JOURNAL OF GUIDANCE, CONTROL, AND DYNAMICS
Vol. 23, No. 2, March–April 2000

Highly Parallelizable Low-Order Dynamics Simulation Algorithm for Multi-Rigid-Body Systems

Kurt S. Anderson* and Shanzhong Duan†
Rensselaer Polytechnic Institute, Troy, New York 12180-3590

A new efficient procedure is presented for the determination of the dynamic equations of motion for complex multibody systems and their subsequent temporal integration using parallel computing. The method is applicable to general systems of rigid bodies, which may contain arbitrary joint types, multiple branches, and/or closed loops. The method is based on the explicit determination of constraint forces at key joint locations and the subsequent highly efficient determination of system state time derivatives. The algorithm uses a novel hybrid direct and iterative solution scheme that allows a substantially higher degree of parallelization than is generally obtainable using the more conventional recursive $\mathcal{O}(N)$ procedures. It is shown that at the coarsest level the parallelization obtainable easily exceeds that indicated by the topology of the system. The procedure can produce a theoretical time optimal $\mathcal{O}(\log_2 N)$ performance for chain systems on computational throughput with $\mathcal{O}(N)$ processors. Numerical results indicate that this procedure performs particularly well relative to other parallel multibody formulations in situations where the number of degrees of freedom are large and the number of bodies that make up the system is greater than the number of computing processors available, the multibody systems involve branches, or the applied forces within the system are rapidly changing or are discontinuous. An estimate for the parallel efficiency of the algorithm is obtained by combining a theoretical bound for parallel complexity with an approximated overhead cost for parallel implementation.

Nomenclature

A^{ik} = matrix representation of both angular acceleration of body ik and the acceleration of its mass center
$\bar{A}^{ik}$ = all terms of A^{ik} explicit in unknown second time derivatives of generalized coordinates q
A_t^{ik} = all terms of accelerations matrix A^{ik} that are not contained in $\bar{A}^{ik}$
a^{ik} = matrix representation (appropriate basis) of acceleration of the center of mass of ith body of the kth subsystem
B_T^{ik} = coefficient matrix that relates constraint loads acting directly on the terminal bodies of subsystem i to body ik
B_0^{ik} = coefficient matrix that relates constraint loads acting directly on base body of subsystem i to body ik
C = system constraint Jacobian
D = portion of velocity level constraint equations that is associated with specified motions
dist$[i]$ = set of cut joint identification numbers that are distal to cut joint i
F_c = global constraint force matrix
$(F_c)_i$ = element of global constraint force matrix F_c associated with the ith (global numbering) cut joint
F^{ik} = matrix representation of resultant force system associated with mass center of body ik that accounts for R^{ik} as well as all inertia force terms of body ik not explicit in $\ddot{q}^{ik}$
$\hat{F}^{ik}$ = composite generalized applied force matrix associated with body ik
F_c^{ik} = matrix representation of the resultant constraint load system associated with the center of mass of body ik
$\hat{F}_c^{ik}$ = composite generalized constraint force matrix associated with body ik
f_c^A = constraint force magnitude acting on some point A
I^{ik} = generalized central inertia matrix of body ik, associated with local ik basis
$\hat{I}^{ik}$ = composite generalized central inertia matrix of body ik
$\mathfrak{I}_{ik}$ = matrix of generalized active forces associated with the degrees of freedom of joint ik
$\mathfrak{I}_{ik}^*$ = matrix of generalized inertia forces associated with the degrees of freedom of joint ik
ik = index associated with kth body of the ith subsystem
M = system mass matrix
M_{kk}^i = diagonal element from kth row of mass matrix associated with subsystem i
m = total number of constraint equations
N = total number of bodies in the system
N^i = total number of bodies in the ith subsystem
n = total number of degrees of freedom for the system
n^i = total number of degrees of freedom for the ith subsystem
P^{ik} = partial velocity (free modes) matrix associated with degrees of freedom of the kth body of subsystem i
pr$[i]$ = set of cut joint identification numbers that are proximal to cut joint i
Q = generalized force matrix
q_I = global generalized coordinate matrix
q_{II} = generalized coordinate first time derivative matrix
q^{ik} = generalized coordinate matrix associated with the degrees of freedom of joint ik
R^{ik} = matrix representation of resultant applied forces system associated with the center of mass of body ik
$r^{i(k-1)k}$ = matrix representation of position vector from mass center of body $i(k-1)$ to the mass center of body ik
$r^{i(k-1)K}\times$ = matrix representation of cross product operation between vector $\vec{r}^{i(k-1)k}$ and any other vector
S^{ik} = linear transformation matrix that converts one force system associated with the center of mass of body ik to an equivalent force system associated with a point of body ik that is instantaneously

Received 15 June 1998; revision received 23 March 1999; accepted for publication 7 July 1999.

*Assistant Professor, Department of Mechanical Engineering, Aeronautical Engineering, and Mechanics; anderk5@rpi.edu. Member AIAA.
†Graduate Student, Department of Mechanical Engineering, Aeronautical Engineering, and Mechanics.

coincident with the mass center of the body proximal to ik

T^{ik} = triangularization operator; linear transformation matrix that triangularizes the equations of motion as one recursively works from body ik to body i pr[k]; used in the construction of composite inertia matrices and composite force matrices

$\hat{T}^{ik}$ = composite triangularization operator

t = time

t_c = constraint torque magnitude

U = identity matrix

v^{ik} = velocity of body ik mass center in the Newtonian reference frame

v_r^{ik} = rth partial velocity of mass center of body ik

α = parallel computation inter processor communication setup cost expressed in terms of floating-point operations (FLOPs)

α^{ik} = matrix representation (appropriate basis) of angular acceleration of ith body of kth subsystem

β = parallel computation inter processor communication data transfer cost expressed in terms of FLOPs/byte

Γ = global constraint force coefficient matrix

$(\Gamma)_{ij}$ = elements of global constraint force coefficient matrix Γ associated with the jth set of constraint loads and the ith set of constraint equations

ζ^{ik} = portion of $\ddot{q}^{ik}$ that is explicit in the unknown constraint load measure numbers

$\bar{\zeta}^{ik}$ = intermediate variable matrix used in the recursive determination of ζ^{ik}

η^{jk} = portion of $\ddot{q}^{ik}$ that is not explicit in constraint load measure numbers

$\bar{\eta}^{jk}$ = intermediate variable matrix used in the recursive determination of η^{jk}

λ = generalized constraint force

ν = general scalar quantity

$\boldsymbol{\nu}$ = general vector quantity or dyadic quantity

Φ = algebraic constraint equation applied to set of subsystems so that they behave as original system

χ^{ik} = orthogonal compliment of partial velocity matrix P^{ik}

Ψ = right-hand side of linear system of constraint equations

$(\Psi)_i$ = elements of right-hand side of linear system of constraint equations Ψ associated with the ith set of algebraic constraint equations

ω^{ik} = angular velocity of body ik in the Newtonian reference frame

ω_r^{ik} = rth partial angular velocity (angular free mode) of body ik

${}^{jl}\omega^{ik}$ = angular velocity of body ik relative to body jl

I. Introduction

THE dynamical analysis determining the behavior of devices and structures that can be modeled as systems of interconnected bodies plays a major role in the design and operation of such devices. As a result, the field of algorithm development for the computer simulation of the dynamic behavior of such systems has received considerable attention since the pioneering work of Hooker and Margulies.[1] In many situations computational efficiency, which manifests itself in the form of computational speed, is of paramount importance. As a result of this need for fast, general simulation capability, significant effort has been put forward by individuals whose interests lay in varied fields such as robotics and aerospace, yielding algorithms that are both general and computationally efficient. Excellent discussions of many of these formulations are presented by Jain[2] and Banerjee.[3]

A. Basic Concepts

Most dynamic formulations generally fall into one of two forms. These forms are the state-space representation

$$\dot{q}_I = q_{II} \tag{1a}$$

$$M(t, q_I)\dot{q}_{II} = Q(t, q_I, q_{II}) \tag{1b}$$

and the descriptor form[4] representation

$$\dot{q}_I - q_{II} = 0 \tag{2a}$$

$$M(t, q_I)\dot{q}_{II} + C^T(t, q_I)\lambda - Q(t, q_I, q_{II}) = 0 \tag{2b}$$

$$\Phi(t, q_I) = 0 \tag{2c}$$

where

$$\left(\frac{\mathrm{d}\Phi}{\mathrm{d}t}\right) = C(t, q_I)q_{II} + D = 0 \tag{2d}$$

In each of these sets of equations, Q is the matrix containing contributions of applied loads, body loads, as well as centripetal and Coriolis acceleration inertia loads contributions, with C being the constraint Jacobian associated with the time derivative of the algebraic constraints conditions (2c).

