symmetry

MDPI

Article

Recomputing Causality Assignments on Lumped Process Models When Adding New Simplification Assumptions

Antonio Belmonte, Juan Garrido *, Jorge E. Jiménez and Francisco Vázquez

Department of Computer Science and Numerical Analysis, University of Cordoba, Campus de Rabanales, 14071 Cordoba, Spain; chgupo@gmail.com (A.B.); jjimenez@uco.es (J.E.J.); fvazquez@uco.es (F.V.)

* Correspondence: juan.garrido@uco.es; Tel.: +34-957-218329

Received: 16 March 2018; Accepted: 5 April 2018; Published: 9 April 2018

Abstract: This paper presents a new algorithm for the resolution of over-constrained lumped process systems, where partial differential equations of a continuous time and space model of the system are reduced into ordinary differential equations with a finite number of parameters and where the model equations outnumber the unknown model variables. Our proposal is aimed at the study and improvement of the algorithm proposed by Hangos-Szerkenyi-Tuza. This new algorithm improves the computational cost and solves some of the internal problems of the aforementioned algorithm in its original formulation. The proposed algorithm is based on parameter relaxation that can be modified easily. It retains the necessary information of the lumped process system to reduce the time cost after introducing changes during the system formulation. It also allows adjustment of the system formulations that change its differential index between simulations.

Keywords: structural analysis; bipartite graph; bipartite matching; model simplification assumptions; over-constrained systems

1. Introduction

There are well known reasons for modeling and simulating in engineering: testing and optimizing the design of a system before its construction, avoiding design errors, enhancing quality, reducing costs, evaluating performances, predicting responses, and so on [1]. Over the past 30 years, several modeling tools and environments have been developed [2], such as Modelica [3], Ascend [4], EcosimPro [5] and so on. Most of them are concerned with the model equations describing the system dynamics, which can be directly introduced by the user or automatically generated by the tool from some previous process description or schematic. Many physical systems are often modeled by lumped process models, particularly in the chemical process domain, where partial differential equations of a continuous time and space model are reduced into a finite set of differential-algebraic equations (DAEs) [6]. The general form of a DAE system has the following expression:

$$\begin{cases} f(x', x, y, t) = 0 \\ g(x, y, t) = 0 \end{cases}, \tag{1}$$

where the first equation is the differential part of the system and the second one is the algebraic part [7]. The x variables are fitted to a vector of states of $\Re^n$, where x' are their time-derivatives and t, the time, is the independent variable. The algebraic variables y, which are not derived, belong to $\Re^m$. Thus, the functions f and g must be defined in some regions such as $f{:}\Omega_f \subseteq \Re^{2n+m+1} \rightarrow \Re^n$ and $g{:}\Omega_g \subseteq \Re^{n+m+1} \rightarrow \Re^m$. The set of equations in (1) must have n + m unknowns, x and y, with n + m equations in order to achieve a system with zero degrees of freedom for solvability. Modeling tools sort the equations and identify the sequence for how variables will be solved during simulation. The incidence matrix denotes which variables appear in which equation. For each variable, the matching solution determines the corresponding equation that solves this variable, that is, the causality. The Block Lower Triangular (BLT) transformation is the classical approach for this step [8] together with Tarjan's algorithm [9]. For semi-explicit index-1 DAE models, such as (2), causalization transforms equations into assignments. In the case of high-index DAE systems, index reduction, usually by the Pantelides' algorithm [10], must be performed to achieve an index-1 DAE or an explicit ODE model prior to matching [11]. However, dynamic simulation frequently results in numerical difficulties in the DAE resolution that can arise as a consequence of the differential index and stiffness of the DAE [12,13]. At the modeling level, an index problem can sometimes be avoided by using an alternative model formulation [14]. Therefore, the analysis of the solvability properties is basic for dynamic simulations.

$$\begin{cases} x' = f(x, y, t) \\ g(x, y, t) = 0 \end{cases} \tag{2}$$

In the modeling process, the DAE system is usually parameterized by the modeler with some values that are assumed to be known during the simulation. These are the design or input variables which influence the system behavior. The selection of the design variables depends highly on the problem or application area and it may greatly influence the mathematical properties of the model equation such as the differential index. Structural analysis methods can use the visual interface of modeling tools to analyze equation-variable patterns to choose the appropriate variables that have zero degrees of freedom or re-sort the DAEs to improve resolution methods [15,16]. In [17], the effect of modeling decisions on structural solvability are studied, and it was found that a change in the system specification can transform an index-1 model into a high-index system. Structural analysis methods traditionally use matrix representation and linear algebraic tools. More recent developments used a graph and combinatorial approach where the problem is addressed as a maximum weight assignment problem [18–20].

In the lumped process, modeling assumptions can be translated into additional mathematical relationships or constraints between model variables, equations or parameters. Adding a new modeling assumption to the system involves an additional equation, and consequently, the set of equations changes and becomes over-constrained. To recover the property of zero degrees of freedom and solve the system, two options can be chosen: removing any equation from the initial systems [21] or relaxing design variables or system parameters. As discussed later, the first option may not be a good alternative due to the logic contradictions of the initial system formulation. The second one converts the selected design variable into a new algebraic unknown to obtain a new system with zero degrees of freedom. The user must decide which design parameters are relaxed and thus, depending on this selection, different solutions can be obtained. This second method does not remove any equation from the original system, only the specification equation corresponding to the relaxed design parameter. In addition, assumption transformations may change the variable structure of the model and induce changes in the causality assignments, which need to be recomputed. Therefore, it is also important to develop algorithms that minimize the computational cost of introducing these assumptions in the initial phases of the modeling tools.

No study has been developed focused on examining the overall effect of modeling assumptions in all modeling processes. In [13], a formal representation of these assumptions is proposed to deduce basic properties of the resultant equations. The simplification assumptions, applied during the modeling phase, may seriously affect the differential index of lumped process models. In [21], adding new model assumptions and their corresponding effects on the differential index of a DAE system are studied by graph transformation. Two algorithms are proposed to follow the changes in the variable-equation assignments and the differential index. However, as will be shown later, Hangos' method may result in contradictions and inconsistent modeling in some cases and therefore, it needs to be improved. In addition, its computational cost can be reduced.

This work proposes such improvement of Hangos' method using the option of relaxing system parameters in the over-constrained system after adding new modeling assumptions. A polynomial time incremental algorithm is developed based on graph theory to analyze the change of differential index in the model simplification phase of the modeling process and reduce the time cost of these assumptions. The proposal can be implemented using either the Bellman-Ford or Dijkstra algorithms together with a representation of the information by minimum heaps or linked list [22,23]. According to that, the best algorithm has an almost linear time complexity if the number of assumption equations added to the system is small. The paper is structured as follows: a brief background on graph theory is given in Section 2. Section 3 describes Hangos' method and demonstrates its limitations by means of an example. The basis of the new proposed method is introduced in Section 4 using the same illustrative example. Then, the proposed algorithm is developed and discussed. Conclusions are summarized in Section 5.

2. Bipartite Graph Preliminzaries

Next, the minimal background on bipartite graphs used to perform the proposed algorithm is discussed. A bipartite graph, denoted as B(L, R, E), is an undirected graph where the vertex set is decomposed into two disjointed sets, L and R, such that every edge $e \in E$ has a vertex in L and R. In this work, DAE systems are represented as bipartite graphs where the first set consists of equation vertices and the second one consists of variable vertices. A matching in a bipartite graph is a set $M \subseteq E$, such that for every vertex v there is an edge $e \in M$ incident to v at most; no edge in a matching can share a node. The cardinality of matching M is the number of edges covered by M. A maximum matching is a matching of maximum cardinality. A perfect matching M is a matching in which for every vertex v there is only one edge of M incident to v. Using graph theory, the equation-variable assignment problem in a DAE system can be done executing a maximum matching algorithm. If the maximum matching association includes all variables and equations (a perfect matching), the system is structurally nonsingular.

