Nonlinear Dyn (2012) 67:2779–2797
DOI 10.1007/s11071-011-0188-y

ORIGINAL PAPER

Multibody graph transformations and analysis

Part I: Tree topology systems

Abhinandan Jain

Received: 12 September 2010 / Accepted: 29 July 2011 / Published online: 3 September 2011

Abstract This two-part paper uses graph transformation methods to develop methods for partitioning, aggregating, and constraint embedding for multibody systems. This first part focuses on tree-topology systems and reviews the key notion of *spatial kernel operator* (*SKO*) *models* for such systems. It develops systematic and rigorous techniques for partitioning SKO models in terms of the SKO models of the component subsystems based on the *path-induced* property of the component subgraphs. It shows that the sparsity structure of key matrix operators and the mass matrix for the multibody system can be described using partitioning transformations. Subsequently, the notions of node contractions and subgraph aggregation and their role in coarsening graphs are discussed. It is shown that the tree property of a graph is preserved after subgraph aggregation if and only if the subgraph satisfies an *aggregation condition*. These graph theory ideas are used to develop SKO models for the aggregated tree multibody systems.

Keywords Multibody systems · Graph theory · Algorithms

A. Jain (✉)
Jet Propulsion Laboratory, California Institute of Technology, 4800 Oak Grove Drive, Pasadena, CA 91109, USA
e-mail: Abhi.Jain@jpl.nasa.gov

1 Introduction

The investigation of the structural properties of multibody system dynamics remains an active area of research. Graph techniques have been developed for the systematic formulation of the equations of motion [20, 24, 28, 30]. The sparsity structure of the equations of motion have been exploited to develop efficient dynamics computational algorithms [3, 7, 8, 23, 25]. Furthermore, analytical techniques such as system-level mass matrix factorization, and diagonalizing coordinate transformations have also been explored [1, 2, 9, 18, 19, 26] to simplify the dynamics formulations.

Recently, graph techniques have been used to identify general structural properties of tree multibody systems that underlie a broad family of analytical techniques and low-order algorithms for the systems [13–17]. It is seen that tree system dynamics can be described using abstract *spatial kernel operator* (*SKO*) *models*, and that several operator results including analytical mass matrix inversion follow directly from the SKO model structure—independent of the specific details of the system. Furthermore, these analytical results lead to a large family of low-order, scatter/gather recursive computational algorithms for important dynamics problems.

This two-part paper, further develops graph theory concepts and techniques for multibody system dynamics subjected to partitioning and aggregation graph transformations. *Partitioning transformations* decompose the system graph into simpler component subgraphs, while *aggregation transformations* collapse one or more component subgraphs into nodes to obtain coarser system graph representations. For example, a tree-topology system can be decomposed into a simpler tree of interconnected serial-chain segments in the system. In other instances, decompositions can reflect the natural structure of the system, e.g., a mobile platform with robotic arms equipped with multi-fingered hands for grasping and manipulating task objects. Applications of partitioning and decomposition techniques in multibody dynamics include: hierarchical dynamics and control; assembly of the overall equations of motion from those of simpler component subsystems; organizing dynamics computations (e.g., mass matrix, forward dynamics) from those of the component sub-systems for serial or parallel computations. Furthermore, partitioning techniques can help identify the sparsity structure of key dynamics matrices associated with the system dynamics that can be exploited to improve computational efficiency.

The contributions of this part are in the use of graph theory techniques to systematically study and derive results on the effect of partitioning and aggregation transformations on the SKO models for tree-topology systems. We derive the explicit partitioned structure of the system SKO model that is induced by partitioning transformations of the system graph. Since subgraph aggregation can destroy the tree graph structure, we derive rigorous sufficient conditions for tree structure preservation after the application of aggregation transformations. Preservation of the tree-topology structure leads to the natural question about the structure of the SKO model for the aggregated system. We derive explicit expressions for these aggregated SKO models. Our focus on SKO models is motivated by the easy availability of the large family of analytical and algorithmic techniques for these models. Furthermore, we show how the sparsity structure of key matrices and operators associated with tree-topology systems can be completely understood by applying the partitioning techniques developed in this paper. Part 2 of this paper [12] applies the aggregation transformation ideas developed here to develop constraint embedding techniques that extend the notion of SKO models to non-tree topology multibody systems.

The paper is organized as follows. Section 2 provides an overview of key graph theory ideas and the notion of SKO models. Section 3 focuses on partitioning transformations for SKO models. Toward this, the graph theory concepts of *induced* and *path-induced* subgraphs are introduced and their role in partitioning of abstract graphs is discussed. These ideas are used to partition dynamics models for tree multibody systems in Sect. 4. Section 5 studies the notion of *subgraph aggregation* for coarsening graphs. These ideas are subsequently used to apply aggregation transformations to dynamics models in Sect. 6. The aggregation techniques are then used to analyze the sparsity structure of the SKO model operators as well as the mass matrix in Sect. 7. The second part of this two-part paper uses the techniques derived here to develop a *constraint-embedding* technique that extends the notion of SKO models to nontree multibody systems.

2 Background: SKO models for tree multibody systems

Directed graphs (digraphs) provide natural mathematical constructs for describing the topology and connectivity of bodies in a multibody system. This section contains a brief review of key graph concepts and their use for multibody dynamics modeling.

2.1 Directed graphs and trees

A **graph** is a collection of **nodes**, and **edges** connecting pairs of nodes. A **directed graph** (also known as a **digraph**) is a graph where the edges have direction, i.e., an edge from one node to another is not the same as an edge in the reverse direction [27]. Each edge in a digraph defines a parent/child relationship between the node pair of that edge. Nodes defining an edge are said to be **adjacent** nodes. The node from which the edge emanates is referred to as the **parent node**, and the destination node is said to be the **child node**. The set of parent nodes of the kth node is denoted $\wp(k)$, and the set of its children nodes is $\complement(k)$. Nodes without parent nodes are referred to as **root nodes**. Digraphs can have zero, one, or multiple root nodes. We assume that there is at most a single edge in the same direction between any pair of nodes, i.e., parallel edges between a pair of nodes are not allowed.

A node, j, is said to be the **ancestor** of another node, i, if there is a directed path from the node j to the node i. We use the notation $i \prec j$ (or equivalently $j \succ i$) to indicate that node j is an ancestor of

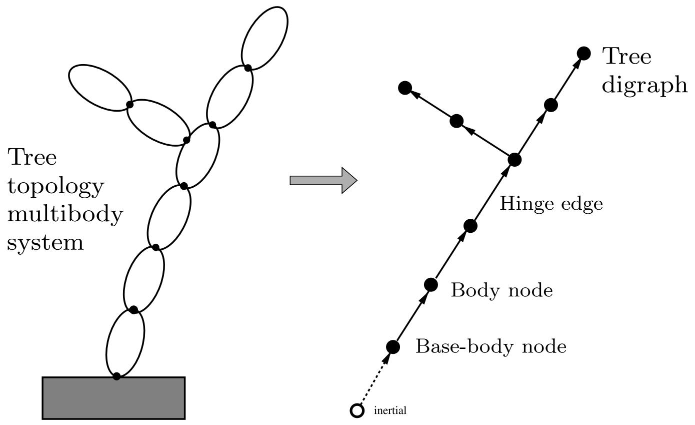


Fig. 1 The correspondence between a tree-topology multibody system and its standard tree digraph.

node i. The notation $i \not\prec j$ implies that node j is not an ancestor of node i. Node i is said to be the **descendant** of node j if j is an ancestor of node i. A pair of nodes, i and j, are said to be **related** if one of them is the ancestor of the other; otherwise they are said to be **unrelated**.

1. A **connected digraph** is a digraph such that there is an undirected path connecting any pair of nodes, i.e., it is a digraph without disjoint components.
2. A **rooted digraph** is a connected digraph with a single root node that is the ancestor of every other node in the digraph.
3. A **directed acyclic graph (DAG)** is a connected digraph without any directed cycles, i.e., there is no directed path from any node back to itself.
4. A **simply connected graph**, or a **polytree** is a DAG in which nodes are not multiply connected, i.e., there is at most one directed path between any pair of nodes.
5. A **tree**, is a polytree where a node has at most one parent node, i.e., $\wp(k)$ contains at most one node for any node k. All trees have a *unique root node*.
6. A **serial-chain** is a tree where each node has at most one child. Unlike trees, serial-chains have the stronger property that all node pairs are *related*, i.e., for any pair of nodes in the serial-chain, one of the nodes is necessarily an ancestor of the other.
7. A **forest** is a collection of disjoint trees. Removing an edge or a node from a tree converts it into a forest. Adding a common root node to the independent trees in a forest converts them into a single tree.

A tree is said to be **canonical**[1] if the index of a parent node is always greater than the index of its child node, i.e., $\wp(k) > k$ for any node k. Every rooted digraph has a **spanning tree**, i.e., a tree that contains all the nodes in the digraph and whose edges belong to the digraph. The edges removed to convert a rooted digraph into its spanning tree are referred to as **cut-edges**. The **adjacency matrix**, $\mathbb{S}$, for a digraph with n nodes is an $n \times n$ matrix where the (k, j) element is 1 if there is an edge between the kth and jth nodes, and zero otherwise.

2.2 Multibody digraphs

The **standard digraph** for a multibody system is a digraph with the inertial frame as the root node, and all the links in the system as the remaining nodes in the digraph. Thus, an n-link multibody system has a standard digraphwith $n+1$ nodes.

The edges in the digraphare defined by the motion constraints among the bodies, and between the bodies and the inertial frame. Thus, each hinge is represented by an edge, with the edges oriented from the inboard to the outboard body. Additional edges are assigned to other nonhinge motion constraints in the system. All motion constraints with respect to the inertial frame are defined so that edges from the inertial frame node to the link nodes are directed away from the inertial frame root node. These assignments result in a *rooted digraph* representation for the multibody system. Figure 1 illustrates the tree multibody system and its stan-

[1]More precisely, the canonical property of a tree depends only on the way indices are assigned to the nodes, and not the topological structure of the tree itself.

dard digraph. The convention is to depict the inertial frame node as unfilled, and the edges from the node as dashed lines. Multibody systems are classified as follows, based on the topology of their standard digraph:

- systems with tree standard digraphs are referred to as **tree-topology systems**;
- systems with serial-chain standard digraphs are referred to as the familiar **serial-chain systems**. They are special cases of tree-topology systems;
- systems with nontree standard digraphs are referred to as **closed-chain** or **constrained systems**. These digraphs can have directed cycles and/or multiply-connected nodes. Recall that every rooted digraph can be decomposed (nonuniquely) into a spanning tree together with a set of cut-edges. A decomposition into a spanning tree with $n+1$ nodes and a set of cut-edges is often used when working with closed-chain systems.

2.3 Equations of motion for tree-topology, rigid body systems

Consider a canonical n-link rigid body serial-chain system. The tip link is denoted link 1 and the base-body link as link n. The associated serial-chain tree associated with this graph is a strictly canonical tree with the parent/child relationship given by $\wp(k) = k+1$.

Using 6-dimensional spatial coordinate-free notation with $\mathcal{V}(k) \in \mathcal{R}^6$ denoting the spatial velocity (angular and linear) of the kth link, $\dot{\boldsymbol{\theta}}(k)$ the kth hinge generalized velocities, $H^*(k)$ the kth hinge map matrix, $\mathfrak{l}(k+1,k)$ the vector between the $(k+1)$th and kth body frames, and

$$\phi^*(k+1,k) = \begin{pmatrix} \mathbf{I} & \widetilde{\mathfrak{l}}(k+1,k) \\ \mathbf{0} & \mathbf{I} \end{pmatrix} \in \mathcal{R}^{6\times 6} \tag{1}$$

the rigid body transformation matrix, the link-to-link spatial velocity relationship between the $(k+1)$th links and its child body k can be expressed as [7, 11]:

$$\mathcal{V}(k) = \phi^*(k+1,k)\mathcal{V}(k+1) + H^*(k)\dot{\boldsymbol{\theta}}(k) \tag{2}$$

The overall velocity degrees of freedomfor the system is denoted $\mathcal{N}$ and is defined as the sum of all the individual hinge degrees of freedom.

