Conditional Modeling, Part II: Solving Using a Complementarity Formulation and a Boundary Crossing Algorithm

V. Rico-Ramírez and A. W. Westerberg[1]

Department of Chemical Engineering and
the Institute for Complex Engineered Systems
Carnegie Mellon University, Pittsburgh, PA 15213

Abstract: In this second paper on conditional modeling, we investigate the solving of conditional models. First, we present a new complementarity formulation for representing algebraic systems of disjunctive equations and, then, we discuss a boundary crossing algorithm proposed by Zaher (1995).

Our complementarity formulation not only establishes the complementarity condition among equations belonging to different disjunctive terms but also enforces simultaneous satisfaction of all of the equations appearing within the same disjunctive term. This approach performed reliably on several example problems where the number of equations in each disjunctive term is small.

Zaher (1995) proposed the boundary crossing algorithm as an alternative to using MINLP techniques for solving. This algorithm involves the execution of several well differentiated activities including logical analysis and continuous reconfiguration of the equations constituting the problem, calculation of Newton-like steps, and the calculation of subgradient steps. We describe the practical implementation of the boundary crossing algorithm as a conditional modeling solution tool. In such an implementation, we have integrated the entities created and/or used to perform each of the activities in an object oriented solving engine: the conditional modeling solver CMSlv.

We solve several examples of conditional models in chemical engineering by using both of the approaches for solving and identify the advantages and disadvantages of such approaches.

Keywords: Conditional models, complementarity formulation, modeling, disjunctive modeling, boundary crossing.

1. Author to whom all correspondence should be addressed

1 INTRODUCTION

In Part I (Rico-Ramirez et al, 1997) we investigated the modeling capabilities to permit the formulation of such models within an equation-based environment. In this paper, we continue our study of conditional modeling by examining two approaches for solving them.

Conditional models exist when the defining equations of the system depend on where the model solution lies. A conditional model, as defined by Grossmann and Turkay (1996), consists of a system of equations expressed by two sets: a globally defined invariant set of equations and a variant (or locally defined) set of conditional equations which are expressed as disjunctions. Grossmann and Turkay (1996) show that a conditional model can be represented as the system of disjunctive equations:

$$\begin{aligned} & \underline{h}(\underline{x}) = 0 \\ & \bigvee_{i \in D_k} \begin{bmatrix} r_{i_{jk}}(\underline{x}) = 0 \\ g_{i_{lk}}(\underline{x}) \le 0 \end{bmatrix} k \in K \qquad \begin{matrix} \forall j \in [1 \ldots \beta_k] \\ \forall l \in [1 \ldots \gamma_k] \end{matrix} \\ & \underline{x} \in R^n \end{aligned} \tag{1}$$

where $\underline{h}(\underline{x})$ and $\underline{r}_{i_k}(\underline{x})$ represent the vectors of the invariant and the variant sets of equations respectively, K represents the set of disjunctions and the index i is used to indicate the i-th disjunctive term in each disjunction D_k. The vector $\underline{h}(\underline{x})$ is m-dimensional, and it is assumed that $\underline{r}_{i_k}(\underline{x})$ is β_k-dimensional and that $\underline{g}_{i_k}(\underline{x})$ is γ_k-dimensional, for all i in D_k. The equations $\underline{h}(\underline{x})$ can be said to be defined over the entire feasible region, while the inequalities $\underline{g}_{i_k}(\underline{x})$ define the domain of validity of each variant set of equations $\underline{r}_{i_k}(\underline{x})$. In that way, each variant set of equations is confined to some subregion resulting from the dissection of the feasible region (Zaher, 1991). The solution to the system will be given by the vector $\hat{\underline{x}}$ satisfying the invariant set $\underline{h}(\underline{x})$ and exactly one set of equations for each of the disjunctions, providing that the corresponding set of inequalities is satisfied. Grossmann and Turkay (1996) also address the existence and uniqueness of the solution for the linear case.

It is important to recognize that modeling is a user dependent task. Frequently, different modelers produce different formulation for the same problem and, as a consequence, different approaches and techniques may be used to find a solution to it.

One can formulate conditional models as mixed-integer programming problems. The literature reports several MINLP approaches to the solution of conditional models. Grossmann and Turkay (1996) show that, in the case of linear equations, the solution of a conditional model can be found by solving a MILP problem that is often solvable as a relaxed LP. Also, based on the works of Raman and Grossmann (1993, 1994), Turkay and Grossmann (1996) address the solution of disjunctive set of nonlinear programming problems. They show that the use of logic can improve the efficiency and robustness of MINLP algorithms in the solution of structural optimization problems.

Zaher (1995) makes the following observations about the MINLP approach (big-M formulation) to the solution of conditional models:

1. One can expect that, during the solution of a MINLP, the algorithm will jump from one subregion to another which is remote from it, leading to the poor initialization of some of the nonlinear subproblems (See Figure 1).
2. The nonlinear subproblems insist on converging the equations of what might be an incorrect subregion.
3. This method remains as a rigorous general purpose algorithm designed for problems where discrete decision making cannot be avoided.

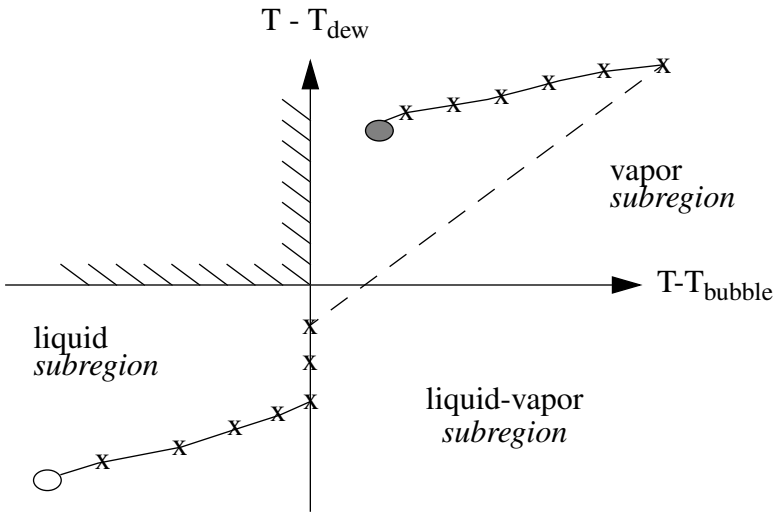
Besides the mixed-integer formulation, some alternative approaches exist which, in principle, can avoid the need of discrete decisions. This work focuses on two of such approaches, a new complementarity representation of conditional models and a boundary crossing algorithm proposed by Zaher (1995).

2 COMPLEMENTARITY APPROACH

Over the last thirty years, the class of problems known as complementarity problems has become increasingly popular as a tool for addressing practical problems arising in mathematical programming, economics, engineering, and the sciences (Billups, 1995; Ferris and Pang, 1995). Several works have documented the basic theory, algorithms and applications of complementarity problems. Dirske and Ferris (1995b) give examples of how to formulate many popular problems as mixed complementarity problems (MCP). Billups (1995) describes the standard forms for the

different classes of complementarity problems and proposes strategies which enhances the robustness of Newton-based methods for solving these problems. More (1994) formulates the complementarity problem as a nonlinear least square problem and gives convergence properties for his approach. In this work, we describe an extension to the standard complementarity formulation (Billups, 1995) for the representation of conditional models.

FIGURE 1 MINLP approach to the solution of conditional models. Note the jump from the liquid to the vapor region, leading to poor initial conditions from which to converge the vapor model equations.



2.1 PROBLEM FORMULATION

In general, the nonlinear complementarity problem is expressed as the following set of equations and inequality constraints:

$$\begin{gathered} x_j \cdot r_j(\underline{x}) = 0 \\ x_j \geq 0 \qquad r_j(\underline{x}) \geq 0 \\ \forall j \in \{1 \ldots n\} \end{gathered} \tag{2}$$

where n is the dimensionality of the vectors $\underline{x}$ and $\underline{r(x)}$. There is certain lack of symmetry in this formulation. One of the functions is quite arbitrary while the other is the vector of variables. Many

commonly occurring problems have a more general form:

$$\begin{gathered} r_{1_j}(\underline{x}) \cdot r_{2_j}(\underline{x}) = 0 \\ r_{1_j}(\underline{x}) \geq 0 \qquad r_{2_j}(\underline{x}) \geq 0 \\ \forall j \in \{1 \ldots n\} \end{gathered} \tag{3}$$

which is called the vertical nonlinear complementarity problem (Ferris and Pang, 1995). It is possible, of course, to have more than two vectors of functions in the above equations:

$$\begin{gathered} r_{1_j}(\underline{x}) \cdot r_{2_j}(\underline{x}) \ldots r_{k_j}(\underline{x}) = 0 \\ r_{i_j}(\underline{x}) \geq 0 \qquad \forall i \in [1 \ldots k], \forall j \in \{1 \ldots n\} \end{gathered} \tag{4}$$

For the case of a *single conditional equation* in a disjunctive statement, Zaher,(1995) show that there exits an equivalent representation by using a standard complementarity formulation as follows:

$$\begin{bmatrix} r_i + g_i(\underline{x}) < 0 \\ r_i = 0 \end{bmatrix} \vee \begin{bmatrix} r_i + g_i(\underline{x}) \geq 0 \\ g_i(\underline{x}) = 0 \end{bmatrix} \Leftrightarrow \begin{gathered} r_i \cdot g_i(\underline{x}) = 0 \\ r_i \geq 0 \qquad g_i(\underline{x}) \leq 0 \end{gathered} \tag{5}$$

A typical example of this equivalence can be found while representing the complementarity equations arising from the Karush-Kuhn-Tucker conditions of an optimization problem. There are also cases in which physicochemical transitions are complementary in nature and can be represented by such a formulation. For instance, the adiabatic compressible flow described by Zaher (1995) can be represented by an equivalent complementarity representation:

$$\begin{bmatrix} P_d - P_f < M_f - 1 \\ M_f - 1 = 0 \end{bmatrix} \vee \begin{bmatrix} P_d - P_f \geq M_f - 1 \\ P_d - P_f = 0 \end{bmatrix} \Leftrightarrow \begin{gathered} (M_f - 1) \cdot (P_d - P_f) = 0 \\ M_f \leq 1 \qquad P_f \geq P_d \end{gathered} \tag{6}$$

On the other hand, if the disjunctive statement has *more than one equation in each disjunctive term*, as the example of the heat exchanger given also by Zaher (1995), the standard complementarity formulation is not equivalent to the disjunctive representation:

/ **(7)**

Notice that in the disjunctive representation all the equations belonging to the same disjunctive term have to be satisfied simultaneously, a restriction which is not represented by the standard complementarity formulation. Next, we propose the representation of disjunctive sets of algebraic equations as a complementarity problem. The formulation we describe not only establishes the complementarity condition among alternative sets of equations, but it also enforces simultaneous satisfaction of all the equations in the solution set. It is important to mention that we are aware of many disadvantages associated with this representation, which we consider at the end of this section. Our motivation is that a complementarity formulation only requires the solution of one square system of nonlinear equations, avoiding the complications encountered in procedural techniques such as the boundary crossing algorithm and the discrete decision making of a MINLP solution. Before going further in the description of our approach, in Figure 2 we explain the relevant terminology employed in this section.

2.1.1 Complementarity Representation of a Conditional Model

As in most of the complementarity approaches reported in the literature, in this paper we assume the nondegeneracy of the solution to the complementarity problem. With this assumption we leave out all those cases in which equations belonging to different terms of the same disjunctive statement are simultaneously satisfied. In addition, we will assume that the range of the conditional equations is positive; that is, we assume that we can rearrange the system of equations so that the residuals of the conditional equations belonging to a non-solution disjunctive term will always be positive.

The complementarity formulation presented here is an extension of that presented by More(1994), who reformulates the nonlinear complementarity problem (2) as:

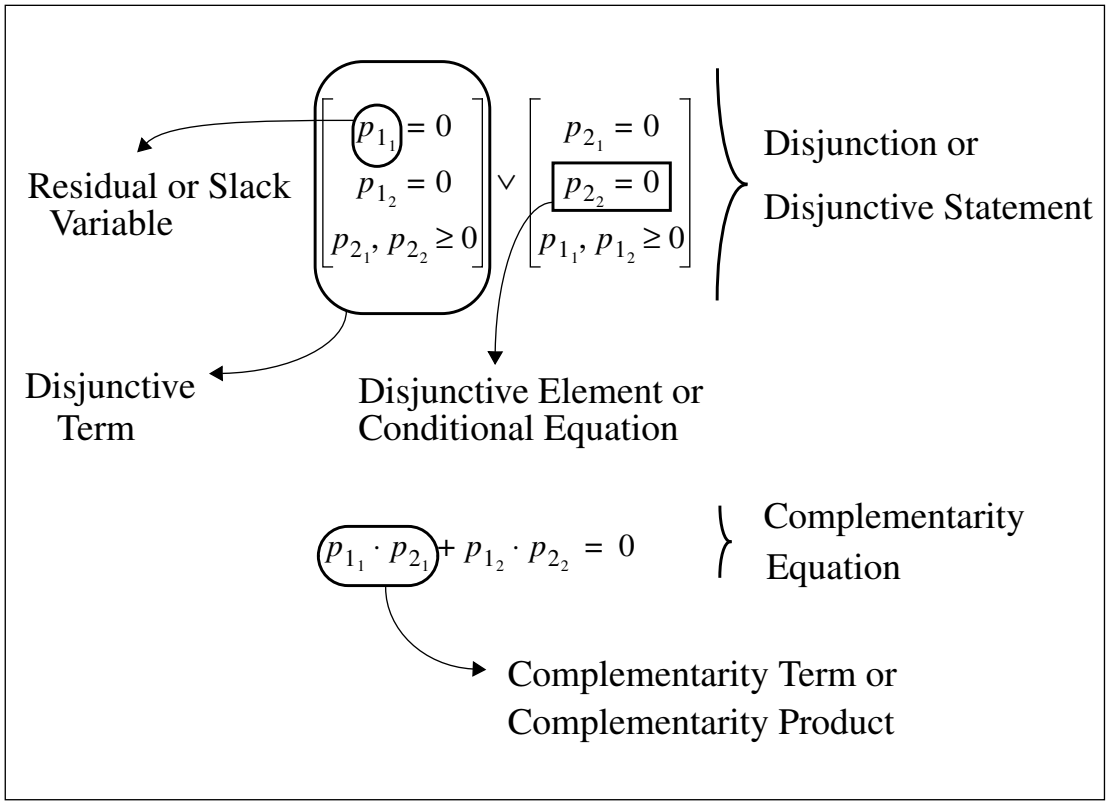
$$\begin{aligned} \underline{r}(\underline{x}) &= \underline{p} \\ P \cdot \underline{x} &= \underline{0} \\ \underline{x} \geq 0 \qquad & \qquad \underline{p} \geq 0 \end{aligned} \tag{8}$$

where $\underline{p}$ is a vector of residual variables and P is the diagonal matrix diag(p_i).

For the purpose of illustration, consider the example of the heat exchanger given by (7). Defining positive residuals p_{i_j} for each of the equations:

$$\begin{aligned} \phi_1 &= p_{1_1} \qquad & 1 - \sum_{i \in C} x_{i_2} &= p_{2_1} \\ \eta_2 - 0.5 &= p_{1_2} \qquad & 0.5 - \eta_1 &= p_{2_2} \end{aligned} \tag{9}$$

FIGURE 2 Description of our terminology.



we then represent the disjunctive statement in terms of these residuals:

$$\begin{bmatrix} p_{1_1} = 0 \\ p_{1_2} = 0 \\ p_{2_1}, p_{2_2} \geq 0 \end{bmatrix} \vee \begin{bmatrix} p_{2_1} = 0 \\ p_{2_2} = 0 \\ p_{1_1}, p_{1_2} \geq 0 \end{bmatrix} \tag{10}$$

Notice that we have been arranged the conditional equations in (9) in such a way that all the variables p_{i_j} are always positive. In order to do that, we used the physical insight given by the nature of the problem. In term of the residual variables p_{i_j}, the standard complementarity formulation is given by:

$$\begin{aligned} p_{1_1} \cdot p_{2_1} &= 0 \\ p_{1_2} \cdot p_{2_2} &= 0 \\ p_{i_j} \geq 0 \qquad i \in [1 \ldots 2] \qquad & j \in [1 \ldots 2] \end{aligned} \tag{11}$$

The disjunctive statement in (10) requires either both p_{1_1} and p_{1_2} or both p_{2_1} and p_{2_2} simultaneously to be zero. That is not a restriction included in the standard complementarity formulation (11). We propose the following formulation to represent the disjunctive statement instead:

$$\begin{aligned} p_{1_1} \cdot p_{2_1} + p_{1_2} \cdot p_{2_2} &= 0 \\ p_{1_1} \cdot p_{2_2} + p_{1_2} \cdot p_{2_1} &= 0 \\ p_{i_j} \geq 0 \qquad i \in [1 \ldots 2] \qquad & j \in [1 \ldots 2] \end{aligned} \tag{12}$$

Since the variables p_{i_j} are all positive, the set of equations in (12) not only contains the complementarity condition given by the standard representation (11) but also enforces the simultaneous satisfaction of all the equations defined in the same terms of the disjunction. Also, it is important to realize that the inequality constraints in (12) are only bounds in the residual variables, and we can use them to guide our search for a solution to the resulting square system of equations:

- The original disjunctive statement in (7) provides 2 equations in either of its cases.

$$\begin{bmatrix} \sum_{i \in C} x_{i_2} + \phi_1 < 1 \\ \phi_1 = 0 \\ \eta_2 = 0.5 \end{bmatrix} \vee \begin{bmatrix} \sum_{i \in C} x_{i_2} + \phi_1 \geq 1 \\ \sum_{i \in C} x_{i_2} = 1 \\ \eta_1 = 0.5 \end{bmatrix}$$

- The complementarity formulation in (9) and (12) provides 6 equations but introduces 4 new variables p_{i_j}, also a net of 2 equations.

$$\begin{aligned} \phi_1 &= p_{1_1} \\ \eta_2 - 0.5 &= p_{1_2} \\ 1 - \sum_{i \in C} x_{i_2} &= p_{2_1} \\ 0.5 - \eta_1 &= p_{2_2} \\ p_{1_1} \cdot p_{2_1} + p_{1_2} \cdot p_{2_2} &= 0 \\ p_{1_1} \cdot p_{2_2} + p_{1_2} \cdot p_{2_1} &= 0 \end{aligned}$$

For cases in which the domain of validity is given by inequality constraints, we reformulate the problem by adding two slacks to the inequality and express the complementarity condition between these slack variables. Consider for example the laminar-turbulent flow transition given by:

$$\begin{bmatrix} Re = 64/f \\ Re \leq 2100 \end{bmatrix} \vee \begin{bmatrix} Re = (0.206307/f)^4 \\ Re > 2100 \end{bmatrix} \quad \textbf{(13)}$$

If we define residuals for the equalities and slack variables for the inequalities:

$$\begin{aligned} Re - 64/f &= p_{1_1} \qquad Re - (0.206307/f)^4 = p_{2_1} \\ Re &= 2100 + p_{1_2} - p_{2_2} \end{aligned} \quad \textbf{(14)}$$

then the disjunctive statement in terms of residuals and slacks and, therefore, the complementarity equations, are exactly the same as those given by (12) for the previous example. Notice that the inequalities become an active part of the system of equations. Also, one of the complementarity

equations contains a complementarity product between the two slacks in the inequality, a requirement to avoid multiple solutions.