Equations (1) are formed in terms of independent position and velocity state variables q_I and q_{II}, respectively. These state variables represent a set of the minimum possible number of variables needed to characterize the state of the system. By comparison, in Eq. (2) the variables q_I and q_{II} are redundant, most often representing the position, the orientation, and their respective derivatives in some absolute coordinate system. Because of this redundancy, generalized constraint forces λ must be applied to the system equations of motion (2b) to enforce the algebraic constraint equations (2c).

Much effort has been expended by various investigators, some pursuing the state-space representation,[3,5–20] whereas others are pursuing the descriptor representation;[21–23] with each of these basic representations offering its own strengths and weaknesses. In the descriptor case, the mass matrix M is of relatively large dimension, but may be extremely sparse. The effort required to generate these equations is small; the approach lends itself well to problems involving closed kinematical loops; and due to the sparseness of M, special sparse matrix routines can in turn be used in the solution of Eq. (2) for $\dot{q}_{II}$.

The generation of the equations of motion in the state-space representation, by comparison, is only truly straightforward in the case of tree-configured (no closed loops) systems. In these cases the use of a state-space representation may be advantageous because the state-space variables selected are of specific interest. Also, the equations in state-space form, unlike those in the descriptor counterpart, can be directly treated by standard ordinary differential equation methods and are of much lower dimensionality than their descriptor counterparts. However, the resulting mass matrix M appearing in Eq. (1) is normally densely populated, and thus, the construction of Eq. (1b) and the subsequent solution for $\dot{q}_{II}$ by traditional methods can be computationally expensive.

Application of state-space formulations to systems containing closed kinematic chains results in a set of partially reduced (containing some redundant variables) equations representing a tree-configured system that would arise if the closed loops are cut in a prescribed manner. At this time, the development of the equations generally proceeds in one of two ways. One is an augmentation approach in which closed loops are cut, associated unknown generalized constraint forces appear explicitly, and the system equations are augmented with the algebraic constraint equations. The other technique is to use the constraint equations to eliminate the dependent/redundant state derivatives and thereby produce a system involving only independent state derivatives.

In the effort to improve simulation turn around, and thereby to increase the overall speed of the computer simulation of the multibody dynamics system's behavior, a variety of algorithms and solution techniques have been developed. In many cases lower computational order formulations for the equations of motion have been sought based on both the full descriptor formulation and state-space formulations. These algorithms of low computational order have evolved greatly. From simple chain systems as addressed by Armstrong[5] in 1979, to more complex robotic systems,[6–9] to open-loop tree systems of rigid bodies,[10,13,14,17] the inclusion of closed loops or

other forms of geometric and/or motion constraints[15,17,18,24,25] have evolved to the generalization that any or all bodies of the system may be flexible.[3,19,26,27] However, until relatively recently the development of such algorithms was limited to application on purely sequential-architecture computer systems.

B. Previous Work Involving Parallel Computing in Multibody Dynamics Analysis

With the advent of parallel computing, researchers began to rethink the use of previously developed dynamic simulation procedures and how the benefits of concurrent processing might be realized. Initially, work was mostly confined to the parallelization of the existing procedures that had been originally designed for exclusively sequential applications. In 1987, Kasahara et al.[28] developed a parallelized form of Walker and Orin's[8] method 3 with a parallel Gauss–Jordan procedure to simulate a robot manipulator.

Lee[27] developed a global procedure based on Kane and Levinson's method[29] for the simulation of the behavior of large flexible space structures. This procedure was designed from its inception to develop the equations in such a way as to allow the exploitation of the concurrent processing to the maximum degree on a multiple instruction multiple data- (MIMD-)type machine. However, due to the underlying $\mathcal{O}(N^3)$ nature of the formulation, the best Lee was able to obtain was a worse than $\mathcal{O}(N^3)/N_P$ performance with N_P processors.

Lee and Chang[30] also developed parallel approaches based on the methods of Walker and Orin.[8] The first of these was based on Walker and Orin's method 3 with parallelization accomplished using techniques presented by Lathrop[31] for evaluating the mass matrix and the subsequent solution for the state derivatives using a parallelized Cholesky factorization method designed for implementation on an array processor. The results indicated $\mathcal{O}(N)$ complexity on turnaround time with $\mathcal{O}(N^2)$ processors.

Anderson[17] pursued increased simulation speed through the elimination of needless calculations via the production of problem specific simulation code derived using his own low-order algorithms, while also trying to exploit the inherent concurrency of the problem. This simulation code was from its inception designed for parallel implementation on an available distributed architecture MIMD computing system.[32] The resulting procedure could be applied to general multi-rigid-body systems that could involve an arbitrary number of branches, and/or closed loops, and specified motions. At the coarsest level, parallelism was strictly a function of the mechanical systems topology. In this and subsequent work,[33] it was also demonstrated that significantly greater parallelism could be realized with a modified form of the $\mathcal{O}(N)$ algorithm than was indicated by the topology of the system.[18,19] This added parallelism was obtained through the explicit determination of the constraint loads that existed at the system joints that were cut to produce the largely independent subsystems and forms the basis for much of the work presented here.

Hwang et al.[34] did leading work that was based on their own recursive variational formulation (Bae et al.[16]) that yielded $\mathcal{O}(N)$ performance when confined to purely sequential processing. This work could be applied to general multibody systems involving closed loops with the parallelization largely being limited to performing calculation along independent branches of the system's spanning tree.

In subsequent work by Chung and Haug,[35] a recursive variational $\mathcal{O}(N^3)$ approach was used to formulate the equations of motion. Static scheduling algorithms were employed to distribute the computations evenly over the processors of the shared memory machines being used. In this work, the simulation of the high-mobility multipurpose wheeled vehicle was sped up by a factor of 4.4 over that obtained with a single processor, using eight processors on an Alliant FX/8.

Substantial work has also been done toward the efficient simulation of multibody systems that yield stiff systems of equations. Eichberger et al.[36] developed an $\mathcal{O}(N)$-residual formulation designed specifically for parallel implementation using an implicit integration scheme and parallel computing.

Work by Fijany and Bejczy[37,38] indicates that the more traditional $\mathcal{O}(N^3)$ formulations provide the highest degree of parallelism and conceivably produce the most efficient (time optimal/minimum turnaround time per integration step) parallel computation provided that an ideal communication and a sufficient number of processors are available. Specifically, a turnaround performance of $\mathcal{O}(\log_2 N)$ can be achieved with $\mathcal{O}(N^3)$ processors. Arguably, such an approach would not be practical for many simulation applications because the necessary number of processors would be unavailable and the communications overhead costs would more than offset such fine grain parallelization.

A better approach was the development of a procedure that is both time optimal [yields $\mathcal{O}(\log_2 N)$ performance] and processor optimal [requires only $\mathcal{O}(N)$ processors]. Work by Sharf and D'Eleuterio[39] and Fijany et al.[40] present procedures based on a full descriptor formulation for simple open chains of bodies directed to this end. The procedures set up a sparse system of linear equations that, for the simple chain systems considered, result in a block tridiagonal constraint force (CF) matrix. The resulting system of equations for the unknown interbody CFs is then solved by various iterative parallel methods. These CFs were in turn used in the determination of the state derivative values that are integrated temporally to obtain updated values for the system state. By assuming that there is ideal communication and that the number of iterations does not increase with N, these procedures result in a theoretical lower bound on simulation turnaround time corresponding to an overall computational performance of $\mathcal{O}(\log_2 N)$ with $\mathcal{O}(N)$ processors.

It is possible (or even likely) that in many situations the systems to be modeled are multibranched and sufficiently large that the number of available processor N_P is significantly smaller than the number of bodies N in the system ($N_P \ll N$). In these situations significant sequential calculations are mandated. Here, still better performance might be possible if for these sequential portions of the simulations one exploits algorithms with superior sequential performance, relative to that of the full descriptor $\mathcal{O}(N)$ and iterative pseudo-$\mathcal{O}(N)$ procedures.[39,40] Moreover, the performance of these full descriptor-based methods degrades profoundly relative to that associated with simple chain systems due to numerical issues associated with the greater coefficient matrix bandwidth and to the more dispersed eigenstructure associated with these more complex systems. Sequential low-order state-space formulations, by comparison, are not as adversely affected by this increase in mechanical system complexity and, thus, may be used to some advantage.

This paper presents such an approach. The procedure is a novel hybrid iterative-direct approach for the simulation of dynamic behavior in complex multibody systems using MIMD parallel computing. The algorithm may be applied to general multi-rigid-body systems arbitrarily possessing many branches and/or closed loops. The procedure produces true noniterative $\mathcal{O}(n)$ performance when used sequentially on a single processor. At a finer level, the algorithm provides parallelization of an N body system to N processors providing pseudo-$\mathcal{O}(\log_2 N)$ computational performance when applied to chain systems. Furthermore, the approach maintains many of the desirable characteristics of the sequential $\mathcal{O}(N)$ procedures, when beneficial, that often allows it to perform in a superior manner [relative to that of the state-space $\mathcal{O}(n)$ or full descriptor formulations] on these branched systems given realistic communications costs and iterative solver performance.

