Transactions of Mathematics and Its Applications (2020) **00**, 1–36
doi:10.1093/imatrm/tnaa003

The Pantelides algorithm for delay differential-algebraic equations

Ines Ahrens* and Benjamin Unger
Institut für Mathematik, Technische Universität Berlin, Str. des 17. Juni 136, 10623 Berlin, Federal Republic of Germany
*Corresponding author: ahrens@math.tu-berlin.de

[Received on 8 August 2019; revised on 24 March 2020; accepted on 8 June 2020]

We present a graph-theoretical approach that can detect which equations of a delay differential-algebraic equation (DDAE) need to be differentiated or shifted to construct a solution of the DDAE. Our approach exploits the observation that differentiation and shifting are very similar from a structural point of view, which allows us to generalize the Pantelides algorithm for differential-algebraic equations to the DDAE setting. The primary tool for the extension is the introduction of equivalence classes in the graph of the DDAE, which also allows us to derive a necessary and sufficient criterion for the termination of the new algorithm.

Keywords: delay differential-algebraic equation, structural analysis, Pantelides algorithm.

1. Introduction

We study *delay differential-algebraic equations* (DDAEs) of the form

$$F(t, x(t), \dot{x}(t), x(t-\tau)) = 0 \tag{1.1}$$

on the time interval $[0, T)$ with $\tau > 0$. The DDAE (1.1) is equipped with an (functional) initial condition of the form

$$x(t) = \varphi(t) \qquad \text{for } t \in [-\tau, 0].$$

Differential equations with delay arise in various applications, where the current rate of change does not only depend on the current state but also on past information, for instance, population dynamics, infection disease models and laser dynamics. We refer to Erneux (2009) and the references therein. Besides, if a dynamical system is controlled based on the current state of the system—a so-called feedback control law—then such a controller often requires some time to measure the current state and to compute the control action, which again results in a delay equation. On the other hand, modern modeling packages such as Modelica[1] or Matlab/Simulink[2] automatically generate dynamical systems with constraints that arise due to interface conditions or conservation laws. As a consequence we have to deal with equations of the form (1.1), which combine the features of *delay differential equations* (DDEs) (Bellman & Cooke, 1963; Hale & Verduyn Lunel, 1993; Gluesing-Luerssen, 2002; Bellen & Zennaro, 2003) and *differential-algebraic equations* (DAEs) (Petzold, 1982; Kunkel & Mehrmann, 2006). Further

[1] https://www.modelica.org/
[2] https://www.mathworks.com/

applications of DDAEs are the analysis of hybrid numerical-experimental models arising in real-time dynamic substructuring (Unger, 2020), blood flow models (Borsche *et al.*, 2019), analysis (Altmann *et al.*, 2019) and construction (Bellen *et al.*, 1999, 2000) of numerical time integration schemes and the realization of transport phenomenon (Schulze & Unger, 2016; Schulze *et al.*, 2018; Fosong *et al.*, 2019). Let us emphasize that our framework is not restricted to a single time delay, since multiple commensurate delays can be rewritten in the form (1.1) by introducing new variables (Ha, 2015).

One of the main difficulties for DDAEs is the fact that solutions may depend on derivatives of the function F in (1.1) as well as on evaluations of F at future time points (Campbell, 1995; Ha & Mehrmann, 2012), i.e., one has to study the interplay of the differentiation operator $\mathrm{d}/\mathrm{d}t$ and the (time) shift operator Δ_τ (Ha *et al.*, 2014), defined via $(\Delta_\tau x)(t) = x(t+\tau)$. In particular it is important to understand which equations in (1.1) need to be differentiated and which need to be shifted. For linear systems this is investigated for instance in Campbell (1995); Ha & Mehrmann (2012, 2016); Ha *et al.* (2014); Unger (2018); Trenn & Unger (2019). Note that already in the linear case, existence and uniqueness results for DDAEs require a distributional solution concept (Trenn & Unger, 2019). To obtain continuous solutions one either has to impose restrictions on the DDAEs (Ha & Mehrmann, 2012, 2016; Unger, 2018) or on the history function (Ha, 2018; Unger, 2018). For nonlinear DDAEs solutions are establish for certain classed in Ascher & Petzold (1995); Unger (2020).

The main idea to establish solutions of (1.1), as described in the literature above, is via integration on successive time intervals $[i\tau, (i+1)\tau)$, the so-called *method of steps* (Campbell, 1980; Bellen & Zennaro, 2003; Ha & Mehrmann, 2016). At each interval one thus has to solve the DAE that is obtained by substituting the delayed variable in (1.1) with the already computed solution. A necessary condition for this procedure to succeed is that the DAE is regular. Unfortunately, this is not necessarily satisfied even if the initial value problem for (1.1) is uniquely solvable (Ha & Mehrmann, 2012). Instead, we seek a reformulation of (1.1) such that the DAE that has to be solved in each interval is regular and has small index. In the literature such a reformulation is constructed either with a compress-and-shift algorithm (Campbell, 1995; Trenn & Unger, 2019) or via a combined shift and derivative array (Ha & Mehrmann, 2016). In both cases one has to shift certain equations and differentiate certain equations.

In this paper we propose a graph-theoretical approach to determine which equations need to be differentiated and which equations need to be shifted. To this end we revisit the *Pantelides* algorithm (Pantelides, 1988) for DAEs in Section 2, a methodology that uses the information which variable appears in which equation to determine which equations need to be differentiated. Our main contributions are the following:

1. We introduce equivalence classes (cf. Definition 2.1) in the graph of an equation, which allows us to derive a criterion—see Theorem 2.8—when the Pantelides algorithm terminates. We show that this criterion is equivalent to the criterion presented in (Pantelides, 1988) and thus provides another interpretation that enables us to extend the algorithm to the DDAE case.

2. In Section 3.1 we illustrate that, from a structural point of view, the operations differentiation and shifting are similar, such that we can essentially use the same procedure to determine the equations that need to be shifted or differentiated, respectively.

3. We introduce a shifting and differentiation graph in Definition 3.5 and show that differentiation does not affect the shifting graph (Proposition 3.8). As a consequence we follow the strategy in Campbell (1995); Trenn & Unger (2019) and first determine the equations that need to be shifted. It turns out that during this procedure some of the equations need to be differentiated and we propose linear integer programs to determine which equations need to be differentiated how

many times. Theorem 3.13 details that each integer program is equivalent to a standard linear program that can be solved with standard methods.

4. Our methodology is summarized in Algorithm 5. We conclude this manuscript with a detailed analysis of Algorithm 5 in Section 5: we show in Theorem 5.1 that Algorithm 5 terminates if and only if the DDAE is structurally nonsingular with respect to a certain equivalence relation. Moreover, Theorem 5.4 details in which cases Algorithm 5 concludes that no equations need to be shifted and that in this situation it produces the same result as if we apply the original Pantelides algorithm to the DDAE (1.1) where we replace $\Delta_{-\tau}x$ with a function parameter λ.

We are convinced that our methodology can be combined with a dummy derivative approach (Mattsson & Söderlind, 1993) or with algebraic regularization techniques as proposed in Scholz & Steinbrecher (2016a,b), and thus serves as the first step to a numerical method for general nonlinear DDAEs. We emphasize that—similar to the original Pantelides algorithm (Pantelides, 1988)—our method not always determines the correct number of differentiations and shifts. For DAEs this fact is accounted for with a success check (Pryce, 2001; Pryce *et al.*, 2015) and an algebraic analysis (Scholz & Steinbrecher, 2016a). A similar validation of the results for DDAEs is ongoing research.

Notation

The natural numbers, the non-negative integers and the reals are denoted by $\mathbb{N}$, $\mathbb{N}_0$ and $\mathbb{R}$, respectively. For a set X the power set of X is denoted by $\mathscr{P}(X)$. For a differentiable function $f\colon \mathbb{I} \to \mathbb{R}^n$ we use the notation $\dot{f} := \frac{\mathrm{d}}{\mathrm{d}}f$ to denote the derivative from the right[3] with respect to the (time) variable t. Similarly, the shift operator Δ_τ is defined as $(\Delta_\tau f)(t) = f(t+\tau)$. As a consequence the DDAE (1.1) can be conveniently written as

$$F(t, x, \dot{x}, \Delta_{-\tau}x) = 0.$$

The partial derivative of F with respect to x, $\dot{x}$ and $\Delta_{-\tau}x$ is denoted by $\frac{\partial F}{\partial x}$, $\frac{\partial F}{\partial \dot{x}}$ and $\frac{\partial F}{\partial \Delta_{-\tau}x}$, respectively. The union of two sets A and B is denoted by $A \cup B$. If, moreover, the intersection $A \cap B$ of A and B is empty, then we write $A \dot{\cup} B$. The number of elements that are contained in the set A is denoted by $|A|$. For an equivalence relation $R \subseteq A \times A$ we denote the set of equivalence classes by A/R. For a subset $\widehat{A} \subseteq A$ we define $\widehat{A}/R := \widehat{A}/\widehat{R}$ with

$$\widehat{R} := \left\{(x, y) \in R \mid x, y \in \widehat{A}\right\}.$$

2. Preliminary results

A standard approach to solve differential equations with delay is the method of steps (Bellen & Zennaro, 2003). In the context of DDAEs this requires to solve a sequence of initial value problems with a DAE, see for instance Unger (2018). More precisely we study the DAE

$$F(t, x, \dot{x}) = 0, \tag{2.1a}$$

[3] For delay equations it is standard to use the derivative from the right (Hale & Verduyn Lunel, 1993; Campbell, 1995) since the history function may not be linked smoothly to the solution of the associated initial value problem.

where $F\colon \mathbb{I} \times \mathbb{D}_x \times \mathbb{D}_{\dot{x}} \to \mathbb{R}^n$, $\mathbb{I} \subseteq \mathbb{R}$ is an interval and $\mathbb{D}_x, \mathbb{D}_{\dot{x}} \subseteq \mathbb{R}^n$ are open, together with the initial condition

$$x(t_0) = x_0. \tag{2.1b}$$

One of the main differences of DAEs compared to *ordinary differential equations* (ODEs) is that the partial derivative of F with respect to $\dot{x}$, denoted via $\frac{\partial F}{\partial \dot{x}}$, may be singular. It is well known that in general solutions of (2.1) depend on derivatives of (2.1a), see for instance Petzold (1982); Brenan *et al.* (1996); Kunkel & Mehrmann (2006). Following Campbell (1987) we introduce the so-called *derivative array* of level ℓ

$$\mathscr{D}_\ell(t, x, \dot{x}, \ldots, x^{(\ell+1)}) := \begin{bmatrix} F(t,x,\dot{x}) \\ \frac{\mathrm{d}}{\mathrm{d}t}F(t,x,\dot{x}) \\ \vdots \\ \frac{\mathrm{d}^\ell}{\mathrm{d}t^\ell}F(t,x,\dot{x}) \end{bmatrix},$$

where $\frac{\mathrm{d}}{\mathrm{d}t}F(t,x,\dot{x}) := \frac{\partial F}{\partial t}(t,x,\dot{x}) + \frac{\partial F}{\partial x}(t,x,\dot{x})\dot{x} + \frac{\partial F}{\partial \dot{x}}(t,x,\dot{x})\ddot{x}$, where $\frac{\partial F}{\partial y}$ denotes the partial derivative of F with respect to $y \in \{t, x, \dot{x}\}$. The derivative array can be used to determine the solution of (2.1) for large enough ℓ. The *strangeness index* (Kunkel & Mehrmann, 1998, 2006) for instance uses $\mathscr{D}_\ell$ to determine matrix functions, which can be used to construct a reformulated version of (2.1a) that is suitable for numerical time integration. Although this procedure is tailored to numerical methods it suffers from two issues: first, in a large-scale setting, the computation of the matrix functions may become computationally expensive, since all equations in (2.1a) are differentiated instead of only the required equations. Second, the computation of the matrix functions requires several numerical challenging rank decisions.

To overcome these issues Scholz & Steinbrecher (2016a,b) propose to combine such algebraic regularization techniques with so-called structural analysis tools, such as the Σ-method (Pryce, 2001; Pryce *et al.*, 2015) and the Pantelides algorithm (Pantelides, 1988). Hereby, structure is understood as the information which variable appears in which equations. In other words the structure defines a bipartite graph $G = (V_\mathrm{E} \dot{\cup} V_\mathrm{V}, E)$ where the set of vertices $V_\mathrm{E} \dot{\cup} V_\mathrm{V}$ is given by the set of equations V_E and the set of variables V_V. An edge $\{F, x\} \in E$ implies that the variable $x \in V_\mathrm{V}$ appears in the equation $F \in V_\mathrm{E}$ (see Definition 2.1). Note that we need to distinguish between a variable and the name of the variable in the definition of the bipartite graph. We, therefore, introduce the language

$$L := \left\{ \Delta_{k\tau} \xi_j^{(i)} \mid \xi \in \{F, x\},\, j \in \mathbb{N},\, i \in \mathbb{N}_0,\, k \in \{-1\} \cup \mathbb{N}_0 \right\},$$

which accounts for differentiation and also for shifting, which is required for the upcoming analysis of the DDAE (1.1). For notational convenience we write ξ_j instead of $\xi_j^{(0)}$, $\dot{\xi}_j$ instead of $\xi_j^{(1)}$ and $\xi_j^{(i)}$ instead of $\Delta_{0\tau}\xi_j^{(i)}$. To map a vector of variable or equation names (as defined in L) to a set containing these elements we define

$$\mathrm{set}\colon L^N \to \mathscr{P}(L), \qquad (a_1, \ldots, a_N) \mapsto \{a_1, \ldots, a_N\}.$$

For the remainder of this paper we identify a variable by its name and do not distinguish between both.

The main idea of the Pantelides algorithm (Pantelides, 1988) is to answer the question, which equation determines which variable. More precisely we are interested in matching each equation with a *highest derivative*, i.e., a variable $x_i^{(k)}$ such that $x_i^{(k)}$ occurs in this equation but $x_i^{(k+\ell)}$ for $\ell > 0$ does not occur in any equation. For instance in an ODE of the form $\dot{x} = f(t, x)$ we can simply assign the ith equation to $\dot{x}_i$. The matching of an equation to a highest derivative is possible only if we can match each equation to a different variable regardless of the order of differentiation. Thus, it is desirable to group variables of different differentiation levels, which motivates the following definition.

DEFINITION 2.1 Consider the equation

$$F(t, \vartheta) = 0 \tag{2.2}$$

with $F\colon \mathbb{I} \times \mathbb{D}_\vartheta \to \mathbb{R}^M$ and $\mathbb{D}_\vartheta \subseteq \mathbb{R}^N$.

1. The set

$$\Theta := \{\theta \in \operatorname{set}(\vartheta) \mid \theta \text{ appears in } F\}$$

is called the *set of variables* of the equation (2.2).

2. Let Θ denote the set of variables of (2.2) and $\widetilde{\Theta} \subseteq \Theta$. The *graph* of (2.2) over the equivalence relation $R \subseteq \widetilde{\Theta} \times \widetilde{\Theta}$ is defined to be the bipartite graph $G = (V_\mathrm{E} \dot{\cup} V_\mathrm{V}, E)$, where $V_\mathrm{E} := \operatorname{set}(F)$ is the *set of equations*, $V_\mathrm{V} := \widetilde{\Theta}/R$ denotes the *set of variables with respect to* R and

$$E := \left\{\{V, w\} \;\middle|\; V \in V_\mathrm{V}, w \in V_\mathrm{E} : \text{there exists } v \in V \text{ which appears in } w\right\}. \tag{2.3}$$

3. Let Θ denote the set of variables for the DAE (2.1a) (with $\vartheta^T = \begin{bmatrix} x^T & \dot{x}^T \end{bmatrix}$) and

$$R := \{(\theta, \theta) \mid \theta \in \Theta\} \subseteq \Theta \times \Theta.$$

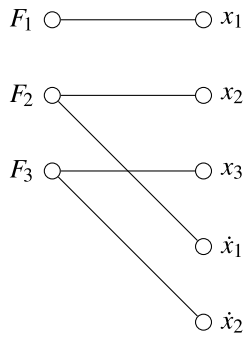
The bipartite graph $G = (V_\mathrm{E} \dot{\cup} V_\mathrm{V}, E)$ is called *graph of the DAE* (2.1a) if G is the graph of (2.1a) over R.

REMARK 2.2 Note that if a variable appears in an equation, that equation may not depend on that variable. For example, in the expression $\sin^2(x_1) + \cos^2(x_1)$ the variable x_1 appears but the expression does not depend on x_1 (Tan *et al.*, 2017). In general it is not possible to determine if an expression truly depends on a variable (Richardson, 1968).

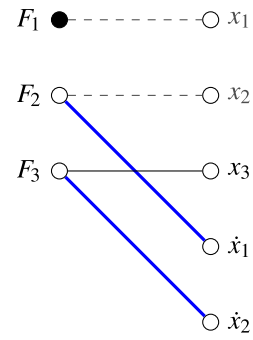
Each equivalence class in the set of variables with respect to R for a graph of a DAE contains one variable. Thus for DAEs, we can identify the set of variables and the set of variables with respect to R with each other. Although such an identification is of course also possible for DDAEs it turns out that nontrivial equivalence relation provide additional insights in the DDAE case (see the forthcoming Section 3). Nevertheless, we do not distinguish between a set of variables and a set of variables with respect to R since it is implied by the context.