Given the equation system in (3), the corresponding bipartite graph is shown in Figure 1a where the edges which are part of a perfect matching are in bold. The form of the equations f_i in (3) is irrelevant. For structural analysis, only the equation-variable relationship is considered. This optimal assignment is not unique. From this initial matching, an alternative representation, the directed graph, can be derived. In a directed graph C(U, F) induced by the bipartite graph B(L, R, E), the set of nodes is $\{u_1, \dots, u_n\}$ and (u_i, u_j) is a directed edge if and only if $i \neq j$ and $(l_i, r_j) \in E$ [24]. The directed graph obtained from the bipartite graph of Figure 1b is also shown on the right.

$$\begin{aligned} f_1(y_1, y_2, y_4) &= 0 \\ f_2(y_2, y_3) &= 0 \\ f_3(y_2, y_3, y_4) &= 0 \\ f_4(y_2, y_4) &= 0 \end{aligned} \tag{3}$$

An alternating path in a bipartite graph B(L, R, E), with a matching M, is a path that interchanges the edges of M and M^c. A vertex v is free in M if there is no edge of M incident to v. An augmented path in M is an alternating path that starts and ends in a free vertex. The symmetric difference of two sets, A and B, is defined as $A \Delta B = (A - B) \cup (B - A)$. The cardinality of a set A is denoted as $|A|$. A weight function in a graph G(V,E) is a function $w : E \rightarrow \Re$. The weight of $C \subseteq E$ is defined as $w(C) = \sum_{e \in C} w(e)$. If p is a path in a bipartite graph with weight function w, the cost of the path p is defined as $\alpha(p) = \sum_{e \in M \cap p} w(e) - \sum_{e \in M^c \cap p} w(e)$. If M is a matching and p is an augmented path into M, then, $w(M \Delta p) = w(M) - \alpha(p)$.

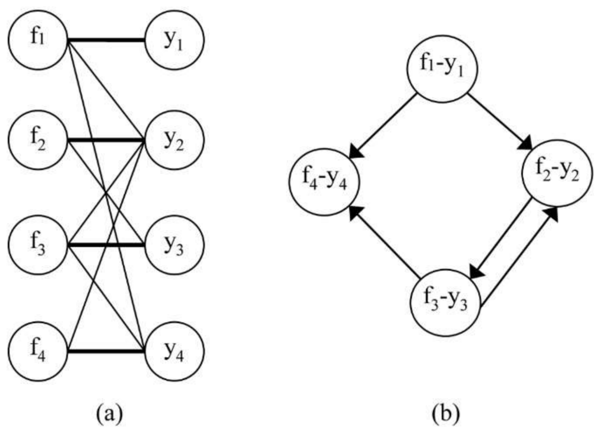

Figure 1. A bipartite graph (**a**) and the corresponding directed graph (**b**).

The problems of finding maximum matching and perfect matching if one exists are two of the most fundamental algorithmic graph problems. Several time polynomial algorithms have been developed. The first developments run in time $O(n^{5/2})$ [25–27]. More recent proposals reduce the time complexity [24,28,29]. Most of them are based on a search for augmenting paths, such as the depth first search (DFS), the breadth first search (BFS) or a combination of both strategies. There also another approaches which employ a push-relabel strategy designed for maximum flow problems [30].

The proposed method is based on a search for augmenting paths. The following Theorems 1–3 establish the importance of the augmented graph in bipartite graphs [23,27,31], and allow the definition of the following Algorithm 1 to compute the matching of maximum weight onto a bipartite graph B(L, R, E) with weight function w [32].

Theorem 1. *If M is a matching and p is an augmented path into M, then M Δ p is a matching of cardinality* $|M| + 1$.

Theorem 2. *M is a maximum matching if and only if M does not have an augmented path.*

Theorem 3. *If B(L, R, E) is a bipartite graph with weight function w, M is a matching of maximum weight for all matchings of cardinality* $|M|$ *and p is an augmented path of minimum cost, then, M Δ p is a matching of maximum weight among all the matchings of cardinality* $|M| + 1$. *Moreover,* $w(M \Delta p) = w(M) - \alpha(p)$.

First, a directed bipartite graph is calculated using the weight function z, and two new free vertices s and t are added. In step 3, the augmented path of minimum cost in G_M can be computed by finding a shortest path from s to t in G_M. There are several classic algorithms for this calculation with polynomial time [22,23]. The Bellman-Ford algorithm computes the shortest path in time $O(V \cdot E)$ which yields an overall running time of $O(V^2 \cdot E)$. If there are no negative weights, the Dijkstra's algorithm can be used improving the time cost. Using an implementation with Fibonacci heaps, the overall cost is $O(V \cdot E + V^2 \cdot \log(V))$.

Algorithm 1. *Computing matching of maximum weight on a bipartite graph.*

```
Input: A bipartite graph $B(L, R, E)$ and weight function $w$.
Output: Matching of maximum weight.
1: Construction of a direct graph $G_M(V, E_M)$ with function weight $z$.
    1.1: $V = L \cup R$
    1.2: $E_M$ is defined by:
        If the undirected edge $(l, r) \in M^c$ then
            the directed edge $(l, r) \in E_M$.
        else if the undirected edge $(l, r) \in M$ then
            the directed edge $(r, l) \in E_M$.
        end if
    That is, the set of edges $E_M$ is: $E_M = \{(l, r)/(l, r) \in M^c\} \cup \{(r, l)/(l, r) \in M\}$
    1.3: The weight function $z$ is defined:
        If $(l, r) \in E_M$ then
            $z(l, r) = w(l, r)$.
        else if $(r, l) \in E_M$ then
            $z(r, l) = - w((l, r)$
        end if
    1.4: Append a new vertex $s$ linked to all free vertexes of $L$.
    1.5: Append a new vertex $t$ which is the destination of all free vertexes of $R$.
2: $M := \emptyset$
3: while there exists an augmented path $p$ of minimum cost into $G_M$ do
    3.1: $M := M \Delta p$
    3.2: Apply steps 1.2 and 1.3 to $G_M$.
  end while
```

3. Hangos' Method for New Model Assumptions

In this section, Hangos' method for dealing with new model assumptions is briefly described [21]. Then, its limitations are illustrated by means of a process example. The Hangos method transforms a semi-explicit DAE in the form of (2) into an integral system of equations as follows:

$$\begin{cases} u = f(x, y, t, p) \\ x = x_0 + \int_0^t u \cdot dt \\ g(x, y, t, p) = 0 \\ \\ p = spec \\ x_0 = spec \end{cases} \tag{4}$$

where p is the vector of design parameters (or design variables) and x_0 is the initial condition vector of x. They must be specified by the user. Note that the first equation in (2) has been changed to the two first equations in (4), and a new vector of variables u has been introduced. The parameters p and x_0 are specified as constants in the two last equations of system (4), that is, specification equations. For each new assumption equation added to the system, Hangos' method removes another one from the initial system (4) to obtain a new zero degrees of freedom system. Hangos removes algebraic equations, such as the third one in (4), or specification equations. The assumption equations fit into one of the two following types: a constant algebraic variable y_i or a steady state differential variable, that is, $u_i = 0$. In [21], two algorithms are introduced to obtain a new matching transformed system from the initial assignment of equation-variable when new assumption equations are included.

3.1. Hangos' Algorithms

Let B(L, R, E) be a bipartite graph corresponding to a physical system of equations and variables where L is the set of equations, R is the set of variables and E is the set of edges of the graph B. There is an edge from $l \in L$ to $r \in R$ if the equation l contains the variable r. Let F be a perfect matching to B. If new assumption equations are added to the system without adding new variables, the task is to find a new matching with the least amount of change in F. Hangos' method is based on writing the system equations in an integral form, reassigning the equation-variable associations by adding the design variables D and state variables S to the graph to achieve a new perfect matching. In [21], Hangos defines two algorithms, depending on the number of equations added to the system:

1. Hangos' Algorithm 1: it is applied when only one assumption equation e is added. It has the following features:

   - The free vertexes on the new graph are 'e' and variables from $D \cup S$.
   - The shortest path from 'e' to any variable of $D \cup S$ is found by a Breadth First Search (BFS).
   - The resulting matching is that having the greatest number of edges shared with F.
   - The computational cost is O(V + E), where V is the number of vertex in the graph.

2. Hangos' Algorithm 2: it is used when several assumption equations Q are added, and its properties are:

   - The vertexes of the new graph become: $L' = L \cup Q$ and $R' = R \cup D \cup S$.
   - The weight function w: $E' \rightarrow \Re$ is defined as follows:

$$w(e) = \begin{cases} n+1 & \text{if } e \in F \\ n & \text{if } e \in E' - F \end{cases}. \tag{5}$$

     The purpose of this function is to provide higher weights to the edges of F to obtain a solution with the maximum number of edges shared with F.
   - The maximum matching of maximum weight for w has the greatest number of edges shared with F.