II. Algorithm Formulation

A. Notation and Preliminaries

In the following derivation a vector dyadic representation is used. A scalar is represented by a letter, and a vector or dyadic by a boldface letter. For example, given some arbitrary quantity v, the associated scalar, vector, and dyadic quantities would be represented by v, $\boldsymbol{v}$, and $\boldsymbol{\nu}$, respectively. Matrix quantities will be represented by a letter. For instance, the matrix associate with the vector $\boldsymbol{\nu}$ would be represented by ν, where $\nu \in \Re^{3\times 1}$.

Consider the general multibody system shown in Fig. 1. For the purpose of this derivation, we will limit our attention to cutting this system down to a set of subsystems with constraint loads applied only at the base and terminal bodies of each subsystem, as shown. In all of the equations that follow, the superscripts ik indicates that the quantity in question is associated with the kth body of the ith

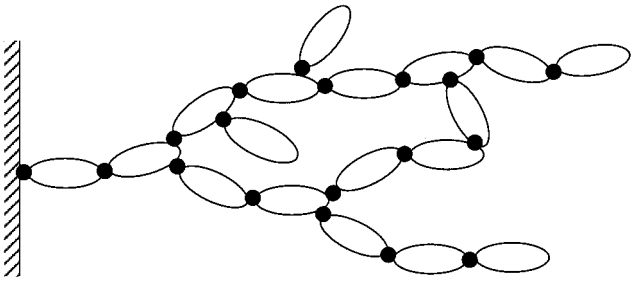


Fig. 1a General multibody system.

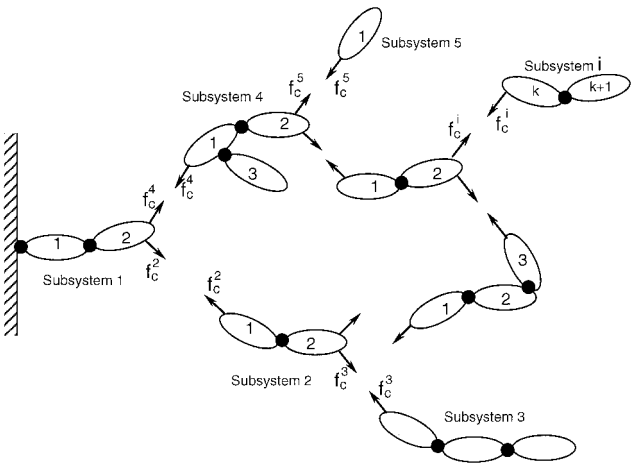


Fig. 1b One of many possible associated subchain systems.

subsystem. To ensure that the set of subsystems continues to behave as the original interconnected system, each subsystem interacts with its proximal (parent or inboard) subsystem and its distal (children or outboard) subsystems through constraint forces $\boldsymbol{f}_c$ and constraint moments $\boldsymbol{t}_c$. Within each subsystem the numbering of each body increases sequentially from 1 for the base body to N_i for the terminal body. In this manner we define the acceleration matrix associated with the kth body of the ith subsystem relative to an observer fixed in a Newtonian reference frame to be

$$A^{ik} \equiv \begin{bmatrix} \alpha^{ik} \\ a^{ik} \end{bmatrix} \qquad \in \Re^{6\times 1} \tag{3}$$

where $\alpha^{ik} \in \Re^{3\times 1}$ and $a^{ik} \in \Re^{3\times 1}$ are the matrix representations of the angular acceleration of body k and the acceleration of the center of mass of body k of the subsystem i, respectively.

We now write A^{ik} as

$$A^{ik} = \bar{A}^{ik} + A_t^{ik} \tag{4}$$

where $\bar{A}^{ik}$ is that portion of A^{ik} that is explicit in the unknown second derivatives of generalized coordinates $\ddot{q}$ that we wish to determine and subsequently integrate temporally, whereas A_t^{ik} represents all of the remaining acceleration terms.

It is now possible to write A^{ik} recursively in terms of the acceleration matrices associated with the proximal body ik, body $i(k-1)$, specifically,

$$A^{ik} = S^{ik}\bar{A}^{i(k-1)} + P^{ik}\ddot{q}^{ik} + A_t^{i(k-1)} + {}^{i(k-1)}A_t^{ik} \tag{5}$$

with S^{ik} defined as

$$S^{ik} = \begin{bmatrix} U & r^{i(k-1)k}\times \\ 0 & U \end{bmatrix} \qquad \in \Re^{6\times 6} \tag{6}$$

where U is an identity matrix of the appropriate dimension and $r^{i(k-1)k}\times \in \Re^{3\times 3}$ is the matrix representation of the vector cross product operator between the position vector $\boldsymbol{r}^{i(k-1)k}$ from the center of mass of body $i(k-1)$ to the center of mass of the next body in the subsystem, body ik, with another vector. With the transformation matrix so specified, the system of forces $\boldsymbol{f}^A$ and moments $\boldsymbol{t}^A$ acting at some point A of a rigid body can be related to an equivalent system of forces acting at point B of the same rigid body by the relation

$$R^B = \begin{bmatrix} U & r^{AB}\times \\ 0 & U \end{bmatrix} R^A \tag{7}$$

where $\boldsymbol{r}^{AB}$ is the position vector from point A to point B and $\boldsymbol{R}^A$ is the resultant force system acting through point A of the body given by

$$R^A = \begin{bmatrix} t^A \\ f^A \end{bmatrix} \qquad \in \Re^{6\times 1} \tag{8}$$

The matrix P^{ik} contains the partial velocities $\boldsymbol{v}_r^{ik}$ of the center of mass and the partial angular velocities $\boldsymbol{\omega}_r^{ik}$ of body ik that are associated with joint ik connecting body $(k-1)$ to body k in subsystem i.

Specifically,

$$P^{ik} = \begin{bmatrix} \omega_r^{ik} \\ v_r^{ik} \end{bmatrix} \qquad \in \Re^{6\times \nu^{ik}} \tag{9}$$

with ν^{jk} the number of degrees of freedom associated with the joint that connects body $(k-1)$ to body k in subsystem i. Finally, $\ddot{q}^{ik} \in \Re^{\nu^{ik}}$ is the matrix of the second time derivatives of the generalized coordinates associated with the joint connecting body $(k-1)$ to body k of the ith subsystem.

B. Equations of Motion with Explicit Constraint Forces

Define the matrices $I^{ik} \in \Re^{6\times 6}$ to be the 6×6 rigid-body central moment of inertia matrix associated with body ik. Define F^{ik} as

$$F^{ik} = R^{ik} - I^{ik}A_t^{ik} \tag{10}$$

where

$$R^{ik} = \begin{bmatrix} \Sigma\ t^{ik} \\ \Sigma\ f^{ik} \end{bmatrix} \qquad \in \Re^{6\times 1} \tag{11}$$

is the resultant force system of all forces and moments, except for constraint forces and moments, that act on body ik, and, similarly, define

$$F_c^{ik} = \begin{bmatrix} \Sigma\ t_c^{ik} \\ \Sigma\ f_c^{ik} \end{bmatrix} \qquad \in \Re^{6\times 1} \tag{12}$$

as the matrix of constraint moments and constraint forces acting directly on body k of subsystem i. Then Kane's dynamic equations[20] may be written as

$$\mathfrak{I}_{ik} + \mathfrak{I}_{ik}^* = (P^{ik})^T(-\hat{I}^{ik}\bar{A}^{ik} + \hat{F}^{ik} + \hat{F}_c^{ik}) = 0, \qquad \in \Re^{\nu^{jk}\times 1} \tag{13}$$

The quantities $\mathfrak{I}_{ik}$ and $\mathfrak{I}_{ik}^*$ are affiliated with the generalized speeds that are associated with the joint ik degrees of freedom, with the quantities $\hat{I}^{ik}$, $\hat{F}^{ik}$, and $\hat{F}_c^{ik}$ defined recursively as

$$\hat{I}^{ik} = I^{ik} + T^{i(k+1)}\hat{I}^{i(k+1)}(S^{i(k+1)})^T \tag{14}$$

$$\hat{F}^{ik} = F^{ik} + T^{i(k+1)}\hat{F}^{i(k+1)} \tag{15}$$

$$\hat{F}_c^{ik} = T^{i(k+1)}\hat{F}_c^{i(k+1)} \tag{16}$$

and

$$\hat{T}^{ik} = T^{i(k+1)}\hat{T}^{i(k+1)} \tag{17}$$

For the special case for which body k is a terminal body N^i for the subsystem, we have

$$\hat{I}^{in_i} = I^{in_i} \tag{18}$$

$$\hat{F}^{in_i} = F^{in_i} \tag{19}$$

$$\hat{F}_c^{in_i} = F_c^{in_i} \tag{20}$$

and

$$\hat{T}^{in_i} = T^{in_i} \tag{21}$$

The quantity T^{ik} appearing in Eqs. (14–17) is the triangularization matrix[12,17] that performs the task of recursively triangularizing the systems mass matrix as it is being formed. This matrix is given by

$$T^{ik} = S^{ik}\left\{U - (1/M_{kk}^i)\hat{I}^{ik}P^{ik}(P^{ik})^T\right\} \tag{22}$$