EXAMPLE 2.3 In the DAE

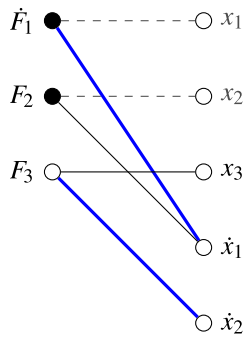
$$0 = x_1 + f_1, \qquad \dot{x}_1 = x_2 + f_2, \qquad \dot{x}_2 = x_3 + f_3 \tag{2.4}$$



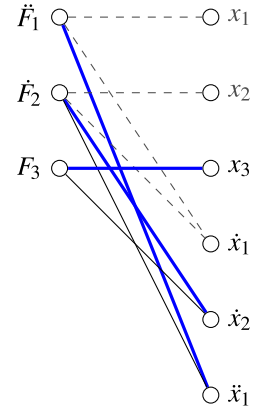
(a) Graph of the DAE



(b) Assignment after deleting x_1 and x_2



(c) Graph for (2.4) after differentiating F_1



(d) Graph for (2.4) after differentiating F_1 twice and F_2 once

FIG. 1. Visualization of Theorem 2.3.

we denote the ith equation by F_i for $i = 1, 2, 3$. The graph $G = (V_\mathrm{E} \dot{\cup} V_\mathrm{V}, E)$ of the DAE (2.4) given by

$$\begin{aligned} V_\mathrm{E} &:= \{F_1, F_2, F_3\} \subseteq \mathscr{P}(L), \\ V_\mathrm{V} &:= \{\{x_1\}, \{x_2\}, \{x_3\}, \{\dot{x}_1\}, \{\dot{x}_3\}\} \subseteq \mathscr{P}(L), \\ E &:= \{\{F_1, \{x_1\}\}, \{F_2, \{x_2\}\}, \{F_2, \{\dot{x}_1\}\}, \{F_3, \{x_3\}\}, \{F_3, \{\dot{x}_2\}\}\} \end{aligned}$$

is visualized in Fig. 1(a) (where we represent the equivalence classes by listing all its elements). By standard abuse of notation we do not distinguish between a graph and its visualization.

In the language of graph theory the assignment of an equation to a variable can be expressed as a *matching* $\mathscr{M} \subseteq E$ (in the literature also called *assignment*) in a graph (V, E), which is a subset of edges, such that for all vertices $v \in V$ with $v \in e_1 \in \mathscr{M}$ and $v \in e_2 \in \mathscr{M}$ we have $e_1 = e_2$, i.e., no vertex appears in more than one edge. An edge $e = \{v_1, v_2\} \in \mathscr{M}$ is called a *matching edge* and we say that v_1 and v_2 are *matched*. A vertex that does not occur in any matching edge is called an *exposed vertex* with

respect to the matching. In a bipartite graph $G = (V_E \dot\cup V_V, E)$ a *maximal matching* is a matching such that no $F \in V_E$ is exposed.

EXAMPLE 2.4 In the graph in Example 2.3 the variables x_1, x_2 are not highest derivatives such that we can (temporarily) delete these variables from the graph, which we visualize by using dashed gray edges in Fig. 1(b). A matching is for instance given by $\mathscr{M} = \{\{F_2, \{\dot x_1\}\}, \{F_3, \{\dot x_2\}\}\}$, which we depict in Fig. 1(b) via thick blue line. We immediately notice that F_1 is an exposed vertex (independent of any possible matching and indicated by a black vertex) and thus no maximal matching is possible.

REMARK 2.5 Let Θ denote the set of variables for the DAE (2.1a) (cf. Definition 2.1). Define

$$\widetilde\Theta := \{\theta \in \Theta \mid \theta \text{ is highest derivative}\} \tag{2.5}$$

and $\widetilde R_{\text{triv}} := \{(\theta, \theta) \mid \theta \in \widetilde\Theta\}$. Then the graph of the DAE that is obtained after deleting all variables that are not highest derivative is the graph of (2.1a) over $\widetilde R_{\text{triv}}$.

Suppose that, after deleting variables that are no highest derivatives, we cannot find a maximal matching and thus have an exposed vertex F_i. Then one of the following applies: the function F_i does not depend on any variable, implying that in general, the DAE is not regular (see Kunkel & Mehrmann, 1998 for further details). Otherwise, the function F_i determines a variable that is not a highest derivative, and thus F_i needs to be differentiated to construct a solution. More precisely we replace F_i by $\frac{\mathrm{d}}{\mathrm{d}t}F_i(t, \vartheta) = 0$ and add possible new variables. Note that all variables that appear in F_i appear with one additional derivative in $\frac{\mathrm{d}}{\mathrm{d}t}F_i(t, \vartheta) = 0$ and thus the original variables cannot be highest derivatives and can thus be ignored in the following.

EXAMPLE 2.6 Continuing with Example 2.4 we replace F_1 with $\frac{\mathrm{d}}{\mathrm{d}t}F_1(t, \vartheta) = 0$, which we denote by $\dot F_1$. Note that there is a (gray) edge in Fig. 1(c) between $\dot F_1$ and x_1, since in our structural approach we do not work with the actual equation but only with the information that F_1 depends on x_1. The chain rule then implies that $\dot F_1$ depends on x_1 and $\dot x_1$. A possible matching is given by $\mathscr{M} = \{\{\dot F_1, \{\dot x_1\}\}, \{F_3, \{\dot x_2\}\}\}$, such that F_2 is an exposed vertex with respect to $\mathscr{M}$. The resulting graph is depicted in Fig. 1. Continuing with our strategy for Theorem 2.6 implies that we need to differentiate equation F_2. This time the situation is slightly different than before, since there exists a coupling between $\dot F_1$ and F_2 via the variable $\dot x_1$ indicated by a black vertex, i.e., we have a path from $\dot F_1$ to F_2 via $\dot x_1$.

A *path* is a sequence of edges $(e_1, \ldots, e_n)$ in a graph $G = (V, E)$ such that $e_i = \{v_{i-1}, v_i\} \in E$, where $v_0, \ldots, v_n \in V$ are distinct. An *alternating path* with respect to a matching is a path with alternating non-matching and matching edges that starts with a non-matching edge. An *augmenting path* with respect to a matching is an alternating path that ends with a non-matching edge. It is easy to see that whenever an augmenting path exists we can find another matching that includes more equations than the previous matching and hence we cannot determine (at this point) that the equations need to be differentiated. Hence, we only have to differentiate whenever we cannot find an augmenting path. In this case we have to differentiate all equations that are connected with the exposed equation via an alternating path. For further details we refer to Pantelides (1988).

EXAMPLE 2.7 In Example 2.6 there exists no augmenting path that starts in F_2 (cf. Fig. 1c). Thus, we have to differentiate F_2 and all equations F_j with an alternating path from F_2 to F_j. In Fig. 1(c) these equations are denoted by a black dot. In this case the path $(F_2, \{\dot x_1\}, \dot F_1)$ implies that we have

to differentiate $\dot{F}_1$ and F_2. Note that in the resulting graph, given in Fig. 2.1(d), we find the maximal matching

$$\mathscr{M} = \left\{\{\ddot{F}_1, \{\ddot{x}_1\}\}, \{\dot{F}_2, \{\dot{x}_2\}\}, \{F_3, \{x_3\}\}\right\}$$

and can thus stop.

Before we present an algorithmic summary of the outlined methodology let us recall the main idea of the Pantelides algorithm, namely to match each equation in (2.1a) with a highest derivative. In more detail we have considered the highest derivative of each variable (for the corresponding equivalence relation see Remark 2.5) to determine which equations need to be differentiated. Let $\{k_1, \ldots, k_m\} \subseteq \{1, \ldots, M\}$ denote the equations that need to be differentiated. Then we define a *subset of equations* of (2.2) via

$$\widehat{F}(t, \widehat{\vartheta}) := \begin{bmatrix} F_{k_1}(t, \vartheta) \\ \vdots \\ F_{k_M}(t, \vartheta) \end{bmatrix} = 0, \tag{2.6}$$

with the understanding that $\widehat{\vartheta}$ contains only the subset of variables of ϑ that appears in any equation of $\widehat{F}$. It is clear that we cannot match each equation in (2.6) with a highest derivative if there are more equations than highest derivative variables, i.e., equivalence classes in $(\mathrm{set}(\hat{\vartheta}) \cap \tilde{\Theta})/R_{\mathrm{equal}}$ with $\tilde{\Theta}$ defined as in (2.5).

Definition 2.8 A *subset* (2.6) of (2.2) is called *structurally singular* with respect to an equivalence class $\widehat{\Theta}/\widehat{R}$ with subset $\widehat{\Theta} \subseteq \mathrm{set}(\widehat{\vartheta})$ and relation $\widehat{R}$ if

$$\left|\mathrm{set}(\widehat{F})\right| > \left|\widehat{\Theta}/\widehat{R}\right|.$$

The *equation* (2.2) is called *structurally singular* with respect to an equivalence class Θ/R with subset $\Theta \subseteq \mathrm{set}(\vartheta)$ if it contains a structurally singular subset (2.6) with respect to $(\Theta \cap \mathrm{set}(\widehat{\vartheta}))/R$.[4] A structurally singular subset (2.6) is called *minimal structural singular* (MSS) with respect to an equivalence class Θ/R if none of its proper subsets (2.6) is structurally singular with respect to $(\Theta \cap \mathrm{set}(\widehat{\vartheta}))/R$.

Remark 2.9 The notion of structural singularity of a subset of equations is defined in Pantelides (1988). In contrast to Pantelides (1988) we restrict ourselves to the set of variables that are actually contained within this subset of equations.

Proposition 2.10 Consider the equation (2.2), let Θ denote the set of variables of (2.2). For a subset $\tilde{\Theta} \subseteq \Theta$ with equivalence relation $R \subseteq \tilde{\Theta} \times \tilde{\Theta}$ consider the graph G of (2.2) over $\tilde{\Theta}/R$ with a matching $\mathscr{M}$. Let F_j be an exposed node. Then one of the following is true:

1. There exists an augmenting path starting in F_j.
2. The set of equations $\widehat{F}(t, \widehat{\vartheta}) = 0$ associated with

$$C_{F_j} := \left\{F_k \in V_{\mathrm{E}} \mid \text{there exists an alternating path between } F_j \text{ and } F_k \text{ in } G\right\} \tag{2.7}$$

is MSS with respect to $(\tilde{\Theta} \cap \mathrm{set}(\hat{\vartheta}))/R$.

[4] Recall that we use $(\Theta \cap \mathrm{set}(\widehat{\vartheta}))/R$ to denote the set of equivalence classes $(\Theta \cap \mathrm{set}(\widehat{\vartheta}))/\widehat{R}$ with the restricted equivalence relation $\widehat{R} := \{(x, y) \in R \mid x, y \in \Theta \cap \mathrm{set}(\widehat{\vartheta})\}$.

Proof. Since all equations in C_{F_j} have an alternating path to F_j we conclude that F_j is the only exposed equation in C_{F_j}. This implies

$$|C_{F_j}| - 1 \leq \left| (\tilde{\Theta} \cap \operatorname{set}(\widehat{\vartheta}))/R \right|. \tag{2.8}$$

Suppose that there is no augmenting path that starts in F_j. Assume that the inequality in (2.8) is strict. Then there exists at least one equivalence class in $(\tilde{\Theta} \cap \operatorname{set}(\widehat{\vartheta}))/R$ that is exposed. By definition of $(\tilde{\Theta} \cap \operatorname{set}(\widehat{\vartheta}))/R$ and C_{F_j} this implies that there exists an augmenting path that starts in F_j, a contradiction. We conclude that we have equality in (2.8), and thus C_{F_j} is structurally singular.

Consider a proper subset $\tilde{C}_{F_j} \subsetneq C_{F_j}$. If $F_j \notin \tilde{C}_{F_j}$ then all $F_k \in \tilde{C}_{F_j}$ are contained in a matching edge. Thus, the set is structurally nonsingular. If $F_j \in \tilde{C}_{F_j}$ there exists an edge between a variable $x_k \in (\tilde{\Theta} \cap \operatorname{set}(\widehat{\vartheta}))/R$ and an equation $F_k \in \tilde{C}_{F_j}$ which is not a matching edge. Thus, $\tilde{C}_{F_j}$ is structurally nonsingular. This completes the proof. □

Proposition 2.10 allows us to implement the outlined strategy as follows. Suppose that we have already found a matching for the first $j-1$ equations. For the jth equation we try to find an augmenting path. If there exists an augmenting path then we can directly generate a matching that includes the first j equations. If not, then we differentiate all equations within the set C_{F_j} defined in (2.7). Note that, to determine that there is no augmenting path, we have to check all alternating paths and hence the set C_{F_j} is automatically generated when we check for an augmenting path. One way to achieve this is via a recursive, depth-first search algorithm (Duff, 1981; West, 2001). The details are given in Algorithm 1. Here we use the notation $\mathtt{x_i}, \mathtt{F_j}$ on the one hand as vertices in V_{V} and V_{E} and on the other hand as indices referring to those vertices. The complete algorithm, which is known as the Pantelides algorithm (Pantelides, 1988), is presented in Algorithm 2.

Note that in contrast to the original algorithm (Pantelides, 1988) we replace equations that are differentiated and do not add the equations to the graph. All variables occurring in the replaced equation are deleted. Thus, the replaced equation has no edges and can be removed without any changes for the algorithm. This has the advantage that we do not need to keep track, which equation was obtained by differentiating another equation and similarly for the variables. We emphasize that we only replace equations in our algorithm. If we want to use the results for numerical time-integration methods then we can either construct a reformulation of the DAE that explicitly contains all algebraic constraints from a reduced derivative array (Steinbrecher, 2006), where only derivatives of equations are added, which are determined by the Pantelides algorithm, or we solve the overdetermined system that results from adding all equations that are differentiated to the original system (as for instance in the dummy derivative approach (Mattsson & Söderlind, 1993) or the least-squares approach (Steinbrecher, 2006)). Note that we use a similar notation in Algorithm 2 as in Algorithm 1 for vertices in V_{V} and V_{E}. The vertex $\mathtt{x_{i,k}}$ represents the variable $x_i^{(k)}$.

Since Algorithm 2 is an iterative algorithm we have to answer the question whether Algorithm 2 terminates. We immediately observe that if Algorithm 2 terminates then we have no exposed equation and thus each equation is matched with a highest derivative. We conclude that a necessary condition for termination of Algorithm 2 is that the DAE (2.1a) is structurally nonsingular with respect to Θ/R_{equal} with Θ denoting the set of variables of (2.1a) and R_{equal} the equivalence relation

$$R_{\text{equal}} := \left\{ (\theta, \tilde{\theta}) \in \Theta \times \Theta \;\middle|\; \exists \xi \in \operatorname{set}(x),\, p, q \in \mathbb{N}_0 \text{ such that } \theta = \xi^{(p)},\, \tilde{\theta} = \xi^{(q)} \right\}. \tag{2.9}$$

The following result shows that in fact the structural nonsingularity is also a sufficient condition.

Algorithm 1 Augmentpath

Input: bipartite graph $\mathtt{G} = (\mathtt{V_E} \dot\cup \mathtt{V_V}, \mathtt{E})$ with $\mathtt{V_E} = \{\mathtt{F_1}, \ldots, \mathtt{F_m}\}$ and $\mathtt{V_V} = \{\mathtt{x_1}, \ldots, \mathtt{x_n}\}$

$\mathtt{F_j} \in \mathtt{V_E}$ ▷ starting vertex

`assign` ▷ matching

`colorV`, `colorE` ▷ vertices needed to be differentiated if `pathfound == false`

Output: `pathfound`, `assign`, `colorV`, `colorE`

```
pathfound ← false
colorE(F_j) ← 1
for x_i ∈ V_V with {x_i, F_j} ∈ E and colorV(x_i) == 0 do
    colorV(x_i) ← 1
    F_k ← assign(x_i)
    if F_k == 0 then
        pathfound ← true
    else
        (pathfound, assign, colorV, colorE) ← Algorithm1(G, F_k, assign, colorV, colorE)
    end if
    if pathfound == true then
        assign(x_i) ← F_j
        return pathfound, assign, colorV, colorE
    end if
end for
```

THEOREM 2.11 Consider the DAE (2.1a) and let Θ denote its set of variables and R_{equal} the equivalence relation in (2.9). Then Algorithm 2 applied to the DAE (2.1a) terminates if and only if (2.1a) is structurally nonsingular with respect to Θ/R_{equal}.

Proof. We show that the structural nonsingularity is equivalent to the termination criterion that is presented in the original paper (Pantelides, 1988). To this end let us rewrite (2.1a) by splitting the variable x into $\text{set}(x) = \text{set}(y) \dot\cup \text{set}(z)$ such that

- for all $\xi \in \text{set}(y)$ it holds that $\dot\xi$ appears in F and
- for all $\xi \in \text{set}(z)$ it holds that $\dot\xi$ does not appears in F.