The result of applying our formulation is a square system of nonlinear equations (including complementarity equations) subject to the positiveness of the variables p_{i_j}.

The proposed complementarity representation has the following properties:

1. The number of complementarity equations is equal to the number of equations in each term of the disjunction to maintain the same number of degrees of freedom as in the original problem.
2. Every residual is multiplied by every residual in all of the other terms of the disjunction. Thus, we will ensure the simultaneous satisfaction of all the equations in at least one of the disjunctive terms and avoid spurious solutions to the problem.
3. In the example, there are several ways in which we could have accommodated the four complementarity terms in the two complementarity equations. The way in which we have distributed the bilinear terms over the complementarity equations is intended to decrease the possibility of having numerical singularities in the Jacobian of the system while using an iterative scheme based on Newton and quasi-Newton methods. We must avoid having two residual variables from the same disjunctive term being multiplied in two complementarity equations by the same set of residual variables from another disjunctive term. Examine Figure 3. We show a set of poorly formulated complementarity equations and the rows of the Jacobian corresponding to those equations. Note that the equations in Figure 3 contain the same four complementarity terms as the formulation given by (12), but they are grouped differently. In the case presented, if the solution to the problem is $p_{2_1} = p_{2_2} = 0$, rows 1 and 2 of the Jacobian become numerically dependent as the Newton method approaches the solution. If $p_{2_1} + p_{2_2} \sim 0$, row 2 is a multiple of row 1 by the factor p_{1_2}/p_{1_1}. This situation does not occur for the complementarity equations we propose in (12).

Next, we formally describe how to obtain a complementarity formulation including all of the properties outlined above. We first consider the case in which the disjunctive statement contains two terms and then we extend the analysis to any number of disjunctive terms.

FIGURE 3 Numerical singularities in complementarity equations.

$$\begin{matrix} p_{1_1} \cdot p_{2_1} + p_{1_1} \cdot p_{2_2} = 0 \\ p_{1_2} \cdot p_{2_1} + p_{1_2} \cdot p_{2_2} = 0 \end{matrix} \Rightarrow \begin{matrix} p_{1_1} \cdot (p_{2_1} + p_{2_2}) = 0 \\ p_{1_2} \cdot (p_{2_1} + p_{2_2}) = 0 \end{matrix}$$

	p_{1_1}	p_{2_1}	p_{1_2}	p_{2_2}
1	$p_{2_1} + p_{2_2}$	p_{1_1}	0	p_{1_1}
2	0	p_{1_2}	$p_{2_1} + p_{2_2}$	p_{1_2}

2.1.2 Representing the Disjunctive Statements in terms of Positive Residual or Slack Variables

Before generating the set of complementarity equations, it is necessary to define positive residual variables (or slack variables for inequalities) for the conditional equations and to represent the disjunctive statements in terms of these positive variables. This task has to be accomplished disregarding the number of terms in each disjunctive statement. As a matter of fact, one of our assumptions here is that the system of equations can be rearranged to obtain this representation. For instance, for the simplest case of one disjunction with two terms (the index k is omitted for simplicity), given the disjunctive set of algebraic equations:

$$\begin{gathered} \underline{h}(\underline{x}) = 0 \\ \begin{bmatrix} r_{1_j}(\underline{x}) = 0 \\ g_l(\underline{x}) \le 0 \end{bmatrix} \vee \begin{bmatrix} r_{2_j}(\underline{x}) = 0 \\ g_l(\underline{x}) \ge 0 \end{bmatrix} \qquad \begin{matrix} \forall j \in [1 \ldots \beta] \\ \forall l \in [1 \ldots \gamma] \end{matrix} \end{gathered} \tag{15}$$

we reformulate the problem as:

$$
\begin{aligned}
\underline{h}(\underline{x}) &= 0 \\
r_{1_j}(\underline{x}) - p_{1_j} &= 0 \\
r_{2_j}(\underline{x}) - p_{2_j} &= 0 \qquad \forall j \in [1 \ldots \beta],\ l \in [\beta + 1 \ldots \beta + \gamma] \\
g_{l-\beta}(\underline{x}) - p_{1_l} + p_{2_l} &= 0 \\
\begin{bmatrix} p_{1_q} = 0 \\ p_{2_q} \geq 0 \end{bmatrix} &\vee \begin{bmatrix} p_{2_q} = 0 \\ p_{1_q} \geq 0 \end{bmatrix} \qquad \forall q \in [1 \ldots \beta + \gamma]
\end{aligned}
\tag{16}
$$

2.1.3 2-Term Disjunctive Statements

The set of complementarity equations in terms of the positive residual variables p_{i_q} equivalent to the disjunctive statement given in (16) is:

$$
\begin{aligned}
\sum_{t=1}^{\beta+\gamma-s} p_{1_t} \cdot p_{2_{t+s}} + \sum_{t=\beta+\gamma-s+1}^{\beta+\gamma} p_{1_t} \cdot p_{2_{t+s-\beta-\gamma}} &= 0 \qquad \forall s \in [0 \ldots \beta + \gamma - 1] \\
p_{i_q} \geq 0 \qquad \forall i \in [1 \ldots 2],\ q &\in [1 \ldots \beta + \gamma]
\end{aligned}
\tag{17}
$$

Hence, the resulting nonlinear system of equations is:

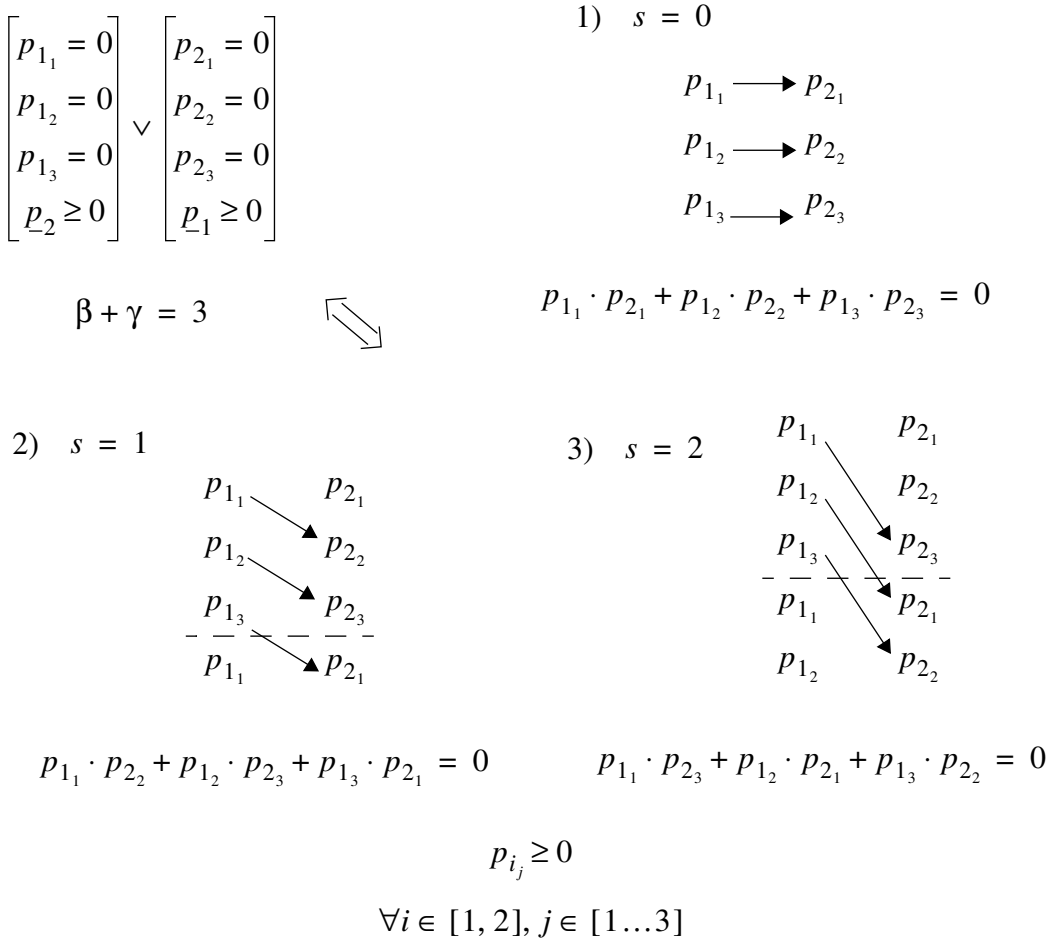
$$
\begin{aligned}
h(\underline{x}) &= 0 \\
\begin{pmatrix} r_{1_j}(\underline{x}) - p_{1_j} \\ r_{2_j}(\underline{x}) - p_{2_j} \\ g_{l-\beta}(\underline{x}) - p_{1_l} + p_{2_l} \end{pmatrix} &= 0 \qquad \forall j \in [1 \ldots \beta],\ l \in [\beta + 1 \ldots \beta + \gamma] \\
\sum_{t=1}^{\beta+\gamma-s} p_{1_t} \cdot p_{2_{t+s}} + \sum_{t=\beta+\gamma-s+1}^{\beta+\gamma} p_{1_t} \cdot p_{2_{t+s-\beta-\gamma}} &= 0 \qquad \forall s \in [0 \ldots \beta + \gamma - 1] \\
p_{i_q} \geq 0 \qquad \forall i \in [1 \ldots 2],\ q &\in [1 \ldots \beta + \gamma]
\end{aligned}
\tag{18}
$$

Note that the resulting nonlinear system of equations is square, and is subject to the bounds in the residual and slack variables. This is illustrated in Appendix A. Appendix A also illustrates how the number of equations of the problem grows with the reformulation of a conditional model as a complementarity problem.

The generation of the complementarity equations given by (17) is illustrated in Figure 4. Basically, in a complementarity equation each residual variable of one disjunctive term is

multiplied by one residual variable of the other disjunctive term, and, for successive complementarity equations, the order of the residual variables in the second term is successively shifted by one.

FIGURE 4 Generation of complementarity equations in a two-term disjunction.



$$p_{i_j} \geq 0$$

$$\forall i \in [1, 2],\ j \in [1 \ldots 3]$$

Every possible complementarity term resulting from the multiplication of positive residual variables belonging to different disjunctive terms is included in complementarity equations (17). This feature ensures that, in order to satisfy the complementarity equations, all the residual variables of at least one disjunctive term have to be simultaneously zero. The proof that (17) satisfies at least one disjunctive term is as follows. Assume that, in each of the disjunctive terms,

there is only one nonzero residual variable. Since all of the possible complementarity terms exist in the complementarity equations, a complementarity term containing those residual variables must exist. Find the complementarity term that contains just these nonzero residual variables. Since the product of those variables will be nonnegative, that term will force the complementarity equation in which it exists to be greater than zero, i.e., it will not be satisfied. To be satisfied, at least one of the residual variables in the complementarity term must be zero, contradicting our original assumption. Thus, in order to satisfy the complementarity equations, at least one disjunctive term must have all its residual variables equal to zero. In other words, a complete set of conditional equations in at least one of the disjunctive terms will be simultaneously satisfied.

As a consequence of the previous analysis, if the complete set of nonlinear equations (including the complementarity equations) is satisfied, then the solution vector $\hat{x}$ will correspond to a consistent solution to the conditional model. Moreover, if we assume uniqueness of the solution, then the vector $\hat{x}$ will be the unique solution to the conditional model.

Another property of the complementarity set of equations given by (17) is that, in every complementarity equation, all the residual variables appear, and each of them appears only once. That is intended to decrease the possibility of having numerical singularities in the Jacobian of the system as explained before. In fact, by analyzing the Jacobian of the formulation (17) under the assumption of nondegeneracy of the solution, it can be shown that the possibility of having numerical singularities is eliminated. If nondegeneracy occurs, the residuals in the equations of the disjunctive set not corresponding to the solution are expected to be different from zero, and, therefore, they will provide a pivot in the jacobian matrix for all the complementarity equations (note that the number of positive residuals is equal to the number of complementarity equations).

2.1.4 Generalization to any number of terms in the Disjunctive Statement

When the disjunctions contain more than two terms, we generate the complementarity equations by applying recursively the same equation given in (17). We assume again that we can obtain a disjunctive statement in terms of positive residual variables. Consider the following simple example of a disjunction with three terms:

$$\begin{bmatrix} p_{1_1} = 0 \\ p_{1_2} = 0 \\ \underline{p}_2, \underline{p}_3 \geq 0 \end{bmatrix} \vee \begin{bmatrix} p_{2_1} = 0 \\ p_{2_2} = 0 \\ \underline{p}_1, \underline{p}_3 \geq 0 \end{bmatrix} \vee \begin{bmatrix} p_{3_1} = 0 \\ p_{3_2} = 0 \\ \underline{p}_1, \underline{p}_2 \geq 0 \end{bmatrix} \tag{19}$$

We apply equation (17) to the first two disjunctive terms:

$$\begin{bmatrix} p_{1_1} = 0 \\ p_{1_2} = 0 \end{bmatrix} \vee \begin{bmatrix} p_{2_1} = 0 \\ p_{2_2} = 0 \end{bmatrix} \vee \begin{bmatrix} p_{3_1} = 0 \\ p_{3_2} = 0 \end{bmatrix} \Leftrightarrow \begin{bmatrix} p_{1_1} \cdot p_{2_1} + p_{1_2} \cdot p_{2_2} = 0 \\ p_{1_1} \cdot p_{2_2} + p_{1_2} \cdot p_{2_1} = 0 \end{bmatrix} \vee \begin{bmatrix} p_{3_1} = 0 \\ p_{3_2} = 0 \end{bmatrix} \tag{20}$$
$$p_{i_j} \geq 0 \qquad \forall i \in [1 \ldots 3],\, j \in [1 \ldots 2]$$

and do it again for the resulting two-term disjunction:

$$\begin{aligned} (p_{1_1} \cdot p_{2_1} + p_{1_2} \cdot p_{2_2}) \cdot p_{3_1} + (p_{1_1} \cdot p_{2_2} + p_{1_2} \cdot p_{2_1}) \cdot p_{3_2} &= 0 \\ (p_{1_1} \cdot p_{2_1} + p_{1_2} \cdot p_{2_2}) \cdot p_{3_2} + (p_{1_1} \cdot p_{2_2} + p_{1_2} \cdot p_{2_1}) \cdot p_{3_1} &= 0 \\ p_{i_j} \geq 0 \qquad \forall i \in [1 \ldots 3],\, j \in [1 \ldots 2] & \end{aligned} \tag{21}$$

A complementarity set of equations obtained in this form still will include the properties outlined before for a two-term disjunction:

4. It will result in a square system of equations.
5. In order to satisfy the complementarity equations, all the equations of at least one set of conditional equations have to be simultaneously satisfied.
6. Under the assumption of nondegeneracy, the Jacobian of the system of equations can be shown to be nonsingular.

The reasoning employed to prove the previous statements is the same as that in the case of a two-term disjunction discussed above.

2.1.5 About the Complementarity Formulation

The complementarity formulation is not without problems:

1. First of all, the problem grows quickly. If (β+γ) is the number of equations in each term of the disjunction and D is the number of terms in the disjunction, the number of equations

representing the disjunctive statement in the complementarity formulation is $(\beta+\gamma)(D+1)$. That number includes both the complementarity equations and the equations defining the positive residual variables.