Proceeding in this manner, one works recursively inward from the terminal body of each subsystem to its associated base body. The unknown second derivatives of the system's generalized coordinates can be expressed in the form

$$\ddot{q}^{ik} = \eta^{jk} + \zeta^{ik} \tag{23}$$

where ζ^{ik} contains all terms that are explicit in the unknown constraint load magnitudes t_c and f_c. The quantity η^{ik} can be determined recursively using the relations

$$\eta^{ik} = \left[(P^{ik})^T/M_{kk}^i\right]\left[-\hat{I}^{ik}(S^{ik})^T\bar{\eta}^{i(k-1)} + \hat{F}^{ik}\right]$$

$$(k = 1, 2, \ldots, N^i) \tag{24}$$

and

$$\bar{\eta}^{ik} = (S^{ik})^T\bar{\eta}^{i(k-1)} + P^{ik}\eta^{ik} \qquad (k = 1, 2, \ldots, N^i) \tag{25}$$

with

$$\bar{\eta}^{i0} = 0 \qquad \in \Re^{6\times 1} \tag{26}$$

Now the base body of a subsystem will have constraint forces and/or moments acting directly on it if this base body interacts with another subsystem. In such a case the quantity ζ^{i1} would involve constraint loads $f_c^{iN^i}$ and $t_c^{iN^i}$ that act on the subsystem terminal body N^i and the constraint loads f_c^{i0} and t_c^{i0} that act on the base body of the ith subsystem from its proximal subsystem. In such a situation the quantity ζ^{i1} would be given as

$$\zeta^{i1} = \left[(P^{i1})^T/M_{11}^i\right]\left[\hat{T}^{i1}F_c^{in^i} + F_c^{i0}\right] \tag{27}$$

or for a general body k of subsystem i

$$\zeta^{ik} = B_n^{ik}F_c^{in^i} + B_0^{ik}F_c^{i0} \tag{28}$$

with

$$B_T^{ik} = \left[(P^{ik})^T/M_{kk}^i\right]\left[-\hat{I}^{ik}(S^{ik})^T P^{i(k-1)}B_T^{i(k-1)} + \hat{T}^{ik}\right]$$

$$\in \Re^{1\times 6} \tag{29}$$

and

$$B_0^{ik} = -\left[(P^{ik})^T/M_{kk}^i\right]\hat{I}^{ik}(S^{ik})^T P^{i(k-1)}B_0^{i(k-1)} \qquad \in \Re^{1\times 6} \tag{30}$$

The displacement constraints that must be enforced between the terminal body N^i of subsystem i and the base body of adjacent subsystem $(i+1)$ are differentiated twice with respect to time to yield the acceleration constraints

$$A^{(i+1)1} = P^{(i+1)1}\ddot{q}^{(i+1)1} + A_t^{i(N^i+1)}$$

$$= \bar{A}^{iN^i} + P^{i(N^i+1)}\ddot{q}^{i(N^i+1)} + A_t^{i(N^i+1)} \tag{31}$$

where $q^{i(N^i+1)}$ are the generalized coordinates that describe the rigid-body degrees of freedom of the base body of subsystem $(i+1)$ relative to the terminal body N^i of subsystem i. These are the degrees of freedom allowed by the joint N^i that would connect these two bodies in the actual system and that were cut to produce subsystem $(i+1)$. These generalized coordinates $q^{i(N^i+1)}$ form a redundant set of coordinates in this system of subchains. As a consequence, if one considers all subsystems, then Eqs. (23) and (31) represent a system of $n+m$ equations and $n+2m$ unknowns, where n is the total number of degrees of freedom possessed by the entire set of subsystems and m is the total number of constraint loads that must act on this system of subsystems to ensure that this system behaves as the original uncut system. To make this system of equations solvable, it is necessary to eliminate m unknowns from the system of equations given by Eqs. (23) and (31). This end is achieved by eliminating $\ddot{q}^{i(N^i+1)}$, $(i = 0, 1, 2, \ldots, J_c - 1)$, where J_c is the total number of joints that are cut in the system, from all Eqs. (31). This is accomplished through the determination of the matrix $\chi^{i(N^i+1)}$ that is defined as the orthogonal complement to the partials matrix $P^{i(N^i+1)}$ and is easily determined from joint N^i+1 definition/specification information. Now

$$\chi^{i(N^i+1)}P^{i(N^i+1)} = 0 \tag{32}$$

so that multiplying Eq. (31) through by $\chi^{i(N^i+1)}$ yields

$$\chi^{i(N^i+1)}P^{(i+1)1}\ddot{q}^{(i+1)1} + \chi^{i(N^i+1)}A_t^{(i+1)1}$$

$$= \chi^{i(N^i+1)}\bar{A}^{iN^i} + \chi^{i(N^i+1)}P^{i(N^i+1)}\ddot{q}^{i(N^i+1)}$$

$$+ \chi^{i(N^i+1)}A_t^{i(N^i+1)} = \chi^{i(N^i+1)}\bar{A}^{iN^i} + \chi^{i(N^i+1)}A_t^{i(N^i+1)} \tag{33}$$

Each of the elements of the matrices appearing in Eq. (33) can be formed in a recursive manner at a fixed cost, with the formulation of Eq. (33) requiring $\mathcal{O}(n^i)$ operations overall. Through a simple mapping, one may easily convert all local coordinates q^{ki} and constraint forces f_c^{ki} to associated global quantities q^j and f_c^j. Then, recursively use Eqs. (24–26) to evaluate all elements of η appearing in Eq. (23) for the entire system. Similarly, definition (12) and Eqs. (27–30) may be recursively used to produce simple expressions for each value of ζ that is linear in the unknown constraint load measure numbers f_c and t_c and is, in turn, substituted in to Eq. (23). Substituting the resulting Eqs. (23) into Eq. (33) yields a set of equations associated with the ith cut joint of the form

$$(\Gamma)_{ii}(F_c)_i + (\Gamma)_{ij}(F_c)_j + (\Gamma)_{ik}(F_c)_k + (\Gamma)_{il}(F_c)_l = (\Psi)_i \tag{34}$$

where:

i = cut joint currently under consideration, (i)
j = cut joint proximal to cut joint i, (pr[i])
k = cut joints distal to cut joint i, (dist[i])
l = cut joints distal to proximal cut joint of i, (dist[$pr[i]$])

and (Γ_{ij}) are coefficients that will become the elements associated with the jth cut joint in the ith set of constraints.

Equations (34) may in turn be assembled for all cut joints into a linear system of global equations for the constraint load magnitudes $F_c \in \Re^{m\times 1}$, containing all m of the constraint loads acting on each of the subsystems as

$$\Gamma F_c = \Psi \tag{35}$$

The numerical values associated with each element of the matrices Γ and Ψ appearing in Eq. (35) that are determined using this largely state-space $\mathcal{O}(n)$ approach are equivalent (if infinite computing precision) to those elements appearing in $[C\,M^{-1}C^T]$ and $[C\,M^{-1}Q + \dot{C}q_{II} - \dot{D}]$, respectively, from

$$[CM^{-1}C^T]\lambda = CM^{-1}Q + \dot{C}q_{II} - \dot{D} \tag{36}$$

which is generated by more traditional means from Eqs. (2b), the time derivative of (2d), and is discussed in Refs. 41 and 42. This equivalence of the element numerical values arises because the state-space recursive $\mathcal{O}(n)$ procedure is computationally, but not implementationally, equivalent to the formulation of Eqs. (2) with their

subsequent decomposition and back substitution via Gaussian elimination. The matrix $\Gamma = [C\,M^{-1}\,C^T]$ is symmetric positive definite for all possible matrices M associated with this formulation, that is, a system cut into tree structure subsystems, and C having full row rank.[41] The constraint Jacobian C will have full row rank for all tree systems, but may lose rank for certain system configurations involving closed loops and nonholonomic constraints at those specific instances when one or more rows of C become numerically dependent.[43] For most real mechanical systems, these instances are the exception and not the rule and may be identified, anticipated, checked for, and dealt with when they arise.

Equation (35) may then be efficiently solved for the system constraint load magnitudes F_c, which are then appropriately substituted into Eqs. (27) and (28). Equations (27–30) are then used with Eqs. (23–26) to determine the generalized coordinate second time derivatives $\ddot{q} \in \Re^{n\times 1}$ associated with all n generalized coordinates of the system. The elements of the matrices $\ddot{q}$ and $\dot{q}$ are then integrated temporally to obtain the updated values for the state variables $\dot{q}$ and q, respectively. The process then repeats for the next simulation time step.