3.2. Problems of Hangos' Method

In order to analyze and show the problems that can arise using Hangos' method, the open evaporator of Figure 2 is considered. This example is also suggested by Hangos in [13,21].

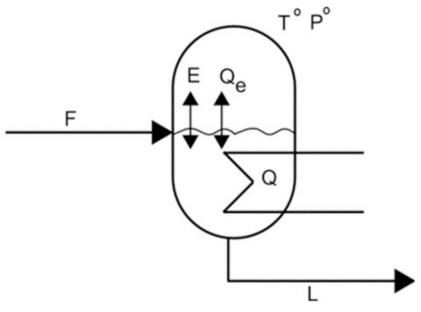

Figure 2. Open evaporator.

The DAE system of this model is given by the system of equations in (6), where M is the mass of the fluid in the vessel, U is the fluid internal energy, T is the fluid temperature, E is the amount of fluid evaporated into the environment, Q_e is the energy transferred by the fluid to the environment and P* is the fluid vapor pressure. The following variables are assumed to be constant: fluid enthalpy

h_L, liquid to gas enthalpy h_{LV}, heat transfer coefficient k_{LV}, specific heat C_e, atmospheric pressure P^0, atmospheric temperature T^0 and an invertible thermodynamic function H(.). The design variables are the heat flux Q transferred to the reactor, the inflow F to the evaporator and the outflow L; they must be specified by the modeler.

$$\begin{array}{ll} f_1: & dM/dt = F - L - E \\ f_2: & dU/dt = F \cdot h_F - L \cdot h_L - E \cdot h_{LV} + Q - Q_e \\ f_3: & E = k_{LV} \cdot (P^* - P^0) \\ f_4: & P^* = H(T) \\ f_5: & Q_e = q_{LV} \cdot (T - T^0) \\ f_6: & U = c_e \cdot M \cdot T \\ f_7: & Q = spec \\ f_8: & L = spec \\ f_9: & F = spec \end{array} \tag{6}$$

Equations f_1 and f_2 are the differential parts of the DAE and belong to the input/output mass and heat flows of the system. The equations f_3, f_4, f_5 and f_6 constitute the algebraic part of the system. Equation f_3 expresses the amount of fluid evaporated into the environment. The thermodynamic equation f_4 relates vapor pressure to temperature. Equation f_5 is used to model the heat flow transferred to the environment, and the state equation f_6 relates U, M and T. Equations from f_7 to f_9 are the user specifications for the design variables.

To transform the DAE system of Equation (6) into an integral one, the following new variables are introduced: the overall mass flow of the reactor m = dM/dt, the overall energy flow u = dU/dt, and the initial values M_0 and U_0 for the mass and energy into the reactor, respectively. Therefore, the system becomes:

$$\begin{array}{ll} f_1: & m = F - L - E \\ f_2: & u = F \cdot h_F - L \cdot h_L - E \cdot h_{LV} + Q - Q_e \\ f_3: & E = k_{LV} \cdot (P^* - P^0) \\ f_4: & P^* = H(T) \\ f_5: & Q_e = q_{LV} \cdot (T - T^0) \\ f_6: & U = c_e \cdot M \cdot T \\ f_7: & Q = spec \\ f_8: & L = spec \\ f_9: & F = spec \\ f_{10}: & M = M_0 + \int m \cdot dt \\ f_{11}: & U = U_0 + \int u \cdot dt \\ f_{12}: & M_0 = spec \\ f_{13}: & U_0 = spec \end{array} . \tag{7}$$

Each variable is assigned to the equation in which it is solved. Therefore, the system causality is {f_1-m, f_2-u, f_3-E, f_4-P*, f_5-Q_e, f_6-T, f_7-Q, f_8-L, f_9-F, f_{10}-M, f_{11}-U, f_{12}-M_0, f_{13}-U_0}. All this information is collected into an equation-variable directed graph where every node is labeled as "equation-variable", as it is shown in Figure 3. An edge goes from node 1 to node 2, if node 2 computes a variable which also belongs to the equation of node 1.

Next, two examples are considered adding a new model assumption. These examples illustrate problems and contradictions that can arise with Hangos' method.

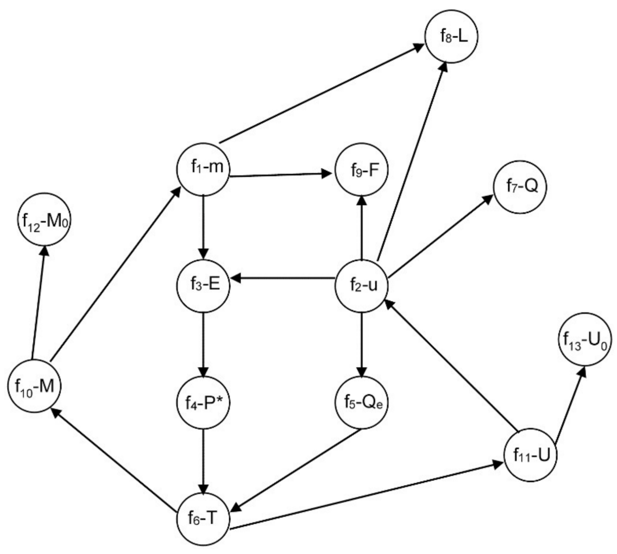

Figure 3. Equation-variable graph of system (7).

3.2.1. Example 1: Removing Initial Conditions

Suppose that M is constant in the integral system of Equation (7), that is, the following assumption equation is added to the system:

$$f_{14}: \quad m = dM/dt = 0. \tag{8}$$

Each time the user adds a new equation, the graph is enlarged with a node. However, the system has zero degrees of freedom if the number of equations is equal to the number of unknowns. Hangos' method proposes to remove equation f_{12} of (7) which is related with the initial condition of M_0. Thus, the variable M_0 is unassigned. This situation is illustrated in the equation-variable graph of Figure 4, where the modified nodes and the new nodes are filled in gray. The new causality assignment is {f_1-E, f_2-u, f_3-P*, f_4-T, f_5-Q_e, f_6-M, f_7-Q, f_8-L, f_9-F, f_{10}-M_0, f_{11}-U, f_{13}-U_0, f_{14}-m}. In the graph, the assignments changes are highilighted with ellipses in dashed line.

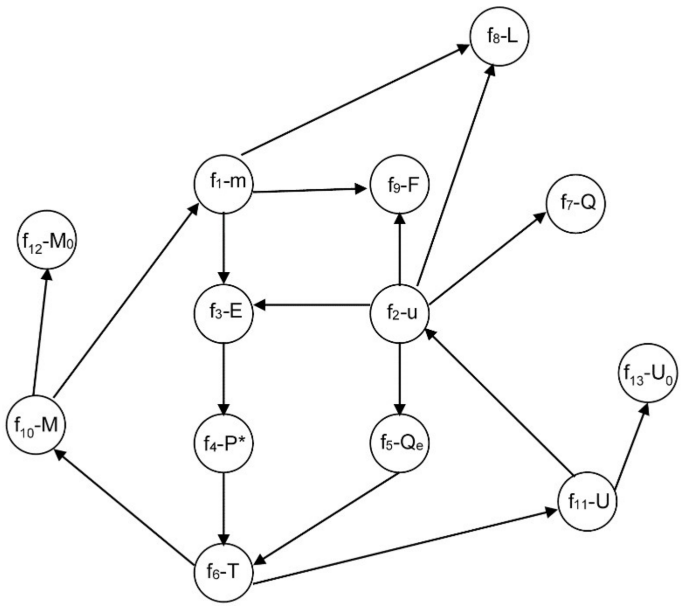

Figure 4. Equation-variable graph under condition m = 0.