2. Numerical singularities still arise for cases in which the solution resides on a boundary. That is the main reason for the assumption of nondegeneracy.
3. The reformulation of a conditional model as a complementarity problem is restricted to cases in which the range of the conditional equations is positive. We consider that as the major restriction of this approach. In principle, a cure for this limitation is to define complementarity variables p_{i_q} which can take negative and positive values, and then to use the square of these complementarity variables in the complementarity equations. However, it is known that the use of square terms in complementarity equations is not convenient since the possibility of introducing numerical singularities is even greater. Recall that the derivative of the square of a variable with value zero is also zero.
4. Regardless the number of disjunctive terms in a disjunction, the complementarity equations are 100% dense in the residual variables defined in the conditional equations representing such a disjunction. This creates an upper bound in the problem that can be tackled.
5. The number of bilinear terms (or terms including products among variables) incorporated in each equation also grows with the number of equations in each term of the disjunction. The combinatorial nature of the problem is encapsulated here.
6. The performance of optimization techniques is badly affected by the introduction of nonconvexities (multiplication among variables) to the system of equations.

In general, we are presenting this approach as a very favorable alternative when the number of equations in each disjunctive term is small.

3 A BOUNDARY CROSSING APPROACH

Conventional process models consist of a set of variables and a unique set of m equations that related them. On the other hand, conditional models constitute a means to formulate alternative sets of m equations depending on the values of the modeling variables. The system of equations of the model is different for each of the alternatives. Thus, the main difficulty while dealing with conditional models is that their solution involves simultaneously selecting the equations to be

solved and solving them.

While dealing with conventional models, one generally applies iterative nonlinear techniques in order to solve the equations simultaneously. Newton-like numerical techniques are the methods most commonly applied to find the solution of conventional models. Such methods require both continuity and differentiability of the system of equations over the entire feasible region in order to guarantee convergence. In conditional models, however, because of the loss of continuous functionality between the surfaces of neighboring subregions in the search space, Newton-like iterative techniques cannot be applied if the solution to the system requires a step from one subregion to another.

3.1 THE BOUNDARY CROSSING ALGORITHM

Zaher (1991) introduced a boundary crossing algorithm as an alternative approach to the solution of conditional models. In the most recent of his works, Zaher (1995) describes the boundary crossing algorithm in the context of an optimization problem. He recognizes the problem of solving a conditional model as one of nondifferentiable optimization, and proposes an augmented-Lagrangian SQP approach for the solution of such a problem. A thorough report of the results obtained by that approach is in preparation. In this work, we focus in the boundary crossing algorithm as a technique for the solution of a fully determined nonlinear system of equations. Here we show that, by isolating in boundaries the nondifferentiable functionality of conditional models, the solution of such problems can be tackled by an algorithm mainly based on the use of conventional nonlinear techniques (such as Newton-based methods).

3.1.1 The Formulation

In conditional models, the equations within each alternative set are confined to a subset of the feasible region. In the boundary crossing algorithm, a set of boundary expressions of the form $\underline{b}(\underline{x}) \geq 0$ is used to associate each set of conditional equations to each of the subregions. Each boundary expression partitions the feasible region into two neighboring subregions, each of the subregions characterized by either the satisfaction or non-satisfaction of the expression. Thus, each subregion in the feasible region is identified by a unique combination of truth values for all of the boundary expressions. In this way, the partitioning of the feasible region into subregions is combinatorial. If l is the number of boundary expression (i.e. $b_i(\underline{x}) \geq 0$ where $i \in \{1 \ldots l\}$), the

number of subregions is 2^l. For most of the conditional models, though, it is physically not meaningful for the solution to lie in some of the subregions. These subregions, therefore, should be removed from the feasible region and no conditional equations need to be associated to them. One must emphasize that the boundary expressions are not inequality constraints but rather are used to associate values of the modeling variables with the appropriate set of equations. A way of identifying a specific subregion is by using a set s where $s \in \{1 \ldots l\}$. The set s contains the indices of the boundary expressions which must be satisfied in order for the solution to reside in that specific subregion. If R is the set of all of the subsets s belonging to the feasible region with which m conditional equations are to be associated in the form $\underline{r}_s(\underline{x}) = 0$, then the conditional model can be formulated in the following disjunctive representation:

$$\bigvee_{s \in R} \begin{bmatrix} b_i(\underline{x}) \geq 0, \forall i \in s \\ b_i(\underline{x}) < 0, \forall i \in \{1 \ldots l\} - s \\ \underline{r}_s(\underline{x}) = 0 \end{bmatrix} \tag{22}$$

Note that, when the number of boundary expressions is zero, only one subregion is created using $s = \varnothing$ and the previous formulation reduces to a conventional nonlinear system of equations.

Since frequently a large number of equations are common to all of the subregions, it is convenient to include those invariant equations outside the disjunction:

$$\begin{gathered} \underline{g}(\underline{x}) = 0 \\ \bigvee_{s \in R} \begin{bmatrix} b_i(\underline{x}) \geq 0, \forall i \in s \\ b_i(\underline{x}) < 0, \forall i \in \{1 \ldots l\} - s \\ \underline{r}_s(\underline{x}) = 0 \end{bmatrix} \end{gathered} \tag{23}$$

where now $\underline{g}(\underline{x})$ is m_g dimensional and $\underline{r}_s(\underline{x})$ is m_r dimensional such that $m_g + m_r = m$. Note that the formulations given by (22) and (23) represent one disjunction over all of the subregions in the feasible region of a conditional model. Such a global disjunctive statement can be generated from the combination of all of the individual disjunctions in the problem. However, it is not necessary to formulate the conditional model strictly in the form given by (22) or (23) in order to solve it by using the boundary crossing algorithm. A representation such as (1), in which each

individual disjunction is explicitly formulated, is also suitable. Note, however, that the domain of validity of the conditional equations can be expressed in terms of a set of boundary expressions, as required by the boundary crossing algorithm, instead of a set of inequality constraints as given by (1).

3.1.2 The Solution Algorithm

The goal of a conditional model solver is to find a solution to the disjunctive system of equations (23) (or the set of equations given by (1)), a solution which consists in a vector of variables satisfying the invariant set of equations and exactly one set of conditional equations, providing that the truth values of the boundary expressions are consistent with the set of variant equations constituting the solution.

The most popular iterative numerical techniques for the solution of a system of nonlinear equations involve strategies designed to enforce descent of some objective function with each iteration. This objective function is generally chosen as the square norm of the residuals of the model equations, assuming the equations are all appropriately scaled. Hence, for each subregion the objective function $\phi_s(\underline{x})$ is defined by:

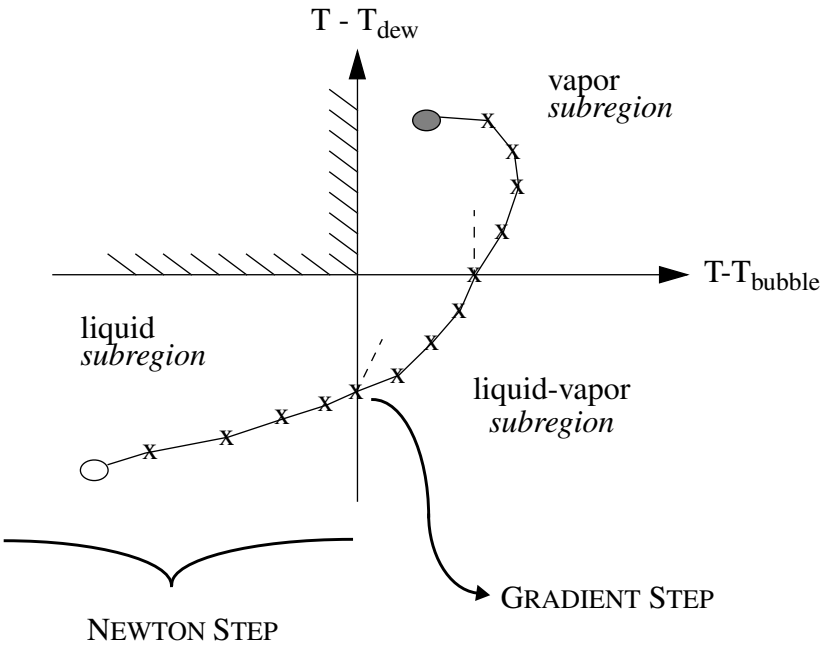
$$\underline{f}_s(\underline{x}) = \begin{bmatrix} \underline{g}(\underline{x}) \\ \underline{r}_s(\underline{x}) \end{bmatrix}$$
$$\phi_s(\underline{x}) = \frac{1}{2} \cdot \underline{f}_s^T(\underline{x}) \cdot \underline{f}_s(\underline{x}) \qquad \textbf{(24)}$$

In the boundary crossing algorithm, all of the points where the functionality is nondifferentiable are characterized by boundary hyper-surfaces as given by the boundary expressions $\underline{b}(\underline{x}) \geq 0$. Therefore, the equations within each subregion form a continuous and differentiable objective function. Hence, a Newton-based numerical technique may find the solution when the initial point is internal to the correct subregion. On the other hand, if one places the initial point inside an incorrect subregion, one could expect to encounter a subregion boundary in a finite number of iterations.

According to the above discussion, the boundary crossing algorithm applies a Newton-like step for all the points internal to a subregion, taking advantage of the superlinear convergence feature

of such a step. If during the iterative process a boundary is reached, then a Newton-like step cannot be taken because of the nondifferentiability of the objective function at that point. A boundary crossing based on subgradient methods (linear convergence methods) takes place instead. Once off the boundary, the superlinear convergence steps resume using a possibly new set of equations. Figure 5 illustrates the boundary crossing algorithm.

FIGURE 5 The Boundary Crossing Algorithm in a simple flash equilibrium calculation.



3.1.3 Boundary Crossing

In the boundary crossing algorithm, because of the lack of differentiability at a boundary, an analogue of the steepest descent vector is derived in order to advance the solution progress into the correct subregion.

While solving continuous and differentiable models, the steepest descent direction at some point with values $\underline{a}$ for the variables is in the opposite direction of the gradient of the objective function at that point. Hence, a vector $\underline{d}$ with a direction of steepest descent direction is a vector which solves the following problem:

$$\begin{aligned} min \quad & \nabla_x \phi(\underline{a})^T \cdot \underline{d} \\ s.t. \quad & \underline{d}^T \cdot \underline{d} \leq 1 \end{aligned} \tag{25}$$

When a descent direction exists at $\underline{a}$, the solution vector $\underline{d}$ has the same direction as $-\nabla_x \phi(\underline{a})$ but it is of unit length. In other words, the steepest descent direction for a differentiable function is the direction which yields a minimum in the directional derivative.

For a conditional model, each subregion has a different value for the directional derivative at a boundary. For that case, Zaher (1991) defines the directional derivative as the maximum over all of the subregion directional derivatives. If B is the set of all of the subregions in the neighborhood of a boundary, then the directional derivative for the problem at such boundary is given by:

$$\begin{aligned} max \quad & \nabla_x \phi_i(\underline{a})^T \cdot \underline{d} \\ s.t. \quad & i \in B \end{aligned} \tag{26}$$

Hence, the minimum of this directional derivative constitutes the steepest descent direction at the boundary:

$$\begin{aligned} min \quad & \begin{pmatrix} max & \nabla_x \phi_i(\underline{a})^T \cdot \underline{d} \\ s.t. & i \in B \end{pmatrix} \\ s.t. \quad & \underline{d}^T \cdot \underline{d} \leq 1 \end{aligned} \tag{27}$$

The definition of directional derivative given by (26) aims for a descent direction with respect to all the subregions neighboring the boundary. By doing that, one will prevent a cycling situation in which a loop of connected subregions are visited continually. The problem given by (27) is recognized as a mini-max problem. Because of the convexity of this problem, one can derive the dual problem to obtain:

$$\begin{aligned} min \quad & \underline{d}^T \cdot \underline{d} \\ s.t. \quad & \underline{d} = -N \cdot \underline{\alpha} \\ & \underline{e}^T \cdot \underline{\alpha} = 1 \\ & \alpha_i \geq 0, \forall i \in B \end{aligned} \tag{28}$$

where $\underline{e}$ is a vector with all its elements equal to one and N is a matrix whose columns are the vectors of gradients of the subregions in the neighborhood of the current boundary. That is, column i of N is the vector $\nabla_x \phi_i(\underline{a})$, $\forall i \in B$. Also, both $\underline{e}$ and $\underline{\alpha}$ are vectors with dimension equal to the number of subregions neighboring the current boundary. The derivation of equation (28) from (27) has been described by Zaher (1991,1995) and it is given in Appendix B. If all the gradients of the subregions cannot be confined to one side of the boundary hyper-surface, the result of (28) is a minimum of zero and we terminate the algorithm. We terminate because no descent direction with respect to all the neighboring subregions has been found. Otherwise, If the result of (28) is nonzero, the linear combination of the elements of the vector $\underline{\alpha}$ is used to generate the steepest descent direction, along which one should move away from the boundary.

3.1.4 Assumption of Continuity in the Boundary Crossing Algorithm

In the boundary crossing algorithm, one has to assume the continuity of the conditional model. In conditional models, continuity is ensured as follows: if the solution to the equations of an arbitrary subregion lies on a boundary, then that solution must satisfy the equations of all of the subregions neighboring that boundary.

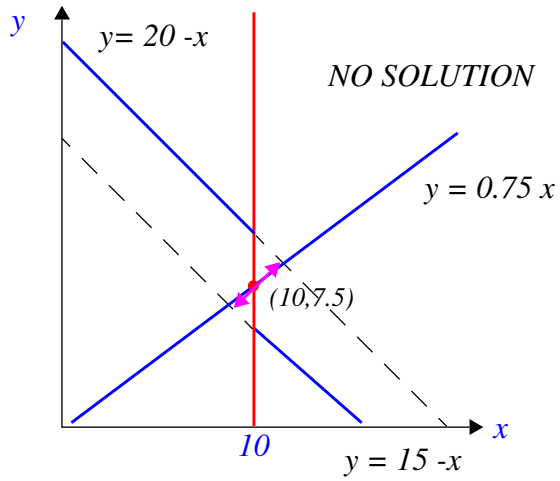
The reason for this assumption of continuity is best described by analyzing Figure 6. Figure 6 presents two cases in which continuity is not preserved in the conditional model and discovers the potential difficulties of such cases. In case (a) there is no solution to the problem, and in case (b) the conditional model has two solutions. Note that, in both cases in Figure 6, the vectors of gradients of the subregions neighboring the boundary are of opposite direction at the point *(x=10,y=7.5)*. If we solve problem (28) for such a point at the boundary, the solution of (28) would be zero in both cases, and the boundary crossing algorithm would terminate. Hence, for the case (b), the boundary crossing algorithm would terminate without finding any of the two existing solutions to the conditional model. A possible cure for this complication would be to search in each neighboring subregion for a solution interior to any of them. Such an approach has been considered throughout this research, but it is not yet a part of the implementation that we describe in this paper.

4 IMPLEMENTING THE BOUNDARY CROSSING ALGORITHM IN AN EQUATION BASED ENVIRONMENT

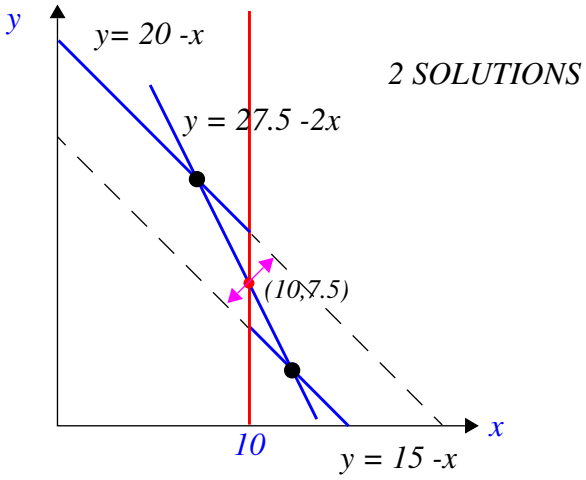
In this section, we present some of the details of the implementation of the boundary crossing algorithm in an equation-based environment.

FIGURE 6 Vectors of gradients are of opposite direction at a boundary.