Now, we introduce **stacked** vectors needed to define system level relationships. The stacked vectors $\mathcal{V}$ and θ are defined as

$$\mathcal{V} \triangleq \operatorname{col}\left\{\mathcal{V}(k)\right\}_{k=1}^{n} = \begin{bmatrix} \mathcal{V}(1) \\ \mathcal{V}(2) \\ \vdots \\ \mathcal{V}(n) \end{bmatrix} \in \mathcal{R}^{6n}, \quad \text{and}$$

$$\theta \triangleq \operatorname{col}\left\{\theta(k)\right\}_{k=1}^{n} = \begin{bmatrix} \theta(1) \\ \theta(2) \\ \vdots \\ \theta(n) \end{bmatrix} \in \mathcal{R}^{\mathcal{N}} \tag{3}$$

The $\mathcal{V}$ stacked vector consists of the component body-level $\mathcal{V}(k)$ spatial velocity vectors combined into a single large vector. Correspondingly, the θ stacked vector consists of the component body-level $\theta(k)$ generalized coordinates combined into a single large vector. The link-level (2) relationship can now be expressed equivalently at the system level [11] as

$$\mathcal{V} = \mathcal{E}_\phi^*\mathcal{V} + H^*\dot{\boldsymbol{\theta}} \tag{4}$$

where the *spatial operator* $\mathcal{E}_\phi \in \mathcal{R}^{6n\times 6n}$ is defined as

$$\begin{aligned}\mathcal{E}_\phi &= \begin{pmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \phi(2,1) & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \phi(3,2) & \dots & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \dots & \phi(n,n-1) & \mathbf{0} \end{pmatrix} \\ &= \sum_{k=1}^{n-1} \mathrm{e}_{k+1}\phi(k+1,k)\mathrm{e}_k^* \end{aligned} \tag{5}$$

$\mathrm{e}_k \in \mathcal{R}^{6n\times 6}$ denotes a block-vector containing zero entries except for the kth slot which is a 6×6 identity matrix. The block-diagonal $H \in \mathcal{R}^{\mathcal{N}\times 6n}$ spatial operator is defined as follows:

$$\begin{aligned} H &\triangleq \operatorname{diag}\left\{H(k)\right\}_{k=1}^{n} \\ &= \begin{pmatrix} H(1) & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & H(2) & \dots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \dots & H(n) \end{pmatrix} \in \mathcal{R}^{\mathcal{N}\times 6n} \end{aligned} \tag{6}$$

The $\mathcal{E}_\phi$ spatial operator is in fact a BWA[2] matrix for the system digraph, and is referred to as a *spatial ker-*

[2]A **block-weighted adjacency (BWA) matrix** [14] for a digraph is its adjacency matrix with block matrix entries. The

nel operator (*SKO*) for the system [14]. Observe that (4) is an implicit equation with $\mathcal{V}$ appearing on the both sides. However, it is explicit at the component-level, i.e., if we look at the kth row of this matrix equation, we obtain back the explicit (2) form.

The key difference between serial-chain and tree-topology multibody systems is that, in a tree system, bodies can have multiple children bodies. Examining the kinematic velocity relationships across the links, the interlink velocity relationship in (2) generalizes as follows for bodies in a tree-topology system:

$$\mathcal{V}(k) = \phi^*\big(\wp(k), k\big)\mathcal{V}\big(\wp(k)\big) + H^*(k)\dot{\boldsymbol{\theta}}(k) \tag{7}$$

That is, the spatial velocity of the kth link can be expressed as the sum of the rigidly propagated spatial velocity of the parent body and the relative spatial velocity, $H^*(k)\dot{\boldsymbol{\theta}}(k)$, across the kth hinge. Defining the $\mathcal{V}$ and $\dot{\boldsymbol{\theta}}$ stacked vectors and the H spatial operator in the same way as shown earlier, it is easy to verify that the following operator expression is a system-level rearrangement of the component-level velocity relationship in (7)

$$\mathcal{V} = \mathcal{E}_\phi^*\mathcal{V} + H^*\dot{\boldsymbol{\theta}} \tag{8}$$

with $\mathcal{E}_\phi$ is defined more generally as [13, 15]

$$\mathcal{E}_\phi = \sum_{k=1}^{n} \mathrm{e}_{\wp(k)}\phi\big(\wp(k), k\big)\mathrm{e}_k^* \tag{9}$$

The $\mathcal{E}_\phi$ SKO operator continues to be a BWA matrix for the system digraph. This time, the $\phi(\wp(k), k)$ matrices are its 6×6 weight matrices. Observe that (4) and (8) have identical form, even though the former was derived specifically for a canonical serial-chain system, while the latter holds for arbitrary tree-topology systems.

For trees, $\mathcal{E}_\phi$ is nilpotent[3] and, therefore, has a well-defined 1-resolvent,[4] $\phi = (\mathbf{I} - \mathcal{E}_\phi)^{-1}$ [14]. ϕ is referred to as the **spatial propagation operator (SPO)** associated with the $\mathcal{E}_\phi$ SKO operator. Equation (8) can thus be transformed into the following explicit form:

$$\mathcal{V} = \big(\mathbf{I} - \mathcal{E}_\phi^*\big)^{-1} H^*\dot{\boldsymbol{\theta}} = \phi^* H^*\dot{\boldsymbol{\theta}} \tag{10}$$

Differentiating the above leads to the following expression for the link spatial accelerations α:

$$\alpha = \phi^* H^*\ddot{\boldsymbol{\theta}} + \mathfrak{a} \tag{11}$$

where $\mathfrak{a}$ denotes the state-dependent Coriolis acceleration stacked vector.

With $\mathfrak{f}(k) \in \mathcal{R}^6$ denoting the interbody interaction spatial force between the parent $\wp(k)$ and the kth links, the following is the force balance expression for the kth body in a tree-topology system:

$$\mathfrak{f}(k) - \sum_{\forall j \in \mathfrak{C}(k)} \phi(k, j)\mathfrak{f}(j) = M(k)\alpha(k) + \mathfrak{b}(k) \tag{12}$$

where $M(k) \in \mathcal{R}^{6\times 6}$ is the kth link spatial inertia and $\mathfrak{b}(k)$ is the state-dependent gyroscopic spatial force for the kth body. Switching to the system-level stacked vector form, we obtain

$$\mathfrak{f} = \mathcal{E}_\phi\mathfrak{f} + \boldsymbol{M}\alpha + \mathfrak{b} \tag{13}$$

where $\boldsymbol{M}$ is a block-diagonal spatial operator with link spatial inertias along the diagonal, and $\mathfrak{b}$ is the state-dependent stacked vector of gyroscopic terms. Continuing in this mode, the following expressions summarize the equations of motion for tree-topology systems:

$$\begin{aligned}
\mathcal{V} &\overset{(8)}{=} \mathcal{E}_\phi^*\mathcal{V} + H^*\dot{\boldsymbol{\theta}} \\
\alpha &= \mathcal{E}_\phi^*\alpha + H^*\ddot{\boldsymbol{\theta}} + \mathfrak{a} \\
\mathfrak{f} &\overset{(13)}{=} \mathcal{E}_\phi\mathfrak{f} + \boldsymbol{M}\alpha + \mathfrak{b} \\
\mathcal{T} &= H\mathfrak{f}
\end{aligned} \tag{14}$$

$\mathcal{T}$ is the stacked vector of generalized forces. Using $\phi = (\mathbf{I} - \mathcal{E}_\phi)^{-1}$, these expressions can be transformed from implicit ones into the following explicit operator expressions:

$$\begin{aligned}
\mathcal{V} &= \phi^* H^*\dot{\boldsymbol{\theta}} \\
\alpha &= \phi^*(H^*\ddot{\boldsymbol{\theta}} + \mathfrak{a}) \\
\mathfrak{f} &= \phi(\boldsymbol{M}\alpha + \mathfrak{b}) \\
\mathcal{T} &= H\mathfrak{f}
\end{aligned} \tag{15}$$

(k, j) element of a BWAmatrix is a $m_k \times m_j$ weight matrix, $w(j, k)$, when there is an edge between the kth and jth nodes. Here, m_k denotes the weight dimension associated with the kth link.

[3]A matrix A is nilpotent if $A^n = \mathbf{0}$ for some finite n.

[4]A **1-resolvent** of a matrix A is defined as the $(\mathbf{I} - A)^{-1}$ matrix.

Combining the expressions in (15) leads to

$$\mathcal{T} = \mathcal{M}(\theta)\ddot{\boldsymbol{\theta}} + \mathcal{C}(\theta, \dot{\boldsymbol{\theta}}) \tag{16}$$

where

$$\begin{aligned} \mathcal{M}(\theta) &= H\phi\boldsymbol{M}\phi^* H^* \in \mathcal{R}^{\mathcal{N}\times\mathcal{N}} \quad \text{and} \\ \mathcal{C}(\theta, \dot{\boldsymbol{\theta}}) &\triangleq H\phi(\boldsymbol{M}\phi^*\mathfrak{a} + \mathfrak{b}) \in \mathcal{R}^{\mathcal{N}} \end{aligned} \tag{17}$$

$\mathcal{M} \in \mathcal{R}^{\mathcal{N}\times\mathcal{N}}$ denotes the mass matrix for the tree-topology system, and $\mathcal{C} \in \mathcal{R}^{\mathcal{N}}$ is the vector nonlinear Coriolis and gyroscopic velocity dependent terms.

2.4 SKO models for tree systems

References [13–15] have shown that SKO and SPO operators such as $\mathcal{E}_\phi$ and ϕ occur in internal-coordinate tree multibody kinematics and dynamics formulations. The occurrence of these operators is invariant to specific details such as body indexing, rigid/flexible links, regular or flexible/geared joints etc. Only the details of the weight matrices change with the system—not the SKO and SPO 1-resolvent properties. In general, the weight matrices can be nonsquare, noninvertible, and of nonuniform size. We formalize this general property with the definition of SKO models for multibody systems.

An **SKO model** for an n-links tree-topology multibody system consists of the following:

1. A tree digraph reflecting the bodies and their connectivity in the system.
2. An $\mathcal{E}_{\mathbb{A}}$ SKO operator and associated SPO operator, $\mathbb{A} \triangleq (\mathbf{I} - \mathcal{E}_{\mathbb{A}})^{-1}$.
3. A full-rank block-diagonal, joint map matrix operator, H.
4. A block-diagonal and positive-definite spatial inertia operator, $\boldsymbol{M}$.
5. Stacked vectors: $\dot{\boldsymbol{\theta}}$ denoting independent generalized velocities, $\mathcal{T}$ the generalized forces, $\mathcal{V}$ the node velocities, α the node accelerations, $\mathfrak{f}$ the interbody forces, $\mathfrak{a}$ the body Coriolis accelerations, $\mathfrak{b}$ the body gyroscopic forces, $\mathcal{N}$ the number of degrees of freedom, and equations of motion defined as:

$$\begin{aligned} \mathcal{V} &= \mathbb{A}^* H^* \dot{\boldsymbol{\theta}} \\ \alpha &= \mathbb{A}^*(H^*\ddot{\boldsymbol{\theta}} + \mathfrak{a}) \\ \mathfrak{f} &= \mathbb{A}(\boldsymbol{M}\alpha + \mathfrak{b}) \\ \mathcal{T} &= H\mathfrak{f} \end{aligned} \tag{18}$$

Thus,

$$\mathcal{T} = \mathcal{M}(\theta)\ddot{\boldsymbol{\theta}} + \mathcal{C}(\theta, \dot{\boldsymbol{\theta}}) \tag{19}$$

where

$$\begin{aligned} \mathcal{M}(\theta) &\triangleq H\mathbb{A}\boldsymbol{M}\mathbb{A}^* H^* \in \mathcal{R}^{\mathcal{N}\times\mathcal{N}} \quad \text{and} \\ \mathcal{C}(\theta, \dot{\boldsymbol{\theta}}) &\triangleq H\mathbb{A}(\boldsymbol{M}\mathbb{A}^*\mathfrak{a} + \mathfrak{b}) \in \mathcal{R}^{\mathcal{N}} \end{aligned} \tag{20}$$

$\mathcal{M}$ is the symmetric and positive-definite mass matrix for the tree system and $\mathcal{C}$ is the vector of nonlinear Coriolis and gyroscopic velocity dependent terms. Equation (20) is the *Newton–Euler operator factorization* of the mass matrix.

SKO models are also referred to by the $(H, \mathbb{A}, \boldsymbol{M})$ triplet of operators that define them.

References [13, 15] show that several analytical techniques and efficient algorithms follow directly from the SKO model structure for multibody systems. Examples of these include:

- Recursive $O(\mathcal{N})$ procedures for computing SPO operator and stacked vector products.
- General $O(\mathcal{N})$ Newton–Euler inverse dynamics algorithms.
- Solutions for the forward Lyapunov equations and decomposition of $\mathbb{A}X\mathbb{B}^*$ operator product.[5]
- General $O(\mathcal{N}^2)$ algorithm for computing the mass matrix.
- Solution for the backward Lyapunov equations and decomposition of $\mathbb{A}^*X\mathbb{B}$ operator product.
- Recursive algorithms for computing the $\mathbb{A}^*X\mathbb{B}$ operator product.
- The general $O(\mathcal{N})$ articulated body inertia solution of the Riccati equation.
- The alternative Innovations Operator Factorization of the mass matrix.
- An analytical expression for the inverse of the mass matrix.
- The analytical expression for the determinant of the mass matrix.
- The general $O(\mathcal{N})$ AB forward dynamics algorithm.