Hangos states that the system has a perfect matching; hence, the system differential index is one. Nevertheless, analyzing the new assignments, the following implications are obtained:

1. From equation f_{14} (8), m = 0, and consequently, from f_1 it is obtained that E is a function of F and L: $E = F - L \Rightarrow E = E(F,L)$.
2. Similarly, from equation f_3 with P^0 and k_{LV} constant, P* is a function of F and L: $P^* = P^0 + E/k_{LV} \Rightarrow P^* = P^*(F,L)$.
3. From f_4, and assuming H invertible: $P^* = H(T) \Rightarrow T = H^{-1}(P^*) \Rightarrow T = T(F,L)$.
4. From f_5 with T^0 and q_{LV} constant: $Q_e = q_{LV} \cdot (T - T^0) \Rightarrow Q_e = Q_e(T) \Rightarrow Q_e = Q_e(F,L)$.
5. By equation f_6 with c_e and M constant: $U = c_e \cdot M \cdot T \Rightarrow U = U(T) \Rightarrow U = U(F,L)$.
6. Finally, from f_2 with h_F, h_L, h_{LV} constant: $dU/dt = F \cdot h_F - L \cdot h_L - E \cdot h_{LV} + Q - Q_e \Rightarrow dU/dt - F \cdot h_F + L \cdot h_L + E \cdot h_{LV} + Q_e = Q$.

Analyzing the last equation, the left side is a function of F and L design variables, and the right side only depends on Q, which is another design variable specified separately. Thus, a contradiction appears because the addition of Equation (8) makes the system over-constrained. To solve this new system, the number of equations must be the same as the number of unknowns. Therefore, another equation should be removed. Conversely, considering this system in a strict way, the initial conditions M_0 and U_0 are considered variables. If this integral system expects to be compatible with a differential one, the time derivatives of these two variables must be zero. Removing the equation f_{12}: M_0 = spec implies that the system from Example 1 does not satisfy $M_{0'} = 0$ and, therefore, this system is different from the original one.

3.2.2. Example 2: Removing an Arbitrary Specification Equation

Let us suppose that M is constant again; however, the equation f_7 is selected to be removed from Equation (7). This is equivalent to relax the design variable Q, which means that the design variable Q turns into an unknown algebraic variable. The changed equation-variable assignments are given by: f_{14}-m, f_1-E, f_3-P*, f_4-T, f_6-U, f_{11}-u, f_2-Q, as shown in Figure 5. Sorting the equations and highlighting the variables that are computed by each equation in square brackets, the new system can be written as in Equation (9).

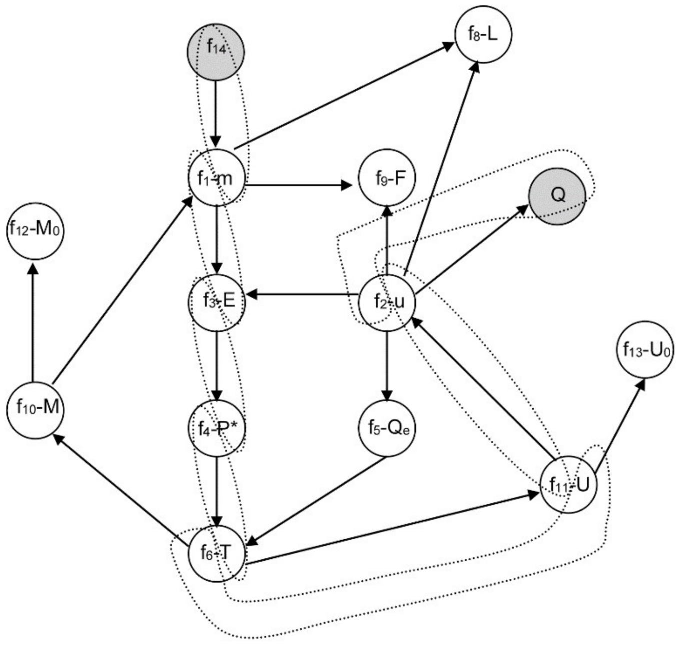

Figure 5. Equation-variable graph with the new causality assignment relaxing variable Q.

$$\begin{aligned}
f_{14}: &\quad [m] = 0 \\
f_9: &\quad [F] = \text{spec} \\
f_8: &\quad [L] = \text{spec} \\
f_1: &\quad m = F - L - [E] \\
f_3: &\quad E = k_{LV} \cdot ([P^*] - P^0) \\
f_4: &\quad P^* = H([T]) \\
f_{12}: &\quad [M_0] = \text{spec} \\
f_{10}: &\quad [M] = M_0 + \textstyle\int m \cdot dt \\
f_6: &\quad [U] = c_e \cdot M \cdot T \\
f_{13}: &\quad [U_0] = \text{spec} \\
f_{11}: &\quad U = U_0 + \textstyle\int [u] \cdot dt \\
f_5: &\quad [Q_e] = q_{LV} \cdot (T - T^0) \\
f_2: &\quad u = F \cdot h_F - L \cdot h_L - E \cdot h_{LV} + [Q] - Q_e
\end{aligned} \tag{9}$$

Note that the system has a perfect matching, so Hangos' method affirms that the system differential index is one. However, examining equation f_{11}, the variable u is under integration. To isolate it, there are two alternatives:

1. Derive equation f_{11}. In this case, equations f_{13}, f_6, f_{10}, f_{12}, f_4, f_3, f_1, f_8 and f_9 must also be derived. After that, a perfect matching is obtained and the index is two, contradicting the previous statement of index one.
2. Solve integral f_{11} by a quadrature method. In this case, it can be thought that no derivative is necessary. However, applying some rules such as the trapezoidal rule, the following results are obtained:

$$U(t) = U_0 + \int_0^t u(t) \cdot dt \Rightarrow U(t+h) = U_0 + \int_0^{t+h} u(t) \cdot dt \tag{10}$$

To decrease errors, these two formulas are subtracted:

$$U(t+h) - U(t) = \int_t^{t+h} u(t) \cdot dt. \tag{11}$$

Solving the above integral by the trapezoidal rule:

$$U(t+h) - U(t) = h\,(u(t+h) - u(t))/2 \tag{12}$$

and isolating the term u(t + h), the following expression is achieved:

$$u(t+h) = -u(t) + 2 \cdot (U(t+h) - U(t))/h. \tag{13}$$

The last expression is the numeric derivative of U(t), which shows that the system solution could not converge. Thus, this second alternative is not a good general alternative.

4. Proposed Method by Relaxation of Design Variables

From the examples of the previous section, it can be concluded that Hangos' method is incomplete and needs to be improved. Modeling assumptions may induce changes in computational properties of the model and modifications in the variable-equation assignments. If the set of design variables is not changed properly, the index of the model can be increased or the zero degree of freedom will not be satisfied. In the next section, a modification of Hangos' method is proposed. This modification relaxes a design variable for each new assumption equation and restores a system with zero degrees of freedom. Two kinds of model assumptions are considered: steady state equations and algebraic equations. First, the basis of the new procedure is introduced and exemplified. The effect of the model assumptions is described as a graph transformation acting on the structure graph of the model. Then, a new algorithm is proposed based on a graph approach and the fundamentals of this proposal are developed.

4.1. Basis of the Proposed Method

This section discusses how Hangos' algorithm can be modified when a new algebraic or steady equation is introduced into the DAE system. The following notation is used:

- The capital letters (I, J, K, . . .) indicate design variables.
- The lowercase letters (a, b, c, . . .) denote the equation identifier and are placed to the left of the equation-variable node.
- The lowercase letters (u, v, w, . . .) represent system variables, which are placed to the right of the graph node. In particular, x and s are states and u, v and w denote algebraic variables.
- An arrow marked with an asterisk indicates that there is a path from the first node to the last one.

To restore the property of zero degrees of freedom after adding a new equation, it is proposed to reduce in one the number of design variables specified by the user, that is relaxing a design variable by removing its corresponding specification equation. After this modification, two vertices of the associated graph remain without matching edges and a new assignment can be constructed starting from the previous one. This minimizes the analysis time, making the modeling tool even more responsive to the user. Then, a closest maximum assignment is performed. This assignment has the largest number of matching edges in common with the original full assignment.

4.1.1. Steady Equation Addition

When a state variable is assumed at steady state, a new equation d is added with the form d: $x' = 0$. To relax a design variable, there are two possibilities: connecting the new equation d with the selected design variable I with a straight path, or linking the equation through a state variable s.

In the first case, connecting the new equation with the design variable with a straight path (Figure 6a), the new causality assignment is completed by combining every equation with a variable of the next one and combining the last equation with the design variable I, as shown in Figure 6b. The final assignment causality is shown in Figure 6c. Note that this new system has a differential index of one.

