathtt{l}(j)}$ and $\mathbf{a}_{\mathtt{R}}^{\mathtt{C},\mathtt{l}(j)}$ qualitatively are regarded as translational and rotational constraint accelerations of body $\mathtt{l}(j)$, while $\mathbf{\Pi}_{\mathtt{T}}^{j,\mathtt{l}(j)}$ and $\mathbf{\Pi}_{\mathtt{R}}^{j,\mathtt{l}(j)}$ are the projection of these two accelerations onto the range of the jacobian matrices associated with joint j; i.e., the jacobian matrices that correspond to the translation and rotation components of the $\mathtt{l}(j)$ body, respectively.

Half of the computational effort[1] to obtain $\hat{\mathbf{e}}_k^{(j)}$ and $\hat{e}_k^{(j)}$ is done by the thread assigned to body b_i in three stages, where b_i is such that $j \in \zeta(b_i)$. The first stage computes $\mathbf{f}^{\mathtt{C},b_i}$ and $\mathbf{n}^{\mathtt{C},b_i}$, in a process that takes $6\mathcal{C}(b_i)$ multiplications and $6(\mathcal{C}(b_i) - \mathcal{J}(b_i))$ additions, where the constraint index of body b_i is defined as $\mathcal{C}(b_i) = \sum_{l\in\zeta(b_i)} m_l$. The second stage computes the constraint accelerations $\mathbf{a}_{\mathtt{T}}^{\mathtt{C},b_i}$ and $\mathbf{a}_{\mathtt{R}}^{\mathtt{C},b_i}$, which for rigid bodies requires a total of 6 more multiplications. Note that $\hat{e}_k^{(j)} = \mathbf{f}^{\mathtt{C},b_i\,T}\mathbf{a}_{\mathtt{T}}^{\mathtt{C},b_i} + \mathbf{n}^{\mathtt{C},b_i\,T}\mathbf{a}_{\mathtt{R}}^{\mathtt{C},b_i}$ is obtained after the first two stages at a price of 6 multiplications and 5 additions, and that the first two stages are build-up stages and they are body b_i specific rather than being specific to a particular joint $l \in \zeta(b_i)$. It is only the third stage that projects the body constraint accelerations that is joint specific. To

[1] The other half is done by the thread associated with the second body to which the joint j is connected.

carry out the projection of body b_i constraint acceleration on each of the adjacent joints, a total of $6\mathcal{C}(b_i)$ multiplications and $5\mathcal{C}(b_i)$ additions are required. Thus, for body b_i the final tally for stages one through three results in a number of $12\mathcal{C}(b_i)+12$ multiplications and $11\mathcal{C}(b_i)-5\mathcal{J}(b_i)+5$ additions, which indicates that the computational effort per body and per iteration is linear in the constraint index corresponding to that body.

The end goal of the algorithm is the computation of the body accelerations. This computation can be carried out in parallel for all bodies

$$\dot{\mathbf{v}}_{b_i} = \mathbf{M}_{b_i}^{-1}\mathbf{f}^{b_i} - \mathbf{a}_{\mathtt{T}}^{\mathtt{C},b_i} \qquad \dot{\bar{\omega}}_{b_i} = \bar{\mathbf{J}}_{b_i}^{-1}\hat{\mathbf{n}}^{b_i} - \mathbf{a}_{\mathtt{R}}^{\mathtt{C},b_i} \tag{35}$$

where according to Eq.9, $\mathbf{f}^{b_i}$ and $\hat{\mathbf{n}}^{b_i}$ are the components corresponding to body b_i in the active force $\mathbf{f}$ and torque $\hat{\mathbf{n}}$, respectively.

Note that for the iterative solver the existence of closed loops in the model is irrelevant. Each body-thread in the iterative solver manipulates information regarding the associated body and its adjacent joints, and disregards any global connectivity properties associated with the model at large. Therefore, the computational effort per body-thread and per iteration depends on the body constraint index, which suggests that load balancing is obtained when the bodies have identical or close constraint indexes $\mathcal{C}(b_i)$. Thus, the topology index reduction employed for the direct solver helps the iterative solver as well, as it both reduces the load per body and balances the thread load.

6. Conclusions

A solution method is proposed that in the Cartesian formulation framework exploits the potential for parallelism both at the equation formulation and numerical solution levels. The multi-threaded attribute of the proposed method draws upon the mapping of each body on a simulation thread, and it is the cornerstone of the proposed solution method. Each body-thread starts with the computation of specific kinetic and kinematic quantities, and continues through the numerical solution; i.e., through the iterative solvers, and numerical integration. The good scalability of the solution method is anticipated to support efficient simulation of complex mechanical systems such as trucks, tracked vehicles, and other large closed-loop distributed, i.e., non-chain like mechanical systems.

The acceleration computation, and in the context of a coordinate partitioning approach, the dependent position and velocity stabilization rely on the solution of positive definite reduced systems. Sharing an identical

structure, these reduced systems are solved via preconditioned conjugate gradient iterative algorithms. For large models, in a distributed computational framework managed by the Message Passing Interface (MPI) standard [20], the preconditioning of the iterative solvers can be delegated to different machines. For open-loop topologies, the preconditioning task is shown to require order $O(N_J)$ effort, provided a topology index reduction is first applied to models that display topology indexes higher than 4. In addition to improving the efficiency in preconditioning, the topology index reduction balances and reduces the thread-load in the iterative algorithms.