These results require no assumptions on the SKO model regarding the SKO weight matrices, the components of the other spatial operators, or the structure of

[5]Here, $\mathbb{A}$ and $\mathbb{B}$ denote an arbitrary pair of SPO operators, and X a compatible block diagonal matrix, for the SKO model.

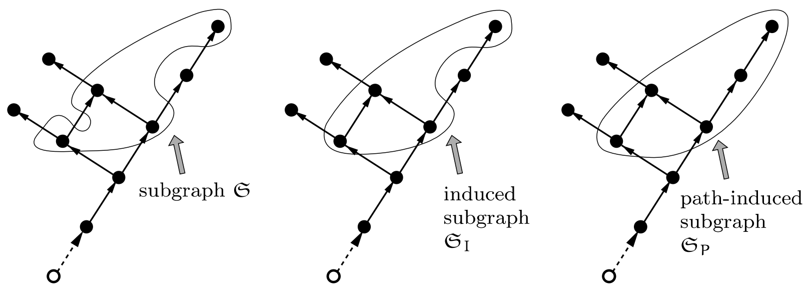

subgraph $\mathfrak{S}$
induced subgraph $\mathfrak{S}_I$
path-induced subgraph $\mathfrak{S}_P$

Fig. 2 Examples of a subgraph $\mathfrak{S}$ and its induced and path-induced sub-graphs $\mathfrak{S}_I$ and $\mathfrak{S}_P$, respectively

the tree digraph. Thus, any multibody system formulation satisfying the requirements of the SKO model has available to it the full spectrum of these techniques and efficient algorithms.

3 Partitioning digraphs

In this section, we study the partitioning of SKO models for tree multibody systems. We begin with a discussion of partitioning concepts from graph theory and subsequently apply them to multibody systems.

A **subgraph** $\mathfrak{S}$ of a digraph, $\mathfrak{T}$, is defined as a digraph containing a subset of the nodes and edges in $\mathfrak{T}$. $\mathfrak{S}$ is said to be **induced** if all edges in $\mathfrak{T}$ connecting node pairs in $\mathfrak{S}$ are also in $\mathfrak{S}$ [4, 27]. In other words, $\mathfrak{S}$ is induced if all node pairs in $\mathfrak{S}$ that are adjacent in $\mathfrak{T}$ are also adjacent in $\mathfrak{S}$. Thus, induced subgraphs preserve the *adjacency* property for node pairs, so that a pair of nodes in an induced subgraph are adjacent if and only if they are adjacent in the parent digraph. The **induced subgraph** $\mathfrak{S}_I$ for a subgraph $\mathfrak{S}$ is the minimal subgraph containing $\mathfrak{S}$ that is also an induced subgraph. A subgraph, and its induced subgraph, contain the same nodes, and differ only in the edges they contain.

A subgraph $\mathfrak{S}$ of $\mathfrak{T}$ is said to be **path-induced** if it contains all the paths (nodes and edges) in $\mathfrak{T}$ that connect node pairs in $\mathfrak{S}$. A path-induced subgraph has no missing nodes or edges for paths in $\mathfrak{T}$ that connect the nodes in $\mathfrak{S}$. Thus path-induced sub-graphs preserve the *relatedness* property for node pairs, so that a pair of nodes in a path-induced subgraph are *related* if and only if they are related in the parent digraph. The **path-induced subgraph** $\mathfrak{S}_P$ for a subgraph $\mathfrak{S}$ is the minimal subgraph of $\mathfrak{T}$ containing $\mathfrak{S}$ that is path-induced. Figure 2 illustrates a subgraph, and its induced and path-induced subgraphs. The path-induced property applies even to disconnected subgraphs. A path-induced subgraph, will generally contain more nodes and edges than the original subgraph or its induced subgraph. That is,

$$\mathfrak{S} \subseteq \mathfrak{S}_I \subseteq \mathfrak{S}_P \tag{21}$$

The following corollary shows that nontree path-induced sub-graphs are "complete", in the sense that if such a subgraph contains part of a loop, then it must necessarily contain all the nodes and edges in the full loop, or, if it contains multiply-connected nodes, then all nodes and edges on paths connecting these nodes must also belong to the subgraph.

Corollary 3.1 (Non-tree path-induced sub-graphs) *Let $\mathfrak{S}$ be a path-induced subgraph of a digraph $\mathfrak{T}$.*

1. *If $\mathfrak{S}$ contains an edge that is a part of a directed cycle in $\mathfrak{T}$, then the full cycle must also be in $\mathfrak{S}$.*
2. *If $\mathfrak{S}$ contains a pair of multiply-connected nodes (i.e., nodes connected by more than one path), then all the paths connecting them must also be in $\mathfrak{S}$.*

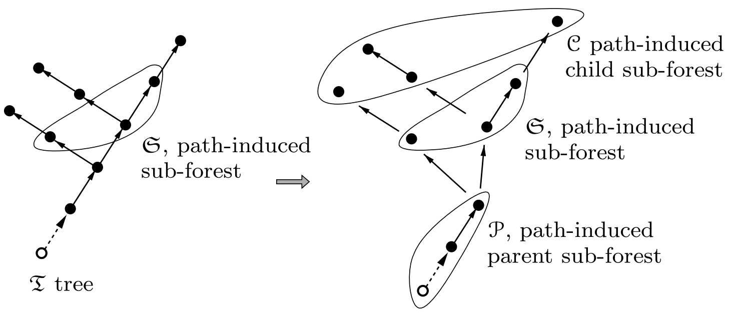
Proof

1. Since a directed cycle containing an edge, represents a directed path connecting the node pair for the edge, the path (and the cycle) must belong to $\mathfrak{S}$ since it is path-induced.
2. Similarly, all paths connecting a pair of nodes in $\mathfrak{S}$ must be in the subgraph since it is path-induced, and the result follows.

□

3.1 Partitioning by path-induced subgraphs

We now turn our attention to partitioning of digraphs. The following lemma shows that path-induced sub-graphs are special in that they partition digraphs into component sub-graphs that are themselves path-induced.



$\mathfrak{S}$, path-induced sub-forest
$\mathfrak{T}$ tree
$\mathfrak{C}$ path-induced child sub-forest
$\mathfrak{S}$, path-induced sub-forest
$\mathfrak{P}$, path-induced parent sub-forest

Fig. 3 Illustration of the disjoint partitioning of a $\mathfrak{T}$ tree by a path-induced sub-forest $\mathfrak{S}$ into parent $\mathfrak{P}$, and child $\mathfrak{C}$ path-induced sub-forests

Lemma 1 (Induced partitions of digraphs) *Assume that $\mathfrak{S}$ is a path-induced subgraph of a digraph $\mathfrak{T}$. Define the $\mathfrak{C}$ and $\mathfrak{P}$ sub-graphs as follows*:

- $\mathfrak{C}$ **child subgraph** *is the induced subgraph for the set of nodes that are not in $\mathfrak{S}$, but are descendants of the nodes in $\mathfrak{S}$, and*
- $\mathfrak{P}$ **parent subgraph** *is the induced subgraph for the remaining nodes that are neither in $\mathfrak{S}$ nor in $\mathfrak{C}$.*

The $\mathfrak{S}$, $\mathfrak{C}$ and $\mathfrak{P}$ subgraphs represent a disjoint partitioning of the nodes in $\mathfrak{T}$. The following properties hold for the $\mathfrak{C}$ and $\mathfrak{P}$ subgraphs:

1. $\mathfrak{C}$ *is path-induced.*
2. $\mathfrak{P}$ *is path-induced.*

Proof

1. Assume that there are nodes $i \succ j \succ k$ where i, $k \in \mathfrak{C}$. Since j is a descendant of i, it must belong to either $\mathfrak{S}$ or $\mathfrak{C}$. We will show that node j must necessarily belong to $\mathfrak{C}$. First, since $i \in \mathfrak{C}$, by definition there is a node $l \in \mathfrak{S}$ which is an ancestor of node i, i.e., $l \succ i \succ j$.
   If node $j \in \mathfrak{S}$, we would have $l \succ i \succ j$. Since $i \notin \mathfrak{S}$, this would contradict the assumption that $\mathfrak{S}$ is path-induced. Hence, $j \notin \mathfrak{S}$ and $j \in \mathfrak{C}$. Therefore, $\mathfrak{C}$ is path-induced.
2. Assume that there are nodes $i \succ j \succ k$, where $i, k \in \mathfrak{P}$. We will show that node j must necessarily be in $\mathfrak{P}$.
   If $j \in \mathfrak{S}$, then, since node k is its child, k must be either in $\mathfrak{S}$ or $\mathfrak{C}$, which would contradict our assumption that $k \in \mathfrak{P}$. Thus $j \notin \mathfrak{S}$.
   If instead $j \in \mathfrak{C}$, then there must a node $l \in \mathfrak{S}$ that is the ancestor of j, i.e., $l \succ j$. This would imply that $l \succ k$ as well, since $j \succ k$. Since k will then be a descendant of $l \in \mathfrak{S}$, it must belong to either $\mathfrak{S}$ or $\mathfrak{C}$. This would contradict our assumption that $k \in \mathfrak{P}$. Hence, node j must be in $\mathfrak{P}$. This establishes that $\mathfrak{P}$ is path-induced. □

$\mathfrak{P}$ contains nodes that are ancestors of the nodes in $\mathfrak{S}$ but are not themselves in $\mathfrak{S}$, together with all nodes that are *unrelated* with any of the nodes in $\mathfrak{S}$. While all nodes in $\mathfrak{T}$ belong to one of the $\mathfrak{S}$, $\mathfrak{C}$ and $\mathfrak{P}$ subgraphs, connecting edges between these subgraphs do not belong to any of the subgraphs.

4 Partitioning SKO models

When $\mathfrak{T}$ is a *tree* digraph, a path-induced subgraph of $\mathfrak{T}$ is in fact a *path-induced sub-forest*, i.e., it is a collection of one or more disjoint path-induced subtrees. Thus, in the partitioning described in Lemma 1, $\mathfrak{S}$, $\mathfrak{C}$, and $\mathfrak{P}$ are all path-induced subforests. This is illustrated in Fig. 3. The partitioning process can be continued on the component sub-forests to further subdivide them into finer-grain path-induced subforests, if desired.

4.1 Partitioning SKO model operators

When $\mathfrak{T}$ is the tree digraph for an SKO model, the disjoint partitioning induced by a path-induced subgraph $\mathfrak{S}$ also partitions the bodies in the multibody system into disjoint component multibody systems, corresponding to the $\mathfrak{S}$, $\mathfrak{C}$, and $\mathfrak{P}$ path-induced subforests. Since these are subforests, each of the component systems have well-defined SKO spatial operators, denoted $\mathcal{E}_{\mathbb{A}_{\mathfrak{S}}}$, $\mathcal{E}_{\mathbb{A}_{\mathfrak{C}}}$, and $\mathcal{E}_{\mathbb{A}_{\mathfrak{P}}}$, respectively.

For ease of exposition, we assume that $\mathfrak{T}$ is a *canonical* tree. The $\mathfrak{S}$, $\mathfrak{C}$, and $\mathfrak{P}$ sub-graphs are then also canonical. Therefore, the $\mathcal{E}_{\mathbb{A}}$, $\mathcal{E}_{\mathbb{A}_{\mathfrak{S}}}$, $\mathcal{E}_{\mathbb{A}_{\mathfrak{C}}}$, and $\mathcal{E}_{\mathbb{A}_{\mathfrak{P}}}$ SKO operators are all strictly lower-triangular. The system-level $\mathcal{E}_{\mathbb{A}}$ SKO operator can be expressed in the following block-partitioned form:

$$\mathcal{E}_{\mathbb{A}} = \begin{pmatrix} \mathcal{E}_{\mathbb{A}_{\mathcal{C}}} & \mathbf{0} & \mathbf{0} \\ \mathcal{B}_{\mathfrak{S}} & \mathcal{E}_{\mathbb{A}_{\mathfrak{S}}} & \mathbf{0} \\ \mathbf{0} & E_{\mathfrak{S}} & \mathcal{E}_{\mathbb{A}_{\mathcal{P}}} \end{pmatrix} \tag{22}$$

The SKO operators of the individual subforests form the block-diagonal elements of $\mathcal{E}_{\mathbb{A}}$, with:

1. The $\mathcal{E}_{\mathbb{A}_{\mathcal{C}}}$ SKO operator for $\mathcal{C}$—with the smallest indices—being in the upper left corner.
2. The $\mathcal{E}_{\mathbb{A}_{\mathcal{P}}}$ SKO operator for $\mathcal{P}$—with the largest indices—being in the lower right corner.
3. The $\mathcal{E}_{\mathbb{A}_{\mathfrak{S}}}$ SKO operator for $\mathfrak{S}$ being in the middle.
4. $\mathcal{B}_{\mathfrak{S}}$ is a *connector* block, whose nonzero elements are for the parent/child edges between the nodes in $\mathfrak{S}$ and their children in $\mathcal{C}$.
5. $E_{\mathfrak{S}}$ is also a *connector* block, whose nonzero elements are for the parent/child edges between the nodes in $\mathcal{P}$ and their children in $\mathfrak{S}$.
6. The lower-left block is zero because none of the nodes in $\mathcal{C}$ are the children of the nodes in $\mathcal{P}$.