Using the new variables we can rewrite (2.1a) as

$$F(t, y, \dot y, z) = 0. \tag{2.10a}$$

We introduce a coupling between y and $\dot y$ by adding formal equations

$$G_i(y_i, \dot y_i) = 0 \qquad \text{for } i = 1, \ldots, n_y \tag{2.10b}$$

for $y = [y_i]_{i=1,\ldots,n_y}$ and $G = [G_i]_{i=1,\ldots,n_y}$. The system of equations (2.10) is called the *extended DAE*. In Pantelides (1988) it is shown that Algorithm 2 applied to the DAE (2.1a) terminates if and only if the extended DAE (2.10) is structurally nonsingular with respect to $\text{set}(y, \dot y, z)/R_{\text{triv}}$ with

$R_{\text{triv}} := \{(\xi, \xi) \mid \xi \in \text{set}(y, \dot{y}, z)\}$. It therefore suffices to show that structural nonsingularity of (2.10a) with respect to $\text{set}(y, \dot{y}, z)/R_{\text{equal}}$ is equivalent to structural nonsingularity of the extended DAE (2.10) with respect to $\text{set}(y, \dot{y}, z)/R_{\text{triv}}$.

Algorithm 2 Pantelides algorithm for DAEs

Input: graph $\mathtt{G} = (\mathtt{V_E} \dot{\cup} \mathtt{V_V}, \mathtt{E})$, $\mathtt{V_E} = \left\{\mathtt{F}_{\mathtt{j},\ell} \mid \mathtt{F}_{\mathtt{j}}^{(\ell)} \in \mathtt{V_E}\right\}$, $\mathtt{V_V} = \left\{\mathtt{x}_{\mathtt{i},\mathtt{k}} \mid \mathtt{x}_{\mathtt{i}}^{(\mathtt{k})} \in \mathtt{V_V}\right\}$

Output: graph $\mathtt{G} = (\mathtt{V_E} \dot{\cup} \mathtt{V_V}, \mathtt{E})$

```
assign(x_{i,k}) ← 0 for each x_{i,k} ∈ V_V
for r = 1, ..., M do
    (p, q) ← (r, 0)
    repeat
        Ẽ ← E \ {{x_{i,k}, F_{j,ℓ}} ∈ E | x_{i,k+s} ∈ V_V for some s > 0}
        Ṽ_V ← V_V \ {x_{i,k} | x_{i,k+s} ∈ V_V for some s > 0}
        colorV(x_{i,k}) ← 0 for each x_{i,k} ∈ Ṽ_V
        colorE(F_{j,ℓ}) ← 0 for each F_{j,ℓ} ∈ V_E
        (pathfound, assign, colorV, colorE) ← Algorithm1((V_E ∪̇ Ṽ_V, Ẽ), F_{p,q},
          assign, colorV, colorE)                          ▷ apply Augmentpath
        if pathfound == false then
            V_V ← V_V ∪ {x_{i,k+1} | colorV(x_{i,k}) == 1}
            for each (j, ℓ) s. t. colorE(F_{j,ℓ}) == 1 do
                F_{j,ℓ} ← F_{j,ℓ+1}
                E ← E ∪ {{x_{i,k+1}, F_{j,ℓ}} | {x_{i,k}, F_{j,ℓ}} ∈ E}
            end for
            for each i with colorV(x_{i,k}) == 1 and F_{j,ℓ} == assign(x_{i,k}) do
                assign(x_{i,k+1}) ← F_{j,ℓ}
            end for
        end if
    until pathfound == true
    pathfound ← false
end for
```

Assume first that the extended DAE (2.10) is structurally nonsingular with respect to $\text{set}(y, \dot{y}, z)/R_{\text{triv}}$. Consider an arbitrary subset of (2.10a)

$$\widehat{F}(t, \eta, \gamma, \dot{\gamma}, \dot{\mu}, \widehat{z}) = 0, \tag{2.11}$$

with $\text{set}(\eta, \gamma, \mu) \subseteq \text{set}(y)$ and $\text{set}(\widehat{z}) \subseteq \text{set}(z)$. Let $\widehat{G}(\gamma, \dot{\gamma}) = 0$ denote the subset of (2.10b) that corresponds to the variable γ. Since the extended DAE is structurally nonsingular we obtain

$$\left|\text{set}\left(\widehat{F}, \widehat{G}\right)\right| \leq \left|\text{set}\left(\eta, \gamma, \dot{\gamma}, \dot{\mu}, \widehat{z}\right)/R_{\text{triv}}\right|.$$

This is equivalent to

$$\left|\operatorname{set}\left(\widehat{F}\right)\right| \leq \left|\operatorname{set}\left(\eta, \gamma, \dot{\mu}, \widehat{z}\right)\right| = \left|\operatorname{set}(\eta, \gamma, \dot{\gamma}, \dot{\mu}, \widehat{z})/R_{\text{equal}}\right|$$

and thus (2.10a) is structurally nonsingular with respect to Θ/R_{equal}.

Conversely, suppose that the DAE (2.10a) is structurally nonsingular with respect to $\operatorname{set}(y, \dot{y}, z)/R_{\text{equal}}$. A subset of (2.10) is given by

$$\begin{aligned} \widehat{F}(t, \eta, \widehat{\eta}, \gamma, \dot{\gamma}, \widehat{\gamma}, \dot{\widehat{\gamma}}, \dot{\mu}, \dot{\widehat{\mu}}, \widehat{z}) &= 0, \\ \widehat{G}(\widehat{\eta}, \dot{\widehat{\eta}}, \widehat{\gamma}, \dot{\widehat{\gamma}}, \widehat{\mu}, \dot{\widehat{\mu}}, \widehat{\zeta}, \dot{\widehat{\zeta}}) &= 0, \end{aligned} \tag{2.12}$$

with $\operatorname{set}(\eta, \widehat{\eta}, \gamma, \widehat{\gamma}, \mu, \widehat{\mu}, \widehat{\zeta}) \subseteq \operatorname{set}(y)$ and $\operatorname{set}(\widehat{z}) \subseteq \operatorname{set}(z)$. Since the DAE (2.10a) is structurally nonsingular with respect to $\operatorname{set}(y, \dot{y}, z)/R_{\text{equal}}$ we obtain

$$\left|\operatorname{set}\left(\widehat{F}\right)\right| \leq \left|\operatorname{set}\left(\eta, \widehat{\eta}, \gamma, \dot{\gamma}, \widehat{\gamma}, \dot{\widehat{\gamma}}, \dot{\mu}, \dot{\widehat{\mu}}, \widehat{z}\right)/R_{\text{equal}}\right| = \left|\operatorname{set}\left(\eta, \widehat{\eta}, \gamma, \widehat{\gamma}, \dot{\mu}, \dot{\widehat{\mu}}, \widehat{z}\right)\right|,$$

which immediately implies

$$\left|\operatorname{set}\left(\widehat{F}, \widehat{G}\right)\right| \leq \left|\operatorname{set}\left(\eta, \widehat{\eta}, \dot{\widehat{\eta}}, \gamma, \widehat{\gamma}, \dot{\widehat{\gamma}}, \dot{\mu}, \widehat{\mu}, \dot{\widehat{\mu}}, \widehat{\zeta}, \dot{\widehat{\zeta}}, \widehat{z}\right)\right|.$$

We conclude that (2.12) is structurally nonsingular with respect to R_{triv}, which completes the proof. □

Even if Algorithm 2 terminates this is not a guarantee that the correct number of differentiations for each equation is determined. In fact, the Pantelides algorithm relies heavily on the so-called sparsity pattern of the DAE, i.e., each equation depends only on few variables. In particular Algorithm 2 is not invariant under bijective transformations of the DAE (cf. Scholz & Steinbrecher, 2016a, Remark 5).

EXAMPLE 2.12 The Pantelides algorithm applied to the DAE

$$\dot{x}_1 + \dot{x}_2 = x_2 + f_1, \qquad 0 = x_1 + x_2 + f_2$$

determines that the second equation needs to be differentiated one time. If we however add the first equation to the second equation then we can match the first equation with $\dot{x}_1$ and the second equation with $\dot{x}_2$ and thus neither of the equations is differentiated.

Also, it may be the case that the Pantelides algorithm concludes that equations need to be differentiated several times, although this is not necessary (Reissig *et al.*, 2000). One possible reason for this is that the DAE is close to a high-index DAE and thus the numerical solution may be more difficult to obtain than the (potentially) small index suggests (see Reissig *et al.*, 2000 for further details). In any case it is possible to check that the equations are differentiated sufficiently often by applying a success check (Pryce, 2001; Pryce *et al.*, 2015) and an algebraic criterion (Scholz & Steinbrecher, 2016a). Similar arguments apply to the relation between solvability and structural singularity.

EXAMPLE 2.13 Consider the DAE

$$\begin{bmatrix} 1 & a \\ a & a \end{bmatrix} \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 1 & a \\ a & a \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} f_1 \\ f_2 \end{bmatrix}.$$

For $a = 1$ the DAE is structurally nonsingular. However, solutions can only exist if $f_1 - f_2 \equiv 0$. On the other hand, if $a = 0$, then the DAE is structurally singular. Still, the DAE possesses a solution if $f_2 \equiv 0$. Note that small perturbations of the nonzero matrix coefficients and the inhomogeneities f_1 and f_2 do not affect the structural singularity but may change the solvability of the DAE.

3. The Pantelides algorithm for DDAEs

It is well known that for DDAEs of the form (1.1) it is not sufficient to differentiate equations to obtain solutions, but in addition, some equations need to be shifted (Campbell, 1995; Ha & Mehrmann, 2012, 2016; Trenn & Unger, 2019).

EXAMPLE 3.1 Consider the DDAE

$$\dot{x}_1 = f_1, \qquad 0 = x_1 - \Delta_{-\tau} x_2 + f_2. \tag{3.1}$$

If we want to construct a solution for the corresponding initial value problem with the method of steps (Bellen & Zennaro, 2003) then in the interval $[0, \tau)$ we have to solve the DAE

$$\dot{x}_1 = f_1, \quad 0 = x_1 + \tilde{f}_2 \tag{3.2}$$

with $\tilde{f}_2 = f_2 + \varphi_2(t - \tau)$. Clearly, this DAE is not regular (it does not depend on x_2 and the two equations might contradict each other). If we however add the first equation to the differentiated second equation and shift the resulting equation we obtain the DDAE

$$\dot{x}_1 = f_1, \qquad \dot{x}_2 = \Delta_\tau f_2 + \Delta_\tau \dot{f}_1$$

and it is easy to see that for any consistent initial value the DAE that we have to solve in the interval $[0, \tau)$ is uniquely solvable.

As a direct consequence a structural analysis for DDAEs should reveal which equations need to be shifted and which equations need to be differentiated. We thus have to understand the similarities and differences between differentiating and shifting from a structural point of view.

3.1 *Differentiation vs. Shifting*

To understand the effect of the two operators $\frac{\mathrm{d}}{\mathrm{d}t}$ and $\Delta_{-\tau}$ from a structural point of view we start with the following simple example.

EXAMPLE 3.2 Let $N \in \mathbb{R}^{n \times n}$ be a nilpotent matrix with $N^\nu = 0$ and $N^{\nu-1} \neq 0$ for some $\nu \in \mathbb{N}$ and consider the two equations

$$N \frac{\mathrm{d}}{\mathrm{d}t} x = x + f, \tag{3.3a}$$

$$Ny = \Delta_{-\tau} y + g, \tag{3.3b}$$

(a) Initial graph for (3.4) (b) Graph for (3.4) after shifting F_1

FIG. 2. Graphs obtained by applying Algorithm 2 to Example 3.3.

which can be understood as special cases of the Weierstraß canonical form (Gantmacher, 1959) for a regular matrix pencil. It is easy to see that the solutions are of the form

$$x = -\sum_{i=0}^{\nu-1} N^i \left(\frac{\mathrm{d}}{\mathrm{d}t}\right)^i f \qquad \text{and} \qquad y = -\sum_{i=0}^{\nu-1} N^i \left(\Delta_\tau\right)^i \Delta_\tau g. \tag{3.3c}$$

Apart from the operator applied to the inhomogeneity both solutions are the same. Thus, the variable $\dot{x}$ has in the DAE the same role as the variable y in the difference equation (3.3b). In other words, in the DAE, the variable with the highest derivative is $\dot{x}$, while in the delay equation, the variable $y = \Delta_0 y$ is the variable with the highest shift, in the sense that $0 > -1$.

Motivated by Example 3.2 we can use the Pantelides algorithm (Algorithm 2) for difference equations by replacing differentiation by shifting in Algorithm 2. In a difference equation we want to assign each equation to a different variable with *highest shift*, meaning that $\Delta_{k\tau} x$ with $k \geq 0$ occurs in some equation but $\Delta_{(k+\ell)\tau} x$ for $\ell > 0$ does not. Note that by our definition the variable $\Delta_{-\tau} x$ can never be a variable with *highest shift*, since this would imply that we can solve directly for this variable without shifting. This can also be seen in the solution formula (3.3c) for the difference equation, which requires an additional shift of the inhomogeneity g.

EXAMPLE 3.3 The graph of the scalar difference equation

$$0 = \Delta_{-\tau} x_1 + f_1 \tag{3.4}$$

is visualized in Fig. 2(a). By definition $\Delta_{-\tau} x_1$ is not a highest shift. Thus, the edge to $\Delta_{-\tau} x_1$ is gray and dashed. By replacing differentiating with shifting in Algorithm 2 we shift F_1 once (cf. Fig. 2(b)). Note that there is no edge between $\Delta_\tau F_1$ and $\Delta_{-\tau} x_1$ since there is no chain rule for the shift operator.

3.2 *The shifting and differentiation graph*

As outlined in Theorem 3.1 some equations in the DDAE (1.1) need to be shifted and some need to be differentiated. Motivated by the observation (cf. Section 3.1) that shifting and differentiating are quite similar from a structural point of view we construct two graphs: one to determine which equations need to be shifted and one to determine which equations need to be differentiated. In the shifting graph we do not want to match equations with delayed variables, since we cannot (directly) solve for these variables. More precisely we do not want to match equations with variables that are not highest shifts and thus delete these variables in the shifting graph. On the other hand, we need to make sure that we do not match one equation with a variable and another equation with the derivative of that variable. This is illustrated in the following example.

(a) without grouping (b) with grouping

FIG. 3. Different shifting graphs for Example 3.4.

EXAMPLE 3.4 Consider again the DDAE (3.1). Deleting the variable $\Delta_\tau x_2$ results in the graph Fig. 3(a) with maximal matching $M = \{\{F_1, \{x_1\}\}, \{F_2, \{\dot{x}_1\}\}\}$. Note that the maximal matching is possible since we match both equations with x_1 respectively its derivative $\dot{x}_1$. If we instead merge the variables x_1 and $\dot{x}_1$ into a single vertex we obtain the graph in Fig. 3(b), which indicates that we have to shift some of the equations (in agreement with the observation in Example 3.1).

To ensure that we do not match different equations with the same variable (with different differentiation levels) we need to group the corresponding variables as in Example 3.4. To merge variables of different differentiation levels we use the equivalence classes introduced in the graph for a DAE in Definition 2.1. Instead of imposing the trivial relation we introduce a new relation. To formalize this consider the equation

$$F\left(t, \Delta_{-\tau}x, \Delta_{-\tau}\dot{x}, \ldots, \Delta_{-\tau}x^{(\rho)}, \ldots, \Delta_{\kappa\tau}x, \Delta_{\kappa\tau}\dot{x}, \ldots, \Delta_{\kappa\tau}x^{(\rho)}\right) = 0, \tag{3.5}$$

which may be understood as a generalization of (1.1) that appears after shifting and/or differentiation of some of the equations.

DEFINITION 3.5 Consider the DDAE (3.5) and let Θ denote the set of variables for (3.5) (with $\vartheta^T = \begin{bmatrix}\Delta_{-\tau}x^T & \ldots & \Delta_{\kappa\tau}x^{(\rho)^T}\end{bmatrix}$).

- The variables $\theta, \widetilde{\theta} \in \Theta$ are called *shifting similar* if there exists $\xi \in \mathrm{set}(x)$ and integers $k \in \mathbb{Z}$, $p, q \in \mathbb{N}_0$ such that $\theta = \Delta_{k\tau}\xi^{(p)}$ and $\widetilde{\theta} = \Delta_{k\tau}\xi^{(q)}$. Shifting similarity introduces an equivalence relation on Θ, which we denote by R_{shift}. The graph of (3.5) over R_{shift} is called *shifting graph*.
- The variables $\theta, \widetilde{\theta} \in \Theta$ are called *differential similar* if there exists $\xi \in \mathrm{set}(x)$ and integers $p, k, \ell \in \mathbb{N}_0$ such that $\theta = \Delta_{k\tau}\xi^{(p)}$ and $\widetilde{\theta} = \Delta_{\ell\tau}\xi^{(p)}$. Differential similarity introduces an equivalence relation on $\widetilde{\Theta}$ with

  $$\widetilde{\Theta} = \{\theta \in \Theta \mid \text{there exists } x \in \Theta, k, p \in \mathbb{N}_0 \text{ such that } \theta = \Delta_{k\tau}x^{(p)}\},$$

  which we denote by R_{diff}. The graph of (3.5) over R_{diff} is called *differentiation graph*.
- The graph of (3.5) over the trivial equivalence relation $R := \{(\theta, \theta) \mid \theta \in \Theta\}$ is called *graph of the DDAE*.