III. Preliminary Results and Discussion

The form of Eq. (33) is such that when the system of constraint force equations are assembled into Eqs. (35) there is only direct interaction between nearest neighbor subsystems through the constraint loads. As a result of this modest coupling in the constraint load magnitudes that appear in Eqs. (35), the matrix $\Gamma \in \Re^{m\times m}$ often will be sparsely populated for systems involving many subsystems [however, Eq. (34) indicates that very heavily branched systems may produce a Γ that is densely populated]. Because of the symmetric, preponderantly positive definite structure of Γ, Eq. (35) lends itself to solution by parallel iterative approaches such as parallel implementation preconditioned conjugate gradient (PCG) methods. In the simplest case where the multibody system does not involve branches or closed loops, the resulting coefficient matrix Γ is block tridiagonal. If the system involves constraint loads that are associated with different branches of the tree system, then additional off-diagonal coupling blocks of terms will occur as dictated by Eq. (34).

If we confine our attention for the time being to the situation where the number of available processors N_P is significantly less than the total number of bodies in the system N, then significant portions of the simulation will be performed sequentially and the distribution of the computational work load over the N_P processor may be achieved in a variety of ways. One extreme is to cut every body producing N subsystems, each consisting of a single body. If the joints were each a single-degree-of-freedom joint, this would then result in a system of $5N$ constraint loads. The constraint load magnitudes would be solved for and then substituted back into the resulting Newton–Euler equations to determine the generalized coordinate second time derivatives. For this full descriptor representation, the real contribution in parallelization is in the parallel construction of the coefficient matrix Γ and the subsequent parallel solution of the linear system of Eqs. (35) for the unknown constraint load magnitudes F_C. This construction lends itself very well to a coarse grain parallelization of $\mathcal{O}(N)$. This construction also produces pseudo-$\mathcal{O}(n)$ sequential performance [$\mathcal{O}(n^\gamma)$, $1.17 < \gamma < 1.25$ in Ref. 39 for simple chain systems], where as a rule the number of iterations, manifested in γ, increases with the condition number and, thus, indirectly with the dimension and form of Γ. By comparison, for chain systems, true $\mathcal{O}(n)$ performance is possible when using direct solvers designed to take full advantage of the symmetric block tridiagonal structure of Eqs. (35).

On the positive side, this full descriptor approach is the easiest to implement because all bodies are treated in an identical manner, and the equations of motion are particularly simple for a single body. Also, the parallelization process is largely transparent because it frees the analysts to make use of currently existing parallel methods for dealing with this linear (often sparse) system of equations. This full descriptor approach also finds all of the interbody loads that exist within the system, and knowledge of these loads may be desired. Unfortunately, there can be negative aspects of this full descriptor approach.

1) If an iterative solver is used, the increased condition number of Γ, associated with larger, more complex systems (particularly with closed loops), increases the number of iterations required to converge to the solution of Eq. (36). This can profoundly degrade solver performance.

2) The greater bandwidth of Γ associated with branched and closed-loop systems results from the increase in the number operations per each iteration in the solution of Eq. (36).

3) The number of iterations required in the convergence to a solution for Eq. (36) is greatly dependent on the quality of (error in) the initial values F_{C_0} used for the first iteration. Moreover, it may not be possible to effectively estimate F_{C_0} in systems involving discontinuous applied forces.

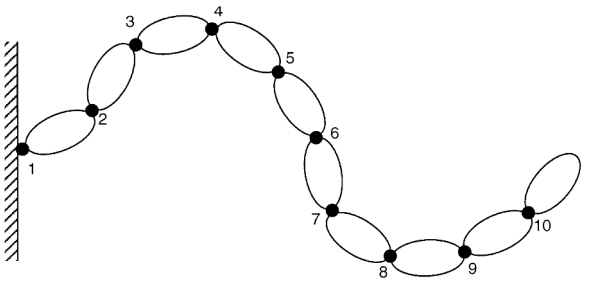
4) This construction requires the determination of all of the interbody constraint loads. Consequently, the computational cost of determining these loads must be paid whether knowledge of these loads is desired from the simulation or not.

5) Because algebraic constraints that exist between all of the bodies of the system are enforced at the acceleration level, and not at the displacement level, zero eigenvalues are introduced for each of these constraints, which will result in an eventual unstable growth in constraint violation error if no constraint error stabilization is employed.[17,18,42]

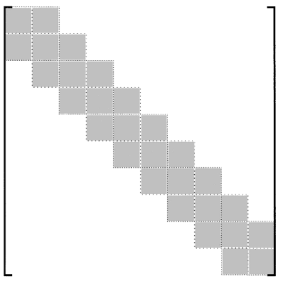
At the other extreme is an implementation where only as many joints are cut as are necessary to obtain N_P processor parallelization. For the case of a simple unconstrained chain, this would require $N_p - 1$ such cuts. The resulting N_P subsystems, to a significant degree, would then have their associated equations of motion sequentially formed and solved on each processor using a state-space direct $\mathcal{O}(n)$ procedure. This $\mathcal{O}(n)$ procedure is a direct method that is more efficient in sequential application than the descriptor $\mathcal{O}(N)$ procedure discussed in the preceding paragraph.[38–40] The resulting overall procedure can be considered a hybrid state-space descriptor procedure in which the $5 \times (N_P - 1)$ ($< 5N$) constraint loads are determined from the parallel solution of Eqs. (35). This approach will often require fewer computational operations overall than offered by either the descriptor approach, despite the greater cost associated with the determination of each element of Γ and Ψ, or a straight state-space $\mathcal{O}(n)$ approach. This is because the solution of Eq. (35) associated with this hybrid formulation will require fewer operations per step due to the smaller dimension of Γ, the lower condition number, and the generally more narrow bandwidth than that for the full descriptor case. Furthermore, the constraint violation drift is not as much a problem with the hybrid formulation because it will only exist at these fewer cut joints. Finally, other than the constraint loads that must be calculated at the cut joints, only those interbody interaction forces for which information is desired need be calculated, and this expense is slight.

Figure 2 shows an example for a simple spatial chain system consisting of 10 bodies and indicates the form of this example's coefficient matrix Γ for the cases where 1) every joint in the system is cut and 2) where only joints 4 and 7 are cut. If, in each of these cases, interbody joints are restricted to single-degree-of-freedom joints, then the resulting individual shaded blocks of elements appearing in these representations of Γ are each of dimension 5×5. We would expect that the solution of Eqs. (35) for F_C in case 2 would require fewer iterations due to the smaller condition number for this case than for case 1. Thus, the merit of this hybrid approach largely depends on the degree to which its added cost in generating the individual elements Γ and Ψ, relative to the full descriptor approach, offsets these matrices's smaller dimension and the lower cost of solving Eqs. (35).

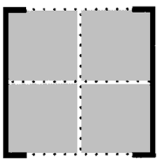
Similarly, Fig. 3 shows the effect that branches and closed loops may have on the form of the coefficient matrix Γ, where the bandwidth that results is a function of the joint numbering convention used. As seen in Fig. 3, the presence of branches, in particular closed loops, can significantly alter the form of Γ, making Eqs. (35) computationally more expense to solve for F_c. For case 1, with every joint being cut, Γ is large, has a decidedly broader bandwidth, and has a higher condition number than that for a chain system with a comparable number of bodies. Because of the presence of the closed loop, it is possible that, for certain predictable system



a) 10 body chain

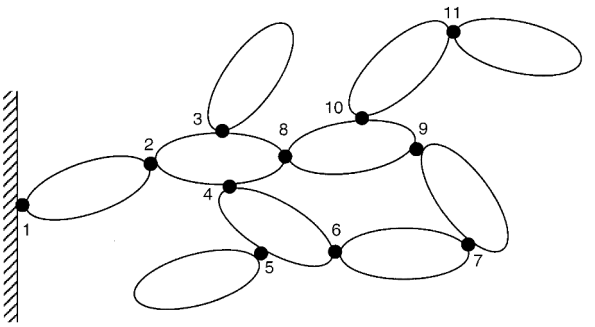


b) Case 1, Γ with every joint of system cut

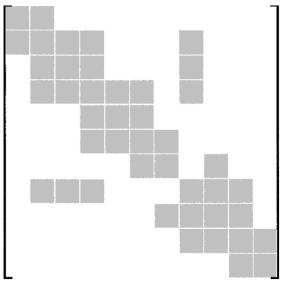


c) Case 2, only joints 4 and 7 cut

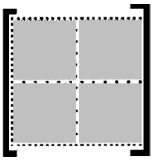
Fig. 2 Example: 10-body chain system.



a) More general 10-body system with branches and closed loops



b) Case 1, Γ when every joint is cut



c) Case 2, Γ when only joints 6 and 8 are cut

Fig. 3 System with branches and closed loops.

configurations, Γ will not be positive definite or will be very poorly conditioned. In such instances, implementation of other strategies such as quasi-minimal residual for non-positive definite systems or implementation of generalized singular valued decomposition for poorly conditioned situations may be necessary.