It is worth pointing out that the block lower-triangular structure of $\mathcal{E}_{\mathbb{A}}$ in (22) continues to hold even under the more relaxed condition where the component systems are not necessarily canonical systems themselves, but are only canonical with respect to each other. In this situation, the index of a node in $\mathcal{C}$ whose parent is in $\mathfrak{S}$ is required to be less than that of its parent. Similarly, the index of a node in $\mathfrak{S}$ whose parent is in $\mathcal{P}$ is required to be less than that of its parent.

Since the component systems are subforests, their corresponding SPOoperators are well-defined, and given by:

$$\mathbb{A}_{\mathcal{C}} \triangleq (\mathbf{I} - \mathcal{E}_{\mathbb{A}_{\mathcal{C}}})^{-1}, \qquad \mathbb{A}_{\mathfrak{S}} \triangleq (\mathbf{I} - \mathcal{E}_{\mathbb{A}_{\mathfrak{S}}})^{-1}, \\ \mathbb{A}_{\mathcal{P}} \triangleq (\mathbf{I} - \mathcal{E}_{\mathbb{A}_{\mathcal{P}}})^{-1} \tag{23}$$

The following lemma describes the corresponding partitioned structure of the system level SPO operator $\mathbb{A}$ in terms of the SPO operators of the component systems.

Lemma 2 (Partitioning of the $\mathbb{A}$ SPO operator) *The $\mathbb{A}$ SPO operator for the full system has the following partitioned structure corresponding to the partitioned structure of $\mathcal{E}_{\mathbb{A}}$ in (22):*

$$\mathbb{A} = \begin{pmatrix} \mathbb{A}_{\mathcal{C}} & \mathbf{0} & \mathbf{0} \\ \mathbb{A}_{\mathfrak{S}}\mathcal{B}_{\mathfrak{S}}\mathbb{A}_{\mathcal{C}} & \mathbb{A}_{\mathfrak{S}} & \mathbf{0} \\ \mathbb{A}_{\mathcal{P}}(E_{\mathfrak{S}}\mathbb{A}_{\mathfrak{S}}\mathcal{B}_{\mathfrak{S}})\mathbb{A}_{\mathcal{C}} & \mathbb{A}_{\mathcal{P}}E_{\mathfrak{S}}\mathbb{A}_{\mathfrak{S}} & \mathbb{A}_{\mathcal{P}} \end{pmatrix} \tag{24}$$

Observe that the SPO operators for the component subgraphs are the block-diagonal elements of $\mathbb{A}$.

Proof Start with the partitioned expression for $\mathcal{E}_{\mathbb{A}}$ in (22), and observe that

$$\mathbb{A}^{-1} = (\mathbf{I} - \mathcal{E}_{\mathbb{A}}) \\ \overset{(22)}{=} \begin{pmatrix} \mathbf{I} - \mathcal{E}_{\mathbb{A}_{\mathcal{C}}} & \mathbf{0} & \mathbf{0} \\ -\mathcal{B}_{\mathfrak{S}} & \mathbf{I} - \mathcal{E}_{\mathbb{A}_{\mathfrak{S}}} & \mathbf{0} \\ \mathbf{0} & -E_{\mathfrak{S}} & \mathbf{I} - \mathcal{E}_{\mathbb{A}_{\mathcal{P}}} \end{pmatrix} \\ \overset{(23)}{=} \begin{pmatrix} \mathbb{A}_{\mathcal{C}}^{-1} & \mathbf{0} & \mathbf{0} \\ -\mathcal{B}_{\mathfrak{S}} & \mathbb{A}_{\mathfrak{S}}^{-1} & \mathbf{0} \\ \mathbf{0} & -E_{\mathfrak{S}} & \mathbb{A}_{\mathcal{P}}^{-1} \end{pmatrix}$$

The result follows by verifying that the product of this expression for $\mathbb{A}^{-1}$, with $\mathbb{A}$ in (24), is indeed the identity matrix. □

If the $\mathcal{P}$ subgraph is empty, then the rows and columns for the parent subgraph do not exist in the partitioned structure. Similarly, if $\mathcal{C}$ is empty, then the corresponding columns and rows for the child subgraph do not exist in the partitioned structure.

4.2 Partitioning of an SKO model

The partitioning of the tree digraph for an SKO model also induces the following corresponding partitioning of the system-level block-diagonal H and $\boldsymbol{M}$ operators:

$$H = \begin{pmatrix} H_{\mathcal{C}} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & H_{\mathfrak{S}} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & H_{\mathcal{P}} \end{pmatrix} \quad \text{and} \\ \boldsymbol{M} = \begin{pmatrix} \boldsymbol{M}_{\mathcal{C}} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{M}_{\mathfrak{S}} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \boldsymbol{M}_{\mathcal{P}} \end{pmatrix} \tag{25}$$

Observe that $(H_{\mathcal{C}}, \mathbb{A}_{\mathcal{C}}, \boldsymbol{M}_{\mathcal{C}})$, $(H_{\mathfrak{S}}, \mathbb{A}_{\mathfrak{S}}, \boldsymbol{M}_{\mathfrak{S}})$, and $(H_{\mathcal{P}}, \mathbb{A}_{\mathcal{P}}, \boldsymbol{M}_{\mathcal{P}})$ define SKO models for the component tree-topology multibody systems associated with the $\mathcal{C}$, $\mathfrak{S}$, and $\mathcal{P}$ subgraphs, respectively.

The following corollary looks at the case where a system is partitioned into just outer and inner subsystems, and shows that the mass matrix subblock corresponding to the outer system is simply the mass matrix of the outer subsystem itself.

Corollary 4.1 (Mass matrix invariance of the outer sub-system) *Consider an SKO model partitioned as in* (22), *but with empty* $\mathcal{C}$. *That is, the system is partitioned into inner,* $\mathcal{P}$, *and outer,* $\mathfrak{S}$, *SKO models. Then the overall SKO model mass matrix has the following partitioned structure*:

$$\mathcal{M} = \begin{pmatrix} \mathcal{M}_{\mathfrak{S}} & X \\ X^* & \mathcal{M}_{\mathcal{P}} + Y \end{pmatrix} \tag{26}$$

where

$$\begin{aligned} \mathcal{M}_{\mathfrak{S}} &\triangleq H_{\mathfrak{S}} \mathbb{A}_{\mathfrak{S}} \boldsymbol{M}_{\mathfrak{S}} \mathbb{A}^*_{\mathfrak{S}} H^*_{\mathfrak{S}}, \\ \mathcal{M}_{\mathcal{P}} &\triangleq H_{\mathcal{P}} \mathbb{A}_{\mathcal{P}} \boldsymbol{M}_{\mathcal{P}} \mathbb{A}^*_{\mathcal{P}} H^*_{\mathcal{P}}, \\ X &\triangleq H_{\mathfrak{S}} \mathbb{A}_{\mathfrak{S}} \boldsymbol{M}_{\mathfrak{S}} \mathbb{A}^*_{\mathfrak{S}} E^*_{\mathfrak{S}} \mathbb{A}^*_{\mathcal{P}} H^*_{\mathcal{P}}, \\ Y &\triangleq H_{\mathcal{P}} \mathbb{A}_{\mathcal{P}} E_{\mathfrak{S}} \mathbb{A}_{\mathfrak{S}} \boldsymbol{M}_{\mathfrak{S}} \mathbb{A}^*_{\mathfrak{S}} E^*_{\mathfrak{S}} \mathbb{A}^*_{\mathcal{P}} H^*_{\mathcal{P}} \end{aligned} \tag{27}$$

Proof The expressions in (26) and (27) are obtained by directly evaluating $\mathcal{M} = H\mathbb{A}\boldsymbol{M}\mathbb{A}^*H^*$ using the following component partitioned expressions from (24) and (25):

$$H = \begin{pmatrix} H_{\mathfrak{S}} & \mathbf{0} \\ \mathbf{0} & H_{\mathcal{P}} \end{pmatrix}, \qquad \mathbb{A} = \begin{pmatrix} \mathbb{A}_{\mathfrak{S}} & \mathbf{0} \\ \mathbb{A}_{\mathcal{P}} E_{\mathfrak{S}} \mathbb{A}_{\mathfrak{S}} & \mathbb{A}_{\mathcal{P}} \end{pmatrix},$$

$$\boldsymbol{M} = \begin{pmatrix} \boldsymbol{M}_{\mathfrak{S}} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{M}_{\mathcal{P}} \end{pmatrix}$$ □

Observe that $\mathcal{M}_{\mathfrak{S}}$ and $\mathcal{M}_{\mathcal{P}}$ are the respective mass matrices for the $\mathfrak{S}$ and $\mathcal{P}$ SKO models. Also, observe that the upper right sub-block $\mathcal{M}_{\mathfrak{S}}$ of $\mathcal{M}$ is independent of quantities associated with the $\mathcal{P}$ subgraph, i.e., the elements of the $\mathcal{M}_{\mathfrak{S}}$ mass matrix do not depend upon the properties or generalized coordinates of inboard bodies.

5 Aggregating sub-graphs

In this section, we use the partitioning techniques developed so far to study sub-structuring of SKO models using subgraph aggregation. We begin with the development of the aggregation concepts for digraphs and subsequently apply them to SKO models.

5.1 Edge and node contractions

In graph theory, **edge-contraction** is referred to as the process of collapsing the node pair for an edge, into a single node [27]. The node and edge neighbors of the original pair of nodes become neighbors of the new *aggregated* node. Thus, the parent and children nodes of either of the original nodes (not including the node pair themselves) are the parent and children nodes of the aggregated node in the transformed digraph.

For a tree, the result of each such edge-contraction is once again a tree, with one fewer node. The edge-contraction process can be repeated to aggregate multiple edges in a tree. Thus, aggregating a sub-tree of a tree, using edge-contraction, results in a digraph that is also a tree.

Node-contraction is a more general concept that applies to pairs of nodes that are not necessarily connected by an edge. The node-contraction of a pair of nodes replaces the pair of nodes with a single aggregated node, where all the neighboring nodes and edges of the original pair become neighbors of the new aggregated node. Thus, edge-contraction is equivalent to node-contracting the node pair for an edge. Unlike edge-contractions, node-contractions generally *do not* preserve the tree property of a digraph. Node-contractions can be applied repeatedly to aggregate multiple nodes in a subgraph. Formally, **subgraph aggregation** is defined as the process of transforming a digraph by applying node-contraction to all the nodes in the subgraph. This is illustrated in Fig. 4. It shows examples of trees transformed into multiply-connected and cyclic digraphs, following subgraph aggregation. Later, we will examine the conditions under which the tree property is preserved following subgraph aggregation.

The following lemma shows that aggregating a subgraph or its induced subgraph result in the same transformed digraph.

Lemma 3 (Aggregation of a subgraph and its induced subgraph) *Let* $\mathfrak{S}$ *denote the subgraph of a digraph* $\mathfrak{T}$. *Then the new digraph,* $\mathfrak{T}_{\mathfrak{S}}$, *created by aggregating* $\mathfrak{S}$, *is the same as the one obtained by aggregating the induced subgraph,* $\mathfrak{S}_I$, *i.e.,* $\mathfrak{T}_{\mathfrak{S}} \equiv \mathfrak{T}_{\mathfrak{S}_I}$.