EXAMPLE 3.6 The shifting graph for the DDAE (3.1) is given in Fig. 3(b). For the differentiation graph we have

$$\widetilde{\Theta} = \left\{\{x_1\}, \{\dot{x}_1\}\right\},$$

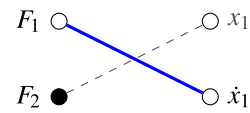


FIG. 4. Differentiation graph for the DDAE (3.1).

such that the differentiation graph is given in Fig. 4. Note that $\{x_2\} \notin \widetilde{\Theta}$.

A question that arises is how differentiation of equations affects the shifting graph and how shifting affects the differentiation graph (cf. Ha *et al.*, 2014 for an example where the order of shifting and differentiation results in different smoothness requirements). We immediately observe that the differentiation graph is affected by shifting, see the following example.

EXAMPLE 3.7 The differentiation graph for the scalar DDAE

$$0 = \Delta_{-\tau} x_1 + f_1 \tag{3.6}$$

does not contain any variable vertices and thus in particular no maximal matching. If we shift (3.6) we obtain

$$0 = x_1 + \Delta_{\tau} f_1,$$

such that the differentiation graph is given by $V = \{F_1, \{x_1\}\}$ and $E = \{\{F_1, \{x_1\}\}\}$.

To understand the effect of differentiation on the shifting graph let us define the system of equations

$$\check{F}\big(t, \Delta_{-\tau} x, \ldots, \Delta_{-\tau} x^{(\rho)}, \Delta_{-\tau} x^{(\rho+1)}, \ldots, \Delta_{\kappa\tau} x, \Delta_{\kappa\tau} \dot{x}, \ldots, \Delta_{\kappa\tau} x^{(\rho+1)}\big) = 0 \tag{3.7}$$

by replacing the ith equation ($i \in \{1, \ldots, M\}$) in (3.5) with its total derivative with respect to t. The shifting graphs of (3.5) and (3.7) are denoted by $G^{\mathrm{s}} := (V_{\mathrm{E}}^{\mathrm{s}} \dot{\cup} V_{\mathrm{V}}^{\mathrm{s}}, E^{\mathrm{s}})$ and $\check{G}^{\mathrm{s}} := (\check{V}_{\mathrm{E}}^{\mathrm{s}} \dot{\cup} \check{V}_{\mathrm{V}}^{\mathrm{s}}, \check{E}^{\mathrm{s}})$, respectively. Let us define the bijection $\phi \colon V_{\mathrm{E}}^{\mathrm{s}} \dot{\cup} V_{\mathrm{V}}^{\mathrm{s}} \to \check{V}_{\mathrm{E}}^{\mathrm{s}} \dot{\cup} \check{V}_{\mathrm{V}}^{\mathrm{s}}$ via

$$\phi(v) = \begin{cases} v \cup \{\dot{w} \mid w \in v\}, & \text{for } v \in V_{\mathrm{V}}^{\mathrm{s}}, \\ v, & \text{for } v \in V_{\mathrm{E}}^{\mathrm{s}} \text{ and } v \neq F_i, \\ \dot{v}, & \text{for } v \in V_{\mathrm{E}}^{\mathrm{s}} \text{ and } v = F_i. \end{cases}$$

Then it is easy to see that $\{u, v\} \in E^{\mathrm{s}}$ if and only if $\{\phi(u), \phi(v)\} \in \check{E}^{\mathrm{s}}$, that is, the graphs G^{s} and $\check{G}^{\mathrm{s}}$ are isomorphic (Jungnickel, 2013, p. 21). In particular we have shown the following result.

PROPOSITION 3.8 Consider the DDAE (3.5). Differentiation of an equation in (3.5) does not affect the shifting graph of (3.5).

Since shifting affects the differentiation graph but differentiation does not affect the shifting graph we conclude that we first determine which equations need to be shifted and afterwards determine which equations should be differentiated.

REMARK 3.9 The strategy to shift first can be understood as replacing the DDAE (1.1) with a transformed DDAE that can be solved with the method of steps. For linear time-invariant DDAEs such a strategy was investigated in Campbell (1995); Trenn & Unger (2019) in a polynomial matrix framework. There the authors use a sequence of row compressions to determine which equations need to be shifted. It should be noted that the row-compression requires differentiation of some of the equations to enable the shift afterward. Fore more details we refer to Campbell (1995); Trenn & Unger (2019) and the upcoming subsection.

3.3 *Differentiating during the shifting step*

If we find (for instance via Algorithm 2) an exposed equation in the shifting graph we conclude that this equation and all equations that are connected with this equation by an alternating path need to be shifted. However, simply shifting all these equations may not be sufficient, since the connection may exist only implicitly via the equivalence class. We illustrate this with the following example.

EXAMPLE 3.10 The shifting graph of Theorem 3.4, i.e., the shifting graph for the DDAE (3.1), is given in Fig. 3(b). If we match the first equation F_1 with the equivalence class $\{x_1, \dot{x}_1\}$ then F_2 is an exposed equation that is connected with F_1 via $\{x_1, \dot{x}_1\}$. Accordingly, we have to shift both equations. If we check the graph of (3.1) (presented in Fig. 3(a)) then we observe that there is no path from F_1 to F_2, i.e., we only obtain a path if we differentiate F_2.

To resolve an (possibly) implicit connection we have to ensure that there also exists a path in the graph of the DDAE and not only in the shifting graph. This can be achieved by differentiating the equation that does not depend on the highest derivative in the equivalence class. To ensure that all equations that need to be shifted are explicitly connected we do not need to resolve all possible implicit connections but only those that ensure a direct path in the graph of the DDAE. Consider the following example.

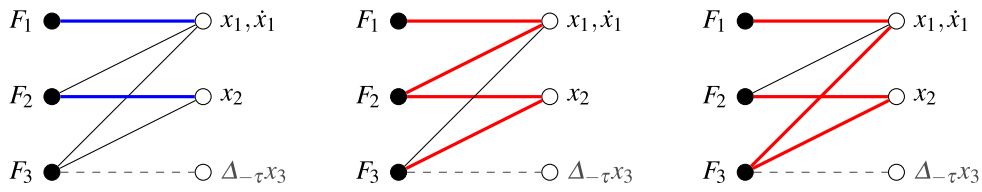
EXAMPLE 3.11 Consider the DDAE

$$\dot{x}_1 = f_1, \qquad \dot{x}_1 = x_2 + f_2, \qquad 0 = x_1 + x_2 + \Delta_{-\tau} x_3 + f_3, \tag{3.8}$$

which can be transformed to the DDAE (3.1) by inserting F_1 in F_2 and then inserting the result into F_3 yielding

$$\dot{x}_1 = f_1, \quad x_1 + \Delta_{-\tau x_3} + f_1 - f_2 + f_3.$$

Constructing the shifting graph of (3.8) and applying the shifting step to it, yields that the equation F_3 is exposed and connected through alternating paths to F_1 and F_2. Thus, all three equations need to be shifted and we need to find a direct connection in the graph of the DDAE. The possible connections of these equations are visualized in Fig. 5(b,c) by red edges.



(a) The shifting graph (b) first possible connection (c) second possible connection

FIG. 5. The shifting graph of (3.11) and possible connections.

We need to identify all possible connections. Given a matching $\mathscr{M}$ suppose that the equation F_j is exposed and the set C_{F_j} as defined in (2.7) is given by

$$C_{F_j} := \left\{ F_k \in V_{\mathrm{E}} \mid \text{there exists an alternating path between } F_j \text{ and } F_k \text{ in } G \right\}.$$

We define a *connection for* F_j to be a set of connected alternating paths (F_i, v_k, F_ℓ) with $(v_k, F_\ell) \in \mathscr{M}$, $(F_i, v_k) \in E \setminus \mathscr{M}$ such that all $(v_k, F_\ell) \in \mathscr{M}$ occur exactly once.

EXAMPLE 3.12 For Example 3.11 we immediately observe that there are two possible connections, which are displayed in Fig. 5(b,c). Note that the connection in Fig. 5(b) is a single alternating path, while the connection in Fig. 5(c) consists of two alternating paths.

For each connection P we need to decide which equations need to be differentiated. Denote by F_{i_1} the exposed equation and by F_{i_k} for $k = 2, \ldots, K$ the equations in $C_{F_{i_1}}$. Let ν_k denote the number how often we have to differentiate the i_kth equation. For each path $(F_{i_{k_j}}, v_j, F_{i_{k_{j+1}}}) \in P$ $(j = 1, \ldots, J)$ with $v_j \in V_{\mathrm{V}}^{\mathrm{s}}$ define $b_j \in \mathbb{Z}$ to be the number how many more times we have to differentiate $F_{i_{k_j}}$ than $F_{i_{k_{j+1}}}$ such that there exists a path to a highest derivative of both equations in the graph of the DDAE. This results in the linear system

$$\nu_{k_j} - \nu_{k_{j+1}} = b_j \qquad j = 1, \ldots, J, \tag{3.9}$$

which we compactly write as $A\nu = b$ with $\nu = [\nu_k] \in \mathbb{N}_0^K$, $b = [b_j] \in \mathbb{Z}^J$ and $A \in \mathbb{Z}^{J \times K}$. Since we do not want to differentiate equations more than necessary we have to solve the linear integer program

$$\begin{aligned} &\min \sum_{k=1}^{K} \nu_k, \\ &\text{such that } A\nu = b, \quad \nu \in \mathbb{N}_0. \end{aligned} \tag{3.10}$$

EXAMPLE 3.13 For the DDAE (3.8) the first connection given in Fig. 5(b) results in the linear program

$$\min \nu_1 + \nu_2 + \nu_3, \quad \text{s.t.} \begin{bmatrix} -1 & 1 & 0 \\ 0 & -1 & 1 \end{bmatrix} \nu = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \quad \nu \in \mathbb{N}_0$$

with the understanding that ν_k determines how often we have to differentiate the F_kth equation. It is easy to see that the 0 solution $\nu_1 = \nu_2 = \nu_3 = 0$ solves (3.10), i. e., we do not have to differentiate any equation. The other connection Fig. 5(c) results in the linear integer program (3.10) given via

$$A = \begin{bmatrix} -1 & 0 & 1 \\ 0 & -1 & 1 \end{bmatrix} \qquad \text{and} \qquad b = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

with the unique solution $\nu_1 = 0$ and $\nu_2 = \nu_3 = 1$. Hence, we have to differentiate F_2 and F_3 to resolve the connection.

A relaxation of the linear integer program (3.10) is the linear program

$$\begin{aligned} &\min \sum_{k=1}^{K} \nu_k, \\ &\text{such that } A\nu = b, \quad \nu \geq 0. \end{aligned} \tag{3.11}$$

In fact, we have the following result.

THEOREM 3.14

1. The linear program (3.11) has a unique solution.
2. The linear integer program (3.10) is solvable if and only if the linear program (3.11) is solvable. In this case the minimizing vector $\nu^\star$ of (3.11) is the unique minimizer of (3.10).

Proof.

1. Recall that J is the number of equations in the linear system $A\nu = b$ and K is the number of unknowns. We immediately observe $J = K - 1$ and $\operatorname{rank}(A) = K - 1$ (the equations F_{i_k} form a connected graph). Thus, A has full row rank and $A\nu = b$ is solvable for each b. Thus, there exists $\widehat{\nu} = [\widehat{\nu}_k] \in \mathbb{R}^K$ such that $A\widehat{\nu} = b$. Define $\alpha^\star = -\min\{\widehat{\nu}_k \mid k = 1, \ldots, K\}$. Let $e := [1, \ldots, 1]^T \in \mathbb{R}^K$. Then the feasible set of (3.11) is

$$\{\widehat{\nu} + \alpha e \mid \alpha \geqslant \alpha^\star\}. \tag{3.12}$$

   We conclude that $\nu^\star = \widehat{\nu} + \alpha^\star e$ is the unique minimizer of (3.11).

2. Using 1 it suffices to show that (3.10) is solvable whenever (3.11) is solvable. Let $\nu^\star = [\nu_k^\star]$ denote the unique minimizer of (3.11). Then the proof of 1 implies $\min\{\nu_k^\star \mid k = 1, \ldots, K\} = 0$. Since the equations (3.9) imply that the difference between two entries of $\nu^\star$ is an integer we conclude $\nu^\star \in \mathbb{N}_0^K$. The result follows from the observation that the feasible set of (3.10) is given by

$$\{\nu^\star + \alpha e \mid \alpha \in \mathbb{N}_0\}.$$

□

Theorem 3.14 implies that we can compute the solution (provided it exists) of the linear integer program (3.10) with standard methods such as the simplex algorithm or interior-point methods (Nelder & Mead, 1965; Karmarkar, 1984). Alternatively, we can exploit the structure of the feasible set

(3.12) and simply compute one solution of the linear system $A\nu = b$ and shift the solution as described in the first part of the proof of Theorem 3.14 1.

In summary we need to solve for each connection one linear program, which is always solvable. If we take of all solutions, the maximal number of differentiations for each equation, all implicit connections in the shifting graph are resolved. This is due to the fact that although we keep only the highest derivative/shift of an equation in our graph, one later uses all intermediate equations as well.

EXAMPLE 3.15 Continuing with Example 3.13 we have to differentiate F_2 and F_3 once. This immediately implies that both connections

$$\{(F_2, \{x_1, \dot{x}_1\}, F_1), (F_3, \{x_2\}, F_2)\} \qquad \text{and} \qquad \{(\dot{F}_3, \{x_1, \dot{x}_1\}, F_1), (\dot{F}_3, \{x_2\}, \dot{F}_2)\}$$

are direct.

The complete algorithm is stated in Algorithm 3. Even though we perform differentiations during the shifting step the differentiations do not influence the shifting graph (cf. Theorem 3.8). Thus, we do not update the shifting graph in the differentiation step but only the graph of the DDAE that is used to identify the number of needed differentiations.

Algorithm 3 The differentiation during the shifting step

Input: graph $\mathtt{G} = (\mathtt{V_E} \dot\cup \mathtt{V_V}, \mathtt{E})$, shifting graph $\mathtt{G^s} = (\mathtt{V_E^s} \dot\cup \mathtt{V_V^s}, \mathtt{E^s})$, `colorV`, `colorE`, `assign`, exposed equation $\mathtt{F_j} \in \mathtt{V_E}$
Output: graph `G`

1: Find all connections $\mathtt{P}^\ell$ for $\mathtt{F_j}$
2: **for** each connection $\mathtt{P}^\ell$ **do**
3: Construct and solve the LP (3.10) with solution ν^ℓ
4: **end for**
5: $\mathtt{colorEdif(F_i)} \leftarrow \arg\max_\ell \nu_i^\ell$ for all `i` ▷ number of differentiations for each equation
6: $\mathtt{G} \leftarrow$ `Algorithm6(G, dif, true, [], colorEdif)`

3.4 *Trimmed linearization*

The differentiation during the shifting step yields a graph of the DDAE, which might contain higher-order derivatives. The Pantelides algorithm is only applicable to equations that contain derivatives up to first-order, namely first-order systems. Thus, we need to reformulate the DDAE as a first-order system. Since we summarize in the shifting graph all variables which only differ by their order of differentiation in the same equivalence class it is sufficient to reformulate the DDAE after the shifting step.

The Pantelides algorithm is based on the information which variable appears in which equation. It, therefore, suffices to introduce new variables for variables that are differentiated more than once. This approach is similar to the first-order formulation of multi-body systems, where only new variables for the velocities are introduced but not for the derivative of the Lagrange multiplier.

REMARK 3.16 The process of reformulating a higher-order system to a first-order system is called linearization in the literature. The variant that we use here, where only variables that actually appear in the equations are replaced, is referred to as a trimmed linearization (Mehrmann & Shi, 2006; Scholz, 2011). If state transformations are allowed, then a minimal number of new variables can be introduced

(Scholz, 2011), which in turn minimizes smoothness requirements and benefits numerical methods. From a structural point of view, such a state transformation is, however, not feasible.

After the shifting step each equation in the shifting graph is matched to a highest shifted variable. Thus, we need to ensure that this property is still fulfilled after performing trimmed linearization.

EXAMPLE 3.17 Applying the shifting step introduced in Section 3.3 to the DDAE

$$x_1\dot{x}_1 = \Delta_{-\tau}x_2 + f_1, \quad \dot{x}_2 = \Delta_{-\tau}x_3 + f_2, \quad 0 = \Delta_{-\tau}x_1 + f_3$$

in Hessenberg form we obtain the shifted and differentiated DDAE

$$\begin{aligned} \Delta_{2\tau}x_1\Delta_{2\tau}\ddot{x}_1 + \Delta_{2\tau}\dot{x}_1^2 &= \Delta_\tau\dot{x}_2 + \Delta_{2\tau}\dot{f}_1, \\ \Delta_\tau\dot{x}_2 &= x_3 + \Delta_\tau f_2, \\ 0 &= \Delta_{2\tau}\ddot{x}_1 + \Delta_{3\tau}\ddot{f}_3, \end{aligned}$$

which contains a second derivative of x_1. To perform the differentiation step we need to reformulate this DDAE as first-order system by introducing

$$y_1 = \dot{x}_1. \tag{3.13}$$

If we replace variables accordingly we obtain the first-order DDAE

$$\begin{aligned} \Delta_{2\tau}x_1\Delta_{2\tau}\dot{y}_1 + \Delta_{2\tau}y_1^2 &= \Delta_\tau\dot{x}_2 + \Delta_{2\tau}\dot{f}_1, \\ \Delta_\tau\dot{x}_2 &= x_3 + \Delta_\tau f_2, \\ 0 &= \Delta_{2\tau}\dot{y}_1 + \Delta_{3\tau}\ddot{f}_3, \end{aligned} \tag{3.14}$$

together with (3.13). No variable that appears in (3.13) is a highest shift and consequently we cannot assign in the shifting graph each equation to an equivalence class. If we shift (3.13) twice the corresponding shifting graph contains a maximal matching.