(a)



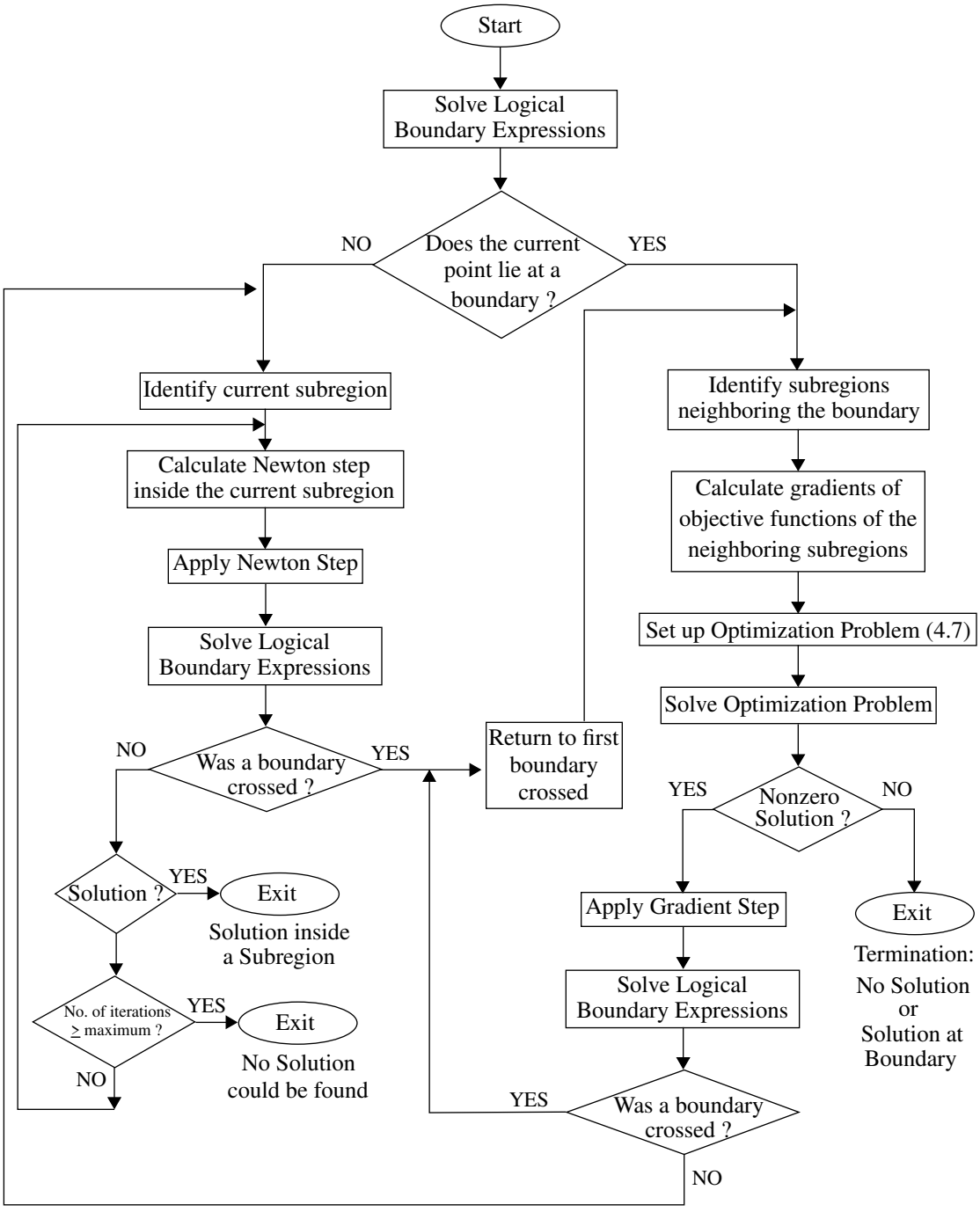
(b)



In Figure 7, we show a flowsheet of the main activities that a solver applying the boundary crossing algorithm must perform. Given the initial guess for the model variables, the algorithm starts by evaluating the logical boundary expressions in order to determine whether that initial guess resides on a boundary or it is internal to any of the subregions:

- If the point resides on a boundary, a subgradient step has to be calculated. In order to do that, the equations and variables of all of the subregions neighboring the boundary have to be identified and differentiated. The problem given by (28) is then generated and solved to obtain the steepest descent direction. If the result of (28) is a minimum of zero, the algorithm terminates concluding that no descent direction could be found for the current point on the boundary. We want to emphasize that the algorithm terminates when the result of the optimization problem is zero because we want to guarantee termination. In such a case, the point on the boundary might not constitute a solution to the conditional model, as we describe in section 3.1.4. If the solution of (28) is nonzero, the subgradient step is taken. After this step, it is necessary to verify is one or several boundaries have been crossed. If so, the subgradient step is reduced until the problem lies exactly on the first boundary crossed, and a subgradient step must be calculated for the new boundary. If no boundary has been crossed, the algorithm proceeds to identify the variables and equations of the current subregion and a nonlinear conventional technique is the used to move towards the solution as explained below.
- If the point does not reside on a boundary, it is necessary to identify the initial subregion. That means that we need to identify the equations and variables constituting the initial problem. The algorithm assumes that the starting point corresponds to a subregion for which the system of equations are defined, i.e. the initial subregion is feasible. An iterative technique is then used in order to generate a Newton-like step towards the solution. After each step, one needs to verify if some of the boundaries have been crossed. If no boundary has been crossed, the numerical technique continues the iterative process in the current subregion until the convergence criteria is satisfied (termination of the algorithm) or until a boundary is encountered. When a boundary has been crossed, the length of the Newton-like step has to be reduced until the system resides exactly on the boundary, and then a boundary crossing has to be performed by using a subgradient step as explained above.

FIGURE 7 Flowsheet of the boundary crossing algorithm implementation.



An efficient implementation of this algorithm in an equation-based environment is hard for the following reasons:

1. The implementation requires the incorporation of modeling tools which enables the user of an equation-based environment to represent conditional models. Mainly, these modeling tools have to be capable of representing alternative sets of equations and the logical conditions given by the boundary expressions.
2. The boundary crossing algorithm involves the performance of several different tasks, including logical analysis, continuous reconfiguration of the equations and variables of the system (switching among alternative subregions), performance of a Newton-like step (which requires the incorporation of a conventional nonlinear technique) and performance of a subgradient step (which requires the solution of an optimization problem). Thus, in the implementation of this technique one has to provide the numerical and algorithmic tools needed to execute each of the previous tasks and to integrate them in a procedural solving engine.

4.1 THE MODELING TOOLS

In this work, we use of the modeling tools described in by Rico-Ramírez et. al. (1997) which allow for the representation of conditional models in an equation based environment Those modeling capabilities meet the representation needs of the boundary crossing algorithm:

1. Boundary expressions can be represented by combining the CONDITION statement and logical relations.
2. Alternative set of model equations can be represented by using the WHEN statement.

These tools have already been incorporated into the ASCEND IV modeling environment (Westerberg et al., 1994; Allan, 1997), hence, such a modeling language was used in this work for representation purposes.

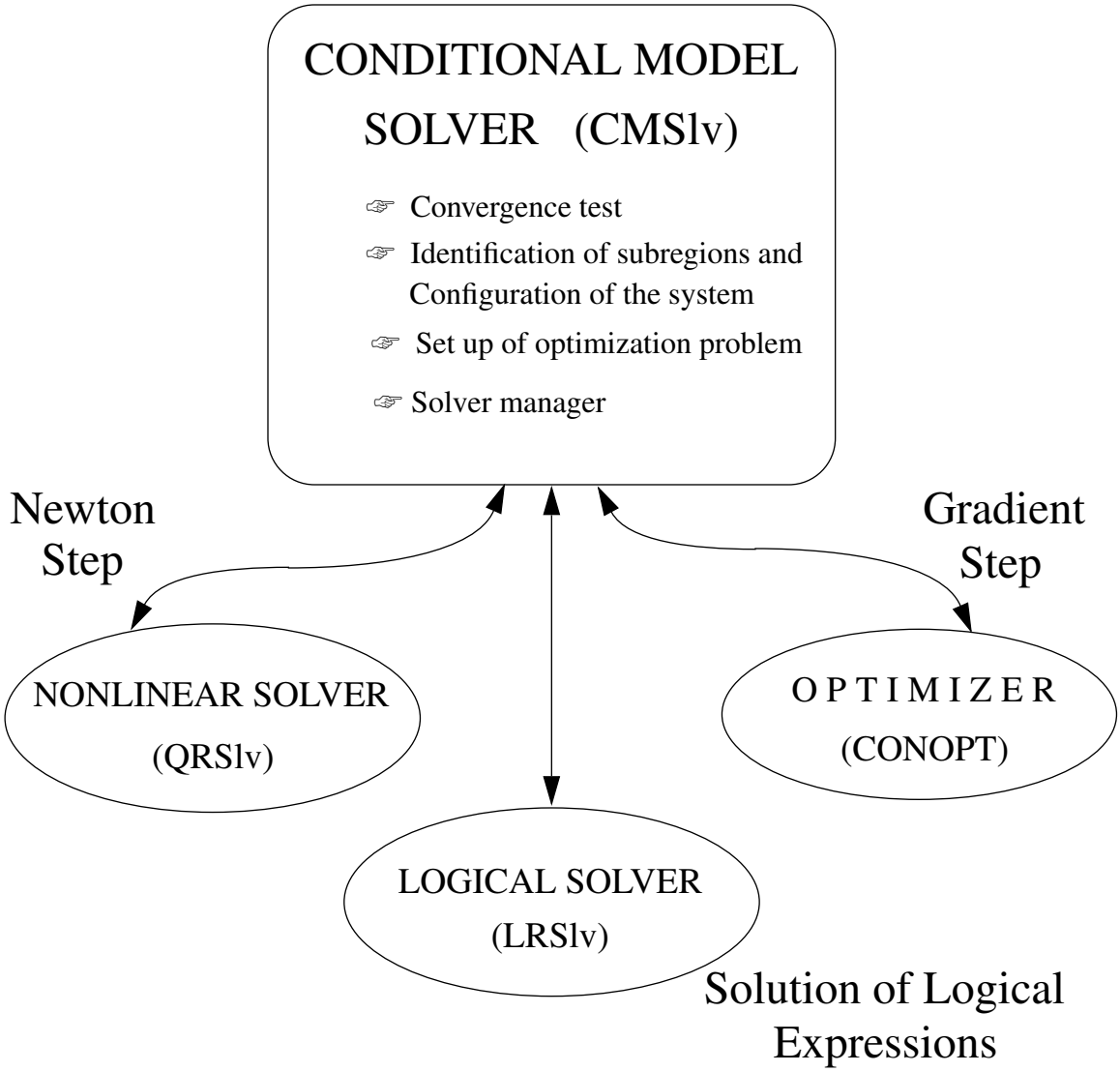
4.2 THE SOLVING ENGINE

We implement our conditional modeling solver in a server-client, object oriented architecture. Figure 8 describes this implementation. This conditional modeling solver (CMSlv) is in charge of the management of external entities which, at the same time, are in charge of executing each of

the specific tasks that the algorithm requires. Particularly, calls to external entities are done for the execution of:

1. The analysis of logical boundary expressions.
2. The calculation of a quadratic converge step inside a subregion (Newton-like iterative technique).
3. The calculation of a subgradient step by solving an optimization subproblem.

FIGURE 8 Object-oriented architecture of the boundary crossing implementation.



In addition, the conditional modeling solver is also in charge of the re/configuration of the appropriate set of equations and variables at each step of the algorithm, convergence tests, and the generation of the optimization subproblem when the system resides on a boundary. In the remainder of this section, we will describe in more detail each of these tasks as well as the external entities created and/or used to provide support to the conditional solver CMSlv.

4.2.1 Logical Analysis

The domain of validity of the alternative sets of equations in a conditional model is given in terms of the truth value of a set of logical boundary expressions. Hence, logical expressions have to be continuously evaluated during the iterative solution scheme. In order to perform this task, we created the logical solver LRSlv. The ASCEND modeling language allows a logical boundary expression to be represented in terms of a combination of:

- Boolean variables.
- Boolean operators.
- The truth value of conditions expressed in term of real variables.

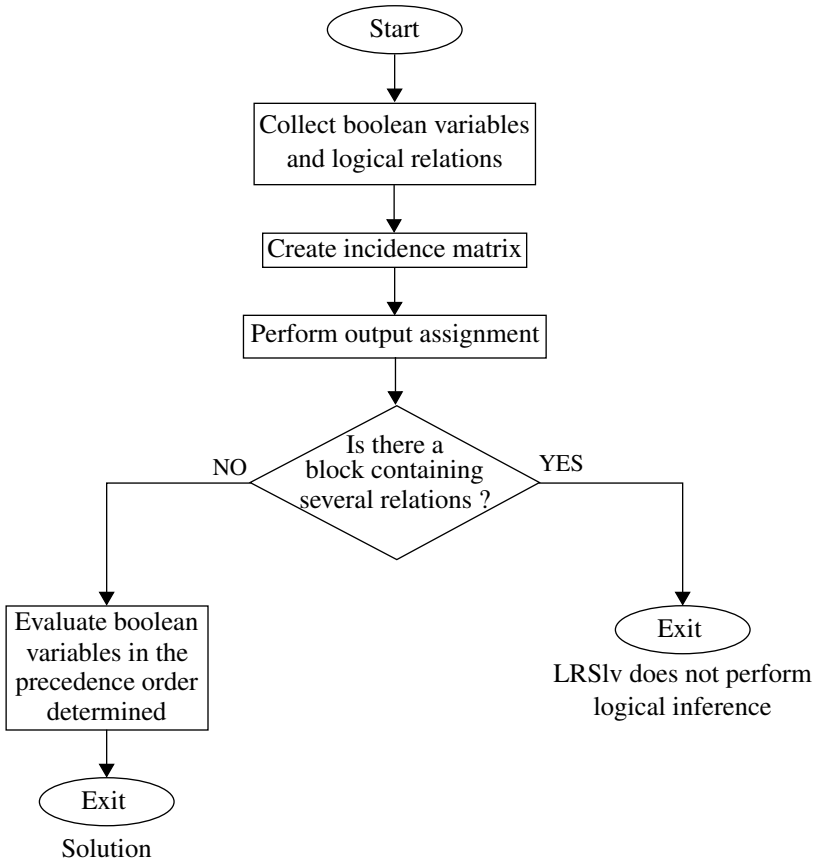
Hence, the logical solver LRSlv has been implemented in such a way that it is able to handle this type of flexibility. Figure 9 shows the flowsheet of the activities performed by LRSlv. In this implementation, we assume that we can explicitly evaluate all the boolean variables in some precedence order. In other words, LRSlv does not perform logical inference. Thus, while finding the value of a boolean variable, we require that the truth values of all the boolean variables and conditions on which each boolean variable depends to have been previously given or previously evaluated. Also, we assume that we can evaluate at any time the truth value of a condition expressed in terms of real variables. In general, both of our assumptions will be met while the boundary crossing algorithm is being used to solve conditional models. LRSlv has been attached as an external entity providing support to CMSlv as explained above.

4.2.2 Reconfiguration of the System of Equations

The conditional solver is capable of determining whether the current point resides on a boundary or is internal to any of the subregions. If the point is internal to any of the subregions, CMSlv must identify the equations and variables constituting the problem in that specific subregion. To perform this task, CMSlv uses the notion of active equation and active variable. A consistent

mechanism to select the structure of the system inside a specific subregion is described in Rico-Ramírez et. al. (1997). That mechanism is used in this work.

FIGURE 9 Implementation of LRSlv.



We emphasize that a change in the configuration of the problem during an iterative solution process may also cause a change in the partitioning of the variables and in the partitioning of the equations. Zaher (1995) developed algorithms for simplifying the partitioning of variables and equations in a conditional model, but those algorithms are not part of this implementation yet.

4.2.3 Conventional Nonlinear Solver: Newton-like Step

When the values of the variables of the model correspond to a point internal to any of the

subregions, the boundary crossing algorithm requires the calculation of a Newton-like step. CMSlv calls an external solver to perform this task. We do not impose any restrictions for the selection of this external solver. CMSlv can be able to interact with any solver which is able to calculate a step based on derivative and function evaluations and on the structural information of the system of equations. In the current implementation of CMSlv, the Newton-like step is calculated by the ASCEND nonlinear solver QRSlv (Westerberg, 1989). In order to obtain the descent of the objective function local to the subregion, QRSlv uses the modification to the Levenberg-Marquardt algorithm given by Westerberg and Director (1979).

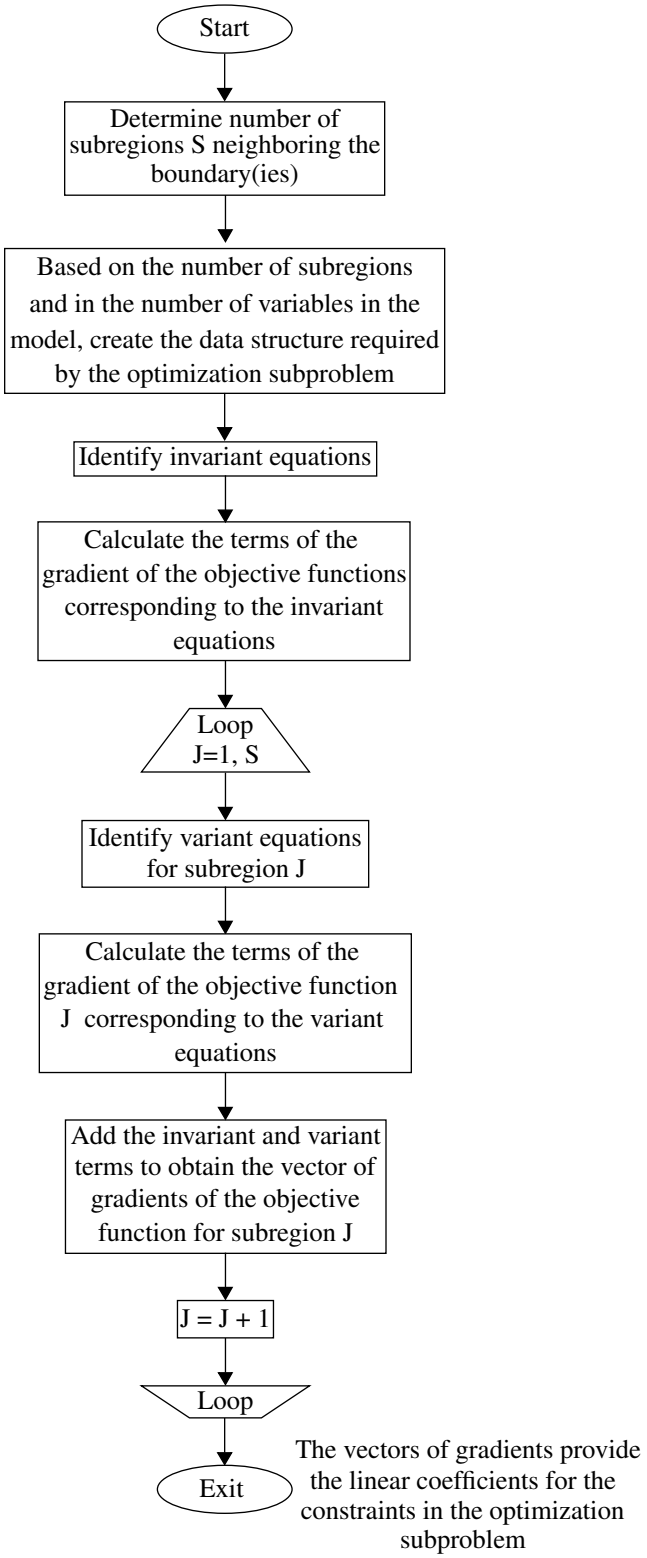
4.2.4 Setting Up the Optimization Subproblem

When CMSlv determines that the current point of the system resides on a boundary, a subgradient step has to be calculated by solving an optimization subproblem. Setting up this optimization subproblem requires the calculation of the gradients of the objective functions of all of the subregions neighboring a boundary. Also, it results in a system of equations which is quite different from the system of equations of any of the subregions of the original problem. That implies that the data structures required to provide the information about this optimization subproblem have to be dynamically created and destroyed. In addition, in order to minimize the number of function and gradient evaluations employed for setting up each optimization subproblem, it is necessary to identify the invariant set of equations (if there is one) and to avoid multiple evaluations for those equations. Figure 10 presents the flowsheet of the activities performed by CMSlv while setting up an optimization subproblem at a boundary.