Proof $\mathfrak{S}$ and $\mathfrak{S}_I$ contain the same nodes, while the latter contains all the edges connecting the nodes as well. Node-contraction of a pair of nodes removes all

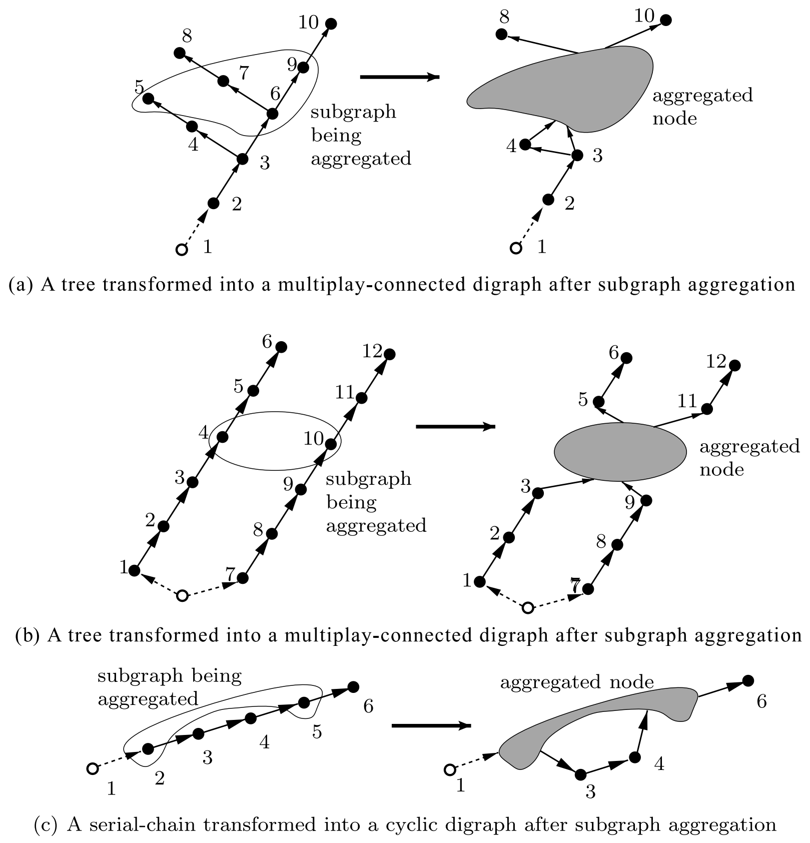

subgraph being aggregated
aggregated node
subgraph being aggregated
aggregated node
subgraph being aggregated
aggregated node

Fig. 4 Examples of subgraph aggregation using node-contraction

(a) A tree transformed into a multiplay-connected digraph after subgraph aggregation

(b) A tree transformed into a multiplay-connected digraph after subgraph aggregation

(c) A serial-chain transformed into a cyclic digraph after subgraph aggregation

edges connecting the node-pair. Thus, aggregating $\mathfrak{S}$ by node-contraction removes all edges connecting the nodes in $\mathfrak{S}$, whether or not the edges are in $\mathfrak{S}$ itself. This implies that the aggregation of $\mathfrak{S}$ or $\mathfrak{S}_I$ results in the same transformed digraph. □

One implication of this lemma is that the nodes and edges missing from the transformed tree, after aggregating $\mathfrak{S}$, are precisely the nodes and edges in the $\mathfrak{S}_I$ induced subgraph of $\mathfrak{S}$.

5.2 Tree preservation after subgraph aggregation

Let $\mathfrak{S}$ denote a subgraph of a *tree* digraph, $\mathfrak{T}$. The aggregation of $\mathfrak{S}$ results in a new $\mathfrak{T}_{\mathfrak{S}}$ digraph, where all the nodes associated with the nodes in $\mathfrak{S}$ are replaced by a single node. We will henceforth refer to this aggregated node, as node $\mathfrak{S}$. Topologically, all the parent nodes of the $\mathfrak{S}$ subgraph, denoted $\wp(\mathfrak{S})$, are now the parents of node $\mathfrak{S}$, and all the children nodes of the $\mathfrak{S}$ subgraph, denoted $\mathfrak{C}(\mathfrak{S})$, are now the children of node $\mathfrak{S}$. In general, $\mathfrak{T}_{\mathfrak{S}}$ obtained by this topological transformation will *not* be a tree.

The following defines the **aggregation condition** for subgraphs.

Assumption 1 (Aggregation condition) *A subgraph $\mathfrak{S}$, of a* tree *digraph $\mathfrak{T}$, is said to satisfy the aggregation condition if*:

1. $\mathfrak{S}$ *is an induced subgraph, i.e.*, $\mathfrak{S} \equiv \mathfrak{S}_I$.
2. $\wp(\mathfrak{S})$ *contains exactly one node—one that is necessarily the ancestor of all the nodes in* $\mathfrak{S}$.

The induced requirement for a $\mathfrak{S}$ satisfying the aggregation condition is a mild one since Lemma 3 states that aggregating a subgraph, or its induced subgraph, lead to the same transformed digraph. Some examples of a tree subgraph $\mathfrak{S}$, satisfying the aggregation condition, are:

- a single node
- a serial-chain segment
- a subtree

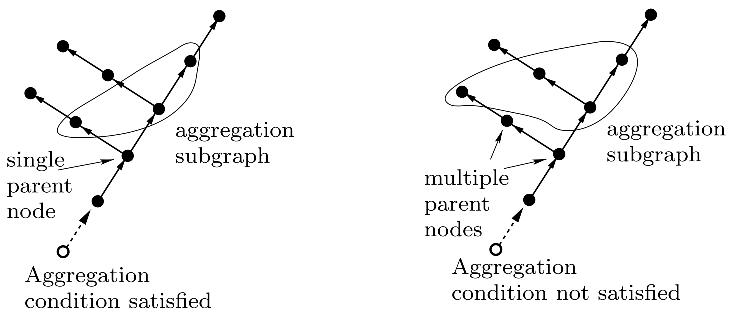


Fig. 5 Examples of path-induced subgraphs that do, and do not, satisfy the aggregation condition

The following lemma shows that any subgraph of a tree satisfying the aggregation condition is a path-induced subgraph.

Lemma 4 (Aggregation condition and path-induced subgraphs) *If a subgraph $\mathfrak{S}$ of a $\mathfrak{T}$ tree satisfies the aggregation condition, then it is path-induced, i.e., $\mathfrak{S} \equiv \mathfrak{S}_P$.*

Proof Since $\mathfrak{S}$ satisfies the aggregation condition, it is an induced subgraph. An induced subgraph of a tree can differ from its path-induced subgraph in the nodes and edges that lie on the paths connecting nodes in $\mathfrak{S}$.

First, let us consider the case where the node sets are the same. Since $\mathfrak{S}$ is induced, there are no missing edges in $\mathfrak{S}$, and, therefore, even the set of edges are exactly the ones in the path-induced subgraph. Hence, $\mathfrak{S}$ and its path-induced subgraph are the same.

Let us now consider the case when the node sets are *not* the same. Then there exists a node k in the path-induced subgraph $\mathfrak{S}_P$ that is not in $\mathfrak{S}$. Node k must lie on a path connecting a pair of nodes in $\mathfrak{S}$. This implies that there is a node l on this path that belongs to $\wp(\mathfrak{S})$. However, since k is in the interior of the path, node l is not the ancestor of all the nodes in $\mathfrak{S}$. This contradicts the assumption that $\mathfrak{S}$ satisfies the aggregation condition, which allows for only a single ancestor node for $\mathfrak{S}$. Hence, $\mathfrak{S}$ must be a path-induced subgraph. □

The converse is not true, i.e., not all path-induced subgraphs satisfy the aggregation condition. Figure 5 illustrates cases of path-induced subgraphs that do, and do not, satisfy the aggregation condition.

The following lemma shows that the aggregation condition is a necessary and sufficient condition for the $\mathfrak{T}_{\mathfrak{S}}$ aggregated digraph to be a tree.

Lemma 5 (Tree property after subgraph aggregation) *Let $\mathfrak{S}$ denote an induced subgraph of a tree digraph $\mathfrak{T}$. Let us assume that $\mathfrak{S}$ is aggregated to create a new $\mathfrak{T}_{\mathfrak{S}}$ digraph with aggregated node $\mathfrak{S}$. Then the aggregated $\mathfrak{T}_{\mathfrak{S}}$ digraph is a tree if and only if the original subgraph $\mathfrak{S}$ satisfies the aggregation condition.*

Proof

1. First, let us assume the $\mathfrak{T}_{\mathfrak{S}}$ is indeed a tree, and we will show that $\wp(\mathfrak{S})$ must contain a single node. Since $\mathfrak{T}_{\mathfrak{S}}$ is a tree, all of its nodes, including node $\mathfrak{S}$, can have at most one parent. Thus, $\wp(\mathfrak{S})$ can have at most one node. The $\mathfrak{S}$ subgraph therefore satisfies the aggregation condition.
2. Now, let us consider the converse case, i.e., assume that $\wp(\mathfrak{S})$ contains a single node that is the ancestor for all the nodes in $\mathfrak{S}$. We will prove that $\mathfrak{T}_{\mathfrak{S}}$ must then be a tree. We need to show that $\mathfrak{T}_{\mathfrak{S}}$ is not a polytree, is not multiply-connected, and does not have directed cycles.
   (a) To show that $\mathfrak{T}_{\mathfrak{S}}$ is not a polytree, we need to focus only on node $\mathfrak{S}$ and show that it cannot have multiple parents. This follows directly from the assumption that $\wp(\mathfrak{S})$ contains only one node.
   (b) To show that $\mathfrak{T}_{\mathfrak{S}}$ is not multiply-connected, we need to focus only on the node $\mathfrak{S}$ and show that there is no node in $\mathfrak{T}_{\mathfrak{S}}$ from which there is more than one directed path to the node $\mathfrak{S}$. Since $\mathfrak{T}$ is a tree, it has no multiply-connected nodes. Thus, the only way for node $\mathfrak{S}$ to be multiply-connected is for multiple paths to end at node $\mathfrak{S}$. This would mean that the node $\mathfrak{S}$ has multiple parents—one from each of the paths that end on it. This would contradict the assumption that $\wp(\mathfrak{S})$ contains a single node.
   (c) Now, we show that $\mathfrak{T}_{\mathfrak{S}}$ cannot have directed cycles. Again, we need to focus on node $\mathfrak{S}$,

and show that it cannot be part of a directed cycle. Assume that such a cycle exists. This implies that there are nodes $k, j \in \mathfrak{S}$, and a node $i \notin \mathfrak{S}$ such that $k \succ i \succ j$. In other words, there is a path from k to j containing node i. This would imply that there is a node, l, on the path from i to j that is a parent node for $\mathfrak{S}$. Since $\wp(\mathfrak{S})$ contains only one node, l must be this ancestor node. This would imply that node l, and hence node i, is an ancestor of k. However we know that k is an ancestor of i. Hence, node $\mathfrak{S}$ is not part of any directed cycles. □

Lemma 5 shows that preservation of the tree property after subgraph aggregation requires that the subgraph satisfy the aggregation condition. Intuitively, this result is not surprising, once we recall that edge-contractions preserve the tree property, and that the aggregation of a path-induced subgraph is equivalent to applying edge-contraction to all the edges in the subgraph.

5.3 The $\mathfrak{S}_A$ aggregation subgraph

The **aggregation subgraph**, for a subgraph $\mathfrak{S}$ of a tree, is defined as the minimal subgraph containing $\mathfrak{S}$ that also satisfies the aggregation condition. The aggregation subgraph of a subgraph $\mathfrak{S}$ is denoted $\mathfrak{S}_A$.

Lemma 6 (Properties of $\mathfrak{S}_A$)

1. *$\mathfrak{S}_A$ is a path-induced subgraph.*
2. *The following containment relation holds*

$$\mathfrak{S} \subseteq \mathfrak{S}_I \subseteq \mathfrak{S}_P \subseteq \mathfrak{S}_A \tag{28}$$

3. *$\mathfrak{S}_A$ is the subforest obtained by deleting the root node from the smallest subtree containing $\mathfrak{S}$.*

Proof

1. By definition, $\mathfrak{S}_A$ satisfies the aggregation condition, and hence by Lemma 4, it is path-induced.
2. By definition, $\mathfrak{S} \subseteq \mathfrak{S}_A$. Hence, the path-induced subgraph of $\mathfrak{S}$, $\mathfrak{S}_P$, is contained in the path-induced subgraph of $\mathfrak{S}_A$, which is $\mathfrak{S}_A$ itself. This establishes the last containment in (28). The rest are restatements of (21).
3. Since $\mathfrak{S}_A$ satisfies the aggregation condition, it has a single parent node. Thus, with the parent node added in, the new subgraph is a subtree. The minimality of the sub-tree follows from the definition of a aggregation subgraph, which is required to be minimal. □

This lemma helps us better understand the relationship between a subgraph, $\mathfrak{S}$, and its induced, path-induced and aggregation subgraphs. Starting with the $\mathfrak{S}$ subgraph, we obtain its induced subgraph $\mathfrak{S}_I$ by adding in the missing edges between the nodes in $\mathfrak{S}$. Furthermore, adding in any missing nodes and edges on paths connecting nodes in $\mathfrak{S}$ leads us to the path-induced $\mathfrak{S}_P$ subgraph. To obtain the $\mathfrak{S}_A$ aggregation subgraph, we need to further grow the path-induced subgraph until we get to the smallest sub-tree containing it, and then drop the root node from this subtree. The aggregation subgraph for a subgraph is important because, as shown by Lemma 5, while the aggregation of a subgraph $\mathfrak{S}$ will generally not preserve the tree property, aggregating $\mathfrak{S}_A$ instead ensures that the aggregated digraph remains a tree. Preserving the tree property is necessary requirement for the aggregated tree to have an SKO model.