By comparison, for case 2 the troublesome closed-loop portion of the system may be dealt with (buried) within the direct $\mathcal{O}(n)$ portion of the hybrid algorithm using techniques such as coordinate partitioning as presented by Wehage.[44] Also, Γ in this case is substantially smaller has a narrower eigenvalue spectrum, and has a reduced bandwidth relative to that for case 1.

A. Algorithm Operations Count and Numerical Example

Consider a system like that shown in Fig. 2, which for this example comprises N bodies connected together to form an unconstrained chain by single-degree-of-freedom revolute joints. For this system, the equations of motion were formulated using both the full descriptor and the hybrid parallelizable $\mathcal{O}(N)$ procedures for the situations where 1, 2, 4, and 8 processors were available. The procedures were then validated for each case by emulating parallel application of each simulation sequential on an SGI workstation. In all cases where iterative solvers were used, the same convergence criterion of $\varepsilon = 10^{-8}$ was enforced. Further, the same temporal integration scheme was used for all simulations. The performance of the algorithm showed itself to be of the form

$$\mathcal{O}\left[(n+m)/N_P + m^{\gamma}/N_P + m^{(\gamma-1)}\log_2 N_P\right] \qquad (37)$$

where

N_P = number of processors available
γ = exponent within which the increase in Γ condition number, number of iterations in the solution of Eq. (35), and the structure of Γ are manifest ($\gamma > 1$)

The first term in Eq. (37) reflects the computation associated the development of the equations of motion, their subsequent solution, and the treatment of the unknown constraint loads acting on the subsystem terminal bodies, within each subsystem via the direct sequential $\mathcal{O}(n)$ portion of algorithm. The second and third terms, exclusive of $\log_2 N_P$, represent the non-communication and interprocessor communication cost portions of the parallel solution of Eq. (35), respectively. Quantity γ appearing in Eq. (37) accounts for increased computational effort (increased number of iterations and increased computations per iteration) associated to the solution of Eq. (35) as the condition number and possibly the bandwidth of the coefficient matrix Γ increase. The appropriate value of γ to be used is very much a function of the nature of the physical system, the iterative solver used, the quality of the estimated values for the unknown constraint loads F_C used in the initial iteration of the solution of Eqs. (35) at each integration step, and the tightness the convergence criteria. In numerical studies performed by the authors values for γ in the range $1.2 < \gamma < 2.04$ were determined depending on the nature of the dynamic system, iterative solver, etc. Given the variability one may expect in γ, as well as the effect on simulation performance due to the temporal integration scheme used, a highly restricted operations count formula for this hybrid algorithm is presented in Table 1. Specifically, the simulation associated with the presented operations count is restricted to the spatial motion of a simple chain system that is connected to an inertially fixed body at one end, with all interbody joints being arbitrarily oriented revolute joints, and $N_P < N$. The presented operations counts represent what performance can be expected for a symbolic implementation of the presented hybrid algorithm for the described chain system. When implementation is parallel, the presented operations counts are further segregated into two portions.

1) One portion contains all direct sequential $\mathcal{O}(n)$ aspects of the algorithm, assembly of the Eqs. (35), and all internode communications cost associated with these operations for a single function evaluation. This effectively represents all noniterative aspects of the algorithm for a single function evaluation of the overall system of equations, exclusive of the solution of Eqs. (35).

2) The other portion contains computational and communications costs associated with a single iteration in the solution of Eqs. (35).

For comparison purposes, operations counts associated with the application of a full descriptor formulation and purely sequential state-space $\mathcal{O}(n)$ approach to this chain system are provided in Table 1.

The quantities α and β appearing in Table 1 representing the communication latency cost and the cost of transferring a single byte of information are hardware, code/data structure, and operating environment dependent. Thus the level of parallelism utilized in any dynamic simulation should greatly depend on the nature of the problem, the computing resources available, and the actual coding practice used. Inspection of Table 1 indicates that the costs associated with the iterative portions of the hybrid and full descriptor algorithms can easily dominate the overall simulation, particularly

Table 1 Operations cost

Description	Computation cost[a]	Communication cost
	Computation and communication cost for hybrid algorithm	
Noniterative aspects of algorithm for single function evaluations	$(N/N_P)(409m + 375a) + [(N_P - 1)/N_P](484m + 323a) - (N/N_P^2)(129m + 35a) - (1/N_P)(20m + 16a)$	$2\alpha + 76\beta$
Iterative aspects of algorithm (single iteration)	$160m + (150 + 3\log_2 N_P)a$	$3\alpha\log_2 N_P + (6\log_2 N_P + 10)\beta$
	Computation and communication for iterative full descriptor formulation	
Noniterative aspects of algorithm for single function evaluation	$(N/N_P)(278m + 251a) + (324m + 214a)$	$2\alpha + 76\beta$
Iterative aspects of algorithm (single iteration)	$(N/N_P)(160m + (150 + 3\log_2 N_P)a)$	$3\alpha\log_2 N_P + (6\log_2 N_P + 10)\beta$
	Computation for purely sequential state-space $\mathcal{O}(n)$ algorithm	
Purely sequential state-space $\mathcal{O}(n)$ algorithm	$N(315m + 280a) - (214m + 165a)$	Not applicable (purely sequential)

[a]Here, m is multiplication cost and a is addition cost.

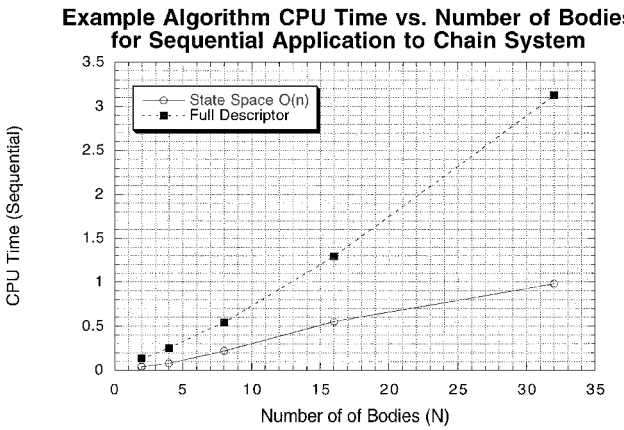


Fig. 4 Example sequential CPU time of state-space $\mathcal{O}(n)$ and full descriptor algorithms.

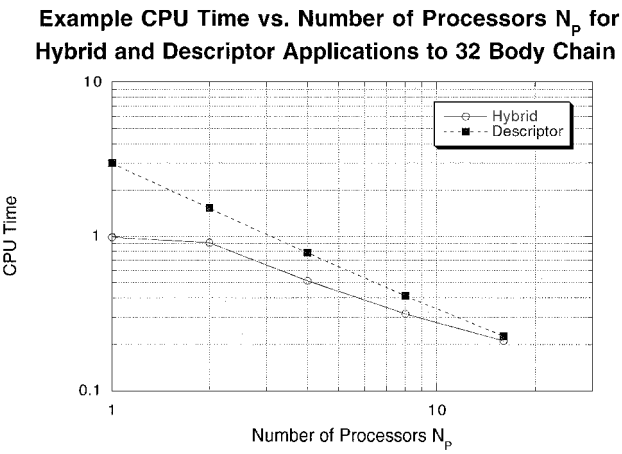


Fig. 5 Example CPU time vs number of processors for 32 body chain.

when communications costs are high. A case in point is the application of these algorithms on an IBM SP2 available to the authors. For the 36-processor configuration used, with a TB3 switch, the latency and byte transfer costs when using the message passing interface[45] were experimentally determined to be $\alpha \sim 6000$ floating-point operations (FLOPs) and $\beta \sim 1.7$ FLOPs/B, respectively. Clearly, such high communications costs greatly restrict the fineness of the grain size that may be effectively implemented. Shared memory machines should in general offer considerably better performance in this regard.

B. Chain Systems

For demonstration purposes, Fig. 4 shows the sequential performance obtained by the algorithm vs the number of bodies in a simple chain system. The chain system is composed of identical bodies with masses and principal inertias of unit magnitude. Also, the characteristic dimensions of each body was also set to a magnitude of 1.0. The simulation time for a single function evaluation as then determined for the cases where the number of bodies in the system was 2, 4, 8, 16, and 32.