4.2.5 Solution of Optimization Subproblems: Subgradient Step

After CMSlv has created the optimization subproblem at a boundary, this subproblem is solved with a call to another external entity. This time, the external solver must be an optimizer. Once again, there are no limitations in the implementation of CMSlv as to which optimizer should be used. The current implementation interacts with the subroutine CONOPT (Drud, 1985), which uses a reduced gradient approach to the solution of the optimization problem. CMSlv provides the values of the derivatives of the linear constraints (elements of the vectors of gradients of the objective functions), the residual of these constraints, and the derivatives and values of the objective function. CONOPT (like QRSlv in the case of the calculation of the Newton-like step) is considered a black box which provides the subgradient step based on that information.

FIGURE 10 Setting up the optimization subproblem



4.2.6 Termination

Termination of the algorithm can occur on a point internal to any subregion or on a point residing on a boundary. It occurs on a boundary when the result of the optimization subproblem is zero, or internal to a subregion when the norm of the residuals of the equations in that subregion is less than a specified tolerance. As we already mentioned, if the algorithm terminates at a point lying on a boundary, such a point on a boundary might not constitute a solution to the conditional model as described in section 3.1.4. CMSlv is in charge of the convergence test after both the subgradient step and the Newton-like step.

4.2.7 About CMSlv

Besides tasks like the reconfiguration of the system, termination tests, setting up of the optimization problem, etc., CMSlv is in charge of managing the logic of the algorithm and interacting with the external solvers. It provides the information required by the external solvers (LRSLv, QRSlv and CONOPT) and processes the results obtained from the calls to them. The conditional modeling solver (CMSlv) and the logical analyzer (LRSlv) have been attached to the ASCEND environment.

5 ILLUSTRATIVE EXAMPLES

We have represented and solved several examples of algebraic systems of disjunctive equations found in the literature by using a complementarity formulation and the boundary crossing algorithm.

Appendix C presents the complementarity equations (or a representative part of them) for each of those examples. In examples 1 through 4 the disjunctive statements contain only two terms, and we generated the complementarity equations by strictly using the formulation proposed in (17). The degree of complexity increases in examples 5 and 6 since the disjunctive statements contain three terms. In those cases, we added two slacks to the conditional equations in one of the disjunctive terms which will allow some of the conditional equations to have a negative residual. In examples 5 and 6 we do not use the general formulation proposed for three-term disjunctions. Instead, we use those examples to show how sometimes the specific structure of the disjunctive statement can be used to simplify the resulting system of complementarity equations. When

solving the complementarity formulation of the conditional models, only one nonlinear system of equations is required. Also, in all the examples the number of equations in each disjunctive term is very small, and, as a consequence, the complexity of the complementarity equations is not as bad as can be expected with problems of larger size.

Also, in Appendix D, we show a representative section of the ASCEND models used by the boundary crossing algorithm for each of the examples.

In all of the conditional models, the initial values of the variables correspond to an incorrect system of equations. We do that in order to test the ability of the boundary crossing algorithm for crossing boundaries until finding the correct subregion and iterating until obtaining the solution to the problem. In particular, the initial values for Example 4 are those defined in Bullard and Biegler (1992). Examples 1 through 3 were introduced by Zaher(1995). In that work, Zaher solves example 1 and 3 as optimization problems in order to test the performance of his approach to sequential quadratic programming. In this work we solve those problems as simulation problems by eliminating the objective function and defining fixed values for the degrees of freedom.

EXAMPLE 1 Fluid Transition (Zaher, 1995).

This example describes the flow of a compressible gas in an adiabatic frictional circular pipe of constant diameter. Nonsmooth functionality occurs due to the possible transition between sonic-subsonic flow at the outlet of the pipe. The alternatives for the solution of the problem are represented by:

$$\begin{bmatrix} P_d - P_f < M_f - 1 \\ M_f - 1 = 0 \end{bmatrix} \vee \begin{bmatrix} P_d - P_f \geq M_f - 1 \\ P_d - P_f = 0 \end{bmatrix}$$

in which one of the terms corresponds to sonic flow (Match number $M_f=1$) and the other to subsonic flow ($P_d=P_f$). The equations describing the thermodynamics are omitted for simplicity and can be found in Zaher (1995). This example involves only one boundary expression (and, therefore, only $2^1=2$ subregions) and contains only one conditional equation in each alternative

set of variant equations.

EXAMPLE 2 Phase Equilibria (Zaher, 1995).

An isothermal flash is applied to a ternary system involving benzene, ethanol and water. According to the phase diagram of this mixture and, depending on the values of pressure and temperature, three phases can be expected to exist simultaneously, an aqueous liquid phase, an organic liquid phase, and a vapor phase. The existence or nonexistence of each phase can be represented as a conditional statement. For instance, to represent the existence of the aqueous phase, the following statement applies:

$$\begin{bmatrix} \sum_{i \in C} y_{i_A} + \phi_A < 1 \\ \phi_A = 0 \end{bmatrix} \vee \begin{bmatrix} \sum_{i \in C} y_{i_A} + \phi_A \geq 1 \\ \sum_{i \in C} y_{i_A} = 1 \end{bmatrix}$$

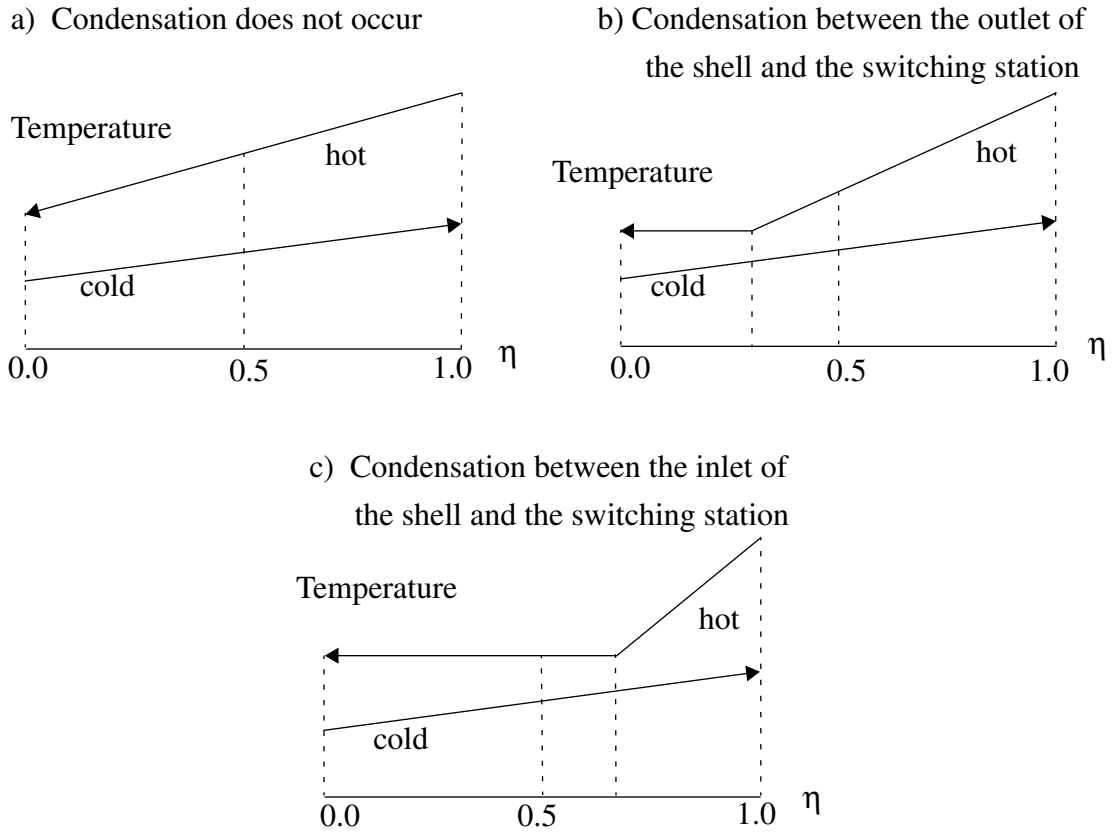
as obtained by Michelsen (1982) and Zaher(1995). For a phase A, ϕ_A represents the fraction of the phase and $\underline{y}_A$ is a vector representing the compositions. Since there are three possible phases, we require three similar disjunctions to represent the behavior. There is only one equation in each disjunction though and, therefore, only one complementarity equation for each of them is required. Also, since we require three disjunctions to represent the behavior, the search space of the boundary crossing algorithm contains $2^3=8$ subregions

EXAMPLE 3 Heat Exchanger (Zaher, 1995).

A detailed explanation of this example can be found in Zaher (1995). It represents a case in which a conditional model contains differential equations that have to be integrated. The approach suggested is to discretize the differential equations and treat the problem as a conditional model with only algebraic equations. To accomplish this, Zaher (1995) introduced a “relay” method: the point in the domain of integration where transition occurs is continuously passed along, as a baton in a relay race, from one element to another by successive contractions and expansions of the individual elements. Switching stations at which the analogous baton transfer occurs must first be positioned. This example is introduced in Figure 11. Three finite elements are chosen with one switching station. To outline the three elements, four positions referenced by the indices {0...3}

are used. The domain of integration is transformed to the dimensionless variable η which varies from zero to one. The difficulty with this model is that, in addition to solving for the temperature profile, the dimension of the finite elements are to be solved for as well.

FIGURE 11 Alternative heat exchanger temperature profiles.



Zaher (1995) shows that the three cases shown in Figure 11 can be represented as a conditional model including the following disjunctive statement:

$$\begin{bmatrix} \sum_{i \in C} x_{i_0} + \phi_0 < 1 \\ \sum_{i \in C} x_{i_2} + \phi_1 < 1 \\ \phi_0 = 0 \\ \eta_1 = 0 \\ \eta_2 = 0.5 \\ \phi_1 = 0 \end{bmatrix} \vee \begin{bmatrix} \sum_{i \in C} x_{i_0} + \phi_0 \geq 1 \\ \sum_{i \in C} x_{i_2} + \phi_1 < 1 \\ \sum_{i \in C} x_{i_0} = 1 \\ \sum_{i \in C} x_{i_1} = 1 \\ \eta_2 = 0.5 \\ \phi_1 = 0 \end{bmatrix} \vee \begin{bmatrix} \sum_{i \in C} x_{i_0} + \phi_0 \geq 1 \\ \sum_{i \in C} x_{i_2} + \phi_1 \geq 1 \\ \sum_{i \in C} x_{i_0} = 1 \\ \sum_{i \in C} x_{i_1} = 1 \\ \eta_1 = 0.5 \\ \sum_{i \in C} x_{i_2} = 1 \end{bmatrix}$$

where ϕ represents the fraction of the hot stream which is condensed and x is a vector representing the composition of the condensation droplets. As described above, the four positions are referenced by the indices $\{0...3\}$. There are 2 boundary expressions in the model which would involve 4 subregions in the search space. However, one of the subregions is infeasible and can be eliminated from the search space.

EXAMPLE 4 Pipeline Network (Bullard and Biegler, 1992)

Consider the pipe network shown in Figure 12 solved previously by Bullard and Biegler (1992). This problem can be described by the system of equations:

$$\sum_j Q_{ij} + \sum_j Q_{ji} = w_i \qquad \forall node\ i$$

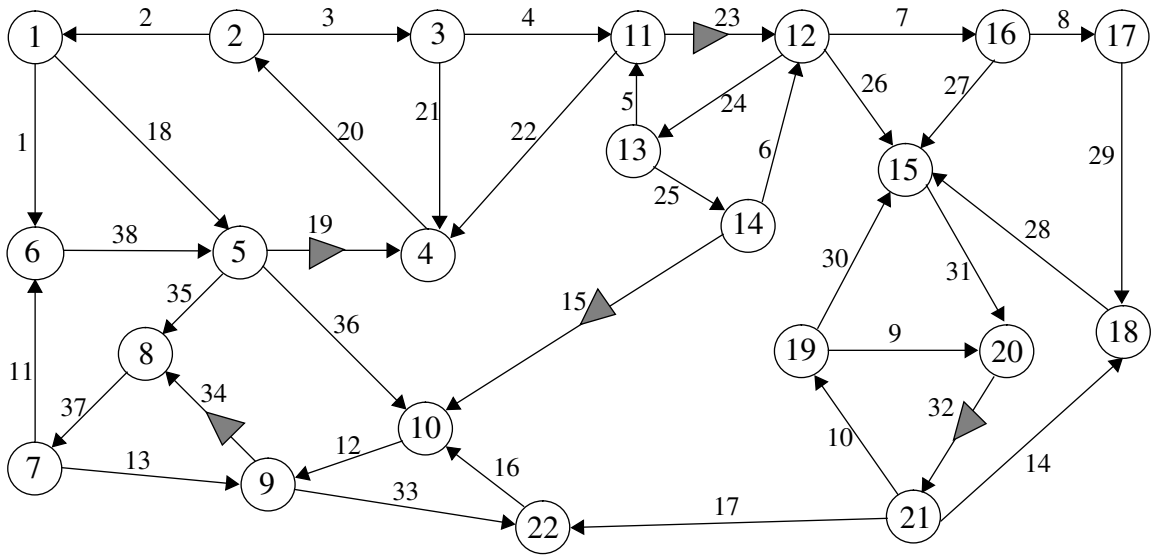
$$H_{ij} = K \cdot sign(Q_{ij}) \cdot {Q_{ij}}^2 \qquad \forall arc\ ij\ without\ valve$$

$$\begin{bmatrix} K \cdot {Q_{ij}}^2 = 0 \\ H_{ij} \leq 0 \end{bmatrix} \vee \begin{bmatrix} K \cdot {Q_{ij}}^2 = H_{ij} \\ H_{ij} \geq 0 \end{bmatrix} \qquad \forall arc\ ij\ with\ valve$$

$$H_{ij} = P_i - P_j \qquad \forall arc\ ij$$

$$Q_{ij} \geq 0 \qquad \forall arc\ ij\ with\ valve$$

FIGURE 12 Pipe network with five check valves.



The first equation is a flow balance around each node, the second is the Hazen-Williams relation for pipes with no valve, and the third is the relation between pressure drop and flowrate. Notice that an equivalent disjunctive representation for the Hazen-Williams relations can be given by:

$$\begin{bmatrix} H_{ij} = -K \cdot Q_{ij}^{2} \\ Q_{ij} \le 0 \end{bmatrix} \vee \begin{bmatrix} H_{ij} = K \cdot Q_{ij}^{2} \\ Q_{ij} \ge 0 \end{bmatrix} \qquad \forall arc\ ij\ without\ valve$$

All pipes are 100 ft long, 6 inches in diameter, and with a roughness $\varepsilon = 0.01$ in, and the fluid is water: $\rho = 62.4$/lbm/ft3, $\mu = 1$ cP. Pressures and inflow/outflow rates specifications are given in Table 1 . Rates not specified are equal to zero (except the one in node 17 which is an unknown). Pressures not specified are unknowns in the problem. The starting point and converged flowrates are given in Table 2 .

Since the problem contains 38 pipes, and, therefore, 38 boundaries are defined, the search space is constituted by 2^{38}=2.748779x10^{11} potential subregions. Hence, this example represents a case in which the combinatorial nature of the problem could have a severe effect in the performance of the boundary crossing algorithm. On the other hand, in the solution of the problem as a complementarity problem, the solution of only one nonlinear problem is required and a set of only

two complementarity equations is defined for each disjunction (each pipe).