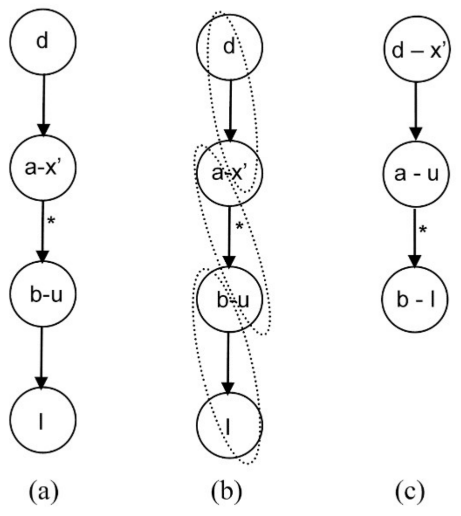

Figure 6. Equation d linked with variable I with a straight path (**a**); matching problem (**b**) and final causality assignment (**c**).

For instance, let suppose again that the assumption equation f_{14} in (8) is added to the original DAE system in (6) in the process example of Section 3.2. The original set of equation is preferred instead of the integral one proposed by Hangos in (7), in order to avoid possible inconsistency problems. The selected design variable to relax is F, so the specification equation f_9 is removed from (6). In this case, there is a direct path from f_{14} to F. Figure 7 shows the causality assignment of the initial system in the corresponding equation-variable graph and the changed causality assignments of the modified system which are represented by dotted lines. The new causality assignment is {f_{14}-M′, f_1-F, f_2-U′, f_3-E, f_4-P*, f_5-Qe, f_6-T, f_7-Q, f_8-L}.

The second possibility consists of linking equation d indirectly with the selected design variable I through the state variable s, see Figure 8. There is a differential edge from node s to node c-s′ represented as a dashed arrow. In this case, an index reduction algorithm, such as Pantelides' algorithm [10], must be applied. This algorithm shows that the path from d to node s must be derived. Equations involved in this path must also be derived. The new system causality is shown in Figure 8c. Thus, in this situation, the differential index is higher than one.

Consider again the previous example of Section 3.2 and suppose now that the selected design variable to relax is Q. Then, the specification equation f_7 is removed. The path from equation f_{14} to variable Q is shown in Figure 9. Note that there is a differential edge from node U to node f_2-U′. After applying Pantelides' algorithm to reduce the differential index of the system, the new causality assignment is {f_{14}-M′, f_1-E, f_3-P*, f_4-T, f_6-U, f'_{10}-M″, f'_1-E′, f'_3-P*′, f'_4-T′, f'_6-U′, f_2-Q, f_5-Q_e, f_8-L, f_9-F}. The variables are classified as state (M), derivative (M′) and algebraic (E, Q, P*, T, Q_e, U, U′, M″, E′, P*′, T′). The new system has zero degrees of freedom, a perfect matching and a differential index of one. Thus, the original system has a differential index of two.

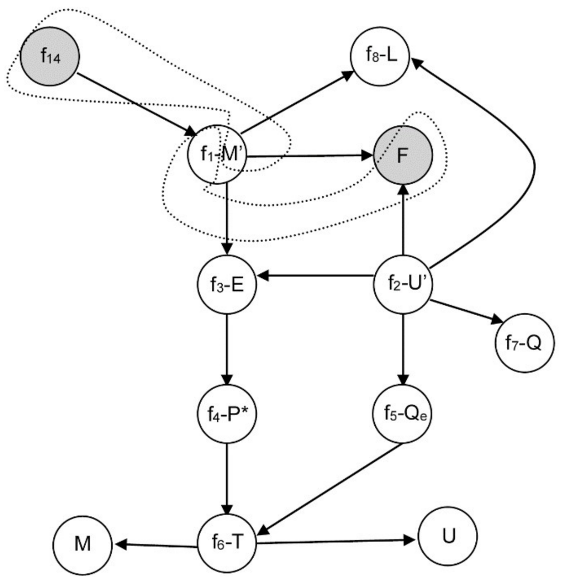

Figure 7. Causality assignment after relaxing F.

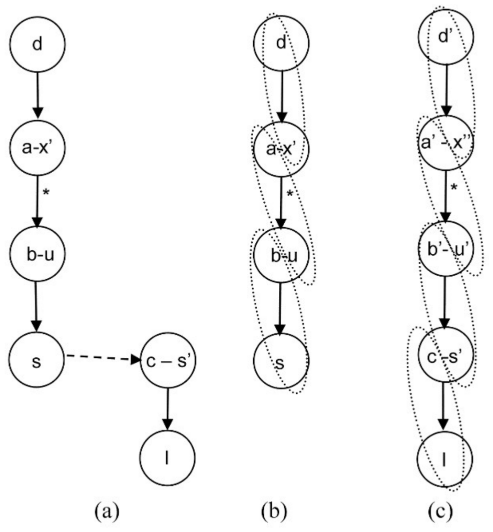

Figure 8. Equation d linked to variable I through a differential path (**a**), and causality assignments before path derivation (**b**) and after derivation with differential index of two (**c**).

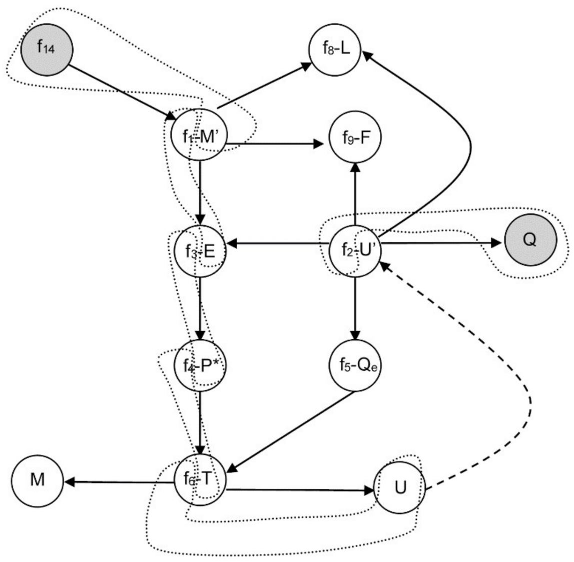

Figure 9. Causality assignment after relaxing Q.

4.1.2. Algebraic Equation Addition

Suppose that an algebraic equation d is added to the DAE system with the form d: z = constant. In this situation, the same alternatives as those of previous subsections are found. Figure 10 shows a straight path from d to design variable I and the new causality assignment. Note that the final system has a perfect matching.

In the second case, equation d is linked with a design variable I by a path through a state variable s (Figure 11). To obtain a perfect matching, it is necessary to apply an index reduction algorithm, such as Pantelides' algorithm, as shown in Figure 11c. Therefore, in this example, the differential index of the system is two.

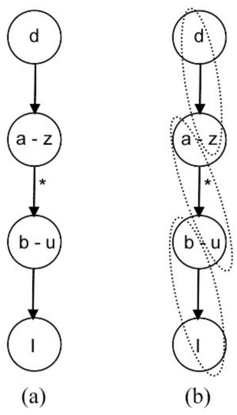

Figure 10. Linking equation d to design variable I by a straight path (**a**); and solution of the assignment problem (**b**).

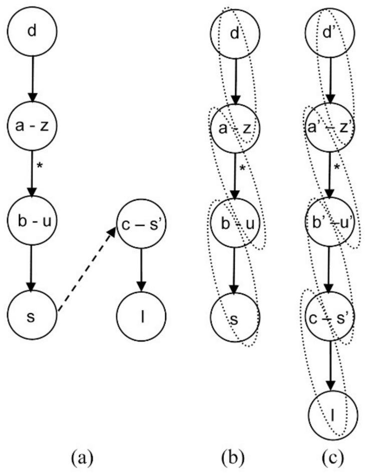

Figure 11. Equation d linked with design variable I through the state variable s (**a**), and causality assignments before path derivation (**b**) and after derivation with differential index of two (**c**).

4.1.3. General Case

The most general assumption appears when an equation with several algebraic variables is added. In this case, it is enough to find a link from any of the paths that begin at vertex d to the selected design variable I. As an example, let us suppose that the equation d is added with the form d: z = y. In Figure 12, two alternatives for finding the design variable that solves this problem are shown.

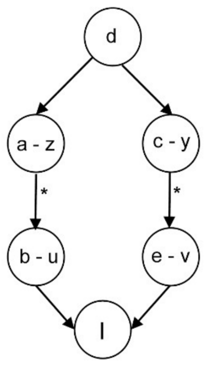

Figure 12. Several paths linking with the design variable I.