6 Transforming SKO models via aggregation

6.1 SKO operators after body aggregation

Let $\mathfrak{T}$ be a tree digraph associated with a multibody system. We assume without loss in generality that it is canonical. Let $\mathfrak{S}$ denote a subgraph satisfying the aggregation condition. By Lemma 5, the aggregated $\mathfrak{T}_\mathfrak{S}$ digraph, with aggregated node $\mathfrak{S}$, is also a tree. Let us now develop an SKO model for the system with the $\mathfrak{T}_\mathfrak{S}$ aggregated tree.

To define SKO operators for $\mathfrak{T}_\mathfrak{S}$, we first need to assign weight dimensions to the nodes in $\mathfrak{T}_\mathfrak{S}$. All nodes unaffected by the aggregation process inherit the weight dimensions from the nodes in $\mathfrak{T}$. The weight dimension for node $\mathfrak{S}$ is defined as the sum of the weight dimensions of all the nodes in the $\mathfrak{S}$ subgraph, i.e.,

$$m_\mathfrak{S} \triangleq \sum_{k\in\mathfrak{S}} m_k \tag{29}$$

Since the $\mathfrak{S}$ subgraph satisfies the aggregation condition it is path-induced. From Lemma 1, $\mathfrak{S}$ induces a disjoint partitioning of $\mathfrak{T}$ into path-induced parent $\mathcal{P}$ and child $\mathcal{C}$ subgraphs, respectively. These partitioned subgraphs allow us to express the $\mathcal{E}_\phi$ SKO operator for $\mathfrak{T}$ in the partitioned form shown in (22):

$$\mathcal{E}_{\mathbb{A}} = \begin{pmatrix} \mathcal{E}_{\mathbb{A}_{\mathcal{C}}} & \mathbf{0} & \mathbf{0} \\ \mathcal{B}_{\mathfrak{S}} & \mathcal{E}_{\mathbb{A}_{\mathfrak{S}}} & \mathbf{0} \\ \mathbf{0} & E_{\mathfrak{S}} & \mathcal{E}_{\mathbb{A}_{\mathcal{P}}} \end{pmatrix} \tag{30}$$

Lemma 7 (SKO operator for the $\mathfrak{T}_{\mathfrak{S}}$ tree) *With $\mathfrak{S}$ satisfying the aggregation condition, define the $\mathcal{E}_{\mathbb{A}_a}$ matrix using the partitioned sub-blocks of* (30) *as follows*:

$$\mathcal{E}_{\mathbb{A}_a} \triangleq \begin{pmatrix} \mathcal{E}_{\mathbb{A}_{\mathcal{C}}} & \mathbf{0} & \mathbf{0} \\ \mathcal{B}_{\mathfrak{S}} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & E_{\mathfrak{S}}\mathbb{A}_{\mathfrak{S}} & \mathcal{E}_{\mathbb{A}_{\mathcal{P}}} \end{pmatrix} \tag{31}$$

The following facts hold for $\mathcal{E}_{\mathbb{A}_a}$:

1. $\mathcal{E}_{\mathbb{A}_a}$ *is an SKO operator for the original tree, $\mathfrak{T}$.*
2. $\mathcal{E}_{\mathbb{A}_a}$ *is an SKO operator for the aggregated tree, $\mathfrak{T}_{\mathfrak{S}}$.*

Proof

1. Since $\mathfrak{S}$ satisfies the aggregation condition, $\wp(\mathfrak{S})$ contains a single node. Let us denote this node as node j. Comparing the expression for $\mathcal{E}_{\mathbb{A}_a}$ in (31) with that of $\mathcal{E}_{\mathbb{A}}$ in (30), the only term that needs further examination to establish the SKO property is the $E_{\mathfrak{S}}\mathbb{A}_{\mathfrak{S}}$ block, which differs from the $E_{\mathfrak{S}}$ block. Recall that the $E_{\mathfrak{S}}$ connector block contains non-zero entries for the edges connecting nodes in $\mathcal{P}$ to nodes in $\mathfrak{S}$ subgraphs. Therefore, $E_{\mathfrak{S}}$ has the form

$$E_{\mathfrak{S}} = \sum_{l\in\wp(\mathfrak{S})} \mathrm{e}_l \sum_{\substack{k\in\mathfrak{S}\\ \wp(k)=l}} \mathbb{A}(l,k)\mathrm{e}_k^* = \mathrm{e}_j \sum_{\substack{k\in\mathfrak{S}\\ \wp(k)=j}} \mathbb{A}(j,k)\mathrm{e}_k^* \tag{32}$$

In the above equations, we have taken some notational liberties; the e_x vectors represent appropriate size vectors for the $\mathfrak{S}$ and $\mathcal{P}$ subgraphs, instead of the full-sized ones for the $\mathfrak{T}$ tree.

It follows that

$$E_{\mathfrak{S}}\mathbb{A}_{\mathfrak{S}} \overset{(32)}{=} \mathrm{e}_j \sum_{\substack{k,i\in\mathfrak{S}\\ \wp(k)=j}} \mathbb{A}(j,k)\mathrm{e}_k^*\mathbb{A}_{\mathfrak{S}} \tag{33}$$

The $\mathrm{e}_k^*\mathbb{A}_{\mathfrak{S}}$ product is the kth row of $\mathbb{A}_{\mathfrak{S}}$. Since $\mathbb{A}_{\mathfrak{S}}$ is an SPO operator, its kth row $\mathbb{A}(k,i)$ entry is nonzero only when node i is a descendant of the kth node. Hence, (33) can be reexpressed as

$$\begin{aligned} E_{\mathfrak{S}}\mathbb{A}_{\mathfrak{S}} &\overset{(33)}{=} \mathrm{e}_j \sum_{\substack{k\in\mathfrak{S}\\ \wp(k)=j,\ k\succ i}} \mathbb{A}(j,k)\mathbb{A}(k,i)\mathrm{e}_i \\ &= \mathrm{e}_j \sum_{\substack{i\in\mathfrak{S}\\ j\succ i}} \mathbb{A}(j,i)\mathrm{e}_i^* \end{aligned} \tag{34}$$

In the last expression, $j \succ i$, denotes the condition that the jth node is an ancestor of the ith node. From (34), it is clear that only the jth row of $E_{\mathfrak{S}}\mathbb{A}_{\mathfrak{S}}$ is nonzero. The only nonzero entries in this row are for nodes in $\mathfrak{S}$ that are descendants of the jth node.[6] Thus, each column of $E_{\mathfrak{S}}\mathbb{A}_{\mathfrak{S}}$, has at most a single nonzero element, and since the central block of $\mathcal{E}_{\mathbb{A}_a}$ is zero, this implies that the same column has only a single nonzero entry in the full $\mathcal{E}_{\mathbb{A}_a}$ matrix as well. Thus, the single nonzero entry per column requirement for an SKO operator for $\mathfrak{T}$ is satisfied. Hence, $\mathcal{E}_{\mathbb{A}_a}$ is an SKO operator for $\mathfrak{T}$.
2. We have seen that $\mathcal{E}_{\mathbb{A}_a}$ is an SKO operator for $\mathfrak{T}$. For $\mathfrak{T}_{\mathfrak{S}}$, the central rows and columns of $\mathcal{E}_{\mathbb{A}_a}$ correspond to the single $\mathfrak{S}$ node in the $\mathfrak{T}_{\mathfrak{S}}$ tree. For $\mathcal{E}_{\mathbb{A}_a}$ to be an SKO operator for $\mathfrak{T}_{\mathfrak{S}}$, we need to show that it satisfies the SKO operator structural requirements, as represented by (9). That is, $E_{\mathfrak{S}}\mathbb{A}_{\mathfrak{S}}$ must be of the form $\mathrm{e}_j X$ for some X. Equation (34) satisfies this requirement, and, thus, $\mathcal{E}_{\mathbb{A}_a}$ is an SKO operator for $\mathfrak{T}_{\mathfrak{S}}$. □

Having derived the expression for an $\mathcal{E}_{\mathbb{A}_a}$ SKO operator for $\mathfrak{T}_{\mathfrak{S}}$, the following lemma derives the expression for the corresponding SPO operator for $\mathfrak{T}_{\mathfrak{S}}$.

Lemma 8 (The $\mathbb{A}_a$ SPO operator for the $\mathfrak{T}_{\mathfrak{S}}$ tree) *Using the same assumptions and notation from Lemma 7, the SPO operator, $\mathbb{A}_a$, for $\mathfrak{T}_{\mathfrak{S}}$ is given by the following*

[6] In fact all nodes in $\mathfrak{S}$ are descendants of j and, therefore, the row is fully populated.

expression:

$$\mathbb{A}_a \triangleq \mathfrak{J}_a \mathbb{A} \overset{(24)}{=} \begin{pmatrix} \mathbb{A}_{\mathfrak{C}} & \mathbf{0} & \mathbf{0} \\ \mathcal{B}_{\mathfrak{S}} \mathbb{A}_{\mathfrak{C}} & \mathbf{I} & \mathbf{0} \\ \mathbb{A}_{\mathfrak{P}}(E_{\mathfrak{S}} \mathbb{A}_{\mathfrak{S}} \mathcal{B}_{\mathfrak{S}}) \mathbb{A}_{\mathfrak{C}} & \mathbb{A}_{\mathfrak{P}} E_{\mathfrak{S}} \mathbb{A}_{\mathfrak{S}} & \mathbb{A}_{\mathfrak{P}} \end{pmatrix}$$

$$\text{where } \mathfrak{J}_a \triangleq \begin{pmatrix} \mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbb{A}_{\mathfrak{S}}^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} \end{pmatrix} \tag{35}$$

Proof Since Lemma 8 established $\mathcal{E}_{\mathbb{A}_a}$ as an SKO operator for the $\mathfrak{T}_{\mathfrak{S}}$ tree, we need to show that $\mathbb{A}_a = (\mathbf{I} - \mathcal{E}_{\mathbb{A}_a})^{-1}$ for it to be an SPO operator for $\mathfrak{T}_{\mathfrak{S}}$. Now

$$(\mathbf{I} - \mathcal{E}_{\mathbb{A}_a}) \overset{(22)}{=} \begin{pmatrix} \mathbf{I} - \mathcal{E}_{\mathbb{A}_{\mathfrak{C}}} & \mathbf{0} & \mathbf{0} \\ -\mathcal{B}_{\mathfrak{S}} & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & -E_{\mathfrak{S}} \mathbb{A}_{\mathfrak{S}} & \mathbf{I} - \mathcal{E}_{\mathbb{A}_{\mathfrak{P}}} \end{pmatrix}$$

$$\overset{(23)}{=} \begin{pmatrix} \mathbb{A}_{\mathfrak{C}}^{-1} & \mathbf{0} & \mathbf{0} \\ -\mathcal{B}_{\mathfrak{S}} & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & -E_{\mathfrak{S}} \mathbb{A}_{\mathfrak{S}} & \mathbb{A}_{\mathfrak{P}}^{-1} \end{pmatrix}$$

The result follows by verifying that the product of the above, with $\mathbb{A}_a$ in (35), is the identity matrix. □

Node $\mathfrak{S}$ in $\mathfrak{T}_{\mathfrak{S}}$ contains all the bodies associated with the nodes in the original subgraph $\mathfrak{S}$. Unlike regular rigid links, the geometry of node $\mathfrak{S}$ is variable, and depends on the hinge coordinates of the component links within the subgraph $\mathfrak{S}$. Such variable geometry bodies have been used in other dynamics modeling contexts [5, 10, 29].