Table 1: Pressures and inflow/outflow rates for Example 4

Node No.	Pressure P (psig)	Inflow rate w (gpm)
1		897.6
7		1570.9
11		-897.6
17	0	
20		-448.8
22		673.2

EXAMPLE 5 Simple L-V Flash Calculation (King, 1980)

This situation corresponds to a simple equilibrium calculation as given by King (1980). Basically, the problem consists of finding a solution to the well known Rachford-Rice equation:

$$x_i = \frac{z_i}{(K_i - 1)\cdot(V/F) + 1} \qquad y_i = \frac{K_i \cdot z_i}{(K_i - 1)\cdot(V/F) + 1}$$

$$\sum_i y_i - \sum_i x_i = 0 \quad \Longrightarrow \quad f(V/F) = \sum_i \frac{z_i \cdot (K_i - 1)}{(K_i - 1)\cdot(V/F) + 1} = 0$$

In the presence of two phases in equilibrium, the iterative calculation involves applying a convergence procedure until a value of *V/F* is found such that *f(V/F) = 0*. However, it may well happen that the specifications of the problem do not correspond to a system with two phases present. For the case of a Liquid-Vapor equilibrium, King proposes the following criteria to differentiate among the different cases: *f(V/F)* will be positive at *V/F=0* and negative at *V/F=1*. Therefore, if *f(V/F)* is negative at *V/F=0*, the system is subcooled liquid. If *f(V/F)* is positive at *V/F=1*, the system is superheated vapor.

Table 2: Starting point and converged flowrates for Example 4

Pipe No.	Estimated flow $Q^{(0)}$ (gpm)	Converged flow Q* (gpm)
1	-48.5	-223.345
2	-640.7	-894.840
3	393.2	520.818
4	538.9	883.254
5	-32.3	-435.573
6	490.2	180.240
7	649.2	533.254
8	904.7	877.652
9	112.2	315.876
10	232.5	602.421
11	402.3	541.160
12	-420.4	-229.091
13	687.3	585.972
14	621.7	701.302
15	-719.2	0.0
16	52.7	-273.643
17	1199.0	-1303.723
18	305.4	226.105
19	582.8	943.137
20	-247.5	-374.022
21	-145.7	-362.435
22	-684.6	-954.723
23	293.6	504.804
24	-261.3	-255.153
25	-229.0	180.420
26	-395.1	407.123
27	-254.8	-344.398
28	-268.9	-216.346
29	-890.6	-917.648
30	120.3	286.546
31	-8.1	132.925
32	-344.7	0.0
33	473.1	356.88
34	-206.2	0.0
35	-275.0	-443.768
36	351.5	44.552
37	-481.2	-443.768
38	353.9	317.815

This behavior can be represented in term of the following disjunctive statement:

$$\sum_i \frac{z_i \cdot (K_i - 1)}{(K_i - 1) \cdot (V/F) + 1} = R - V/F$$

$$\begin{bmatrix} V/F = 0 \\ R \le 0 \end{bmatrix} \vee \begin{bmatrix} V/F = R \\ 0 \le R \le 1 \end{bmatrix} \vee \begin{bmatrix} V/F = 1 \\ R \ge 1 \end{bmatrix}$$

For testing the proposed formulation, we took a mixture 20% of butane, 50% of pentane and 20% of hexane, at 10 atm, and performed simulations for a broad range of temperatures (150 K to 890 K).

EXAMPLE 6 Linear Mass Balance (Grossmann and Turkay, 1996)

This example, illustrated in Figure 13, represents a problem in which each of the six processing units interconnected in a flowsheet has three operating regions, each region with a different mass balance coefficient in terms of the main product flowrate. The mass balance coefficients and the bounds for each of the flowrates are shown in Table 3 .

Because of the three operating regions, the disjunctive linear mass balance for each of the units is represented by a disjunction containing 3 disjunctive terms. For instance, for the case of unit operation 1, we have:

$$\begin{bmatrix} F_6 = 1.1 \cdot F_7 \\ F_{10} = 0.05 \cdot F_7 \\ 0 \le F_7 \le 50 \end{bmatrix} \vee \begin{bmatrix} F_6 = 1.15 \cdot F_7 \\ F_{10} = 0.1 \cdot F_7 \\ 50 \le F_7 \le 80 \end{bmatrix} \vee \begin{bmatrix} F_6 = 1.2 \cdot F_7 \\ F_{10} = 0.2 \cdot F_7 \\ 80 \le F_7 \le 150 \end{bmatrix}$$

As a consequence of having three disjunctive terms, the complementarity formulation becomes complicated (as in Example 5). Five out of the six units (those containing two equations in each term) can be represented by similar sets of disjunctive equations. For the case of unit 6 (which has only one equation in each disjunctive term), the disjunctive statement and the complementarity equations are similar from those obtained for the example of the simple flash calculation:

$$\begin{bmatrix} F_{14} = 1.15 \cdot F_5 \\ 0 \le F_5 \le 20 \end{bmatrix} \vee \begin{bmatrix} F_{14} = 1.10 \cdot F_5 \\ 20 \le F_5 \le 45 \end{bmatrix} \vee \begin{bmatrix} F_{14} = 1.02 \cdot F_5 \\ 45 \le F_5 \le 75 \end{bmatrix}$$

FIGURE 13 Processing Units for Example 6.

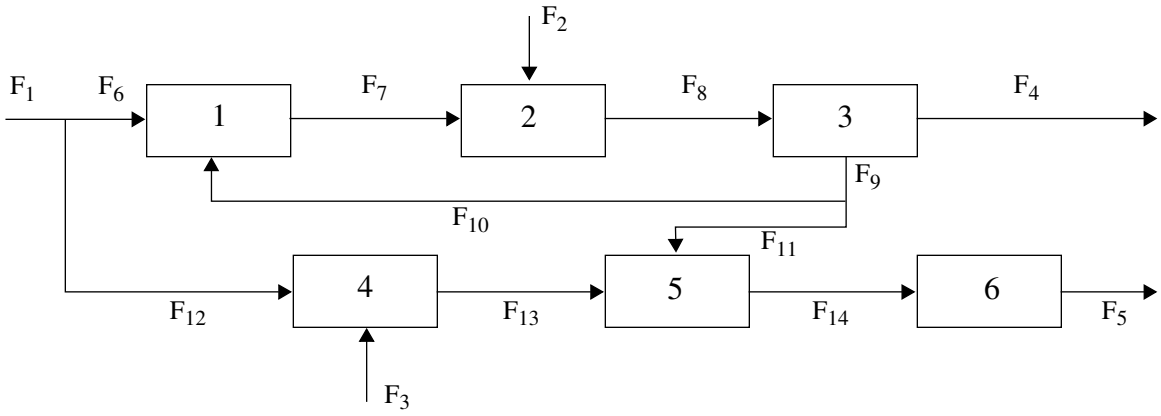


Table 3: Material balance equations for units in Example 6

Unit	Main Product	Interval	Lower Bound	Upper Bound	Mass Balance Coefficient
1	F_7	1	0	50	F_6: 1.10 F_{10}: 0.05
		2	50	80	1.15 0.10
		3	80	150	1.20 0.20
2	F_8	1	0	50	F_2: 0.50 F_7: 0.80
		2	50	100	0.47 0.75
		3	100	150	0.45 0.70
3	F_4	1	0	50	F_8: 1.70 F_9: 0.67
		2	50	110	1.80 0.70
		3	110	180	1.87 0.75
4	F_{13}	1	0	50	F_3: 1.18 F_{12}: 0.23
		2	50	90	1.15 0.25
		3	90	140	1.10 0.30
5	F_{14}	1	0	40	F_{11}: 0.37 F_{13}: 1.20
		2	40	80	0.35 1.25
		3	80	130	0.30 1.30
6	F_5	1	0	20	F_{14}: 1.15
		2	20	45	1.10
		3	45	75	1.02

Even though the original problem is linear, the introduction of the complementarity equations in the complementarity formulation leaves us with one nonlinear system of equations. On the contrary, in the boundary crossing algorithm all the alternative sets of equations are linear, but the search space in the boundary crossing algorithm is constituted by 729 subregions. The starting point used in this work and the converged values of the flowrates are shown in Table 4 .

Table 4: Starting point and converged flowrates for Example 6.

Stream	Starting Point (lbmole/hr)	Converged Value (lbmole/hr)
F_1	47.50	47.5000
F_2	21.25	19.8549
F_3	69.00	57.7545
F_4	25.00	23.3587
F_5	50.00	36.5246
F_6	37.50	34.9447
F_7	34.00	31.7679
F_8	52.50	39.7099
F_9	16.75	15.6504
F_{10}	1.700	1.5884
F_{11}	16.80	14.0620
F_{12}	15.00	12.5553
F_{13}	60.00	50.2213
F_{14}	48.00	40.1770

6 NUMERICAL RESULTS

In this section, we present the numerical results obtained by solving the examples presented in section 5 with our complementarity formulation representing a conditional model and with the boundary crossing algorithm. Some observations are:

- The fixed parameters and constants for Examples 1 through 3 are the same as those giving by Zaher (1995).

- For the fluid transition problem, we ran simulations for values of the diameter of the pipe between 2 cm and 9 cm, such that we could make sure that both of the alternative cases are reached by using the complementarity representation and the boundary crossing algorithm. The number of iterations reported corresponds to the diameter of 5 cm.
- For the example of the heat exchanger, we ran simulations for values of area between 250 ft^2 and 1104 ft^2 and values of flowrates between 250 lbmole/hr and 380 lbmole/hr, ranges analyzed by Zaher while finding an optimal solution. The number of iterations reported is for an area equal to 379.12 ft^2 and flowrate of cooling water equal to 1104.31 lbmole/hr. There is no special reason for the selection of those values.
- For the simple flash calculation, the number of iterations reported is for the value of temperature of 200 K (liquid phase region). The number of iterations was roughly of the same order for other values of temperature spread over the subcooled through the superheated range.

6.1 COMPLEMENTARITY FORMULATION

There exists a whole body of literature for the solution of nonlinear complementarity problems. In one of the most recent approaches, Dirske and Ferris (1995a) developed a non-monotone stabilization scheme for mixed complementarity problems (MCP) and implemented such an approach in the PATH solver. Some other approaches to the solution of MCPs involve the use of quadratic programming based techniques (Billups and Ferris, 1996) and homotopy based algorithms (Billups, 1998). As stated, a common feature of the previous approaches is that they require the complementarity problem to be reformulated as a MCP. However, a systematic reformulation of our complementarity problem (18) into a MCP is yet to be discovered. Such a systematic reformulation is desired because it would make suitable the application of a large number of existing codes and numerical techniques such as the ones described above. Future work should be conducted in this direction.

Since the result of applying the complementarity formulation is one square nonlinear system of equations, our approach here is just to use a conventional nonlinear solver. To solve the examples described in section 5, we used the ASCEND solver QRSlv which applies the modified Levenberg-Marquardt algorithm given by Westerberg and Director (1979). This method is preferred because

it is a least squares solver which may help to overcome the numerical singularities arising from the complementarity formulation.

The number of iterations that we used to obtained the solution of each of these examples is shown in Table 5 .

Table 5: Results obtained by using a complementarity approach

Example	Reference	Number of Equations	Number of Disjunctions	Number of Complementarity Equations	Iterations
Flow Transition (sonic-subsonic)	Zaher (1995)	7	1	1	8
Phase Equilibria	Zaher (1995)	18	3	3	10
Heat exchanger	Zaher (1995)	56	2	4	11
Pipeline network	Bullard and Biegler (1992)	250	38	76	24
Simple L-V flash	King (1980)	23	1	4	25
Linear mass balance	Grossmann and Turkay (1996)	81	6	34	25

6.2 THE BOUNDARY CROSSING ALGORITHM

The number of iterations that we used to obtained the solution of each of the examples is shown in Table 6 . The initial values are the same as those used by the complementarity approach. As mentioned before, the definition of those initial values for the variables ensures that the boundary crossing algorithm starts in an incorrect subregion, which enforces the application of the algorithm for the crossing of boundaries.

6.3 DISCUSSION

Since the hardware and the technique that we are using to get the solution is different, we are comparing neither time nor number of iterations with other works. Still we consider it important to make remarks about some differences of the alternative approaches for solving conditional models.

Table 6: Results

Example	Reference	Number of Equations	Number of Disjunctions	Number of boundary crossings	Number of Iterations
Flow Transition (sonic-subsonic)	Zaher (1995	5	1	1	10
Phase Equilibria	Zaher (1995	12	3	5	17
Heat exchanger	Zaher (1995)	48	2	1	8
Pipeline network	Bullard and Biegler (1992)	98	38	44	106
Simple L-V flash	King (1980)	15	1	1	12
Linear mass balance	Grossmann and Turkay (1996)	13	6	2	8

In general, the performance of both approached is quite acceptable and very encouraging. It is interesting to look at the example of the pipeline network. The result of reformulating the conditional model as a complementarity problem is one nonlinear system, containing 250 equations, 76 of them containing 2 bilinear terms. In the same example, the number of boundaries in the boundary crossing algorithm is 38 (that means 2^{38}=2.748779 x 10^{11} possible subregions) and the nonlinear system to be solved in every subregion would contain 98 equations. As expected, the combinatorial complexity of the example of the pipeline network affects the effectiveness of the boundary crossing algorithm. Still, The solution path crossed only 44 boundaries. Thus with 106 iterations, the average number of Newton steps per region entered is only of the order of two. The combinatorial complications present in examples like this clearly represents a disadvantage for the boundary crossing algorithm with respect to the complementarity formulation. On the other hand, the reverse situation is also possible, and the boundary crossing may be clearly a better option than the complementarity formulation. For example, in a problem with only one disjunction, but 20 equations in each of the terms of the disjunction, we may not be able to solve a nonlinear system in which 20 of the equations contain 20 bilinear terms each. However, the number of subregions in the boundary crossing algorithm would be only 2, and the possibility of applying this algorithm efficiently would be much greater.

In the example of the linear mass balance, Turkay and Grossmann (1996) solve the problem by using a mixed-integer approach. The resulting MILP contains 18 binary variables, 66 continuous variables and 89 linear equations. In the complementarity formulation, the nonlinear system contains 81 equations, 47 are linear, but the remaining 34 contain multiplications among residuals. The size of the problem is very similar, the difference will be in either using a branch and bound search in the MILP or dealing with the complementarity equations in the solution of one square nonlinear system of equations. In the case of the boundary crossing algorithm, the number of possible subregions is 729, but the fact that the equations are linear facilitates the convergence of the algorithm. A clear advantage of the boundary crossing implementation is the problem size. Note the number of equations inside each subregion is always the minimum, and it is not affected by the combinatorial nature of the problem.

7 CONCLUSIONS

We have described and tested two approaches to the solution of conditional models. Both of them represent an alternative to MILP formulations, avoiding discrete decisions.

On the one hand, we propose a new representation of conditional models as complementarity problems. In order to obtain the complementarity formulation, we rely on the assumption that the sets of conditional equations can be rearranged so that the disjunctive statements can be represented in terms of positive residual and slack variables. We show that the formulation described in this paper does not introduce spurious solutions to the problem, and, under the assumption of nondegeneracy, it will not introduce numerical singularities to the Jacobian matrix. We also mentioned some of the weaknesses and advantages of this approach.

On the other hand, following a brief description of the boundary crossing algorithm, we have described the details of its practical implementation. The conditional modeling solver CMSlv is based on an server-client, objected oriented architecture. CMSlv executes a number of activities including:

- Configuration of the systems of variables and equations consistent with the restrictions imposed by the boundary expression.
- Creation of optimization subproblems at boundaries.

- Managing the logic of the algorithm
- Performing termination tests.
- Managing the interaction with external solvers; LRSlv for logical analysis, QRSlv for iterating within a subregion and CONOP for the solution of the optimization subproblems at boundaries.

The CMSlv and LRSlv solver have been attached to the ASCEND environment, whose modeling language provides the tools required for the representation of conditional models in an equation oriented manner.

The number of iterations employed for all of the examples solved here makes the complementarity formulation and the boundary crossing appear as interesting alternatives. Such examples show the scope of application of each of the approaches to solving and reveal their very encouraging performance.

8 ACKNOWLEDGMENTS

This work has been supported by the Engineering Design Research Center at Carnegie Mellon University (NSF Grant No. EEC-8943164), by the Computer Aided Process Design (CAPD) Consortium, and by CONACYT, “Consejo Nacional de Ciencia y Tecnología ”.

9 REFERENCES

Allan, B. A.; A More Reusable Modeling System; PhD thesis, Department of Chemical Engineering, Carnegie Mellon University, Pittsburgh, Pennsylvania,1997.

Billups, S. C.; Algorithms for Complementarity Problems and Generalized Equations. PhD thesis, University of Wisconsin-Madison, August 1995.

Billups, S. C.; A Homotopy Based Algorithm for Mixed Complementarity Problems. Technical Report, Department of Mathematics, University of Colorado, 1998.

Billups, S. C. and Ferris, M. C.; QPCOMP: A Quadratic Programming Based Solver for Mixed Complementarity Problems. Technical Report, Department of Mathematics, University of Colorado, 1996.

Bullard, L. G. and Biegler, L. T.; Iterative Linear Programming Strategies for Constrained Simulation. *Comput. Chem. Eng.*, 15(4):239–254, 1991.

Bullard, L. G. and Biegler, L. T.; Iterated Linear Programming Strategies for Nonsmooth Simulation: Continuous and Mixed-Integer Approach. *Comput. Chem. Eng.*, 16(10), 1992.

Dirske, S. P. and Ferris, M. C.; The PATH Solver: A Non-Monotone Stabilization Scheme for Mixed Complementarity Problems. Optimization Methods and Software, 5, 123-156, 1995a.

Dirske, S. P. and Ferris, M. C.; MCPLIB: A Collection of Nonlinear Mixed Complementarity Problems. Optimization Methods and Software, 5, 319-345,1995b.