4.2. Proposed Algorithm

The suggested procedure is generalized in the proposed Algorithm 2, that relates algebraic and state variables in the assumption equations with design variables through straight paths or through the state variables of the initial system. This algorithm allows the addition of several equations simultaneously and tries to maintain the highest number of intact equation-graph associations of the initial system as possible, that is, the closest assignment. The algorithm can be combined with another index reduction algorithm to solve the causality assignment in one step. In comparison with the Hangos' Algorithm 2, discussed in Section 2.1, the proposal offers some improvements that optimize its computational cost and behavior. Since a maximum weight matching of a bipartite graph can be calculated in polynomial time [27], the closest assignment problem has the same time bound as valid.

A first improvement concerns the weight function w in (5) used by Hangos. It will be proven that there is a family of functions which obtain the same solution as the w in (5) and that they are simpler to compute.

Definition 1. *Let a and b be two real numbers with a > b. Let B(L, R, E) be a bipartite graph with a matching F. We define the weight function w_{ab} as $w_{ab} : E \to \Re$*

$$\mathrm{w_{ab}(e)} = \begin{cases} \mathrm{a} & \text{if } \mathrm{e} \in \mathrm{F} \\ \mathrm{b} & \text{if } \mathrm{e} \in \mathrm{F^c} \end{cases} . \tag{14}$$

Lemma 1. *Let a and b $\in \Re$, with a > b. Let B(L, R, E) be a bipartite graph with a matching F and weight function w_{ab}. Let M be the maximum matching of maximum weight and M' be any other maximum matching, then $|M \cap F| \geq |M' \cap F|$.*

Proof. Because M and M' are maximum matchings: $|M| = |M'|$ and $|M \cap F| + |M \cap F^c| = |M' \cap F| + |M' \cap F^c|$. Then:

$$|M \cap F| - |M' \cap F| = |M' \cap F^c| - |M \cap F^c|. \tag{15}$$

Because $w_{ab}(M) \geq w_{ab}(M') \Rightarrow \sum\limits_{e \in M \cap F} a + \sum\limits_{e \in M \cap F^c} b \geq \sum\limits_{e \in M \cap F} a + \sum\limits_{e \in M \cap F^c} b \Rightarrow a|M \cap F| + b|M \cap F^c| \geq a|M' \cap F| + b|M' \cap F^c|$. By arranging all of the expressions to the left-hand side and gathering the expressions that are preceded by a or b, it follows:

$$a(|M \cap F| - |M' \cap F|) + b(|M \cap F^c| - |M' \cap F^c|) \geq 0. \tag{16}$$

By (15), the following relation is deduced:

$$a(|M \cap F| - |M' \cap F|) - b(|M \cap F| - |M' \cap F|) \geq 0. \tag{17}$$

This implies:

$$(a - b)(|M \cap F| - |M' \cap F|) \geq 0. \tag{18}$$

Because a > b, the expression of the first bracket is a positive number. Therefore, the proposed result is obtained. □

From Lemma 1, any w_{ab} function can be selected to obtain a maximum matching of maximum weight M, which has the maximum number of element shared with F, considering the set of maximum matchings, that is: $\forall M' \in \{M''/M'' \text{ is a maximum matching}\} \Rightarrow |M \cap F| \geq |M' \cap F|$.

Theorem 4. *Let a, b, c and d* $\in \Re$*, such that a > b and c > d. If we let B(L, R, E) be a bipartite graph with a matching F, then M is a maximum matching of maximum weight with* w_{ab} *if and only if M is a maximum matching of maximum weight with* w_{cd}*.*

Proof. $\Rightarrow$ It is assumed that M is a matching of maximum weight for w_{ab}, and it will be proven that this is also true for w_{cd}. Let M' be any maximum matching, and by Lemma 1:

$$|M \cap F| - |M' \cap F| \geq 0. \tag{19}$$

By multiplying (19) by $c - d$, $c > d$, we obtain: $(c - d)(|M \cap F| - |M' \cap F|) \geq 0$, and therefore:

$$c\,(|M \cap F| - |M' \cap F|) - d\,(|M \cap F| - |M' \cap F|) \geq 0. \tag{20}$$

Because M and M' are maximum matchings, it follows:

$$|M \cap F| + |M \cap F^c| = |M' \cap F| + |M' \cap F^c|. \tag{21}$$

By arranging (21):

$$|M \cap F| - |M' \cap F| = |M' \cap F^c| - |M \cap F^c|. \tag{22}$$

By replacing (22) into (20), it follows:

$$c\,(|M \cap F| - |M' \cap F|) + d\,(|M \cap F^c| - |M' \cap F^c|) \geq 0. \tag{23}$$

This implies that

$$c\,|M \cap F| + d\,|M \cap F^c| \geq c\,|M' \cap F| + d\,|M' \cap F^c|. \tag{24}$$

$\Leftarrow$ This result is obtained in a similar manner. $\square$

Theorem 4 allows taking any w_{ab} function as a weight function, with a > b. The simplest function of this family is the characteristic function of the matching F, which is $w_F : E \to \Re$:

$$w_F(e) = \begin{cases} 1 & \text{if } e \in F \\ 0 & \text{if } e \in F^c \end{cases}. \tag{25}$$

As a consequence of Theorem 3, taking a matching of maximum weight in a run, another matching of maximum weight is in the next run. Therefore, if F is considered initially, then the number of iterations is drastically reduced.

The next proposition characterizes a maximum matching of maximum weight regarding to the initial matching F.

Lemma 2. *Let B(L, R, E) be a bipartite graph. Let F and M be two matching with* $|M| > |F|$*. If* $|M \cap F|$ *is the maximum among all matchings with cardinality* $|M|$*, then* $M \Delta F$ *has neither cycles nor alternating paths with even lengths.*

Proof. (by contradiction)

Suppose for the sake of contradiction that $M \Delta F$ has cycles or alternating paths with even lengths.

(1) $M \Delta F$ is decomposed into $(\bigcup_{i=1}^{k} p_i) \cup (\bigcup_{i=1}^{r} d_i) \cup (\bigcup_{i=1}^{r} c_i)$, where:

- All of the sets p_i, d_i and c_i are disjointed.

- p_i represents the augmented paths in F. There are $k = |M| - |F| > 0$ paths of this kind, and all of them have odd length. Moreover, the number of edges of p_i in F + 1 is equal to number of edges of pi in $M - F$, that is:

$$|p_i \cap F| + 1 = |p_i \cap (M - F)| = (|p_i| + 1)/2. \tag{26}$$

- d_i represents an alternating path of even length. The number of edges of d_i in F is equal to the number of edges of d_i in $M - F$:

$$|d_i \cap F| = |d_i \cap (M - F)| = |d_i|/2. \tag{27}$$

- c_i represents cycles, and therefore, their length is even. They fulfill the same equation as d_i, that is:

$$|c_i \cap F| = |c_i \cap (M - F)| = |c_i|/2. \tag{28}$$

(2) Let $M' = F \Delta (\overset{k}{\underset{i=1}{\cup}} p_i)$. If M' is a matching in B and its cardinality is $|M'| = |F| + k = |M|$, then:

$$|M' \cap F| = |F - \overset{k}{\underset{i=1}{\cup}} p_i| = |F| - \sum_{i=1}^{k} |p_i \cap F| = |F| - \sum_{i=1}^{k} \frac{|p_i| - 1}{2}. \tag{29}$$

In contrast,

$$|M \cap F| = |F - (\overset{k}{\underset{i=1}{\cup}} p_i) \cup (\overset{r}{\underset{i=1}{\cup}} d_i) \cup (\overset{r}{\underset{i=1}{\cup}} c_i)| = |F| - \sum_{i=1}^{k} |p_i \cap F| - \sum_{i=1}^{r} |d_i \cap F| - \sum_{i=1}^{s} |c_i \cap F|. \tag{30}$$

(3) According to our hypothesis, if $|M \cap F|$ is maximum among all the matchings of cardinality $|M|$, then: $|M \cap F| \geq |M' \cap F|$. By (29) and (30):

$$|F| - \sum_{i=1}^{k} |p_i \cap F| - \sum_{i=1}^{r} |d_i \cap F| - \sum_{i=1}^{s} |c_i \cap F| \geq |F| - \sum_{i=1}^{k} |p_i \cap F|. \tag{31}$$

Equation (31) can be simplified to:

$$\sum_{i=1}^{r} |d_i \cap F| + \sum_{i=1}^{s} |c_i \cap F| \leq 0. \tag{32}$$

By replacing the Expressions (27) and (28) into (32):

$$\sum_{i=1}^{r} \frac{|d_i|}{2} + \sum_{i=1}^{s} \frac{|c_i|}{2} \leq 0. \tag{33}$$

This expression implies that:

$$|d_i| = 0 \ \forall\, i = 1, \ldots, r \text{ and } |c_i| = 0 \ \forall\, i = 1, \ldots ., s, \tag{34}$$

which is a contradiction.