The following theorem provides a first-order system such that applying the shifting step to it yields no shifts.

THEOREM 3.18 Consider a DDAE that depends on variables that are differentiated more than once, i.e., there exists $i \in \{1, \dots, n\}$ such that

$$q_i := \max\left\{\ell \;\middle|\; \Delta_{k\tau}x_i^{(\ell)} \text{ appears in some equation for } k \geqslant 0\right\} \geqslant 2,$$

and assume that there exists a maximal matching in the shifting graph. Define

$$\ell_p := \max\{k \mid \Delta_{k\tau}x_i^{(q)} \text{ appears in some equation for } q \geqslant p\}.$$

Replace for $1 \leqslant p < q_i$ the variable $\Delta_{k\tau} x_i^{(p)}$ by $\Delta_{k\tau} y_{i_p}$ and $\Delta_{k\tau} x_i^{(q_i)}$ by $\Delta_{k\tau} y_{y_{q_{i-1}}}$. We add the equations

$$\begin{aligned}
\Delta_{\ell_1\tau}\dot{x}_i &= \Delta_{\ell_1\tau} y_{i_1} \\
\Delta_{\ell_2\tau}\dot{y}_{i_1} &= \Delta_{\ell_2\tau} y_{i_2} \\
&\vdots \\
\Delta_{\ell_{q_i-1}\tau}\dot{y}_{i_{q_i-2}} &= \Delta_{\ell_{q_i-1}\tau}\dot{y}_{i_{q_i-1}}
\end{aligned}$$

to the DDAE. Then the new DDAE has a maximal matching in its shifting graph.

Proof. Let $\mathscr{M}$ denote the maximal matching in the shifting graph before we replace variables and add new equations. Then some equation F_j is matched with the equivalence class corresponding to $\Delta_{\ell\tau} x_i$, where $\ell = \max\{\ell_1, \ldots, \ell_{q_i}\} = \ell_1$. Note that by assumption we have introduced $q_i - 1$ new variables yielding $q_i - 1$ new equivalence classes in the shift graph. If after replacing we can still match F_j with the equivalence class corresponding to $\Delta_{\ell\tau} x_i$ then we can construct a maximal matching by assigning each new equation to the equivalence class of the variable on its right-hand side and obtain a maximal matching. Each equation is assigned to a highest shift since $\ell_i \geqslant \ell_{i+1}$ by construction. If after replacing we cannot assign F_j to the equivalence class corresponding to $\Delta_{\ell\tau} x_i$ then at least one of the newly introduced variables must appear in F_j and we can assign F_j to the associated equivalence class of $\Delta_{\ell\tau} y_{i_k}$ that appears on the right-hand side. Thus, we can assign all equations

$$\begin{aligned}
\Delta_{\ell\tau}\dot{x}_i &= \Delta_{\ell\tau} y_{i_1} \\
\Delta_{\ell\tau}\dot{y}_{i_1} &= \Delta_{\ell\tau} y_{i_2} \\
&\vdots \\
\Delta_{\ell\tau}\dot{y}_{i_{k-1}} &= \Delta_{\ell\tau}\dot{y}_{i_k}
\end{aligned}$$

to the left-hand side that are shifted all ℓ times by construction of ℓ_i. The remaining equations can be assigned to the right-hand side that are by construction highest shifts. □

3.5 *The algorithm*

Before we summarize our findings in form of an algorithm let us briefly recap the results of this section.

- We illustrated that we can use the original Pantelides algorithm (Pantelides, 1988) (see Algorithm 2) with a different equivalence class to determine which equations need to be shifted. Recall that we may have to differentiate some equations during the shifting step to resolve implicit connections via equivalence classes (cf. Section 3.3). To account for the additional differentiation we present a slight modification of Algorithm 2 in Algorithm 4. We emphasize that the notation $\xi_{\mathtt{i},\mathtt{k}}$ for $\xi \in \{\mathtt{F}, \mathtt{x}\}$ has a different meaning for shifting and differentiation. In the case of shifting the $\mathtt{k}$ represents the order of shift while in the case of differentiation $\mathtt{k}$ represents the order of differentiation. This enables us to shift in the shifting graph and differentiate in the differentiation graph with the same notation.

Algorithm 4 Generalization of Algorithm 2 for shifting and differentiation

Input: graph $\mathtt{G} = (\mathtt{V_E} \dot\cup \mathtt{V_V}, \mathtt{E})$

denote by $\Delta_{m\tau}\mathtt{F}_j^{(\ell)}$, respectively $\Delta_{n\tau}\mathtt{x}_i^{(k)}$ the elements of $\mathtt{V_E}$, respectively $\mathtt{V_V}$

$\mathtt{type} \in \{\mathtt{shift}, \mathtt{dif}\}$

graph $\mathtt{G}^t = (\mathtt{V}_\mathtt{E}^t \dot\cup \mathtt{V}_\mathtt{V}^t, \mathtt{E}^t)$ of type $\mathtt{type}$

$$\mathtt{V}_\mathtt{E}^t = \begin{cases} \left\{\mathtt{F}_{j,\ell} \mid \Delta_{m\tau}\mathtt{F}_j^{(\ell)} \in \mathtt{V_E}\right\} & \text{if } \mathtt{type} == \mathtt{dif} \\ \left\{\mathtt{F}_{j,\ell} \mid \Delta_{\ell\tau}\mathtt{F}_j^{(m)} \in \mathtt{V_E}\right\} & \text{if } \mathtt{type} == \mathtt{shift} \end{cases}$$

$$\mathtt{V}_\mathtt{V}^t = \begin{cases} \left\{\mathtt{x}_{i,k} \mid \Delta_{n\tau}\mathtt{x}_i^{(k)} \in \mathtt{V_V}\right\} & \text{if } \mathtt{type} == \mathtt{dif} \\ \left\{\mathtt{x}_{i,k} \mid \Delta_{k\tau}\mathtt{x}_i^{(n)} \in \mathtt{V_V}\right\} & \text{if } \mathtt{type} == \mathtt{shift} \end{cases}$$

Output: graph $\mathtt{G} = (\mathtt{V_E} \dot\cup \mathtt{V_V}, \mathtt{E})$

$\mathtt{assign}(\mathtt{x}_{i,k}) \leftarrow 0$ for each $\mathtt{x}_{i,k} \in \mathtt{V}_\mathtt{V}^t$
for $\mathtt{r} = 1, \ldots, \mathtt{M}$ **do**
  $(\mathtt{p}, \mathtt{q}) \leftarrow (\mathtt{r}, 0)$
  repeat
    $\tilde{\mathtt{E}}^t \leftarrow \mathtt{E}^t \setminus \{\{\mathtt{x}_{i,k}, \mathtt{F}_{j,\ell}\} \in \mathtt{E}^t \mid \mathtt{x}_{i,k+s} \in \mathtt{V}_\mathtt{V}^t \text{ for some } \mathtt{s} > 0\}$
    $\tilde{\mathtt{V}}_\mathtt{V}^t \leftarrow \mathtt{V}_\mathtt{V}^t \setminus \{\mathtt{x}_{i,k} \mid \mathtt{x}_{i,k+s} \in \mathtt{V}_\mathtt{V}^t \text{ for some } \mathtt{s} > 0\}$
    $\mathtt{colorV}(\mathtt{x}_{i,k}) \leftarrow 0$ for each $\mathtt{x}_{i,k} \in \tilde{\mathtt{V}}_\mathtt{V}^t$
    $\mathtt{colorE}(\mathtt{F}_{j,\ell}) \leftarrow 0$ for each $\mathtt{F}_{j,\ell} \in \mathtt{V}_\mathtt{E}^t$
    $(\mathtt{pathfound}, \mathtt{assign}, \mathtt{colorV}, \mathtt{colorE}) \leftarrow$ Algorithm1$((\mathtt{V}_\mathtt{E}^t \dot\cup \tilde{\mathtt{V}}_\mathtt{V}^t, \tilde{\mathtt{E}}^t), \mathtt{F}_{p,q}, \mathtt{assign}, \mathtt{colorV}, \mathtt{colorE})$ ▷ apply Augmentpath
    if $\mathtt{pathfound} == \mathtt{false}$ **then**
      if $\mathtt{type} == \mathtt{shift}$ **then**
        $\mathtt{G} \leftarrow \mathtt{Algorithm3}(\mathtt{G}, \mathtt{G}^t, \mathtt{colorV}, \mathtt{colorE}, \mathtt{assign}, \mathtt{F}_{p,q})$ ▷ differentiation during the shifting step
      end if
      $\mathtt{G}^t \leftarrow \mathtt{Algorithm6}(\mathtt{G}^t, \mathtt{type}, \mathtt{boolFull} \leftarrow \mathtt{false}, \mathtt{colorV}, \mathtt{colorE})$ ▷ shift or different the equations
      $\mathtt{G} \leftarrow \mathtt{Algorithm6}(\mathtt{G}, \mathtt{type}, \mathtt{boolFull} \leftarrow \mathtt{true}, \mathtt{colorV}, \mathtt{colorE})$ ▷ shift or different the equations
      for each $\mathtt{i}$ with $\mathtt{colorV}(\mathtt{x}_{i,k}) == 1$ and $\mathtt{F}_{j,\ell} == \mathtt{assign}(\mathtt{x}_{i,k})$ **do**
        $\mathtt{assign}(\mathtt{x}_{i,k+1}) \leftarrow \mathtt{F}_{j,\ell}$
      end for
    end if
  until $\mathtt{pathfound} = \mathtt{true}$
end for

- Due to the structure of the equivalence classes we shift first and then differentiate, see Proposition 3.8. Note that we do not have to resolve implicit connections within the differentiation step since we only determine the maximal number that each equation needs to be shifted and differentiated. Since a possible implicit connection in the differentiation step results from equations that are already shifted the implicit connection is resolved automatically by using the same equation with a smaller shift than the maximal shift that was determined in the shifting step.

- The differentiation during the shifting step might yield higher-order derivatives. Since the Pantelides algorithm can only handle first-order systems the resulting DDAE must be transformed into one. This can be achieved using trimmed linearization.

The complete methodology is summarized in Algorithm 5.

Algorithm 5 Pantelides Algorithm for DDAEs

Input: graph $\mathtt{G} = (\mathtt{V_E} \dot\cup \mathtt{V_V}, \mathtt{E})$, denote by $\Delta_{\mathtt{m}\tau}\mathtt{F}_{\mathtt{j}}^{(\ell)}$, respectively $\Delta_{\mathtt{n}\tau}\mathtt{x}_{\mathtt{i}}^{(\mathtt{k})}$ the elements of $\mathtt{V_E}$, respectively $\mathtt{V_V}$.

Output: graph $\mathtt{G}$

Step 1: Shifting

$\mathtt{V_E^s} \leftarrow \{\mathtt{F}_{\mathtt{j},0} \mid \mathtt{F}_{\mathtt{j}} \in \mathtt{V_E}\}$

$\mathtt{V_V^s} \leftarrow \left\{\mathtt{x}_{\mathtt{i},\mathtt{k}} \mid \Delta_{\mathtt{k}\tau}\mathtt{x}_{\mathtt{i}}^{(\mathtt{n})} \in \mathtt{V_V}\right\}$

$\mathtt{E^s} \leftarrow \{\{\mathtt{x}_{\mathtt{i},\mathtt{k}}, \mathtt{F}_{\mathtt{j},\ell}\} \mid \exists \mathtt{n}, \mathtt{m} \in \mathbb{N}_0 \text{ s.t. } \{\Delta_{\mathtt{k}\tau}\mathtt{x}_{\mathtt{i}}^{(\mathtt{n})}, \Delta_{\ell\tau}\mathtt{F}_{\mathtt{j}}^{(\mathtt{m})}\} \in \mathtt{E}\}$

$\mathtt{G^s} \leftarrow (\mathtt{V_V^s} \dot\cup \mathtt{V_E^s}, \mathtt{E^s})$

$\mathtt{G} \leftarrow \mathtt{Algorithm4}(\mathtt{G}, \mathtt{shift}, \mathtt{G^s})$ ▷ Generalization of the Pantelides algorithm

Step 2: Trimmed Linearization

Add equations to graph $\mathtt{G}$ according to Theorem 3.16

Step 3: Differentiation

$\mathtt{V_E^d} \leftarrow \left\{\mathtt{F}_{\mathtt{j},\ell} \mid \Delta_{\mathtt{m}\tau}\mathtt{F}_{\mathtt{j}}^{(\ell)} \in \tilde{\mathtt{V}}_{\mathtt{E}}\right\}$

$\mathtt{V_V^d} \leftarrow \left\{\mathtt{x}_{\mathtt{i},\mathtt{k}} \mid \Delta_{\mathtt{n}\tau}\mathtt{x}_{\mathtt{i}}^{(\mathtt{k})} \in \tilde{\mathtt{V}}_{\mathtt{V}}\right\}$

$\mathtt{E^d} \leftarrow \{\{\mathtt{x}_{\mathtt{i},\mathtt{k}}, \mathtt{F}_{\mathtt{j},\ell}\} \mid \exists \mathtt{n}, \mathtt{m} \text{ s.t. } \{\Delta_{\mathtt{n}\tau}\mathtt{x}_{\mathtt{i}}^{(\mathtt{k})}, \Delta_{\mathtt{m}\tau}\mathtt{F}_{\mathtt{j}}^{(\ell)}\} \in \tilde{\mathtt{E}}\}$

$\mathtt{G^d} \leftarrow (\mathtt{V_E^d} \dot\cup \mathtt{V_V^d}, \mathtt{E^d})$

$\mathtt{G} \leftarrow \mathtt{Algorithm4}(\mathtt{G}, \mathtt{dif}, \mathtt{G^d})$ ▷ Generalization of the Pantelides algorithm

REMARK 3.19 During the shifting and differentiation step we have to update the graph and the corresponding shifting and differentiation graphs. To unify this procedure the necessary updates of the graph are summarized in Algorithm 6. This algorithm is called in the modified Pantelides algorithm (Algorithm 4).

4. Examples

We illustrate Algorithm 5 with two examples. The first two examples, which we discuss in Sections 4.1 and 4.2, are linear toy examples that illustrates the different steps of Algorithm 5. The third example (cf. Section 4.3) is a nonlinear DDAE that arises in real-time dynamic substructuring (Bursi & Wagg, 2008).