Drud, A.; CONOPT: A GRG Code for Large Sparse Dynamic Nonlinear Optimization Problems. *Mathematical Programming*, 31, 153-191, 1985.

Ferris, M. C. and Pang, J. S. ; Engineering and Economic Applications of Complementarity Problems. Technical report, University of Colorado, 1995.

Grossmann, I. E. and Turkay, M.; Solution of Algebraic Systems of Disjunctive Equations. *Comput. Chem. Eng.*, 20:S339–44, 1996. Suppl. Part A.

King, C. J.; Separation Processes, Chemical Engineering Series, 2nd. Edition, McGrawHill, 77-79, 1980.

Michelsen, M.L.; The Isothermal Flash Problem. Part I: Stability. *Fluid Phase Equilibria*, 9, 1-19, 1982.

More, J. J.; Global Methods for Nonlinear Complementarity Problems. Technical report, Argonne National Laboratory, Mathematics and Computer Science Division, April 1994.

Raman, R. and Grossmann, I. E.; Symbolic Integration of Logic in Mixed-Integer Linear Programming Techniques for Process Synthesis. *Comput. Chem. Eng.*, 17(9):909–927, 1993.

Raman, R. and Grossmann, I. E.; Modeling and Computational Techniques for Logic Based Integer Programming. *Comput. Chem. Eng.*, 18(7):563–578, 1994.

Rico-Ramirez, V., Allan, B. A., and Westerberg, A. W.; Conditional Modeling, Part I: Requirements for an Equation-based Environment, submitted for publication (1997).

Turkay, M. and Grossmann, I. E.; Logic-Based MINLP Algorithms for the Optimal Synthesis of Process Networks. Comput. Chem. Eng., 20(8):959–978, 1996.

Westerberg, A.W., Abbott, K. A., and Allan, B. A.; Plans for ASCEND IV: Our Next Generation Equational-Based Modeling Environment. Boston, Massachusetts, November 1994. AspenWorld'94.

Westerberg, A. W. and Director, S. W.; A Modified Least Square Algorithm for Solving Sparse n x n Sets of Nonlinear Equations. Technical report, Carnegie Mellon University, Engineering Design Research Center, EDRC-06-5-79, January 1979.

Zaher, J. J.; Conditional Modeling. PhD thesis, Department of Chemical Engineering Carnegie

Mellon University, Pittsburgh, Pennsylvania, May 1995.

Zaher, J. J.; Conditional Modeling in an Equation-Based Modeling Environment. The Annual AIChE National Meeting, 1991. paper 138c.

Zaher, J. J.; Conditional Programming. The Annual AIChE National Meeting, March 1993.

APPENDIX A

Number of Complementarity Equations Resulting from the Reformulation of a Conditional Model.

Consider the pipe network solved by Bullard and Biegler (1992). Before the reformulation as complementarity problem, this example is described by the following system of conditional equations:

(29)

$$\sum_j Q_{ij} + \sum_j Q_{ji} = w_i \qquad \forall node\ i \qquad \textbf{(a)}$$

$$\begin{bmatrix} H_{ij} = -K \cdot Q_{ij}^{\,2} \\ Q_{ij} \le 0 \end{bmatrix} \vee \begin{bmatrix} H_{ij} = K \cdot Q_{ij}^{\,2} \\ Q_{ij} \ge 0 \end{bmatrix} \qquad \forall arc\ ij\ without\ valve \qquad \textbf{(b)}$$

$$\begin{bmatrix} K \cdot Q_{ij}^{\,2} = 0 \\ H_{ij} \le 0 \end{bmatrix} \vee \begin{bmatrix} K \cdot Q_{ij}^{\,2} = H_{ij} \\ H_{ij} \ge 0 \end{bmatrix} \qquad \forall arc\ ij\ with\ valve \qquad \textbf{(c)}$$

$$H_{ij} = P_i - P_j \qquad \forall arc\ ij \qquad \textbf{(d)}$$

$$Q_{ij} \ge 0 \qquad \forall arc\ ij\ with\ valve$$

For the pipe network shown in Figure 12, in which there are 38 pipes and 22 nodes, the number of equations in each of the alternatives of the conditional model is given in Table 7 . The number of variables is given in Table 8 .

Table 7: Number of Equations in the Pipeline Network Solved by Bullard and Biegler

Equation	Number of Equations
(29a): Flow Balance for each node	22
(29b): Hazen-Williams, pipes with no valve	33
(29c): Relation pressure drop-flowrate	5
(29d): Pressure drop	38
Total in each alternative of the conditional model	98

Table 8: Number of Variables in the Pipeline Network Solved by Bullard and Biegler

Variables	Number
Flowrates, Q_{ij}	38
Inflow/outflow Rates, w_i	22
Head of Pressure, H_{ij}	38
Pressure at each node, P_i	22
K	1
Total	121

According to Table 7 and Table 8 , there are 121 variables and 98 equations in the conditional model. Hence, there must be 23 specifications in the problem. Such specifications are: the value of K (dimensions of the pipes and the fluid are known), 21 values of w_i (every node but node 17, see Table 1) and 1 value of pressure (pressure at node 17, P_{17}, see Table 1).

After the reformulation of the conditional model as a complementarity problem, the following system of equations results:

(30)

$$\sum_j Q_{ij} + \sum_j Q_{ji} = w_i \qquad \forall node\ i \qquad \textbf{(a)}$$

$$\begin{bmatrix} H_{ij} + K \cdot Q_{ij}^{\,2} = p_{1_{1,ij}} \\ K \cdot Q_{ij}^{\,2} - H_{ij} = p_{2_{1,ij}} \\ Q_{ij} = p_{1_{2,ij}} - p_{2_{2,ij}} \\ p_{1_{1,ij}} \cdot p_{2_{1,ij}} + p_{1_{2,ij}} \cdot p_{2_{2,ij}} = 0 \\ p_{1_{1,ij}} \cdot p_{2_{2,ij}} + p_{1_{2,ij}} \cdot p_{2_{1,ij}} = 0 \end{bmatrix} \qquad \forall arc\ ij\ without\ valve \qquad \textbf{(b)}$$

$$\begin{bmatrix} K \cdot Q^2_{ij} = p_{1_{1,ij}} \\ K \cdot Q^2_{ij} - H_{ij} = p_{2_{1,ij}} \\ H_{ij} = p_{1_{2,ij}} - p_{2_{2,ij}} \\ p_{1_{1,ij}} \cdot p_{2_{1,ij}} + p_{1_{2,ij}} \cdot p_{2_{2,ij}} = 0 \\ p_{1_{1,ij}} \cdot p_{2_{2,ij}} + p_{1_{2,ij}} \cdot p_{2_{1,ij}} = 0 \end{bmatrix} \quad \forall arc\ ij\ with\ valve \qquad \textbf{(c)}$$

$$H_{ij} = P_i - P_j \qquad \forall arc\ ij \qquad \textbf{(d)}$$

$$Q_{ij} \geq 0 \qquad \forall arc\ ij\ with\ valve$$

$$p_{k_{ij,q}} \geq 0 \qquad \forall arc\ ij, k \in [1 \ldots 2], q \in [1 \ldots 2]$$

The number of equations and the number of variables for the complementarity formulation are shown in Table 9 and Table 10 correspondingly.

Table 9: Number of Equations in the Complementarity Formulation of the Pipeline Network Solved by Bullard and Biegler

Equation	Number of Equations
(30a): Flow Balance for each node	22
(30b): Hazen-Williams, pipes with no valve	165
(30c): Relation pressure drop-flowrate	25
(30d): Pressure drop	38
Total in the complementarity formulation	250

The number of equations in the complementarity formulation is 250 (note the agreement with Table 5) and the number of variables is 273. Hence, the values of 23 variables must be specified. However, this number of specifications is exactly the same as the number of specifications

required by the conditional model before the reformulation as a complementarity problem. Note that 250-98=152 new equations and 273-121=152 new variables have been introduced. In summary, the reformulation of a fully specified conditional model as a complementarity problem results in a square system of nonlinear equations, including the complementarity equations.

Table 10: Number of Variables in the Pipeline Network Solved by Bullard and Biegler

Variables	Number
Flowrates, Q_{ij}	38
Inflow/outflow Rates, w_i	22
Head of Pressure, H_{ij}	38
Pressure at each node, P_i	22
K	1
Residual and slack variables, $p_{k_{ij,q}}$	152
Total	273

APPENDIX B

Derivation of the Optimization Subproblem of the Boundary Crossing Algorithm

Zaher (1991,1995) describes the derivation of Equation (28) as follows. Let us start from Equation (27):

$$\begin{aligned} min \quad & \begin{pmatrix} max & \nabla_x \phi_i(\underline{a})^T \cdot \underline{d} \\ s.t. & i \in B \end{pmatrix} \\ s.t. \quad & \underline{d}^T \cdot \underline{d} \leq 1 \end{aligned}$$

where B is the set of all of the subregions in the neighborhood of a boundary, $\nabla_x \phi_i(\underline{a})$ is the gradient vector of the objective function for the neighboring subregion i, and $\underline{d}$ is the descent direction which constitutes the solution to the problem (27). Let us now define the matrix N whose columns are the vectors of gradients of the subregions in the neighborhood of the current boundary. That is, column i of N is the vector $\nabla_x \phi_i(\underline{a})$, $\forall i \in B$. Hence, the inner maximization problem of (27) can be reformulated as finding the value of a scalar β such that:

$$\beta \underline{e} \geq N^T \underline{d} \tag{31}$$

where $\underline{e}$ is a vector with all its elements equal to one. Then, Equation (27) can be rewritten as:

$$\begin{aligned} min \quad & \beta \\ s.t. \quad & \beta \underline{e} \geq N^T \underline{d} \\ & \underline{d}^T \cdot \underline{d} \leq 1 \end{aligned} \tag{32}$$

The real-valued Lagrange function, $L(\beta, \underline{d}, \underline{\alpha}, \pi)$, of problem (32) is given by:

$$L(\beta, \underline{d}, \underline{\alpha}, \pi) = \beta - \underline{\alpha}^T (\beta \underline{e} - N^T d) - \pi(1 - d^T d) \tag{33}$$

where $\underline{\alpha}$ is the vector of multipliers corresponding to first of the inequality constraints in (32) and π is the multiplier (scalar) corresponding to the second inequality of (32). Hence, the dual of the

problem (32) is given by:

$$\begin{aligned} max \quad & \beta - \underline{\alpha}^T(\beta \underline{e} - N^T d) - \pi(1 - d^T d) \\ s.t. \quad & \underline{e}^T \cdot \underline{\alpha} = 1 \\ & 2\pi \underline{d} = -N\underline{\alpha} \\ & \pi \geq 0,\ \alpha_i \geq 0,\ \forall i \in B \end{aligned} \tag{34}$$

The equality constraints of (34) were obtained from the first order conditions of the Lagrangian function with respect to the variables β and $\underline{d}$. After further simplification, (34) reduces to:

$$\begin{aligned} max \quad & -\pi(\underline{d}^T \cdot \underline{d} + 1) \\ & 2\pi \underline{d} = -N \cdot \underline{\alpha} \\ & \underline{e}^T \cdot \underline{\alpha} = 1 \\ & \pi \geq 0,\ \alpha_i \geq 0,\ \forall i \in B \end{aligned} \tag{35}$$

which is equivalent to:

$$\begin{aligned} min \quad & \pi(\underline{d}^T \cdot \underline{d} + 1) \\ & 2\pi \underline{d} = -N \cdot \underline{\alpha} \\ & \underline{e}^T \cdot \underline{\alpha} = 1 \\ & \pi \geq 0,\ \alpha_i \geq 0,\ \forall i \in B \end{aligned} \tag{36}$$

Finally, it should be recognized that, as long as the problem (36) is bounded and the solution is non trivial ($\pi > 0$), the multiplier π will have no effect in the solution vector $\underline{d}$ of the optimization problem (36). Therefore, (36) reduces to equation (28):

$$\begin{aligned} min \quad & \underline{d}^T \cdot \underline{d} \\ s.t. \quad & \underline{d} = -N \cdot \underline{\alpha} \\ & \underline{e}^T \cdot \underline{\alpha} = 1 \\ & \alpha_i \geq 0,\ \forall i \in B \end{aligned}$$

APPENDIX C

Complementarity Equations of the Illustrative Examples.

Example 1 Fluid Transition (Zaher,1995).

A single complementarity equation is required. The definition of the positive residuals and the complementarity equation are given by:

$$\begin{gathered} 1 - M_f = p_{1_1} \\ p_f - p_d = p_{2_1} \\ \begin{bmatrix} p_{1_1} = 0 \\ p_{2_1} \geq 0 \end{bmatrix} \vee \begin{bmatrix} p_{2_1} = 0 \\ p_{1_1} \geq 0 \end{bmatrix} \Rightarrow \begin{matrix} p_{1_1} \cdot p_{2_1} = 0 \\ p_{1_1}, p_{2_1} \geq 0 \end{matrix} \end{gathered} \tag{37}$$

Since the problem contains only one condition in each disjunctive term, our complementarity representation reduces to the standard complementarity formulation.

Example 2 Phase Equilibria (Zaher,1995).

The representation of the existence-non existence of each phase is given by the following set of equations:

$$\begin{gathered} \phi_A = p_{1_1} \\ 1 - \sum_{i \in C} y_{i_A} = p_{2_1} \\ \begin{bmatrix} p_{1_1} = 0 \\ p_{2_1} \geq 0 \end{bmatrix} \vee \begin{bmatrix} p_{2_1} = 0 \\ p_{1_1} \geq 0 \end{bmatrix} \Rightarrow \begin{matrix} p_{1_1} \cdot p_{2_1} = 0 \\ p_{1_1}, p_{2_1} \geq 0 \end{matrix} \end{gathered} \tag{38}$$

There are three possible phases, but there is only one equation in each disjunction, and, therefore, only one complementarity equation for each of phases is required. Each of the complementarity equations is similar to the complementarity equation of Example 1.

Example 3 Heat Exchanger (Zaher,1995)

In order to obtain the complementarity formulation of this example, we start by recognizing that we can decouple the disjunctive statement of Example 3 in two independent disjunctions:

$$\begin{bmatrix} \phi_0 = 0 \\ \eta_1 = 0 \\ \sum_{i \in C} x_{i_0} + \phi_0 < 1 \end{bmatrix} \vee \begin{bmatrix} \sum_{i \in C} x_{i_0} = 1 \\ \sum_{i \in C} x_{i_1} = 1 \\ \sum_{i \in C} x_{i_0} + \phi_0 \geq 1 \end{bmatrix} \tag{39}$$

and

$$\begin{bmatrix} \phi_1 = 0 \\ \eta_2 = 0.5 \\ \sum_{i \in C} x_{i_2} + \phi_1 < 1 \end{bmatrix} \vee \begin{bmatrix} \sum_{i \in C} x_{i_2} = 1 \\ \eta_1 = 0.5 \\ \sum_{i \in C} x_{i_2} + \phi_1 \geq 1 \end{bmatrix} \tag{40}$$

Hence, the complementarity equations for these disjunctions are:

$$\begin{gathered} \phi_0 = p_{1_{11}} \qquad 1 - \sum_{i \in C} x_{i_0} = p_{2_{11}} \\ \eta_1 = p_{1_{21}} \qquad 1 - \sum_{i \in C} x_{i_1} = p_{2_{21}} \\ \begin{bmatrix} p_{1_{11}} = 0 \\ p_{1_{21}} = 0 \end{bmatrix} \vee \begin{bmatrix} p_{2_{11}} = 0 \\ p_{2_{21}} = 0 \end{bmatrix} \Rightarrow \begin{matrix} p_{1_{11}} \cdot p_{2_{11}} + p_{1_{21}} \cdot p_{2_{21}} = 0 \\ p_{1_{11}} \cdot p_{2_{21}} + p_{1_{21}} \cdot p_{2_{11}} = 0 \end{matrix} \\ p_{i_{j1}} \geq 0 \qquad \forall i \in [1 \ldots 2],\ j \in [1 \ldots 2] \end{gathered} \tag{41}$$

and

$$\begin{gathered} \phi_1 = p_{1_{12}} \qquad 1 - \sum_{i \in C} x_{i_2} = p_{2_{12}} \\ \eta_2 - 0.5 = p_{1_{22}} \qquad 0.5 - \eta_1 = p_{2_{22}} \\ \begin{bmatrix} p_{1_{12}} = 0 \\ p_{1_{22}} = 0 \end{bmatrix} \vee \begin{bmatrix} p_{2_{12}} = 0 \\ p_{2_{22}} = 0 \end{bmatrix} \Rightarrow \begin{matrix} p_{1_{12}} \cdot p_{2_{12}} + p_{1_{22}} \cdot p_{2_{22}} = 0 \\ p_{1_{12}} \cdot p_{2_{22}} + p_{1_{22}} \cdot p_{2_{12}} = 0 \end{matrix} \\ p_{i_{j2}} \geq 0 \qquad \forall i \in [1 \ldots 2],\ j \in [1 \ldots 2] \end{gathered} \tag{42}$$

Example 4 Pipeline Network (Bullard and Biegler, 1992).