$\square$

Theorem 5. *Let B(L, R, E) be a bipartite graph. Let F be a matching of B and w_F, the characteristic function of F. Let M be a maximum matching of maximum weight for w_F. If $k = |M| - |F| > 0$, then there are k augmented paths disjoined by vertexes $\{p_1, \ldots, p_k\}$ in F, such that:*

1. $M = F \,\Delta\, p_1 \,\Delta \ldots \Delta\, p_k.$
2. $|M \cap F| = |F| - \sum_i \frac{|p_i|-1}{2}.$
3. $|M \cap F| = \frac{|M|+|F|-\sum_i |p_i|}{2}.$

Proof.

1. It is known by Lemma 1 that the cardinality of M ∩ F is the maximum among all the maximum matchings. After applying Lemma 2, the desired result is obtained.
2. By (26) and (29) of Lemma 2, it follows that:

$$|M \cap F| = |F| - \sum_{i=1}^{k} |p_i \cap F| = |F| - \sum_{i=1}^{k} \frac{|p_i| - 1}{2}. \tag{35}$$

3. By the last expression:

$$|M \cap F| = |F| - \sum_{i=1}^{k} \frac{|p_i| - 1}{2} = |F| - \frac{k}{2} - \frac{1}{2}\sum_{i=1}^{k} |p_i|. \tag{36}$$

As k = | M | − | F |, then:

$$|M \cap F| = |F| - \frac{|M| - |F|}{2} - \frac{1}{2}\sum_{i=1}^{k} |p_i| = \left(|M| + |F| - \sum_{i=1}^{2} |p_i|\right)/2. \tag{37}$$

□

The above theorem states the relation between the cardinals of the augmented path of F, F itself and the maximum matching of maximum weight M.

Corollary 1. *Under the assumptions of Theorem 5, $\sum_{i=1}^{k} |p_i|$ is the lowest among all the sums of any k vertex-disjoint augmenting paths of F.*

Proof. Let $\{q_1, \ldots, q_k\}$ be any k vertex-disjoint augmenting paths of F. The matching $M' = F\,\Delta q_1 \,\Delta \ldots \Delta q_k$ has a cardinality of $|M'| = |F| + k = |M|$. By applying Theorem 5 we deduce that:

$$|M \cap F| = \left(|M| + |F| - \sum_{i} |p_i|\right)/2. \tag{38}$$

In contrast:

$$|M' \cap F| = |(F\Delta \bigcup_{i=1}^{k} q_i) \cap F| = |F - \bigcup_{i=1}^{k} q_i| = |F| - \sum_{i=1}^{k} |q_i \cap F| = |F| - \sum_{i=1}^{k} (|q_i| - 1)/2 = \left(|M| + |F| - \sum_{i} q_i\right)/2. \tag{39}$$

By Lemma 1: $|M \cap F| \geq |M' \cap F|$ and by the relations (38) and (39), then:

$$\frac{|M| + |F| - \sum_i |p_i|}{2} \geq \frac{|M| + |F| - \sum_i q_i}{2}. \tag{40}$$

By operating in the last expression, we deduce that $\sum_{i=1}^{k} |p_i| \leq \sum_{i=1}^{k} |q_i|$. □

Using the previous results, the Algorithm 2 is proposed. It is based on the construction of assignments using graph theory and the Algorithm 1 of Section 2, and is an improvement of Hangos' Algorithm 2. The algorithm expects as input a bipartite graph B(L, R, E) representing the system equations, the weight function w such as that in (25), and an initial matching F before adding modeling assumptions. The complexity to compute the maximum matching starting from an empty matching is O(V·E); however, it is reduced to O(E) when the graph is enlarged by adding only one new node and a previous maximum matching is already known [20]. This algorithm uses the information generated in previous steps of the simulation, so the initial matching F is not discarded, which is an improvement over algorithms that need to start from scratch (these kinds of algorithms have a cost $O(V^2 + V\cdot E)$). This can make the modeling tool more responsive to user specification changes.

If Q is the number of assumption equations, the time complexity varies depending on the algorithm used in the computation of the path of the lowest cost in G_M (step 3.1). When the Bellman-Ford algorithm is employed, the overall time cost is polynomial with O(Q·V·E), since it is performed one time for each new assumption added to the system.

Algorithm 2. *Find the maximum matching such that* $|M \cap F|$ *is highest.*

```
Input: A bipartite graph B(L, R, E), weight function w and initial maximum matching F.
Output: Matching of maximum weight closest to F.
1: M := F
2: Construction of a direct graph $G_M$ (V, $E_M$) with function weight z.
    2.1: $V = L \cup R$
    2.2: $E_M$ is defined by:
        If the undirected edge $(l, r) \in M^c$ then
            the directed edge $(l, r) \in E_M$.
        else if the undirected edge $(l, r) \in M$ then
            the directed edge $(r, l) \in E_M$.
end if
    That is, the set of edges $E_M$ is: $E_M = \{(l, r)/(l, r) \in M^c\} \cup \{(r, l)/(l, r) \in M\}$
    2.3: The weight function z is defined:
        textbfIf $(l, r) \in E_M$ then
            $z(l, r) = w(l, r)$.
        else if $(r, l) \in E_M$ then
            $z(r, l) = -w((l, r)$
        end if
    2.4: Append a new vertex s linked to all free vertexes of L.
    2.5: Append a new vertex t which is the destination of all free vertexes of R.
3: while there exists an augmented path p into $G_M$ do
    3.1: p = an augmented path of lowest cost in $G_M$
    3.2: $M := M \Delta p$
    3.3: Adjust $G_M$ by path p.
  end while
```

The cost of the algorithm can also be improved if it is noted that the edges of G_M can take only values from the set $\{-1, 0, 1\}$. In [26], the authors proposed an algorithm based on Dijkstra's algorithm with linked list or minimal heaps that improves the time complexity of algorithm 1. This algorithm can be used if the weight of the edges is a positive integer in the range $\{0, 1, 2, \dots, W\}$, where W is the highest value of any edge. To apply this algorithm to the proposed one, it is necessary to reduce the set $\{-1, 0, 1\}$ of the weight function in (25) to a set of a positive integers. The next theorem resolves this limitation.

Theorem 6. *Let G(V, E) be a graph with weight function w and let k be any real number. Then, M is a matching of maximum weight among all matchings of cardinality* $|M|$ *for* $w \Leftrightarrow M$*, which is a matching of maximum weight among all the matchings of cardinality* $|M|$ *for* $w_2 = w + k$.

Proof. Let M' be a matching of cardinality $|M|$. According to our hypothesis, $w(M) \geq w(M')$, so:

$$w(M) \geq w(M') \Leftrightarrow w(M) + k\,|M| \geq w(M') + k\,|M| \Leftrightarrow \sum_{e \in M} w(e) + k \cdot |M| \geq \sum_{e \in M'} w(e) + k \cdot |M| \Leftrightarrow \sum_{e \in M} (w(e) + k) \geq \sum_{e \in M'} (w(e) + k) \Leftrightarrow \sum_{e \in M} w_2(e) \geq \sum_{e \in M'} w_2(e) \Leftrightarrow w_2(M) \geq w_2(M').$$

According to (25), the weight function w of graph G_M only takes on values in the set $\{-1, 0, 1\}$. By applying Theorem 6 for k = 1, the problem is reduced to another one where the values of the weight function belong to the set {0, 1, 2}. Thus, Dijkstra's algorithms can be applied to step 3.1 of the proposed algorithm. Using the Dijkstra's algorithm with minimal heaps, the time complexity is $O(Q \cdot (E + V \cdot \log(V)))$. When the Dijkstra's algorithm is implemented with linked list, the overall time cost is reduced to $O(Q \cdot (V + E))$ [22]. In this last case, the algorithm is almost linear because Q is supposed to be small. Therefore, if the cardinality of Q is negligible against V + E, the cost is O(V + E).