Algorithm 6 Shift or differentiate equations

Input: graph $\mathtt{G} = (\mathtt{V}_\mathtt{E} \dot\cup \mathtt{V}_\mathtt{V}, \mathtt{E})$

$\mathtt{type} \in \{\mathtt{shift}, \mathtt{dif}\}$, $\mathtt{boolFull} \in \{\mathtt{true}, \mathtt{false}\}$

$\mathtt{colorV}, \mathtt{colorE}$

Output: graph $\tilde{\mathtt{G}} = (\tilde{\mathtt{V}}_\mathtt{V} \dot\cup \tilde{\mathtt{V}}_\mathtt{E}, \tilde{\mathtt{E}})$

if $\mathtt{boolFull} == \mathtt{false}$ **then**
  $\mathtt{x}_{\mathtt{i},\mathtt{k},0} \leftarrow \mathtt{x}_{\mathtt{i},\mathtt{k}} \in \mathtt{V}_\mathtt{V}$ for all $\mathtt{i}, \mathtt{k}$
  $\mathtt{F}_{\mathtt{j},\ell,0} \leftarrow \mathtt{F}_{\mathtt{j},\ell} \in \mathtt{V}_\mathtt{E}$ for all $\mathtt{j}, \ell$
else if $\mathtt{boolFull} == \mathtt{true}$ and $\mathtt{type} == \mathtt{shift}$ **then**
  $\mathtt{x}_{\mathtt{i},\mathtt{k},\mathtt{n}} \leftarrow \Delta_{\mathtt{k}\tau}\mathtt{x}_\mathtt{i}^{(\mathtt{n})} \in \mathtt{V}_\mathtt{V}$ for all $\mathtt{i}, \mathtt{k}, \mathtt{n}$
  $\mathtt{F}_{\mathtt{j},\ell,\mathtt{m}} \leftarrow \Delta_{\ell\tau}\mathtt{F}_\mathtt{j}^{(\mathtt{m})} \in \mathtt{V}_\mathtt{E}$ for all $\mathtt{j}, \ell, \mathtt{m}$
else
  $\mathtt{x}_{\mathtt{i},\mathtt{k},\mathtt{n}} \leftarrow \Delta_{\mathtt{n}\tau}\mathtt{x}_\mathtt{i}^{(\mathtt{k})} \in \mathtt{V}_\mathtt{V}$ for all $\mathtt{i}, \mathtt{k}, \mathtt{n}$
  $\mathtt{F}_{\mathtt{j},\ell,\mathtt{m}} \leftarrow \Delta_{\mathtt{m}\tau}\mathtt{F}_\mathtt{j}^{(\ell)} \in \mathtt{V}_\mathtt{E}$ for all $\mathtt{j}, \ell, \mathtt{m}$
end if
$\tilde{\mathtt{V}}_\mathtt{V} \leftarrow \{\mathtt{x}_{\mathtt{i},\mathtt{k},\mathtt{n}} \in \mathtt{V}_\mathtt{V}\}$
$\tilde{\mathtt{V}}_\mathtt{E} \leftarrow \{\mathtt{F}_{\mathtt{j},\ell,\mathtt{m}} \in \mathtt{V}_\mathtt{E}\}$
$\tilde{\mathtt{E}} \leftarrow \left\{\{\mathtt{x}_{\mathtt{i},\mathtt{k},\mathtt{n}}, \mathtt{F}_{\mathtt{j},\ell,\mathtt{m}}\} \in \mathtt{E}\right\}$
if $\mathtt{colorV} == []$ **then**
  for each $\mathtt{x}_{\mathtt{i},\mathtt{k},\mathtt{n}} \in \tilde{\mathtt{V}}_\mathtt{V}$ define $\mathtt{colorV}(\mathtt{x}_{\mathtt{i},\mathtt{k},\mathtt{n}}) \leftarrow \max\{\mathtt{q} \mid \mathtt{F}_{\mathtt{j},\ell,\mathtt{m}} \in \tilde{\mathtt{V}}_\mathtt{E}, \{\mathtt{x}_{\mathtt{i},\mathtt{k},\mathtt{n}}, \mathtt{F}_{\mathtt{j},\ell,\mathtt{m}}\} \in \tilde{\mathtt{E}}$ and $\mathtt{colorE}(\mathtt{F}_{\mathtt{j},\ell,\mathtt{m}}) == \mathtt{q}\}$
end if
$\tilde{\mathtt{V}}_\mathtt{V} \leftarrow \tilde{\mathtt{V}}_\mathtt{V} \cup \{\mathtt{x}_{\mathtt{i},\mathtt{k}+\mathtt{p},\mathtt{n}} \mid \mathtt{x}_{\mathtt{i},\mathtt{k},\mathtt{n}} \in \tilde{\mathtt{V}}_\mathtt{V}, \mathtt{colorV}(\mathtt{x}_{\mathtt{i},\mathtt{k},\mathtt{n}}) == \mathtt{q}, 1 \le \mathtt{p} \le \mathtt{q}\}$
for each $(\mathtt{j}, \ell)$ s. t. $\mathtt{colorE}(\mathtt{F}_{\mathtt{j},\ell,\mathtt{m}}) == \mathtt{q}$, $\mathtt{q} > 0$ **do**
  $\mathtt{F}_{\mathtt{j},\ell,\mathtt{m}} \leftarrow \mathtt{F}_{\mathtt{j},\ell+\mathtt{q},\mathtt{m}}$
  if $\mathtt{type} == \mathtt{shift}$ **then**
    $\left\{\mathtt{x}_{\mathtt{i},\mathtt{k},\mathtt{n}}, \mathtt{F}_{\mathtt{j},\ell,\mathtt{m}}\right\} \in \tilde{\mathtt{E}} \leftarrow \left\{\mathtt{x}_{\mathtt{i},\mathtt{k}+1,\mathtt{n}}, \mathtt{F}_{\mathtt{j},\ell,\mathtt{m}}\right\}$ for each $\mathtt{i}, \mathtt{k}, \mathtt{n}$
  else
    $\tilde{\mathtt{E}} \leftarrow \tilde{\mathtt{E}} \cup \left\{\{\mathtt{x}_{\mathtt{i},\mathtt{k}+\mathtt{p},\mathtt{n}}, \mathtt{F}_{\mathtt{j},\ell,\mathtt{m}}\} \mid \mathtt{x}_{\mathtt{i},\mathtt{k}+\mathtt{p},\mathtt{n}} \in \tilde{\mathtt{V}}_\mathtt{V}, \{\mathtt{x}_{\mathtt{i},\mathtt{k},\mathtt{n}}, \mathtt{F}_{\mathtt{j},\ell,\mathtt{m}}\} \in \tilde{\mathtt{E}}, 1 \le \mathtt{p} \le \mathtt{q}\right\}$
  end if
end for
$\tilde{\mathtt{G}} \leftarrow (\tilde{\mathtt{V}}_\mathtt{V} \dot\cup \tilde{\mathtt{V}}_\mathtt{E}, \tilde{\mathtt{E}})$

4.1 *Academic example 1*

The initial shifting graph of the DDAE

$$\begin{aligned} 0 &= x_1 + f_1 \\ \dot{x}_1 &= \Delta_{-\tau} x_2 + f_2 \\ \dot{x}_2 &= x_1 + \Delta_{-\tau} x_3 + f_3 \\ \dot{x}_3 &= x_4 + \Delta_{-\tau} x_1 + f_4 \end{aligned} \tag{4.1}$$

is given in Fig. 6(a) and in the notation of Algorithm 5 by the sets

$$\begin{aligned}
V_{\mathrm{E}}^{\mathrm{s}} &:= \left\{\mathtt{F}_{1,0}, \mathtt{F}_{2,0}, \mathtt{F}_{3,0}, \mathtt{F}_{4,0}\right\},\\
V_{\mathrm{V}}^{\mathrm{s}} &:= \left\{\mathtt{x}_{1,0}, \mathtt{x}_{2,0}, \mathtt{x}_{3,0}, \mathtt{x}_{4,0}, \mathtt{x}_{1,-1}, \mathtt{x}_{2,-1}, \mathtt{x}_{3,-1}\right\},\\
E^{\mathrm{s}} &:= \big\{\{\mathtt{F}_{1,0}, \mathtt{x}_{1,0}\}, \{\mathtt{F}_{2,0}, \mathtt{x}_{1,0}\}, \{\mathtt{F}_{2,0}, \mathtt{x}_{2,-1}\}, \{\mathtt{F}_{3,0}, \mathtt{x}_{1,0}\}, \{\mathtt{F}_{3,0}, \mathtt{x}_{2,0}\},\\
&\qquad \{\mathtt{F}_{3,0}, \mathtt{x}_{3,-1}\}, \{\mathtt{F}_{4,0}, \mathtt{x}_{3,0}\}, \{\mathtt{F}_{4,0}, \mathtt{x}_{4,0}\}, \{\mathtt{F}_{4,0}, \mathtt{x}_{1,-1}\}\big\},
\end{aligned}$$

where each $\mathtt{x}_{i,j}$ denotes one equivalence class. We observe that there is no maximal matching in the shifting graph and we may choose either F_1 or F_2 as exposed (cf. Fig. 6(b)). Since both are connected via a matching we shift F_1 and F_2. Note that F_3 has a path to F_1 and F_2 but the path is not an alternating path. Thus, we do not shift F_3. The equations F_1 and F_2 are only implicitly connected via the equivalence class $\{x_1, \dot{x}_1\}$. Constructing the linear integer program as described in Section 3.3 yields

$$\min \nu_1 + \nu_2 \quad \text{such that} \quad \begin{bmatrix} 1 & -1 \end{bmatrix} \begin{bmatrix} \nu_1 \\ \nu_2 \end{bmatrix} = \begin{bmatrix} 1 \end{bmatrix}, \quad \nu_1, \nu_2 \in \mathbb{N}_0$$

with the solution $\nu_1 = 1$, $\nu_2 = 0$. Consequently, we differentiate F_1. In the resulting graph Fig. 6(c) a possible matching is giving by $\mathscr{M} = \{\{\mathtt{F}_{1,1}, \mathtt{x}_{1,1}\}, \{\mathtt{F}_{2,1}, \mathtt{x}_{2,0}\}, \{\mathtt{F}_{4,0}, \mathtt{x}_{4,0}\}\}$. The matching is not maximal and the exposed vertex $\mathtt{F}_{3,0}$ is connected through an alternating path to $\mathtt{F}_{1,1}$ and $\mathtt{F}_{2,1}$. Thus, we need to shift these three equations. The linear program for these three equations

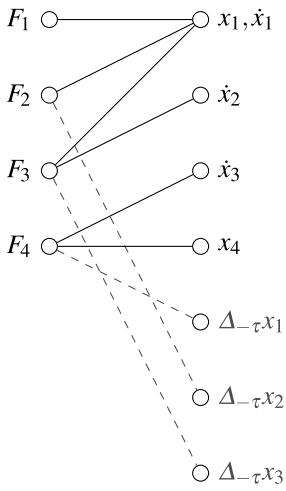
$$\min \nu_1 + \nu_2 + \nu_3 \quad \text{such that} \quad \begin{bmatrix} 0 & 1 & -1 \\ 1 & -1 & 0 \end{bmatrix} \begin{bmatrix} \nu_1 \\ \nu_2 \\ \nu_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad \nu_1, \nu_2, \nu_3 \in \mathbb{N}_0$$

with solution $\nu_1 = \nu_2 = 1$ and $\nu_3 = 0$ yields that we differentiate the first two equations. The resulting shifting graph Fig. 6(d) provides a maximal matching. The differentiation during the shifting step results in a single variable that is differentiated twice or more, namely $\Delta_{2\tau}\ddot{x}_1$. Since the Pantelides algorithm can only handle derivatives up to first order we replace this variable in the trimmed linearization step by $\dot{y}_{1_0}$ and add the equation G_{1_0} given by

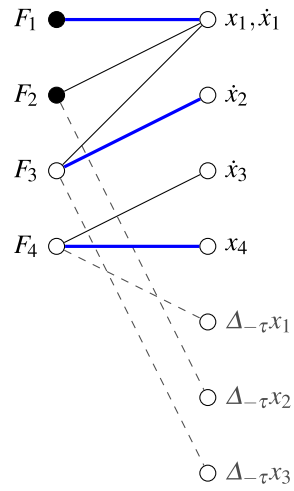
$$y_{1_0} = \dot{x}_1.$$

Since $\Delta_{2\tau}\ddot{x}_1$ appears we shift G_{1_0} twice. To enlarge the full graph by this variable and equation we rename y_{1_0} to x_5 and G_{1_0} to F_5. This yields the graph of the DDAE

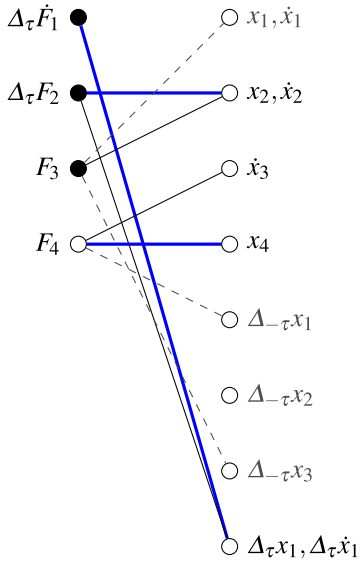
$$\begin{aligned}
\tilde{V}_{\mathrm{E}} &:= \left\{\Delta_{2\tau}\ddot{\mathtt{F}}_1, \Delta_{2\tau}\ddot{\mathtt{F}}_2, \Delta_{2\tau}\ddot{\mathtt{F}}_3, \Delta_{2\tau}\ddot{\mathtt{F}}_4, \Delta_{2\tau}\ddot{\mathtt{F}}_5\right\},\\
\tilde{V}_{\mathrm{V}} &:= \left\{\Delta_{\tau}\mathtt{x}_1, \Delta_{2\tau}\mathtt{x}_1, \Delta_{2\tau}\dot{\mathtt{x}}_1, \Delta_{\tau}\dot{\mathtt{x}}_2, \mathtt{x}_3, \dot{\mathtt{x}}_3, \mathtt{x}_4, \Delta_{2\tau}\mathtt{x}_5, \Delta_{2\tau}\dot{\mathtt{x}}_5\right\},\\
\tilde{E} &:= \big\{\{\Delta_{2\tau}\ddot{\mathtt{F}}_1, \Delta_{2\tau}\mathtt{x}_1\}, \{\Delta_{2\tau}\ddot{\mathtt{F}}_1, \Delta_{2\tau}\mathtt{x}_5\}, \{\Delta_{2\tau}\ddot{\mathtt{F}}_1, \Delta_{2\tau}\dot{\mathtt{x}}_5\}, \{\Delta_{2\tau}\ddot{\mathtt{F}}_2, \Delta_{2\tau}\mathtt{x}_2\}, \{\Delta_{2\tau}\ddot{\mathtt{F}}_2, \Delta_{2\tau}\dot{\mathtt{x}}_2\}\\
&\qquad \{\Delta_{2\tau}\dot{\mathtt{F}}_2, \Delta_{2\tau}\mathtt{x}_5\}, \{\Delta_{2\tau}\dot{\mathtt{F}}_2, \Delta_{2\tau}\mathtt{x}_5\}, \{\Delta_{2\tau}\mathtt{F}_3, \Delta_{2\tau}\mathtt{x}_1\}, \{\Delta_{2\tau}\mathtt{F}_3, \Delta_{2\tau}\dot{\mathtt{x}}_2\}, \{\Delta_{2\tau}\mathtt{F}_3, \mathtt{x}_3\},\\
&\qquad \{\mathtt{F}_4, \dot{\mathtt{x}}_3\}, \{\mathtt{F}_4, \mathtt{x}_4\}, \{\Delta_{2\tau}\mathtt{F}_5, \Delta_{2\tau}\dot{\mathtt{x}}_1\}, \{\Delta_{2\tau}\mathtt{F}_5, \Delta_{2\tau}\mathtt{x}_5\}\big\}.
\end{aligned}$$



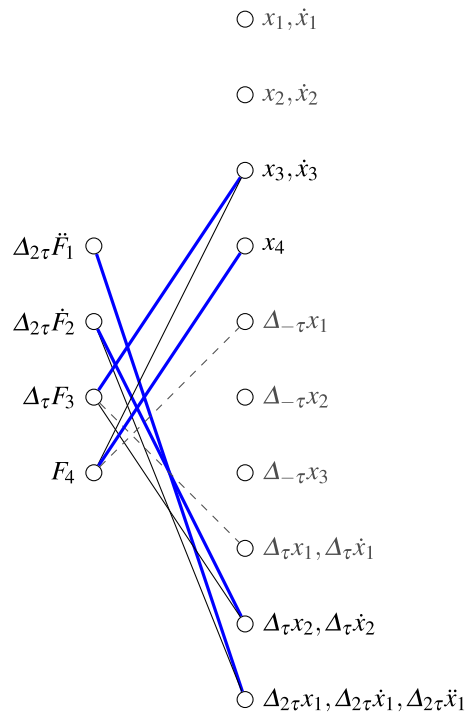
(a) Initial shifting graph



(b) Shifting graph with assignment



(c) Shifting graph after differentiating F_1 and shifting $\dot{F}_1$ and F_2 with assignment



(d) Shifting graph after differentiating $\Delta_\tau \dot{F}_1$, $\Delta_\tau F_2$ and shifting $\Delta_\tau \ddot{F}_1$, $\Delta_\tau \dot{F}_2$ and F_3

FIG. 6. Visualization of the shifting step from Algorithm 5 for the DDAE (4.1).

Now we construct the differentiation graph for the equations $\Delta_{2\tau}\ddot{F}_1, \Delta_{2\tau}\dot{F}_2, \Delta_\tau F_3, F_4$ and $\Delta_{2\tau}\mathtt{F}_5$ given by

$$
\begin{aligned}
V_{\mathrm{E}}^{\mathrm{d}} &:= \left\{\mathtt{F}_{1,2}, \mathtt{F}_{2,1}, \mathtt{F}_{3,0}, \mathtt{F}_{4,0}, \mathtt{F}_{5,0}\right\},\\
V_{\mathrm{V}}^{\mathrm{d}} &:= \left\{\mathtt{x}_{1,0}, \mathtt{x}_{2,0}, \mathtt{x}_{3,0}, \mathtt{x}_{4,0}, \mathtt{x}_{1,1}, \mathtt{x}_{2,1}, \mathtt{x}_{3,1}, \mathtt{x}_{5,0}, \mathtt{x}_{5,1}\right\},\\
E^{\mathrm{d}} &:= \big\{\{\mathtt{F}_{1,2}, \mathtt{x}_{1,0}\}, \{\mathtt{F}_{1,2}, \mathtt{x}_{1,1}\}, \{\mathtt{F}_{1,2}, \mathtt{x}_{5,1}\}, \{\mathtt{F}_{2,1}, \mathtt{x}_{2,0}\}, \{\mathtt{F}_{2,1}, \mathtt{x}_{1,1}\}, \{\mathtt{F}_{2,1}, \mathtt{x}_{2,1}\}, \{\mathtt{F}_{2,1}, \mathtt{x}_{5,1}\},\\
&\qquad \{\mathtt{F}_{3,0}, \mathtt{x}_{1,0}\}, \{\mathtt{F}_{3,0}, \mathtt{x}_{3,0}\}, \{\mathtt{F}_{3,0}, \mathtt{x}_{2,1}\}, \{\mathtt{F}_{4,0}, \mathtt{x}_{4,0}\}, \{\mathtt{F}_{4,0}, \mathtt{x}_{3,1}\}, \{\mathtt{F}_{5,0}, \mathtt{x}_{1,1}\}, \{\mathtt{F}_{5,0}, \mathtt{x}_{5,0}\}\big\},
\end{aligned}
$$

or respectively Fig. 7(a). Note that the variable $\Delta_{-\tau}x_1$ does not occur in the differentiation graph since it is shifted. The vertex $\Delta_\tau F_3$ is exposed and connected through an alternating path to $\Delta_{2\tau}\ddot{F}_1$ and $\Delta_{2\tau}\dot{F}_2$. Thus, we differentiate all three equations that results in Fig. 7(a). The existence of a maximal matching in Fig. 7(a) leads to the termination of Algorithm 5. We conclude that we shift the first, second and fifth equation twice, the third equation once and that we differentiate the first equation three time, the second equation twice and the third equation once.