6.2 SKO model for the $\mathfrak{T}_{\mathfrak{S}}$ aggregated tree

The key difference between the aggregated and the original tree is that the former treats the set of bodies in subgraph $\mathfrak{S}$ as a single body. The aggregation process provides a way of transforming and *substructuring* SKO models for tree-topology multibody systems into coarser SKO models. The aggregation process induces the following partitioning of the $\dot{\boldsymbol{\theta}}$, $\mathcal{V}$, $\mathfrak{f}$ and $\mathcal{T}$ stacked vectors:

$$\dot{\boldsymbol{\theta}} = \begin{bmatrix} \dot{\boldsymbol{\theta}}_{\mathfrak{C}} \\ \dot{\boldsymbol{\theta}}_{\mathfrak{S}} \\ \dot{\boldsymbol{\theta}}_{\mathfrak{P}} \end{bmatrix}, \qquad \mathcal{V} = \begin{bmatrix} \mathcal{V}_{\mathfrak{C}} \\ \mathcal{V}_{\mathfrak{S}} \\ \mathcal{V}_{\mathfrak{P}} \end{bmatrix},$$
$$\mathfrak{f} = \begin{bmatrix} \mathfrak{f}_{\mathfrak{C}} \\ \mathfrak{f}_{\mathfrak{S}} \\ \mathfrak{f}_{\mathfrak{P}} \end{bmatrix}, \qquad \mathcal{T} = \begin{bmatrix} \mathcal{T}_{\mathfrak{C}} \\ \mathcal{T}_{\mathfrak{S}} \\ \mathcal{T}_{\mathfrak{P}} \end{bmatrix} \tag{36}$$

For the $\mathfrak{T}_{\mathfrak{S}}$ aggregated tree, the $\mathcal{V}_{\mathfrak{S}}$, $\dot{\boldsymbol{\theta}}_{\mathfrak{S}}$, etc., subvectors of the $\mathcal{V}$ and $\dot{\boldsymbol{\theta}}$ stacked vectors correspond to the single $\mathfrak{S}$ aggregate link. This partitioning of the system-level $\dot{\boldsymbol{\theta}}$ and $\mathcal{V}$ stacked vectors extends to other stacked vectors such as the Coriolis spatial accelerations vector $\mathfrak{a}$, the gyroscopic spatial forces vector $\mathfrak{b}$, and to the other spatial operators.

Lemma 9 shows that the transformed system with the aggregated tree possesses a well-defined SKO model, and defines the equations of motion for the model.

Lemma 9 (SKO model for an aggregated tree) *Let $(H, \mathbb{A}, \boldsymbol{M})$ denote an SKO model with tree digraph $\mathfrak{T}$. Let $\mathfrak{S}$ be a subgraph of $\mathfrak{T}$ satisfying the aggregation condition. Then the $(H_a, \mathbb{A}_a, \boldsymbol{M})$ spatial operators define an SKO model for the $\mathfrak{T}_{\mathfrak{S}}$ aggregated tree, where*

$$H_a \triangleq H \mathfrak{J}_a^{-1} \overset{(35)}{=} \begin{pmatrix} H_{\mathfrak{C}} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \underline{H}_{\mathfrak{S}} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & H_{\mathfrak{P}} \end{pmatrix} \quad \textit{and} \tag{37}$$
$$\underline{H}_{\mathfrak{S}} \triangleq H_{\mathfrak{S}} \mathbb{A}_{\mathfrak{S}}$$

H_a defines the new joint map matrix for the $\mathfrak{T}_{\mathfrak{S}}$ aggregated tree, with $H_{\mathfrak{C}}$, $H_{\mathfrak{S}}$, and $H_{\mathfrak{P}}$ denoting the component joint map matrices as defined in (25). The transformed version of the equations of motion from (18) are given by:

$$\begin{aligned} &\mathcal{V} = \mathbb{A}_a^* H_a^* \dot{\boldsymbol{\theta}} \\ &\alpha = \mathbb{A}_a^* (H_a^* \ddot{\boldsymbol{\theta}} + \underline{\mathfrak{a}}), \\ &\qquad \text{where } \underline{\mathfrak{a}} \triangleq \mathfrak{J}_a^{-*} \mathfrak{a} = \begin{bmatrix} \mathfrak{a}_{\mathfrak{C}} \\ \underline{\mathfrak{a}}_{\mathfrak{S}} \\ \mathfrak{a}_{\mathfrak{P}} \end{bmatrix}, \ \underline{\mathfrak{a}}_{\mathfrak{S}} \triangleq \mathbb{A}_{\mathfrak{S}}^* \mathfrak{a}_{\mathfrak{S}} \\ &\underline{\mathfrak{f}} = \mathbb{A}_a (\boldsymbol{M} \alpha + \mathfrak{b}), \\ &\qquad \text{where } \underline{\mathfrak{f}} \triangleq \mathfrak{J}_a \mathfrak{f} = \begin{bmatrix} \mathfrak{f}_{\mathfrak{C}} \\ \mathfrak{f}'_{\mathfrak{S}} \\ \mathfrak{f}_{\mathfrak{P}} \end{bmatrix}, \ \mathfrak{f}'_{\mathfrak{S}} \triangleq \mathbb{A}_{\mathfrak{S}}^{-1} \mathfrak{f}_{\mathfrak{S}} \\ &\mathcal{T} = H_a \underline{\mathfrak{f}} \end{aligned} \tag{38}$$

Thus,

$$\mathcal{T} = \mathcal{M} \ddot{\boldsymbol{\theta}} + \mathcal{C} \tag{39}$$

with

$$\mathcal{M} = H_a \mathbb{A}_a \boldsymbol{M} \mathbb{A}_a^* H_a^* \in \mathcal{R}^{\mathcal{N}\times\mathcal{N}} \quad \textit{and}$$
$$\mathcal{C} \triangleq H_a \mathbb{A}_a \left(\boldsymbol{M} \mathbb{A}_a^* \underline{\mathfrak{a}} + \mathfrak{b}\right) \in \mathcal{R}^{\mathcal{N}} \tag{40}$$

The mass matrix expression above represents the Newton–Euler operator factorization of the mass matrix for the SKO model of the aggregated system.

Proof Lemma 8 established that $\mathbb{A}_a$ is the SPO operator for $\mathfrak{T}_{\mathfrak{S}}$. With the central row and column of H_a corresponding to node $\mathfrak{S}$ in $\mathfrak{T}_{\mathfrak{S}}$, H_a is block-diagonal for $\mathfrak{T}_{\mathfrak{S}}$. Lastly, $\boldsymbol{M}$ has remained unchanged in going from $\mathfrak{T}$ to $\mathfrak{T}_{\mathfrak{S}}$. Thus, the structural requirements 1 through 4 for an SKO model in Sect. 2.4, are satisfied for the $\mathfrak{T}_{\mathfrak{S}}$ tree, by the $(H_a, \mathbb{A}_a, \boldsymbol{M})$ operators.

The equations of motion in (38) follow by substituting the expressions for H_a and $\mathbb{A}_a$ from (35) and (37) in the original equations of motion in (18). □

Observe that, while $H_{\mathfrak{S}}$ is block-diagonal, $\underline{H}_{\mathfrak{S}} = H_{\mathfrak{S}} \mathbb{A}_{\mathfrak{S}}$ is no longer block-diagonal. Consequently, H_a is not block-diagonal for $\mathfrak{T}$ and the $(H_a, \mathbb{A}_a, \boldsymbol{M})$ operators do *not* satisfy the structural requirements for an SKO model for the original $\mathfrak{T}$. Nevertheless, the above lemma shows that these operators *do* satisfy the SKO model structural requirements for the $\mathfrak{T}_{\mathfrak{S}}$ tree. This implies that all of the SKO model formulation techniques and algorithms, including mass matrix factorization and inversion, are applicable to the SKO model of the aggregated tree.

The following corollary formally verifies that at the system-level, the mass matrix and Coriolis terms in the equations of motion are identical across the original and the aggregated systems. This of course is to expected, since the aggregation process only changes the form of the equations of motion and not the system dynamics.

Corollary 6.1 (Mass matrix invariance with aggregation) *The mass matrix*, $\mathcal{M}$, *and the Coriolis vector*, $\mathcal{C}$, *of a tree-topology system remain unchanged after subgraph aggregation. In other words*, *the expressions for* $\mathcal{M}$ *and* $\mathcal{C}$ *in* (40) *agree with the quantities defined in* (20), *i.e.*,

$$\mathcal{M} = H\mathbb{A}\boldsymbol{M}\mathbb{A}^* H^* = H_a \mathbb{A}_a \boldsymbol{M} \mathbb{A}_a^* H_a^*$$
$$\mathcal{C} = H\mathbb{A}(\boldsymbol{M}\mathbb{A}^* \mathfrak{a} + \mathfrak{b}) = H_a \mathbb{A}_a \left(\boldsymbol{M} \mathbb{A}_a^* \underline{\mathfrak{a}} + \mathfrak{b}\right) \tag{41}$$

Proof First, we have

$$H\mathbb{A} \overset{(35),(37)}{=} \left(H_a \mathfrak{J}_a^{-1}\right)(\mathfrak{J}_a \mathbb{A}_a) = H_a \mathbb{A}_a \tag{42}$$

Using this directly establishes the $\mathcal{M}$ equalities in (41).

Moreover,

$$\begin{aligned} H\mathbb{A}(\boldsymbol{M}\mathbb{A}^*\mathfrak{a} + \mathfrak{b}) &\overset{(42)}{=} H_a \mathbb{A}_a (\boldsymbol{M}\mathbb{A}^*\mathfrak{a} + \mathfrak{b}) \\ &\overset{(35)}{=} H_a \mathbb{A}_a \left(\boldsymbol{M}\mathbb{A}_a^* \mathfrak{J}_a^{-*}\mathfrak{a} + \mathfrak{b}\right) \\ &\overset{(38)}{=} H_a \mathbb{A}_a (\boldsymbol{M}\mathbb{A}_a^* \underline{\mathfrak{a}} + \mathfrak{b}) \end{aligned}$$

This establishes the $\mathcal{C}$ equalities in (41). □

The aggregation process provides a way to apply substructuring to develop alternative SKO models for a multibody system.

7 SKO model sparsity structure

Researchers have used system-level matrices and operators to analyze and exploit the sparsity structure of the mass matrix to develop efficient computational algorithms for the inverse and forward dynamics problems [3, 7, 21, 23]. In this section, we apply the partitioning and aggregation techniques developed to identify the sparsity structure of the SKO and SPO operators and the mass matrix for tree-topology systems.

Tree and serial-chain subgraphs are always path-induced and satisfy the aggregation condition. Thus any tree system can be successively partitioned using such subgraphs. However, only in serial-chains are all node pairs *related* resulting in the most dense SKO and SPO operators. Thus, serial-chain subgraphs are well suited for analyzing the sparsity structure of SKO models. Toward this, the system digraph can be decomposed into disjoint serial-chain branch segments, which are subsequently aggregated to obtain a coarser tree model for the system. Figure 6 illustrates such a transformation of a tree with five serial-chain segments into a coarser tree with all the serial-chain segments aggregated into nodes.

The sparsity structure of the system's spatial operators (e.g., SPO, mass matrix) directly reflects the structure of the aggregated tree. Subblocks of these matrices are associated with the aggregated serial-chain segment nodes, and are the most dense because all the nodes in a serial-chain are *related*. Equation (43) illustrates the sparsity structure of the system-level $\mathcal{E}_{\mathbb{A}}$

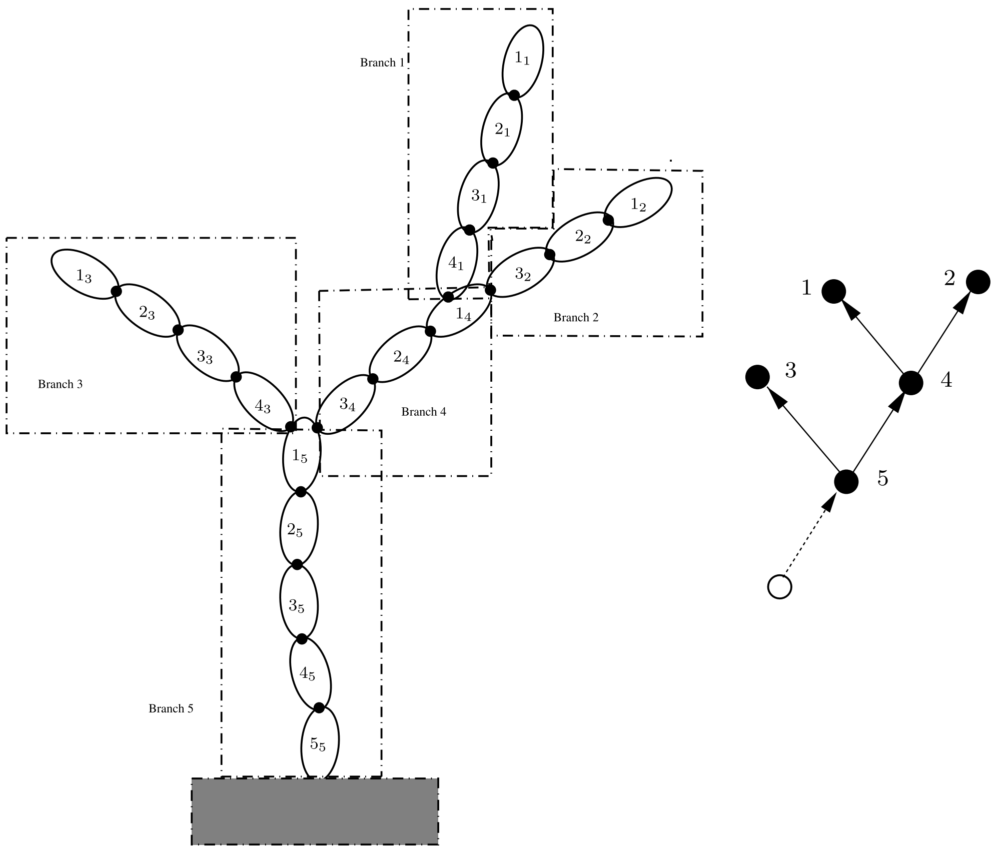


Fig. 6 Illustration of a tree-topology system on the left transformed into the coarser tree on the right by aggregating the serial-chain segments