In terms of open problems, the parallel direct solution of the reduced system remains to be investigated. A faster direct solution leads to more often updates of the preconditioners, which in turn will improve the convergence speed in both the acceleration computation and the dependent coordinate stabilization. Likewise, it remains to further investigate the suitability of the iterative approach for the simulation of mechanical systems with contact and impact in the presence of low to moderate friction, when the topology of the system changes in time.

Acknowledgements

MSC.Software and former Mechanical Dynamics, Inc., are acknowledged for their openness and support that allowed this work to be presented and published outside the boundaries of the company. The author would like to thank Mihai Anitescu, Andrei Schäffer, and Martin Arnold for their suggestions and remarks that indeed helped improve this work.

References

1. Anderson, K.S., and S. Duan: 1999,'A Hybrid Parallelizable Low Order Algorithm for Dynamics of Multi-Rigid-Body Systems: Part I, Chain Systems', *Journal of Mathematical and Computer Modelling* **Vol. 30**, pp.193–215.
2. Ascher, U.M., H. Chin, and S. Reich: 1994,'Stabilization of DAEs and invariant manifolds', *Numerische Mathematik*, **Vol.67 no.2**, pp.131–149.
3. Ascher, U.M. and L. Petzold:*Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations*.SIAM, 1998.
4. Baraff, D.: 1996,'Linear-Time Dynamics using Lagrange Multipliers', *COMPUTER GRAPHICS Proc., Annual Conference Series*, pp.137–146.
5. Baumgarte, J.: 1972, 'Stabilization of constraints and integrals of motion in dynamical systems'*Comp. Meth. in Appl. Mech. and Eng.*, **Vol.1**,pp.1–16.
6. Bertzekas, D.:*Nonlinear Programming*. Belmont, MA: Athena Scientific, 1995.

7. Chandra, R., R. Menon, L. Dagum, D. Kohr, D. Maydan, and J. McDonald: *Parallel Programming in OpenMP*, Morgan Kaufmann Publishers, 2000.
8. Featherstone, R.: 1983,'The calculation of robot dynamics using articulated-body inertias',*International Journal of Robotics Research*, **Vol.2 no.1**,pp.13–30.
9. Featherstone, R.: *Robot Dynamics Algorithms*, Kluwer, 1987.
10. Fuhrer, C., and B. Leimkuhler: 1991, 'Numerical Solution of Differential-Algebraic Equations for Constrained Mechanical Motion, *Numerische Mathematik*, **Vol.59**, pp.55–69.
11. Gear, W.C., G. Gupta, and B. Leimkuhler: 1985, 'Automatic Integration of the Euler-Lagrange Equations with Constraints', *J. Comp. Appl. Math.*, **Vol.12**, pp.77–90.
12. Golub, G.H. and C.F. Van Loan: *Matrix Computations*, Second Edition. The Johns Hopkins University Press, Baltimore, 1989.
13. Haug, E.J.:*Computer Aided Kinematics and Dynamics of Mechanical Systems*. Allyn and Bacon, Boston, London, Sydney, Toronto, 1989.
14. Haug, E. J., D. Negrut, R. Serban, and D. Solis: 1999, 'Numerical Methods for High Speed Vehicle Dynamic Simulation',*Mech. Struct. & Mach.*, **Vol.27**, pp.507–533.
15. Karypis, G., and V. Kumar: 1994, 'A High Performance Sparse Cholesky Factorization Algorithm for Scalable Parallel Computers', *Tech. Report 94-41*, Department of Computer Science, University of Minnesota.
16. Lubich, C. U. Nowak, U. Pöhle, and C. Engstler: 1992 MEXX–Numerical software for the integration of constrained mechanical multibody systems, *Techical Report SC 92–12, ZIB Berlin, Germany.*
17. Lubich, C., U. Nowak, U. Pöhle, and C. Engstler: 1993, 'An Overview of MEXX: Numerical Software for Integration of Multibody Systems',*Advanced Multibody System Dynamics*, W. Schiehlen (ed.), pp.421–426.
18. Markowitz, H.: 1957, 'The elimination form of the inverse and its application to linear programming', *Management Science*, **Vol.3**, pp.255–269.
19. Negrut, D.: 2002, 'On the Issue of Iterative Linear Algorithms for the Multi-Threaded Simulation of Mechanical Systems Represented in Cartesian Coordinates', *Virtual Nonlinear Multibody Systems, NATO Advanced Study Institute*, W. Schiehlen, M. Valasek (eds.), **Vol.1**, pp.144–149.
20. Pacheco, P.: *Parallel Programming with MPI.* Morgan Kaufmann Publishers, 1996.
21. Potra, F., and W.C. Rheinboldt: 1991, 'On the numerical solution of the Euler-Lagrange equations', *J. Mech. Struct. Mach.*, **Vol.19**, pp.1–18.
22. Rothberg, E., and B. Hendrickson: 1998, 'Sparse Matrix Ordering Methods for Interior Point Linear Programming',*Informs. J. Comput.* **Vol.10 no.1**, pp107–113.
23. Schiehlen, W. (ed.): *Advanced Multibody System Dynamics - Simulation and Software Tools*, Kluwer Academic Publishers, Dordrecht,1993.
24. Shabana, A.A.: *Dynamics of Multibody Systems.* Cambridge University Press, Cambridge, New York, Melbourne, 1998.
25. Wehage, R. and E. Haug: 1982,'Generalized Coordinate Partitioning for Dimension Reduction in Analysis of Constrained Dynamic Systems', *ASME J. Mech. Design*, **Vol.104 no.1**, pp.247–255.