5. Conclusions

Often the system under study is described by a system of DAEs. These equations are usually parameterized by the design variables, which are known during the simulation. If a new assumption equation is added to the system, it becomes over-constrained. There are several methods to reduce the new system to a well-constrained one, such as Hangos' method. This method basically consists of removing an equation from the original system to equal the number of equations and variables. However, this procedure requires that the new system contradicts the first one and, in some cases, the solution of the transformed system is far from one which describes the initial system. This paper presented an alternative to Hangos' method which preserves the original formulation of the system. The basis of this new method is to relax design variables, converting them into new algebraic variables. This procedure achieves a solution compatible with that of the initial system.

The example proposed in [21] has been studied with a simulation of a reactor with input and output flows. The differential system has been transformed into an integral one, as Hangos' method shows. Then, a new assumption equation has been added and a new system that is not related to the first one has been obtained. However, by applying the method suggested by the authors, an accurate system with the initial equations is obtained.

Also, an algorithm has been developed based on graph theory, starting from the initial matching of equation-variables. This algorithm has the followings characteristics:

- It relates algebraic and state variables of the assumptions equations to the design variables of the system selected by the user.
- It allows the addition of several assumption equations and obtaining a greater number of common edges with the starting system.
- In the best case, depending on the implementation, the computational cost is O(Q(E + V)), where Q is the number of assumption equations, V the number of vertexes in the bipartite equation-variable graph and E is the number of edges in that graph. Therefore, if the number of assumption equations is small, the cost of the algorithm is linear.
- The algorithm can be combined with an algorithm of index reduction.

Author Contributions: Antonio Belmonte and Juan Garrido conceived and design the method; Jorge E. Jiménez and Francisco Vázquez analyzed the method; and Francisco Vázquez and Juan Garrido wrote the paper.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Garrido, J.; Zafra, A.; Vazquez, F. Object oriented modelling and simulation of hydropower plants with run-of-river scheme: A new simulation tool. *Simul. Model. Pract. Theory* **2009**, *17*, 1748–1767. [CrossRef]
2. Nikolić, D.D. Dae tools: Equation-based object-oriented modelling, simulation and optimisation software. *PeerJ Comput. Sci.* **2016**, *2*, e54. [CrossRef]
3. Fritzson, P.; Engelson, V. Modelica—A Unified object-oriented language for system modeling and simulation. In *ECOOP'98—Object-Oriented Programming*; Springer: New York, NY, USA, 1998; pp. 67–90.
4. Piela, P.; Epperly, T.; Westerberg, K.; Westerberg, A. Ascend—An object-oriented computer environment for modeling and analysis—The modeling language. *Comput. Chem. Eng.* **1991**, *15*, 53–72. [CrossRef]
5. Vázquez, F.; Jiménez, J.; Garrido, J.; Belmonte, A. *Introduction to Modelling and Simulation with Ecosimpro*; Pearson Educacion: Madrid, Spain, 2010; p. 272.
6. Brenan, K.E.; Campbell, S.L.; Petzold, L.R. *Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations*; Society for Industrial and Applied Mathematics: Philadelphia, PA, USA, 1996.
7. Navarro, A.; Vassiliadis, V. Computer algebra systems coming of age: Dynamic simulation and optimization of dae systems in mathematica™. *Comput. Chem. Eng.* **2014**, *62*, 125–138. [CrossRef]
8. Cellier, F.E.; Kofman, E. *Continuous System Simulation*; Springer Science & Business Media: Berlin, Germany, 2006.
9. Tarjan, R. Depth-first search and linear graph algorithms. *SIAM J. Comput.* **1972**, *1*, 146–160. [CrossRef]
10. Pantelides, C.C. The consistent initialization of differential-algebraic systems. *SIAM J. Sci. Comput.* **1988**, *9*, 213–231. [CrossRef]
11. Cafferkey, N.; Provan, G. An analysis of performance-critical properties of modelica models. *IFAC-PapersOnLine* **2015**, *48*, 210–215. [CrossRef]
12. Unger, J.; Kroner, A.; Marquardt, W. Structural-analysis of differential-algebraic equaion systems—Theory and applications. *Comput. Chem. Eng.* **1995**, *19*, 867–882. [CrossRef]
13. Hangos, K.; Cameron, I. A formal representation of assumptions in process modelling. *Comput. Chem. Eng.* **2001**, *25*, 237–255. [CrossRef]
14. Merchan, V.; Esche, E.; Fillinger, S.; Tolksdorf, G.; Wozny, G. Computer-aided process and plant development. A review of common software tools and methods and comparison against an integrated collaborative approach. *Chem. Ing. Tech.* **2016**, *88*, 50–69. [CrossRef]
15. Jensen, A.K. *Generation of Problem Specific Simulation Models within an Integrated Computer Aided System*; CAPEC-DTU: Lyngby, Denmark, 1998.
16. Bogusch, R.; Lohmann, B.; Marquardt, W. Computer-aided process modeling with modkit. *Comput. Chem. Eng.* **2001**, *25*, 963–995. [CrossRef]
17. Moe, H.I. Dynamic Process Simulation: Studies on Modeling and Index Reduction. Ph.D. Thesis, University of Trondheim, Trondheim, Norway, 1995.
18. Murota, K. Systems analysis by graphs and matroids. Structural solvability and controllability. *SIAM Rev.* **1989**, *31*, 502.
19. Leitold, A.; Hangos, K. Structural solvability analysis of dynamic process models. *Comput. Chem. Eng.* **2001**, *25*, 1633–1646. [CrossRef]
20. Soares, R.D.P.; Secchi, A.R. Structural analysis for static and dynamic models. *Math. Comput. Model.* **2012**, *55*, 1051–1067. [CrossRef]
21. Hangos, K.; Szederkenyi, G.; Tuza, Z. The effect of model simplification assumptions on the differential index of lumped process models. *Comput. Chem. Eng.* **2004**, *28*, 129–137. [CrossRef]
22. Cormen, T.H.; Leiserson, C.E.; Rivest, R.L.; Stein, C. *Introduction to Algorithms*, 2nd ed.; McGraw-Hill: New York, NY, USA, 2001.
23. Korte, B.; Vygen, J.; Korte, B.; Vygen, J. *Combinatorial Optimization*; Springer: Berlin, Germany, 2012; Volume 2.
24. Tassa, T. Finding all maximally-matchable edges in a bipartite graph. *Theor. Comput. Sci.* **2012**, *423*, 50–58. [CrossRef]
25. Micali, S.; Vazirani, V.V. An O(v | v | c | E |) algorithm for finding maximum matching in general graphs. In Proceedings of the 21st Annual Symposium on Foundations of Computer Science, Syracuse, NY, USA, 13–15 October 1980; pp. 17–27.
26. Gabow, H.N.; Tarjan, R.E. Faster scaling algorithms for general graph matching problems. *J. ACM* **1991**, *38*, 815–853. [CrossRef]

27. Hopcroft, J.E.; Karp, R.M. An nˆ5/2 algorithm for maximum matchings in bipartite graphs. *SIAM J. Comput.* **1973**, *2*, 225–231. [CrossRef]
28. Mucha, M.; Sankowski, P. Maximum matchings via gaussian elimination. In Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science, Rome, Italy, 17–19 October 2004; pp. 248–255.
29. Goel, A.; Kapralov, M.; Khanna, S. Perfect matchings in O(n\logn) time in regular bipartite graphs. *SIAM J. Comput.* **2013**, *42*, 1392–1404. [CrossRef]
30. Frenkel, J.; Kunze, G.; Fritzson, P. Survey of appropriate matching algorithms for large scale systems of differential algebraic equations. In Proceedings of the 9th International MODELICA Conference, Munich, Germany, 3–5 September 2012; Linköping University Electronic Press: Linköping, Sweden, 2012; pp. 433–442.
31. Berge, C. Two theorems in graph theory. *Proc. Natl. Acad. Sci. USA* **1957**, *43*, 842–844. [CrossRef] [PubMed]
32. Har-Peled, S. Matchings II. Available online: https://courses.engr.illinois.edu/cs473/fa2015/w/lec/lec/31_matchings_II.pdf (accessed on 8 April 2018).