SKO operator via its decomposition in terms of the SKO operators for each of the aggregated serial-chain nodes

$$\mathcal{E}_{\mathbb{A}} = \begin{pmatrix} \mathcal{E}_{\mathbb{A}_1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathcal{E}_{\mathbb{A}_2} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathcal{E}_{\mathbb{A}_3} & \mathbf{0} & \mathbf{0} \\ \mathcal{E}_{\mathbb{A}_{4,1}} & \mathcal{E}_{\mathbb{A}_{4,2}} & \mathbf{0} & \mathcal{E}_{\mathbb{A}_4} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathcal{E}_{\mathbb{A}_{5,3}} & \mathcal{E}_{\mathbb{A}_{5,4}} & \mathcal{E}_{\mathbb{A}_5} \end{pmatrix}. \tag{43}$$

$\mathcal{E}_{\phi_j}$ denotes the SKO operator for the jth branch segment. Also, when the kth branch is a child of the jth branch, the $\mathcal{E}_{\mathbb{A}_{j,k}}$ block denotes the non-zero connector block between them. All other blocks are zero. The structure of $\mathcal{E}_{\mathbb{A}_{j,k}}$ is as follows:

$$\mathcal{E}_{\mathbb{A}_{j,k}} \stackrel{\triangle}{=} \begin{pmatrix} \mathbf{0} & \cdots & \mathbf{0} & \mathbb{A}(1_j, n_k) \\ \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} \\ \vdots & \cdots & \vdots & \vdots \\ \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} \end{pmatrix} \tag{44}$$

where 1_j denotes the tip body on the jth branch that is the parent of the n_k base-body of the kth branch. This partitioned structure of $\mathcal{E}_{\mathbb{A}}$ is a generalization of the partitioned structure in (22). We see that the only nonzero blocks in the lower-triangular part of the $\mathcal{E}_{\mathbb{A}}$ matrix in (43) are for the *adjacent* nodes in the aggregated tree.

With $\phi_j \stackrel{\triangle}{=} (\mathbf{I} - \mathcal{E}_{\phi_j})^{-1}$ denoting the SPO operator for the jth branch, the overall structure of the system level $\mathbb{A}$ SPO operator for Fig. 6 tree is illustrated in (45):

$$\mathbb{A} = \begin{pmatrix} \mathbb{A}_1 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbb{A}_2 & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbb{A}_3 & \mathbf{0} & \mathbf{0} \\ \mathbb{A}_{4,1} & \mathbb{A}_{4,2} & \mathbf{0} & \mathbb{A}_4 & \mathbf{0} \\ \mathbb{A}_{5,1} & \mathbb{A}_{5,2} & \mathbb{A}_{5,3} & \mathbb{A}_{5,4} & \mathbb{A}_5 \end{pmatrix} \tag{45}$$

The $\mathbb{A}_{j,k}$ blocks denote the nonzero connector block between *related* serial-chain segments. All other

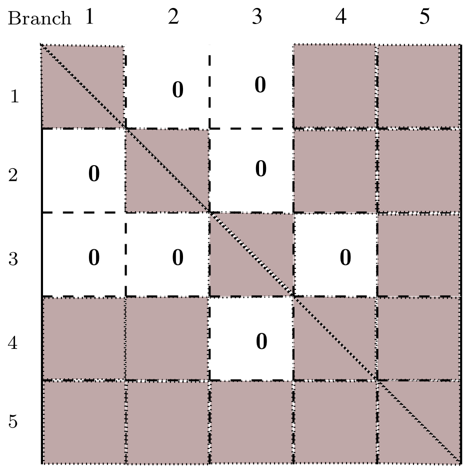


Fig. 7 Structure of the mass matrix $\mathcal{M}$ for the tree topology system in Fig. 6

blocks are zero. This partitioned structure of $\mathcal{E}_{\mathbb{A}}$ is a generalization of the partitioned structure in (24). The general expression for the $\mathbb{A}_{j,k}$ elements is:

$$\mathbb{A}_{j,k} = \mathbb{A}_j \mathcal{E}_{\mathbb{A}_{j,i}} \mathbb{A}_{i,k} \quad \text{with } i \in \mathbb{C}(j),\ i \succeq k$$

In the above, $\mathbb{A}_{k,k} \equiv \mathbb{A}_k$. We see that the only nonzero blocks in the lower-triangular part of the $\mathbb{A}$ matrix in (45) are for the *related* nodes in the aggregated tree.

Figure 7 illustrates the sparsity structure of the mass matrix for the tree-topology system in Fig. 6. As expected, the sparsity structure of the mass matrix mirrors the sparsity of the $\mathbb{A}$ SPO operator and its transpose. The block-diagonal of the mass matrix contains dense blocks, one for each of the branch segments in the tree. The off-diagonal blocks are zero for *unrelated* nodes in the aggregated tree. The topology dependency of the mass matrix's sparsity structure was initially described in [22].

8 Conclusions

SKO models play a pivotal role in the analysis and algorithm development for tree multibody systems. In this paper, we have studied the effect of topological transformations on such SKO models. Path-induced subgraphs are shown to partition tree digraphs into parent and child path-induced subgraphs. We derive the relationship between the SKO models of the original multibody system and the SKO models of the partitioned subsystems. We further show that SKO models can be substructured into coarser SKO models by aggregating path-induced subgraphs into aggregated nodes. We show that the path-induced property for the aggregated subgraph is a necessary and sufficient condition for preserving the tree structure of the aggregated system and derive the SKO model for the aggregated system. Lastly, we use these topological insights to understand the relationship between the topological structure of the system digraph and the sparsity of the system SKO and SPO operators and its mass matrix. In the companion second part, we use these aggregation methods to develop a constraint embedding technique that extends the notion of SKO models from tree to closed-topology systems.

We anticipate future applications of these substructuring ideas in the development of parallel algorithms for distributing computations across component SKO models. Featherstone's *divide and conquer* (*DCA*) forward algorithm [6] exemplifies such a use of a hierarchy of substructured component models to parallelize the solution for the forward dynamics problem. We also expect that partitioning and substructuring ideas to be useful in managing the dynamics computations for humanoid and legged robotic platforms subject to varying topologies and constraints. Furthermore, we anticipate the sparsity structure insights to facilitate the further development of sparsity based computational algorithms.

Acknowledgements The research described in this paper was performed at the Jet Propulsion Laboratory (JPL), California Institute of Technology, under contract with the National Aeronautics and Space Administration.[7] This project was also supported in part by Grant Number RO1GM082896-01A2 from the National Institute of Health.

References

1. Aghili, F.: Simplified Lagrangian mechanical systems with constraints using square-root factorization. In: Multibody Dynamics 2007 ECCOMAS Thematic Conference (2007),
2. Aghili, F.: A gauge-invariant formulation for constrained robotic systems using square-root factorization and unitary transformation. In: IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, pp. 2814–2821 (2008)
3. Baraff, D.: Linear-time dynamics using Lagrange multipliers. In: Proc. ACM SIGGRAPH '96, New Orleans, LA, pp. 137–146 (1996)

[7]©2010 California Institute of Technology. Government sponsorship acknowledged.

4. Cherowitzo, W.: Graph and digraph glossary. Website, http://www-math.cudenver.edu/wcherowi/courses/m4408/glossary.htm/ (2009)
5. Critchley, R., Anderson, K.: A generalized recursive coordinate reduction method for multibody dynamic systems. Int. J. Multiscale Comput. Eng. **12**(2), 181–200 (2003)
6. Featherstone, R.: A divide-and-conquer articulated-body algorithm for parallel $O(\log(n))$ calculation of rigid-body dynamics. Part 1: Basic algorithm. Int. J. Robot. Res. **18**(9), 867–875 (1999)
7. Featherstone, R.: Rigid Body Dynamics Algorithms. Springer, Berlin (2008)
8. Featherstone, R., Orin, D.: Robot dynamics: equations and algorithms. In: IEEE International Conference on Robotics and Automation, San Francisco, CA, pp. 826–834 (2000)
9. Herman, P., Kozlowski, K.: A survey of equations of motion in terms of inertial quasi-velocities for serial manipulators. Arch. Appl. Mech. **76**, 579–614 (2006)
10. Hu, W., Marhefka, D.W., Orin, D.E.: Hybrid kinematic and dynamic simulation of running machines. IEEE Trans. Robot. Autom. **21**(3), 490–497 (2010)
11. Jain, A.: Unified formulation of dynamics for serial rigid multibody systems. J. Guid. Control Dyn. **14**(3), 531–542 (1991)
12. Jain, A.: Graph techniques for multibody partitioning, aggregation and constraint embedding: Part II—Closed-chain constraint embedding. In review (2010). doi:10.1007/s11071-011-0136-x
13. Jain, A.: Graph theoretic structure of multibody system spatial operators. In: The 1st Joint International Conference on Multibody System Dynamics, Lappeenranta, Finland, (2010)
14. Jain, A.: Graph theory roots of spatial operators for kinematics and dynamics. In: 2010 IEEE International Conference on Robotics and Automation, Anchorage, AK (2010)
15. Jain, A.: Robot and Multibody Dynamics: Analysis and Algorithms. Springer, New York (2010)
16. Jain, A.: Graph theoretic foundations of multibody dynamics: Part I—Structural properties. Multibody Syst. Dyn., doi:10.1007/s11044-011-9266-7 (2011)
17. Jain, A.: Graph theoretic foundations of multibody dynamics: Part II—Analysis and algorithms. Multibody Syst. Dyn., doi:10.1007/s11044-011-9267-6 (2011)
18. Jain, A., Rodriguez, G.: Diagonalized Lagrangian robot dynamics. IEEE Trans. Robot. Autom. **11**(4), 571–584 (1995)
19. Junkins, J.L., Schaub, H.: An instantaneous eigenstructure quasivelocity formulation for nonlinear multibody dynamics. J. Astronaut. Sci. **45**(3), 279–295 (1997)
20. McPhee, J.J., Ishac, M.G., Andrews, G.C.: Wittenburg's formulation of multibody dynamics equations from a graph-theoretic perspective. Mech. Mach. Theory **31**(2), 201–213 (1996)
21. Orlandea, N., Chace, M., Calahan, D.: A sparsity-oriented approach to the dynamic analysis and design of mechanical systems—Part 1. ASME J. Eng. Ind. **99**(2), 773–779 (1977)
22. Rodriguez, G., Jain, A., Kreutz-Delgado, K.: Spatial operator algebra for multibody system dynamics. J. Astronaut. Sci. **40**(1), 27–50 (1992)
23. Saha, S.K.: A decomposition of the manipulator inertia matrix. IEEE Trans. Robot. Autom. **13**(2), 301–304 (1997)
24. Shi, P., McPhee, J.J.: Dynamics of flexible multibody systems using virtual work and linear graph theory. Multibody Syst. Dyn. **4**, 355–381 (2000)
25. Siciliano, B., Khatib, O.: Springer Handbook of Robotics. Springer, Berlin (2008)
26. Sinclair, A.J., Hurtado, J.E., Junkins, J.L.: Linear feedback control using quasi velocities. J. Guid. Control Dyn. **29**(6), 1309–1314 (2006)
27. West, D.B.: Introduction to Graph Theory. Prentice Hall, New York (2001)
28. Wittenburg, J.: Dynamics of Systems of Rigid Bodies. Teubner, Stuttgart (1977)
29. Yamane, K., Nakamura, Y.: Dynamics computation of structure-varying kinematic chains and its application to human figures. IEEE Trans. Robot. Autom. **16**(2), 124–134 (2000)
30. Zanganeh, K., Angeles, J.: A formalism for the analysis and design of modular kinematic structures. Int. J. Robot. Res. **17**(7), 720–730 (1998)