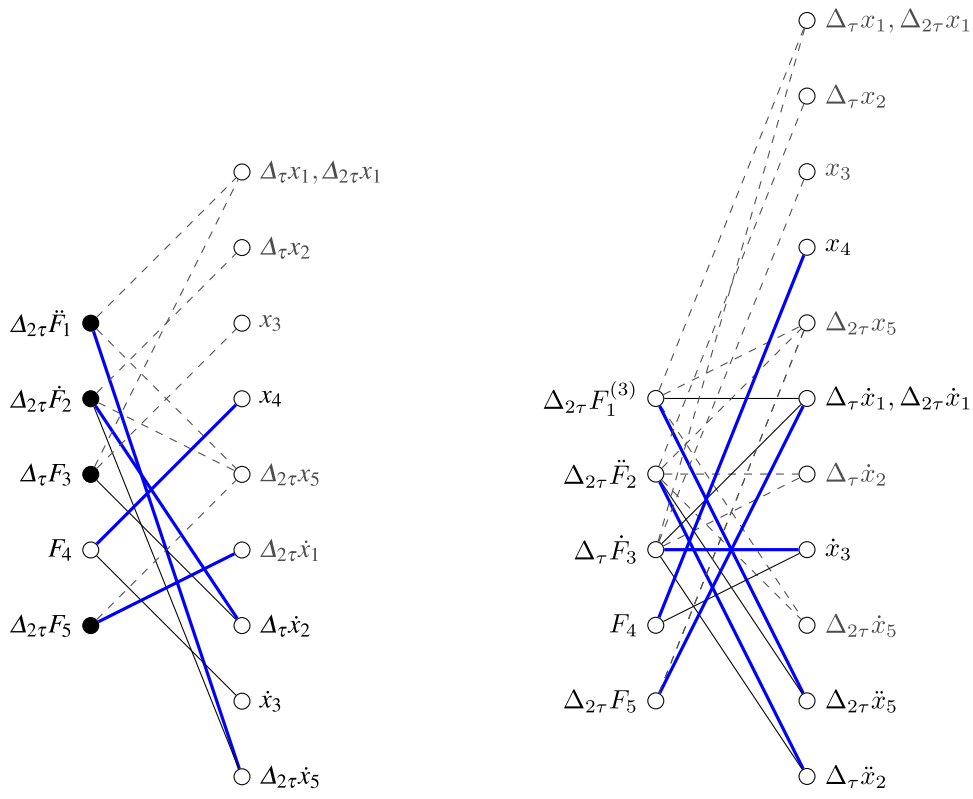
4.2 *Academic example 2*

In this example we finish the already partly executed Pantelides algorithm for DDAEs for (3.8). We have already seen in Example 3.15 that the shifting step results in equations $\Delta_\tau F_1$, $\Delta_\tau \dot{F}_2$ and $\Delta_\tau \dot{F}_3$. We detect in the trimmed linearization step that the only variable, depending on second or higher-order derivative is given by $\Delta_\tau \ddot{\mathtt{x}}_1$. Thus, $\Delta_\tau \ddot{\mathtt{x}}_1$ and $\Delta_\tau \dot{\mathtt{x}}_1$ are replaced by $\Delta_\tau \dot{\mathtt{x}}_4$ and $\Delta_\tau \mathtt{x}_1$, respectively. To add the equation $\Delta_\tau \mathtt{x}_4 = \Delta_\tau \dot{\mathtt{x}}_1$ to the graph the vertex $\Delta_\tau \dot{\mathtt{x}}_1$ is added to the graph. This results in the differentiation graph Fig. 8(a) where $\Delta_\tau F_1$ is exposed and thus differentiated. The resulting graph Fig. 8(b) has a maximal matching and thus Algorithm 5 terminates.

4.3 *Hybrid numerical-experimental model*

In earthquake engineering it is common to use a hybrid numerical-experimental approach to reduce unmanageable costs in testing complex dynamical systems (Bursi & Wagg, 2008). The complete model is subdivided in a numerical part and an experimental component, which interact in real-time. A state delay arises naturally by transferring numerical results via hydraulic actuators to the experiment (Horiuchi *et al.*, 1999). This delay essentially implies that the experiment is delayed compared to the numerical simulation. Here we discuss a coupled oscillator-pendulum system, which is analyzed in Kyrychko *et al.* (2006); Unger (2020) and depicted in Fig. 9.

$$
\begin{aligned}
m_1\ddot{\mathbf{y}}_1 + c\dot{\mathbf{y}}_1 + k\mathbf{y}_1 &= -2(\Delta_{-\tau}\lambda)(\Delta_{-\tau}\mathbf{y}_2 - \Delta_{-\tau}\mathbf{y}_1) - m_2\mathsf{g},\\
m_2\Delta_{-\tau}\ddot{x}_2 &= -2(\Delta_{-\tau}\lambda)\Delta_{-\tau}x_2,\\
m_2\Delta_{-\tau}\ddot{\mathbf{y}}_2 &= -2(\Delta_{-\tau}\lambda)(\Delta_{-\tau}\mathbf{y}_2 - \Delta_{-\tau}\mathbf{y}_1) - m_2\mathsf{g},\\
0 &= \Delta_{-\tau}x_2^2 + (\Delta_{-\tau}\mathbf{y}_2 - \Delta_{-\tau}\mathbf{y}_1)^2 - l^2,
\end{aligned}
$$



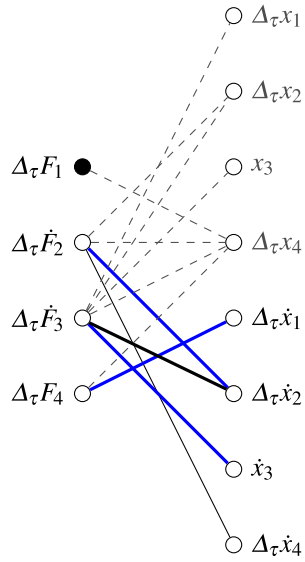
(a) Initial differentiation graph with assignment

(b) Differentiation graph after differentiating $\Delta_{2\tau}\ddot{F}_1$, $\Delta_{2\tau}\dot{F}_2$ and $\Delta_\tau F_3$

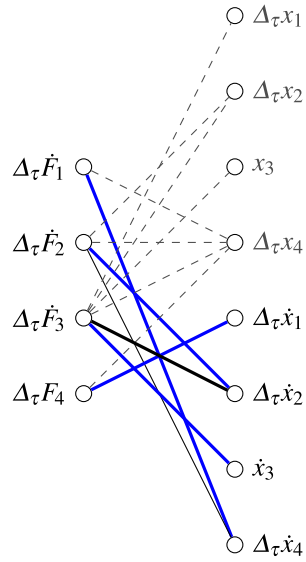
FIG. 7. Visualization of the differentiation step from Algorithm 5 for the DDAE (4.1).

with constants $m_1, c, k, m_2, l, \mathsf{g} > 0$. To obtain a reformulation of (4.2) which is suited for Algorithm 5 we rename λ to x_1, $\mathbf{y}_1$ to x_3, $\mathbf{y}_2$ to x_4 and rewrite (4.2) as first-order DAE given by

$$\begin{aligned}
\dot{x}_2 &= x_5,\\
\dot{x}_3 &= x_6,\\
\dot{x}_4 &= x_7,\\
M\dot{x}_6 + C\dot{x}_3 + Kx_3 &= -2\Delta_{-\tau}x_1(\Delta_{-\tau}x_4 - \Delta_{-\tau}x_3) - m\mathsf{g},\\
m\Delta_{-\tau}\dot{x}_5 &= -2\Delta_{-\tau}x_1\Delta_{-\tau}x_2,\\
m\Delta_{-\tau}\dot{x}_7 &= -2\Delta_{-\tau}x_1(\Delta_{-\tau}x_4 - \Delta_{-\tau}x_3) - m\mathsf{g},\\
0 &= \Delta_{-\tau}x_2^2 + (\Delta_{-\tau}x_4 - \Delta_{-\tau}x_3)^2 - l^2.
\end{aligned}$$

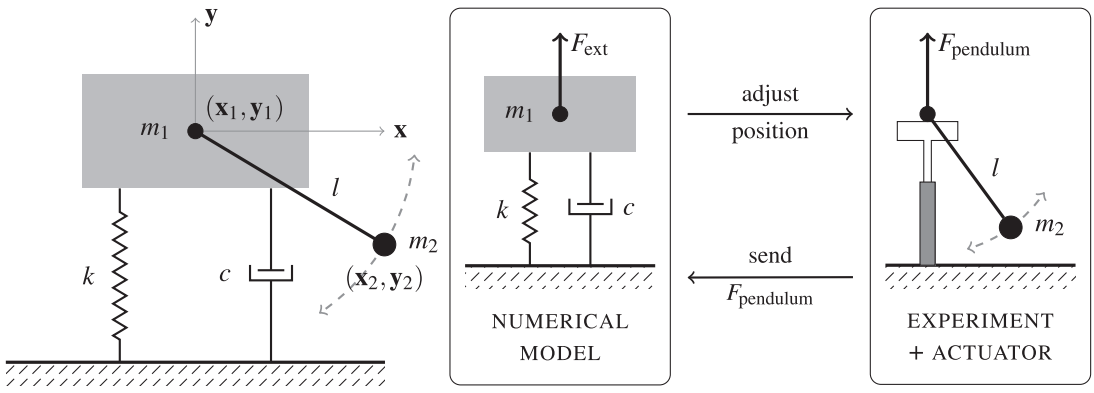


(a) Initial differentiation graph with assignment



(b) Differentiation graph after differentiating $\Delta_\tau F_1$ with assignments

FIG. 8. Visualization of the differentiation step from Algorithm 5 for the DDAE (3.8).



(a) Fully coupled system

(b) Hybrid numerical-experimental setup

FIG. 9. Real-time dynamic substructuring for a coupled oscillator-pendulum system.

Since the equations F_5, F_6 and F_7 only depend on past time points we shift all three equations in the shifting step of Algorithm 5 separately such that the differentiation during the shifting step yields no differentiation. No other equation is shifted. In the differentiation step we obtain the equations $\dot{F}_1$, $\dot{F}_2$, $\dot{F}_3$, F_4, $\Delta_\tau F_5$, $\Delta_\tau F_6$ and $\Delta_\tau \ddot{F}_7$, which is the expected result (Unger, 2020).

5. Termination of the new algorithm

Similarly as for the Pantelides algorithm we have to answer the question under which circumstances Algorithm 5 applied to the DDAE (1.1) terminates. As in the DAE case (cf. Theorem 2.11) we partition the set of variables Θ of (1.1) in such a way that variables that only differ by the level of differentiation or shifting are grouped together. In more detail we consider the equivalence relation

$$R_{\text{equal}} := \left\{ (\theta, \tilde{\theta}) \in \Theta \times \Theta \;\middle|\; \begin{array}{l} \text{there exist } k, \ell \in \mathbb{N},\ p, q \in \mathbb{N}_0 \text{ and } \xi \in \text{set}\,(x) \\ \text{such that } \theta = \Delta_{k\tau}\xi^{(p)},\ \tilde{\theta} = \Delta_{\ell\tau}\xi^{(q)} \end{array} \right\}, \tag{5.1}$$

which by abuse of notation is denoted as the corresponding equivalence relation (2.9) for DAEs. It is easy to see that if Algorithm 5 applied to the DDAE (1.1) terminates then each equation is matched to a different variable and thus (1.1) is structurally nonsingular with respect to Θ/R_{equal}. The following result shows that this is indeed also a sufficient condition.

THEOREM 5.1 Consider the DDAE (1.1) with set of variables Θ and equivalence relation R_{equal} as defined in (5.1). Algorithm 5 applied to (1.1) terminates if and only if (1.1) is structurally nonsingular with respect to Θ/R_{equal}.

We split the proof of Theorem 5.1 in two parts, namely the shifting and differentiation step, which are very similar to the original Pantelides algorithm. Thus, we can extend the termination proof presented in Pantelides (1988) to our setting. The main idea in Pantelides (1988) is to show that differentiating an MSS subset (see Definition 2.8) renders it structurally nonsingular and a structurally nonsingular subset remains structurally nonsingular after differentiation. Let us first show that under the assumptions of Theorem 5.1 the shifting step terminates.

LEMMA 5.2 Consider the DDAE (1.1) with set of variables Θ and equivalence relation R_{equal} as defined in (5.1). The shifting step of Algorithm 5 applied to (1.1) terminates if and only if (1.1) is structurally nonsingular with respect to Θ/R_{equal}.

Proof. Recall that in the shifting step we group variables that are shifting similar, i.e., we work with the equivalence relation R_{s} from Theorem 3.5. By virtue of Theorem 2.10 any subset of equations of the DDAE (1.1) that Algorithm 5, or more precisely Algorithm 4, identifies to be shifted is MSS with respect to Θ/R_{s}. Suppose that the subset

$$\widehat{F}(t, \tilde{x}, \dot{\bar{x}}, \Delta_{-\tau}\widehat{x}) = 0, \tag{5.2}$$

with $\text{set}(\tilde{x}, \bar{x}, \widehat{x}) \subseteq \text{set}(x)$ of the DDAE (1.1) is such a set. Note that here the sets

$$\text{set}(\tilde{x}),\ \text{set}(\bar{x}) \text{ and } \text{set}(\widehat{x})$$

are not necessarily disjoint. More precisely decompose $\widehat{x} = (\widehat{x}_1, \widehat{x}_2)$ such that

$$\text{set}(\widehat{x}) = \text{set}(\widehat{x}_1) \dot{\cup} \text{set}(\widehat{x}_2), \qquad \text{set}(\widehat{x}_2) \cap \text{set}(\bar{x}, \tilde{x}) = \emptyset \qquad \text{and} \qquad \text{set}(\widehat{x}_1) \subseteq \text{set}(\bar{x}, \tilde{x}).$$

Let us assume first that the DDAE (1.1) is structurally nonsingular with respect to Θ/R_{equal}. This implies that $\text{set}(\widehat{x}_2) \neq \emptyset$. Shifting (5.2) results in the equation $\Delta_\tau \widehat{F}$ given by

$$\widehat{F}(t+\tau, \Delta_\tau \tilde{x}, \Delta_\tau \dot{\tilde{x}}, \widehat{x}) = 0.$$

By construction we have

$$\begin{aligned}\left|\text{set}(\Delta_\tau \widehat{F})\right| = \left|\text{set}(\widehat{F})\right| &\leqslant \left|\text{set}\left(\tilde{x}, \dot{\tilde{x}}, \Delta_{-\tau}\widehat{x}\right)/R_{\text{equal}}\right| = \left|\text{set}\left(\tilde{x}, \dot{\tilde{x}}, \Delta_{-\tau}\widehat{x}_2\right)/R_{\text{equal}}\right| \\ &= \left|\text{set}\left(\tilde{x}, \dot{\tilde{x}}, \Delta_{-\tau}\widehat{x}_2\right)/R_{\text{shift}}\right| = \left|\text{set}\left(\Delta_\tau \tilde{x}, \Delta_\tau \dot{\tilde{x}}, \widehat{x}_2\right)/R_{\text{shift}}\right|\end{aligned}$$

showing that $\Delta_\tau \widehat{F}$ is structurally nonsingular with respect to the equivalence relation R_{shift}. Note that this argument is easily extended to subsets of the general equation (3.5) and we conclude that whenever an MSS subset of equations is shifted during the shifting step it becomes structurally nonsingular. In Algorithm 4 we do not only replace $\widehat{F}$ with $\Delta_\tau \widehat{F}$ but may have to differentiate some of the equations. However, Proposition 3.8 implies that this does not affect the shifting graph and hence does not effect the structural nonsingularity. In order to show that the shifting step terminates it suffices to show that any subset $\text{set}(\overline{F}) \subseteq \text{set}(F)$ that is disjoint from $\text{set}(\widehat{F})$ and is structurally nonsingular before we shift (5.2) remains structurally nonsingular after shifting. The only possibility for $\text{set}(\overline{F})$ to become structurally singular is if a subset of $\text{set}(\overline{F})$ has a matching edge to some vertex in $\text{set}(\tilde{x}, \dot{\tilde{x}})/R_s$. This implies that there is an alternating path from the exposed equation in $\widehat{F}$ to one of the equations in $\text{set}(\overline{F})$, a contradiction to $\text{set}(\widehat{F}) \cap \text{set}(\overline{F}) = \emptyset$. We conclude that the shifting step terminates.