The results indicate that, for this simple chain system, the purely sequential performance of a state-space $\mathcal{O}(n)$ approach is better than that offered by the iterative full descriptor approach. In this particular case, the iterative full descriptor pseudo-$\mathcal{O}(n)$ algorithm required a factor of $\sim$4 more time than the state-space $\mathcal{O}(N)$ approach for small N, with the CPU time increasing approximately as $\mathcal{O}(N^{1.42})$ for the iterative procedure. The CPU time vs N obtained indicate that the Jacobi conjugate gradient solver, using the diagonal of Γ for preconditioning and a zero-order hold[46] for determining the unknown values to be used in the initial iteration, performed with the number of iterations increasing approximately as $N^{0.42}$, that is, $\gamma \approx 1.4$. Significantly better results were obtained for the simulation of well-behaved systems (systems where F_c varies unabruptly and smoothly) using a second-order hold[46]; however, the authors were unsuccessful in obtaining the $1.17 < \gamma < 1.25$ performance for a similar system cited in Ref. 39.

In Fig. 5, a sample of the performance of these algorithms as a function of the number of processors N_P is presented. These results were based on data collected from sequential emulation of parallel performance using ideal but unrealistic communications cost parameter values of $\alpha = \beta = 0$. The small decrease in CPU time for the hybrid algorithm when using two processors relative to that obtained for a single processor indicates that for this system the hybrid algorithm offers no advantage relative to direct sequential application of the direct state-space $\mathcal{O}(n)$ algorithm unless more than two processors are used. This is because whenever unknown constraint loads are considered with the state-space $\mathcal{O}(n)$ algorithms a penalty is incurred due to the necessity of carrying out additional computations associated with the determination of each constraint force. For this simple chain system, this added computation offsets any gains made through the use of a second processor. However, the formulation is such that, once this penalty has been incurred by the $\mathcal{O}(n)$ portion of the algorithm, effectively no additional cost is incurred by this portion of the algorithm for subsequent cut joints used in the production of additional subsystems. The only additional cost for subsequent cut joints is confined to the iterative solution of Eq. (35), which grows in dimension with each cut joint. The data points associated with four and eight processors clearly show that real computational gains may be made with the addition of subsequent processors, once this initial parallelization cost of the algorithm has been paid.

The results shown in Fig. 5 also indicate that, for this specific case, the hybrid algorithm does offer superior parallel performance to the full descriptor implementation for $N_P < \frac{1}{2}N$, with the relative performance of the hybrid algorithm increasing as the ratio N_P/N decreases. However, note that the relative merit of the hybrid algorithm will decrease if one can successfully reduce the size of γ in Eq. (37).

C. Branched Systems

For branched systems, the structure of Γ and the performance of the associated iterative solver are considerably different. Consider the branched system shown in Fig. 3. The structure of Γ is heavily a function of the numbering convention used in labeling the cut joints. In this regard the bandwidth should be reduced to a minimum if possible because an increased matrix bandwidth general manifests itself in the form of more computations per iteration in the solution of Eq. (35). To this end, the bandwidth of the Γ matrix can be reduced to a minimum if an optimal joint numbering scheme such as reverse Cuthill–McKee (see Ref. 47) is employed. Further, branched systems tend to have a more dispersed eigenvalue structure that can result in an increased number of iteration in the solution of Eqs. (35). Thus, branched systems, particularly those involving closed loops, with their combination of increased bandwidth, large condition number, and more dispersed eigenvalue structure pose a greater problem in obtaining an economical solution than do comparably sized chain system.

For branched systems, the hybrid algorithm develops the equations of motion and constructs Eqs. (35) in about

$$\mathcal{O}\left[1 + (N_P + N_P^{\gamma})/N_B + N/N_P + N_P^{\gamma-1}\log_2 N_P\right] \qquad (38)$$

operations, where the quantity N_B appearing in Eq. (38) represents the number of branch bodies in the system. A branch body is a subsystem terminal body (having no outboard bodies) that was generated through the cutting of a system joint. This number shrinks from N_P for chain systems to 1 as the system becomes more heavily branched. By comparison, the performance of the full descriptor formulation for a branched system would be approximated by

$$\mathcal{O}\left(1 + \frac{N^2 + N^{(\gamma+1)}}{N_P N_B} + \frac{N}{N_P} + N^{(\gamma-1)}\log_2 N_P\right) \qquad (39)$$

with N_B shrinking from N for chain systems to 1 as the system becomes more heavily branched.

To demonstrate the impact of branching on the performance of the algorithms, consider a very heavily branched N system where bodies $2, 3, \ldots, N$ are each connected to body 1. In such a system $N_B = 1$. This will result in a Γ that is largely and broadly populated, producing no band structure. A major difficulty with Eq. (39) is then the $\mathcal{O}([N^2 + N^{(\gamma+1)}]/N_P)$ term that now appears and that is directly associated with the cost of constructing the now largely populated matrix Γ and the subsequent iterative solution for the constraint forces. In such cases, Eqs. (35) will require $\mathcal{O}(N^3)$ to solve by sequential direct methods, and potentially will perform poorly via iterative methods given the now undesirable structure of Γ.

For $N > N_P$, the hybrid formulation is not as affected because here the heavy coupling will only exist in the Γ matrix associated with these fewer cut joints. This will result in a smaller though equally widely populated Γ matrix, which in light of its reduced size is more economical to produce and the associated Eq. (35) is easier to solve.

IV. Conclusions

A new solution procedure is presented for the computer simulation of the dynamic behavior of complex multibody systems. The algorithm represents an extension to, and generalization of, the recursive multibody analysis and procedures previously developed for use on complex multi-flexible-body systems. The advantage of the proposed scheme is that it cuts joints as necessary to allow for the explicit determination of interbody constraint loads only to the degree necessary to obtain the desired level of parallelism. Further, this parallelism is trivial to identify and, as such, relatively easy to implement. This explicit determination of selected constraint forces allows the analysis of the associated subsystem to proceed in a largely independent-parallel manner with the general modest coupling through the system's cut joint constraint forces. When sequential processing is necessary due to limited availability of processors, gains may be made through the efficient recursive $\mathcal{O}(n)$ direct procedure. The usually sparse linear system of equations in the constraint load measure numbers may be solved using a special iterative parallel procedure such as a PCG method. Thus far, this algorithm has shown itself to be particularly useful in those situations where the system is branched and possesses more bodies than there are available processors.

The presented algorithm has been implemented for some special cases involving simple chains and branched systems and has been validated against results obtained from more conventional independently written sequential $\mathcal{O}(n^3)$ and purely recursive $\mathcal{O}(n)$ formulations. The implementation of this procedure in a general purpose simulation code and the development of improved special iterative parallel methods for efficiently solving for the constraint load magnitudes F_c are areas of ongoing work.

Acknowledgment

The authors wish to acknowledge and thank the National Science Foundation for its support of this work through award ECS-9596237. Special thanks are due to M. Kupferschmid of Rensselaer Polytechnic Institute ACS for his help in obtaining good estimates of the α and β parameter values cited at the end of Sec. III.A.

References

[1]Hooker, W. W., and Margulies, G., "The Dynamical Attitude Equations for an *n*-body Satellite," *Journal of Astronautical Sciences*, Vol. 7, No. 4, 1965, pp. 123–128.

[2]Jain, A., "Unified Formulation of Dynamics for Serial Rigid Multibody Systems," *Journal of Guidance, Control, and Dynamics*, Vol. 14, No. 3, 1991, pp. 531–542.

[3]Banerjee, A. K., "Block-Diagonal Equations for Multibody Elastodynamics with Geometric Stiffness and Constraints," *Journal of Guidance, Control, and Dynamics*, Vol. 16, No. 6, 1993, pp. 1092–1100.

[4]Schwertassek, R., "Reduction of Multibody Simulation Time by Appropriate Formulation of Dynamical System Equations," *Computer-Aided Analysis of Rigid and Flexible Mechanical Systems*, edited by M. F. O. Pereira and J. A. C. Ambrósio, Kluwer, Dordrecht, The Netherlands, 1994, pp. 447–482.

[5]Armstrong, W. W., "Recursive Solution of the Equations of Motion of an N-link Manipulator," *Proceedings of the Fifth World Congress on the Theory of Machines and Mechanisms*, Vol. 2, American Society of Mechanical Engineers, Fairfield, NJ, 1979, pp. 1342–1346.

[6]Luh, J. S. Y., Walker, M. W., and Paul, R. P. C., "On-line Computational Scheme for Mechanical Manipulators," *Journal of Dynamic Systems, Measurements, and Control*, Vol. 102, No. 2, 1980, pp. 69–76.

[7]Hollerbach, J. M., "A Recursive Lagrangian Formulation of Manipulator Dynamics and a Comparative Study of Dynamic Formulation Complexity," *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. SMC-10, No. 11, 1980, pp. 730–736.

[8]Walker, M. W., and Orin, D. E., "Efficient Dynamic Computer Simulation of Robotic Mechanisms," *Journal of Dynamic Systems, Measurements, and Control*, Vol. 104, No. 3, 1982, pp. 205–211.

[9]Featherstone, R., "The Calculation of Robotic Dynamics Using Articulated Body Inertias," *International Journal of Robotics Research*, Vol. 2, No. 1, 1983, pp. 13–30.