1) Arcs with no valve

$$
\begin{gathered}
H_{ij} + K \cdot Q_{ij}^{\,2} = p_{1_1} \\
K \cdot Q_{ij}^{\,2} - H_{ij} = p_{2_1} \\
Q_{ij} = p_{1_2} - p_{2_2} \\
\begin{bmatrix} p_{1_1} = 0 \\ p_{1_2} = 0 \end{bmatrix} \vee \begin{bmatrix} p_{2_1} = 0 \\ p_{2_2} = 0 \end{bmatrix} \Rightarrow \begin{matrix} p_{1_1} \cdot p_{2_1} + p_{1_2} \cdot p_{2_2} = 0 \\ p_{1_1} \cdot p_{2_2} + p_{1_2} \cdot p_{2_1} = 0 \end{matrix} \\
p_{k_q} \geq 0 \qquad \forall k \in [1 \ldots 2], q \in [1 \ldots 2]
\end{gathered}
\tag{43}
$$

2)Arcs with check valve

$$
\begin{gathered}
K \cdot Q^2_{ij} = p_{1_1} \\
K \cdot Q^2_{ij} - H_{ij} = p_{2_1} \\
H_{ij} = p_{1_2} - p_{2_2} \\
\begin{bmatrix} p_{1_1} = 0 \\ p_{1_2} = 0 \end{bmatrix} \vee \begin{bmatrix} p_{2_1} = 0 \\ p_{2_2} = 0 \end{bmatrix} \Rightarrow \begin{matrix} p_{1_1} \cdot p_{2_1} + p_{1_2} \cdot p_{2_2} = 0 \\ p_{1_1} \cdot p_{2_2} + p_{1_2} \cdot p_{2_1} = 0 \end{matrix} \\
p_{k_q} \geq 0 \qquad \forall k \in [1 \ldots 2], q \in [1 \ldots 2]
\end{gathered}
\tag{44}
$$

Example 5 Simple L-V Flash Calculation (King, 1980).

The behavior described in Example 5 can be represented in terms of the following disjunctive statement:

$$
\begin{gathered}
\sum_i \frac{z_i \cdot (K_i - 1)}{(K_i - 1) \cdot (V/F) + 1} = R - V/F \\
\begin{bmatrix} V/F = 0 \\ R \leq 0 \end{bmatrix} \vee \begin{bmatrix} V/F = R \\ 0 \leq R \leq 1 \end{bmatrix} \vee \begin{bmatrix} V/F = 1 \\ R \geq 1 \end{bmatrix}
\end{gathered}
\tag{45}
$$

This example includes a disjunction with three disjunctive terms in it, but a complementarity formulation can still be obtained. Also, in this case, some of the residual variables p will appear in more than one disjunctive term and a representation including an index i for each disjunctive term could be confusing. For that reason, we use the variables p indexed in successive order as follows:

$$\begin{gathered} V/F = p_1 \qquad V/F = R + p_2 - p_3 \qquad V/F = 1 - p_4 \\ R = p_5 - p_6 \qquad R = 1 + p_7 - p_8 \end{gathered} \tag{46}$$

The disjunctive representation in terms of the residuals is:

$$\begin{gathered} \begin{bmatrix} p_1 = 0 \\ p_5 = 0 \\ p_3 = 0 \\ p_7 = 0 \end{bmatrix} \vee \begin{bmatrix} p_2 = 0 \\ p_6 = 0 \\ p_3 = 0 \\ p_7 = 0 \end{bmatrix} \vee \begin{bmatrix} p_2 = 0 \\ p_6 = 0 \\ p_4 = 0 \\ p_8 = 0 \end{bmatrix} \\ p_j \geq 0 \qquad \forall j \in [1 \ldots 8] \end{gathered} \tag{47}$$

Note that several conditions appear in two different disjunctive terms. We could generate the complementarity equations as we described in section 2.1.4. The formulation obtained would still be consistent. However, we took advantage of the structure of the disjunctive representation given above in order to simplify the formulation. The following structure can be identified:

$$\begin{bmatrix} A \\ B \end{bmatrix} \vee \begin{bmatrix} C \\ B \end{bmatrix} \vee \begin{bmatrix} C \\ D \end{bmatrix} \Rightarrow (A \wedge B) \vee (C \wedge B) \vee (C \wedge D) \tag{48}$$

and converting from Disjunctive Normal Form to Conjunctive Normal Form:

$$(A \wedge B) \vee (C \wedge B) \vee (C \wedge D) \Rightarrow (A \vee C) \wedge (B \vee C) \wedge (B \vee D) \tag{49}$$

This derivation tells us that, in order to represent the three-term disjunctive term, we can use the union of three two-term disjunctions:

$$\left\{\begin{bmatrix} p_1 = 0 \\ p_5 = 0 \end{bmatrix} \vee \begin{bmatrix} p_2 = 0 \\ p_6 = 0 \end{bmatrix}\right\} \wedge \left\{\begin{bmatrix} p_3 = 0 \\ p_7 = 0 \end{bmatrix} \vee \begin{bmatrix} p_2 = 0 \\ p_6 = 0 \end{bmatrix}\right\} \wedge \left\{\begin{bmatrix} p_3 = 0 \\ p_7 = 0 \end{bmatrix} \vee \begin{bmatrix} p_4 = 0 \\ p_8 = 0 \end{bmatrix}\right\} \tag{50}$$

$$p_j \geq 0 \qquad \forall j \in [1 \ldots 8]$$

which is equivalent to

$$\left\{\begin{array}{l} p_1 \cdot p_2 + p_5 \cdot p_6 = 0 \\ p_1 \cdot p_6 + p_5 \cdot p_2 = 0 \end{array}\right\} \wedge \left\{\begin{array}{l} p_3 \cdot p_2 + p_7 \cdot p_6 = 0 \\ p_3 \cdot p_6 + p_7 \cdot p_2 = 0 \end{array}\right\} \wedge \left\{\begin{array}{l} p_3 \cdot p_4 + p_7 \cdot p_8 = 0 \\ p_3 \cdot p_8 + p_7 \cdot p_4 = 0 \end{array}\right\} \tag{51}$$

$$p_j \geq 0 \qquad \forall j \in [1 \ldots 8]$$

If we apply that simplification for this example, the number of terms in our complementarity representation decreases significantly: strictly, a disjunction with three terms and four conditions in each term requires $4^3 = 64$ trilinear terms for its representation. On the other hand, the previous derivation tells us that 12 bilinear terms are enough for the representation of this particular problem.

We still need to find out how to distribute the 12 bilinear terms through the 4 complementarity equations required to obtain a square system of equations. Such a distribution is not unique, and we can use any set of complementarity equations which does not introduce numerical singularities (any set in which it can be proved that there is a possible pivot for each complementarity equation in the Jacobian matrix) to the system. The distribution can be as simple as the following:

$$\begin{aligned} p_1 \cdot p_2 + p_5 \cdot p_6 &= 0 \\ p_1 \cdot p_6 + p_5 \cdot p_2 &= 0 \\ p_3 \cdot p_2 + p_7 \cdot p_6 + p_3 \cdot p_4 + p_7 \cdot p_8 &= 0 \\ p_3 \cdot p_6 + p_7 \cdot p_2 + p_3 \cdot p_8 + p_7 \cdot p_4 &= 0 \\ p_j \geq 0 \qquad \forall j \in [1 \ldots 8] & \end{aligned} \tag{52}$$

in which the terms of two disjunctions are joined to generate two of the equations, or a more thoughtful one like:

$$
\begin{aligned}
&p_1 \cdot p_2 + p_3 \cdot p_6 + p_7 \cdot p_8 = 0 \\
&p_1 \cdot p_6 + p_3 \cdot p_2 + p_7 \cdot p_4 = 0 \\
&p_5 \cdot p_2 + p_7 \cdot p_6 + p_3 \cdot p_4 = 0 \\
&p_5 \cdot p_6 + p_7 \cdot p_2 + p_3 \cdot p_8 = 0 \\
&p_j \geq 0 \qquad \forall j \in [1 \ldots 8]
\end{aligned}
\tag{53}
$$

in which the complementarity pairs are arranged to avoid repeated indices in the complementarity equations. When incorporated with the rest of the system, both sets of complementarity equations provide the correct solution to the problem. The number of iterations that we reported in this paper corresponds to the straightforward formulation given first.

Example 6 Linear Mass Balance (Grossmann and Turkay, 1996).

Even though the original disjunctive problem is linear, the introduction of the complementarity equations leaves us with a nonlinear system of equations. Also, as in Example 5, this example contains disjunctions with 3 disjunctive terms. As a consequence, the formulation once again becomes complicated. Units 1 through 5 (those containing two equations in each term) can be represented by similar sets of disjunctive equations. Here we only show the complementarity equations for unit operation 1. The equations for unit operation 2 to 5 are very similar to the equations generated for this one. For the case of unit 6 (which has only one equation in each disjunctive term), the disjunctive statement and the complementarity equations are similar from those obtained for the example of the simple flash calculation. The simplification process for this example is also the same as that given in Example 5.

Definition of the residual and slack variables:

$$
\begin{aligned}
F_6 &= 1.1 \cdot F_7 + p_1 \qquad F_6 = 1.5 \cdot F_7 + p_2 - p_3 \qquad F_6 = 1.2 \cdot F_7 - p_4 \\
F_{10} &= 0.05 \cdot F_7 + p_5 \qquad F_{10} = 0.1 \cdot F_7 + p_6 - p_7 \qquad F_{10} = 0.2 \cdot F_7 - p_8 \\
F_7 &= 50 + p_9 - p_{10} \qquad F_7 = 80 + p_{11} - p_{12}
\end{aligned}
\tag{54}
$$

Disjunctive statement in terms of the residual and slack variables:

$$\begin{bmatrix} p_1 = 0 \\ p_5 = 0 \\ p_9 = 0 \\ p_2 = 0 \\ p_6 = 0 \\ p_{11} = 0 \end{bmatrix} \vee \begin{bmatrix} p_3 = 0 \\ p_7 = 0 \\ p_{10} = 0 \\ p_2 = 0 \\ p_6 = 0 \\ p_{11} = 0 \end{bmatrix} \vee \begin{bmatrix} p_3 = 0 \\ p_7 = 0 \\ p_{10} = 0 \\ p_4 = 0 \\ p_8 = 0 \\ p_{12} = 0 \end{bmatrix} \tag{55}$$
$$p_j \geq 0 \qquad \forall j \in [1 \ldots 12]$$

After converting to Conjunctive Normal Form:

$$\left\{ \begin{bmatrix} p_1 = 0 \\ p_5 = 0 \\ p_9 = 0 \end{bmatrix} \vee \begin{bmatrix} p_3 = 0 \\ p_7 = 0 \\ p_{10} = 0 \end{bmatrix} \right\} \wedge \left\{ \begin{bmatrix} p_2 = 0 \\ p_6 = 0 \\ p_{11} = 0 \end{bmatrix} \vee \begin{bmatrix} p_3 = 0 \\ p_7 = 0 \\ p_{10} = 0 \end{bmatrix} \right\} \wedge \left\{ \begin{bmatrix} p_2 = 0 \\ p_6 = 0 \\ p_{11} = 0 \end{bmatrix} \vee \begin{bmatrix} p_4 = 0 \\ p_8 = 0 \\ p_{12} = 0 \end{bmatrix} \right\} \tag{56}$$
$$p_j \geq 0 \qquad \forall j \in [1 \ldots 12]$$

One possible set of complementarity equations:

$$\begin{aligned} p_1 \cdot p_3 + p_2 \cdot p_4 + p_9 \cdot p_7 + p_6 \cdot p_{10} + p_{11} \cdot p_{12} &= 0 \\ p_1 \cdot p_7 + p_2 \cdot p_8 + p_9 \cdot p_{10} + p_6 \cdot p_3 + p_{11} \cdot p_4 &= 0 \\ p_1 \cdot p_{10} + p_2 \cdot p_{12} + p_9 \cdot p_3 + p_6 \cdot p_7 + p_{11} \cdot p_8 &= 0 \\ p_5 \cdot p_3 + p_6 \cdot p_4 + p_{11} \cdot p_{10} + p_2 \cdot p_7 &= 0 \\ p_5 \cdot p_7 + p_6 \cdot p_8 + p_{11} \cdot p_3 + p_2 \cdot p_{10} &= 0 \\ p_5 \cdot p_{10} + p_6 \cdot p_{12} + p_{11} \cdot p_7 + p_2 \cdot p_3 &= 0 \\ p_j \geq 0 \qquad \forall j \in [1 \ldots 12] & \end{aligned} \tag{57}$$

The number of iterations reported in this paper corresponds to the solution of the system of equations including the set of complementarity equations (57), but some other alternative sets were also successfully used to obtain the solution to the problem.

APPENDIX D

In this appendix, we show a representative section of the ASCEND models for the examples described in section 5.

Example 1 Fluid Transition (Zaher,1995).

```
CONDITION
      bound: (Pd - Pf) * 1.0 {atm^-1} < Mf - 1.0;
END CONDITION;

sonic_flow <==> SATISFIED(bound,1e-08);

non_sonic: Pf = Pd;
sonic: Mf = 1.0;

WHEN (sonic_flow)
      CASE TRUE:
            USE sonic;
      CASE FALSE:
            USE non_sonic;
END WHEN;
```

Example 2 Phase Equilibria (Zaher,1995).

```
components :== ['B','E','W'];
phases :== ['A','O','V'];

CONDITION
      FOR i IN phases CREATE
            bound[i]: SUM[y[i][j] | j IN components ]+ phi[i] >= 1.0;
      END FOR;
END CONDITION;

FOR i IN phases CREATE
      sum[i]: SUM[y[i][j] | j IN components ] = 1.0;
      frac[i]: phi[i] = 0.0;
      exist[i] <==> SATISFIED(bound[i],1e-08);

      WHEN (exist[i])
            CASE TRUE:
                  USE sum[i];
            CASE FALSE:
                  USE frac[i];
      END WHEN;
END FOR;
```

Example 3 Heat Exchanger (Zaher,1995)

```
components :== ['B','P5','H'];

CONDITION
      bound1: SUM[x[0][i] | i IN components] + phi[0] >= 1.0;
      bound2: SUM[x[2][i] | i IN components] + phi[1] >= 1.0;
END CONDITION;

liq_exist[1] <==> SATISFIED(bound1,1e-08);
liq_exist[2] <==> SATISFIED(bound2,1e-08);

sum0: SUM[x[0][i] | i IN components] = 1.0;
frac0: phi[0] = 0.0;

sum2: SUM[x[2][i] | i IN components] = 1.0;
frac1: phi[1] = 0.0;

p1: eta[1] = 0.5;
p2: eta[2] = 0.5;

sum1: SUM[x[1][i] | i IN components] = 1.0;
p12: eta[1] = 0.0;

WHEN (liq_exist[1])
      CASE TRUE:
            USE sum0;
            USE sum1;
      CASE FALSE:
            USE frac0;
            USE p12;
END WHEN;

WHEN (liq_exist[2])
      CASE TRUE:
            USE sum2;
            USE p1;
      CASE FALSE:
            USE frac1;
            USE p2;
END WHEN;
```

Example 4 Pipeline Network (Bullard and Biegler, 1992).

Model of a pipe with no valve.

```
positive_flow     IS_A  boolean_var;
```

```
      CONDITION
            bound: Q >= 0.0 {gpm};
      END CONDITION;

      positive_flow <==> SATISFIED(bound,1e-08 {gpm});

      positive_head: H = k * sqr(Q);
      negative_head: H = -k * sqr(Q);

      WHEN (positive_flow)
            CASE TRUE:
                  USE positive_head;
            CASE FALSE:
                  USE negative_head;
      END WHEN;
```

Example 5 Simple L-V Flash Calculation (King, 1980).

```
(* Rachford-Rice *)

      SUM[ ( z[i] * ( K[i] - 1 ) ) / ( ((K[i]-1)* V_F) + 1 )
                                                | i IN components ] = R - V_F;

(* Conditional Equations *)
      CONDITION
            bound1: R <= 0.0;
            bound2: R <= 1.0;
      END CONDITION;

(* Logical Relations *)
      bol1 <==> SATISFIED(bound1,1e-08);
      bol2 <==> SATISFIED(bound2,1e-08);

(* Variant Equations*)
      liq_eq: V_F = 0.0;
      two_eq: V_F = R;
      vap_eq: V_F = 1.0;

(* Disjunctions *)
      WHEN (bol1,bol2)
            CASE TRUE,TRUE:
                  USE liq_eq;
            CASE FALSE,TRUE:
                  USE two_eq;
            CASE FALSE,FALSE:
                  USE vap_eq;
            OTHERWISE:
      END WHEN;
```

Example 6 Linear Mass Balance (Grossmann and Turkay, 1996).

```
      CONDITION
            bound1: Fmain <= B[1];
            bound2: Fmain >= B[2];
      END CONDITION;

      bol1 <==> SATISFIED(bound1,1e-08{lb_mole/hour});
      bol2 <==> SATISFIED(bound2,1e-08{lb_mole/hour});

(* Variant Equations *)

      eq1a: Fsub1 = a[1] * Fmain;
      eq1b: Fsub2 = b[1] * Fmain;

      eq2a: Fsub1 = a[2] * Fmain;
      eq2b: Fsub2 = b[2] * Fmain;

      eq3a: Fsub1 = a[3] * Fmain;
      eq3b: Fsub2 = b[3] * Fmain;

(* Disjunction  *)

      WHEN (bol1,bol2)
            CASE TRUE,FALSE:
                  USE eq1a;
                  USE eq1b;
            CASE FALSE,FALSE:
                  USE eq2a;
                  USE eq2b;
            CASE FALSE,TRUE:
                  USE eq3a;
                  USE eq3b;
      END WHEN;
```