Conversely, assume that (1.1) is structurally singular with respect to Θ/R_{equal}. Then at some point Algorithm 5 identifies a subset (5.2) of (1.1) that is structurally singular with respect to Θ/R_{equal}. In this case we have $\text{set}(\widehat{x}_2) = \emptyset$ and thus

$$\left|\text{set}(\widehat{F})\right| > \left|\text{set}\left(\tilde{x}, \dot{\tilde{x}}, \Delta_{-\tau}\widehat{x}\right)/R_{\text{equal}}\right| = \left|\text{set}\left(\tilde{x}, \dot{\tilde{x}}, \Delta_{-\tau}\widehat{x}_2\right)/R_{\text{shift}}\right| \geqslant \left|\text{set}\left(\tilde{x}, \dot{\tilde{x}}\right)/R_{\text{shift}}\right|.$$

In this case shifting does not increase the set of equivalence classes and thus the set (5.2) is still structurally singular with respect to the relation R_{shift} and thus results in an infinite loop. □

To show that also the differentiation step in Algorithm 5 terminates we want to apply Theorem 2.11. For this we have to show that the set of equations that we obtain after applying the shifting step and the trimmed linearization is structurally nonsingular with respect to $\widetilde{\Theta}/R_{\text{equal}}$ with

$$\widetilde{\Theta} := \left\{\theta \in \Theta \mid (\theta, \theta) \in R_{\text{diff}}\right\}. \tag{5.3}$$

Note that $\widetilde{\Theta}$ does not include variables depending on $t-\tau$ and hence $\widetilde{\Theta}/R_{\text{equal}}$ is different from Θ/R_{equal}.

LEMMA 5.3 Suppose that (1.1) is structurally nonsingular with respect to Θ/R_{equal}. Let $G = (V_{\text{E}} \dot{\cup} V_{\text{V}}, E)$ with

$$V_{\text{E}} = \left\{\Delta_{\mathtt{m}_1\tau}\mathtt{F}_1^{(\ell_1)}, \ldots, \Delta_{\mathtt{m}_{\mathtt{M}}\tau}\mathtt{F}_{\mathtt{M}}^{(\ell_{\mathtt{M}})}\right\}$$

denote the graph that results from applying the shifting step and the trimmed linearization of Algorithm 5 to (1.1). Define the set of equations

$$0 = \overline{F}(\bar{t}, \bar{x}, \dot{\bar{x}}, \Delta_{-\tau} x) = [\overline{F}_i]_{i=1,\dots,M}(\bar{t}, \bar{x}, \dot{\bar{x}}, \Delta_{-\tau} x) \tag{5.4}$$

with $m := \max_i(\mathbb{m}_i)$, $\overline{\xi} = \left[\Delta_{0\tau}\xi \ \dots \ \Delta_{m\tau}\xi\right]$ for $\xi \in \{t, x\}$ by $\overline{F}_i := \Delta_{\mathbb{m}_i} F_i$. Then (5.4) is structurally nonsingular with respect to $\widetilde{\Theta}/R_{\text{equal}}$ with $\widetilde{\Theta}$ defined in (5.3).

Proof. Since (1.1) is structurally nonsingular with respect to Θ/R_{equal} Algorithm 5 up to Line 6 terminates (cf. Lemma 5.1, Theorem 3.18). Using Theorem 3.18 we can assign each equation $\overline{F}_i$ for $i = 1, \dots M$ to a highest shift. In other words we can assign each equation to a variable which is shifted at least 0 times which yields directly the structural nonsingularity of $\overline{F}$ with respect to $\widetilde{\Theta}/R_{\text{equal}}$. □

Combining the previous results we are now able to proof Theorem 5.1.

Proof of Theorem 5.1. If (1.1) is structurally nonsingular with respect to Θ/R_{equal} Algorithm 5 up to Line 6 terminates (cf. Lemma 5.1, Theorem 3.18). Since the resulting shifted equations (5.4) are structurally nonsingular with respect to $\widetilde{\Theta}/R_{\text{equal}}$ we can use Theorem 2.11 to conclude that Algorithm 5 terminates. The converse direction follows from Lemma 5.1. □

We conclude our analysis by investigating in which cases Algorithm 5 determines that no equation needs to be shifted. The following result shows that this is the case if the DDAE (1.1) is structurally nonsingular with respect to the set of equivalence classes that are obtained by using the relation R_{equal} with a restricted set of variables.

THEOREM 5.4 If the set of equations (1.1) is structurally nonsingular with respect to $\overline{\Theta}/R_{\text{equal}}$ with

$$\overline{\Theta} := \{\xi \in \text{set}(x, \dot{x}) \mid \xi \text{ appears in} F\}$$

then Algorithm 5 applied to (1.1) terminates and determines that no equation is shifted. In this case also Algorithm 2 applied to (1.1) where we replace $\Delta_{-\tau} x$ with a function parameter λ terminates and the resulting graphs of both algorithms are isomorphic.

Proof. First observe that the structural nonsingularity of (1.1) with respect to $\overline{\Theta}/R_{\text{equal}}$ implies structural nonsingularity with respect to Θ/R_{equal}. Thus, Theorem 5.1 ensures that Algorithm 5 terminates if applied to (1.1). Consider a subset

$$\widehat{F}(t, \eta, \gamma, \dot{\gamma}, \dot{\mu}, \Delta_{-\tau}\widehat{x}) = 0$$

of the DDAE (1.1) with $\text{set}(\eta) \dot{\cup} \text{set}(\gamma) \dot{\cup} \text{set}(\mu) \subseteq \text{set}(x)$ and $\text{set}(\widehat{x}) \subseteq \text{set}(x)$. The structural nonsingularity with respect to $\overline{\Theta}/R_{\text{equal}}$ implies

$$\left|\text{set}(\widehat{F})\right| \leqslant \left|\text{set}\left(\eta, \gamma, \dot{\gamma}, \dot{\mu}\right)/R_{\text{equal}}\right| = \left|\text{set}\left(\eta, \gamma, \dot{\gamma}, \dot{\mu}\right)/R_{\text{shift}}\right|,$$

which is the structural nonsingularity of $\widehat{F}$ after deleting variables with lower shifts in the shifting graph. Thus, no equation is shifted. We conclude that Algorithm 5 can be reduced to the differentiation step.

Within the differentiation step, Algorithms 2 and 4 coincide and thus Algorithm 2 terminates with the same result as Algorithm 5. □

We conclude our analysis by emphasizing that the Pantelides algorithm for DDAEs suffers from the same problems as the original Pantelides algorithm (see for instance Example 2.12 and the discussion thereafter). In particular there is no guarantee that the correct number of differentiations and shifts is identified.

EXAMPLE 5.5 Even though the solution of the DDAE

$$0 = x_4 + \Delta_{-\tau} x_1 + f_1, \quad \dot{x}_2 = x_1 + f_1, \quad 0 = x_2 + x_4 + f_3, \quad 0 = x_2 + \Delta_{-\tau} x_3 + f_4$$

depends on the shifted and differentiated equations F_1, F_3, F_4, the Pantelides algorithm for DDAEs determines that we have to shift equations F_1, F_3 and F_4 but do not have to differentiate any equation.

6. Summary

We have presented a method (Algorithm 5) to determine which equations of the DDAE (1.1) need to be shifted and which equations need to be differentiated. The algorithm extends the Pantelides algorithm (Pantelides, 1988) for DAEs to the DDAE case. The main idea that enables the generalization is the introduction of equivalence classes in the bipartite graph associated with the DDAE (1.1). For further details see Definition 2.1. The Pantelides algorithm for DDAEs is divided into the shifting step and the differentiation step. We prove (Proposition 3.8) that differentiation does not affect the shifting graph and hence start with the shifting step. It turns out that already in the shifting step we have to differentiate some equations. To obtain the required number of differentiations for each equation we may have to solve several linear systems, see Section 3.3 for further details. We presented a necessary and sufficient condition for the termination of Algorithm 5 in Theorem 5.1. We foresee that we can combine our framework with a dummy derivative approach (Mattsson & Söderlind, 1993) and an algebraic approach (Scholz & Steinbrecher, 2016a) such that Algorithm 5 can be used with numerical time integration methods.

Since our framework builds upon the Pantelides algorithm it has similar strengths and weaknesses. One of the big advantages of our algorithm (in contrast to the algebraic index reduction procedure in Ha & Mehrmann, 2016) is that the underlying graph-theoretical tools do not suffer from numerical rank decisions and can be computed efficiently also in a large-scale setting. On the other hand, our algorithm is not invariant under equivalence transformations and thus may fail to determine the correct number of shifts and differentiations. In the DAE case this fact is accounted for in the Σ-method (Pryce, 2001; Pryce *et al.*, 2015) by including a success check. It is ongoing research to develop such a success check for DDAEs.

Acknowledgements

The work of IA is supported by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation)—384950143/GRK2433. The work of BU is supported by the DFG Collaborative Research Center 910 *Control of self-organizing nonlinear systems: Theoretical methods and concepts of application*, project number 163436311.

References

Altmann, R., Maier, R. & Unger, B. (2019) Semi-explicit discretization schemes for weakly-coupled elliptic-parabolic problems. ArXiv e-print 1909.03497.

Ascher, U. M. & Petzold, L. R. (1995) The numerical solution of delay-differential-algebraic equations of retarded and neutral type. *SIAM J. Numer. Anal.*, **32**, 1635–1657.

Bellen, A., Guglielmi, N. & Zennaro, M. (1999) On the contractivity and asymptotic stability of systems of delay differential equations of neutral type. *BIT Numer. Math.*, **39**, 1–24.

Bellen, A., Guglielmi, N. & Zennaro, M. (2000) Numerical stability of nonlinear delay differential equations of neutral type. *J. Comput. Appl. Math.*, **125**, 251–263.

Bellen, A. & Zennaro, M. (2003) *Numerical Methods for Delay Differential Equations*. Oxford, United Kingdom: Clarendon Press, p. 410.

Bellman, R. & Cooke, K. (1963) *Differential-Difference Equations*. New York: Academic Press.

Borsche, R., Kocoglu, D. & Trenn, S. (2019) *A distributional solution framework for linear hyperbolic PDEs coupled to switched DAEs. Technical Report*. Bernoulli Institute for Mathematics, CS and AI, University of Groningen.

Brenan, K., Campbell, S. & Petzold, L. (1996) *Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations*. Philadelphia, PA: SIAM.

Bursi, O. & Wagg, D. (eds.) (2008) *Modern Testing Techniques for Structural Systems*. Vienna: Springer.

Campbell, S. L. (1980) Singular linear systems of differential equations with delays. *Appl. Anal.*, **11**, 129–136.

Campbell, S. L. (1987) A general form for solvable linear time varying singular systems of differential equations. *SIAM J. Math. Anal.*, **18**, 1101–1115.

Campbell, S. L. (1995) Nonregular 2D descriptor delay systems. *IMA J. Math. Control Inform.*, **12**, 57–67.

Duff, I. S. (1981) On algorithms for obtaining a maximum transversal. *ACM Trans. Math. Softw.*, **7**, 315–330.

Erneux, T. (2009) *Applied Delay Differential Equations, vol. 4*. New York: Springer, p. 241.

Fosong, E., Schulze, P. & Unger, B. (2019) From time-domain data to low-dimensional structured models. ArXiv e-print 1902.05112.

Gantmacher, F. R. (1959) *The Theory of Matrices*, vol. 2. New York, NY, USA: Chelsea Publishing Company.

Gluesing-Luerssen, H. (2002) *Linear Delay-Differential Systems with Commensurate Delays: An Algebraic Approach*. Berlin: Springer, Berlin.

Ha, P., Mehrmann, V. & Steinbrecher, A. (2014) Analysis of linear variable coefficient delay differential-algebraic equations. *J. Dynam. Differential Equations*, **26**, 889–914.

Ha, P. (2015) *Analysis and numerical solutions of delay differential-algebraic equations*. Dissertation, Technische Universität Berlin.

Ha, P. (2018) Spectral characterizations of solvability and stability for delay differential-algebraic equations. *Acta Math. Vietnamica*, **43**, 715–735.

Ha, P. & Mehrmann, V. (2012) Analysis and reformulation of linear delay differential-algebraic equations. *Electron. J. Linear Algebr.*, **23**, 703–730.

Ha, P. & Mehrmann, V. (2016) Analysis and numerical solution of linear delay differential-algebraic equations. *BIT Numer. Math.*, **56**, 633–657.

Hale, J. K. & Verduyn Lunel, S. M. (1993) *Introduction to Functional Differential Equations*. New York: Springer.

Horiuchi, T., Inoue, M., Konno, T. & Namita, Y. (1999) Real-time hybrid experimental system with actuator delay compensation and its application to a piping system with energy absorber. *Earthq. Eng. Struct. Dyn.*, **28**, 1121–1141.

Jungnickel, D. (2013) *Graphs, Networks and Algorithms*, 4th edn. Berlin: Springer.

Karmarkar, N. (1984) A new polynomial-time algorithm for linear programming. *Combinatorica*, **4**, 373–395.

Kunkel, P. & Mehrmann, V. (1998) Regular solutions of nonlinear differential-algebraic equations and their numerical determination. *Numer. Math.*, **79**, 581–600.

KUNKEL, P. & MEHRMANN, V. (2006) *Differential-Algebraic Equations. Analysis and Numerical Solution*. Zürich, Switzerland: European Mathematical Society.

KYRYCHKO, Y. N., BLYUSS, K. B., GONZALEZ-BUELGA, A., HOGAN, S. J. & WAGG, D. J. (2006) Real-time dynamic substructuring in a coupled oscillator - pendulum system. *Proc. R. Soc. A*, **462**, 1271–1294.

MATTSSON, S. E. & SÖDERLIND, G. (1993) Index reduction in differential-algebraic equations using dummy derivatives. *SIAM J. Sci. Comput.*, **14**, 677–692.

MEHRMANN, V. & SHI, C. (2006) Transformation of high order linear differential-algebraic systems to first order. *Numer. Algorithms*, **42**, 281–307.

NELDER, J. A. & MEAD, R. L. (1965) A simplex method for function minimization. *Comput. J.*, **7**, 308–313.

PANTELIDES, C. C. (1988) The consistent initialization of differential-algebraic systems. *SIAM J. Sci. Stat. Comput.*, **9**, 213–231.

PETZOLD, L. (1982) Differential/algebraic equations are not ODE's. *SIAM J. Sci. Stat. Comput.*, **3**, 367–384.

PRYCE, J. D. (2001) A simple structural analysis method for DAEs. *BIT Numer. Math.*, **41**, 364–394.

PRYCE, J. D., NEDIALKOV, N. S. & TAN, G. (2015) DAESA—a Matlab tool for structural analysis of differential-algebraic equations: theory. *ACM Trans. Math. Softw.*, **41**, 9:1–9:20.

REISSIG, G., MARTINSON, W. S. & BARTON, P. I. (2000) Differential–algebraic equations of index 1 may have an arbitrarily high structural index. *SIAM J. Sci. Comput.*, **21**, 1987–1990.

RICHARDSON, D. (1968) Some undecidable problems involving elementary functions of a real variable. *J. Symb. Log.*, **33**, 514–520.

SCHOLZ, L. (2011) A derivative array approach for linear second order differential-algebraic systems. *Electron. J. Linear Algebra*, **22**, 310–347.

SCHOLZ, L. & STEINBRECHER, A. (2016a) Regularization of DAEs based on the signature method. *BIT Numer. Math.*, **56**, 319–340.

SCHOLZ, L. & STEINBRECHER, A. (2016b) Structural-algebraic regularization for coupled systems of DAEs. *BIT Numer. Math.*, **56**, 777–804.

SCHULZE, P., UNGER, B., BEATTIE, C. & GUGERCIN, S. (2018) Data-driven structured realization. *Linear Algebra Appl.*, **537**, 250–286.

SCHULZE, P. & UNGER, B. (2016) Data-driven interpolation of dynamical systems with delay. *Systems Control Lett.*, **97**, 125–131.

STEINBRECHER, A. (2006) Numerical solution of quasi-linear differential-algebraic equations and industrial simulation of multibody systems. Ph.D. Thesis. Institut für Mathematik, Technische Universität Berlin.

TAN, G., NEDIALKOV, N. S. & PRYCE, J. D. (2017) Conversion methods for improving structural analysis of differential-algebraic systems. *BIT Numer. Math.*, **57**, 845–865.

TRENN, S. & UNGER, B. (2019) Delay regularity of differential-algebraic equations. *Proceedings of the 58th IEEE Conference on Decision Control (CDC) 2019, Nice, France, IEEE pp. 989–994*.

UNGER, B. (2018) Discontinuity propagation in delay differential-algebraic equations. *Electron. J. Linear Algebra*, **34**, 582–601.

UNGER, B. (2020) Delay differential-algebraic equations in real-time dynamic substructuring. ArXiv e-print 2003.10195.

WEST, D. B. (2001) *Introduction to Graph Theory*, 2nd edn. Upper Saddle River: Prentice Hall.