[10]Rosenthal, D. E., and Sherman, M. A., "Symbolic Multibody Equations via Kane's Method," *American Astronomical Society/AIAA Astrodynamics Special Conference*, AIAA Paper 87-2262, Aug. 1987, pp. 1–20.

[11]Rosenthal, D. E., and Sherman, M. A., "High Performance Multibody Simulations via Symbolic Equation Manipulation and Kane's Method," *Journal of Astronautical Sciences*, Vol. 34, No. 3, 1986, pp. 223–239.

[12]Rosenthal, D. E., "An Order *n* Formulation for Robotic Systems," *Journal of Astronautical Sciences*, Vol. 38, No. 4, 1990, pp. 511–529.

[13]Nielan, P. E., "Efficient Computer Simulation of Motions of Multibody Systems," Ph.D. Dissertation, Dept. of Mechanical Engineering, Stanford Univ., Stanford, CA, 1986.

[14]Bae, D. S., and Haug, E. J., "A Recursive Formation for Constrained Mechanical Systems Dynamics: Part I, Open Loop Systems," *Mechanisms, Structures, and Machines*, Vol. 15, No. 3, 1987, pp. 359–382.

[15]Bae, D. S., and Haug, E. J., "A Recursive Formation for Constrained Mechanical Systems Dynamics: Part II, Closed Loop Systems," *Mechanisms, Structures, and Machines*, Vol. 15, No. 4, 1987, pp. 481–506.

[16]Bae, D. S., Kuhl, J. G., and Haug, E. J., *A Recursive Formation for Constrained Mechanical Systems Dynamics: Part III, Parallel Processing Implementation*, Center for Computer Aided Design, Univ. of Iowa, Sept. 1987.

[17]Anderson, K. S., "Recursive Derivation of Explicit Equations of Motion for the Efficient Dynamic/Control Simulation of Large Multibody Systems," Ph.D. Dissertation, Dept. of Mechanical Engineering, Stanford Univ., Stanford, CA, 1990.

[18]Anderson, K. S., "An Order-*N* Formulation for Motion Simulation of

General Constrained Multi-Rigid-Body Systems," *Computers and Structures*, Vol. 43, No. 3, 1992, pp. 565–572.

[19]Anderson, K. S., "An Efficient Formulation for the Modeling of General Multi-Flexible-Body Constrained Systems," *International Journal of Solids and Structures*, Vol. 30, No. 7, 1993, pp. 921–945.

[20]Kane, T. R., and Levenson, D. A., "Dynamics Online: Theory and Implementation with AUTOLEV," OnLine Dynamics, Inc., Sunnyvale, CA, May 1996.

[21]Orlandea, N., Chace, M. A., and Calahan, D. A., "A Sparcity-Oriented Approach to the Dynamics Analysis and Design of Mechanical Systems—Parts 1 and 2," *Journal of Engineering for Industry*, Vol. 99, No. 3, 1977, pp. 773–784.

[22]Chace, M. A., "Methods and Experience in Computer Aided Design of Large-Displacement Mechanical Systems," *Computer Aided Analysis and Optimizations of Mechanical System Dynamics*, edited by E. J. Haug, Springer–Verlag, Berlin, 1984, pp. 233–259.

[23]Lubich, C., Nowak, U., Pöhle, U., and Engstler, C., "MEXX—Numerical Software for the Integration of Constrained Mechanical Systems," Konrad–Zuse–Zentrum für Informationstechnik, SC 92-12, Berlin, 1992.

[24]Petzold, L. R., "Issues in the Numerical Solution of Differential-Algebraic Equations for Mechanical Systems Simulations," *Computer Aided Analysis of Rigid and Flexible Mechanical Systems*, edited by M. F. O. Pereira and J. A. C. Ambrósio, Kluwer, Dordrecht, The Netherlands, 1994, pp. 447–483.

[25]Nikravesh, P. E., and Chung, I. S., "Application of Euler Parameters to the Dynamic Analysis of Three-Dimensional Constrained Systems," *Journal of Mechanism Design*, Vol. 104, No. 4, 1982, pp. 785–791.

[26]Kim, S. S., and Haug, E. J., "Recursive Formation for Flexible Multibody Dynamics: Part I, Open-Loop Systems," *Computer Methods in Applied Mechanics and Engineering*, Vol. 71, No. 3, 1988, pp. 293–314.

[27]Lee, S. S., "Symbolic Generation of Equations of Motion for Dynamics/Control Simulation of Large Flexible Multibody Space Systems," Ph.D. Dissertation, Dept. of Aerospace Engineering, Univ. of California, Los Angeles, CA, 1988, pp. 1–142.

[28]Kashara, H., Fuji, H., Takane, E., and Iwata, M., "Parallel Processing of Robot Motion Simulation," *Proceedings of IFAC 10th World Conference*, Munich, Germany, Pergamon Press, Oxford, July 1987.

[29]Kane, T. R., and Levinson, D. A., *Dynamics: Theory and Application*, McGraw–Hill, New York, 1985, pp. 90–124, 158–169.

[30]Lee, C. S. G., and Chang, P. R., "Efficient Parallel Algorithms for Robot Forward Dynamics," *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 18, No. 2, 1988, pp. 238–251.

[31]Lathrop, L. H., "Parallelism in Manipulator Dynamics," *International Journal of Robotics Research*, Vol. 4, No. 2, 1985, pp. 80–102.

[32]Glück, R., "A Custom Architecture Parallel Processing System for the Space Station," Rept. TRW EML-003, TRW Space and Technology Group, Redondo Beach, CA, May 1989.

[33]Anderson, K. S., "An Efficient Modeling of Constrained Multibody Systems for Application with Parallel Computing," *Zeitschrift für Angewante Mathematic un Mechanik*, Vol. 73, No. 6, 1993, pp. 935–939.

[34]Hwang, R. S., Bae, D. S., Kuhl, J. G., and Haug, E. J., "Parallel Processing for Real-Time Dynamic System Simulation," *Journal of Mechanical Design*, Vol. 112, No. 4, 1990, pp. 520–528.

[35]Chung, S., and Haug, E. J., "Real-Time Simulation of Multibody Dynamics on Shared Memory Multiprocessors," *Journal of Dynamic Systems, Measurement, and Control*, Vol. 115, No. 4, 1993, pp. 627–637.

[36]Eichberger, A., Führer, C., and Schwertassek, R., "The Benefits of Parallel Multibody Simulation," *International Journal for Numerical Methods in Engineering*, Vol. 37, No. 9, 1994, pp. 1557–1572.

[37]Fijany, A., and Bejczy, A. K., "Techniques for Parallel Computation of Mechanical Manipulator Dynamics. Part II: Forward Dynamics," *Advances in Control and Dynamic Systems, Vol. 40: Advances in Robotic Systems and Control*, edited by C. T. Leondes, Academic, New York, 1991, pp. 357–410.

[38]Fijany, A., and Bejczy, A. K., "Parallel Computation of Forward Dynamics of Manipulators," JPL New Technology Rept. NPO-18706, NASA Technical Brief, Vol. 17, No. 12, Item 82, Jet Propulsion Lab., California Inst. of Technology, Pasadena, CA, Dec. 1993.

[39]Sharf, I., and D'Eleuterio, G. M. T., "An Iterative Approach to Multibody Simulations Dynamics for Parallel Implementation," *Journal of Dynamics Systems, Measurement, and Control*, Vol. 115, No. 4, 1993, pp. 730–735.

[40]Fijany, A., Sharf, I., and D'Eleuterio, G. M. T., "Parallel $\mathcal{O}(\log_2 N)$ Algortihms for the Computation of Manipulator Forward Dymanics," *IEEE Transactions Robotics and Automation*, Vol. 11, No. 3, 1995, pp. 389–400.

[41]Haug, E. J., *Intermediate Dynamics*, Prentice–Hall, Upper Saddle River, NJ, 1992, pp. 154–176.

[42]Shabana, A. A., *Computational Dynamics*, Wiley, New York, 1994, pp. 292–309, 326–328.

[43]Amirouche, F. M. L., *Computational Methods in Multibody Dynamics*, Prentice–Hall, Englewood Cliffs, NJ, 1992, pp. 424–452.

[44]Wehage, R. A., "Application of Matrix Partitioning and Recursive Projection to $\mathcal{O}(n)$ Solution of Constrained Equations of Motion," *Trends and Developments in Mechanism*, Vol. 15, No. 2, 1988, pp. 221–230.

[45]Pacheco, P. S., *Parallel Programming with MPI*, Morgan Kaufmann, San Francisco, 1997, pp. 65–396.

[46]Franklin, G. F., and Powell, J. D., *Digital Control of Dynamic Systems*, Addison–Wesley, Reading, MA, 1981, pp. 82–86.

[47]Collins, R. J., "Bandwidth Reduction by Automatic Renumbering," *International Journal of Numerical Methods in Engineering*, Vol. 6, No. 3, 1973, pp. 345–356.