Proceedings of the American Control Conference
San Diego, California • June 1999

A Framework for Control, Fault Detection, State Estimation, and Verification of Hybrid Systems

Domenico Mignone, Alberto Bemporad, Manfred Morari
Institut für Automatik, ETH - Swiss Federal Institute of Technology
ETHZ - ETL, CH 8092 Zürich, Switzerland, tel.+41-1-632 7626 fax +41-1-632 1211
{mignone,bemporad,morari}@aut.ee.ethz.ch
http://www.control.ethz.ch

Abstract

This paper presents a modeling formalism for hybrid systems which allows to formulate and solve several practical problems, such as control, formal verification, state estimation, and fault detection. As an extension to previous works we report a technique that allows to reduce the number of auxiliary binary variables in the modeling phase.

1 Introduction

The concept of *model* of a system is traditionally associated with differential or difference equations, typically derived by physical laws governing the *dynamics* of the system under consideration. Consequently, most of the control theory and tools have been developed for such systems, in particular for systems whose evolution is described by smooth linear or nonlinear state transition functions. On the other hand, in many applications the system to be controlled is also constituted by parts described by *logic*, such as for instance on/off switches or valves, gears or speed selectors, evolutions dependent on if-then-else rules. Often, the control of these systems is left to schemes based on heuristic rules inferred from practical plant operation.

Recently, in the literature researchers started dealing with *hybrid systems*, namely hierarchical systems constituted by dynamical components at the lower level, governed by upper level logical/discrete components [9, 5]. However, in some applications a precise distinction between different hierarchic levels is not possible, especially when dynamical and logical facts are dramatically interdependent. For such a class of systems it is even not clear how to obtain models systematically.

This paper proposes a framework for modeling, controlling, estimating, and verifying models of systems described by interacting physical laws, logic rules, and operating constraints. According to techniques described e.g. in [17, 6, 13], propositional logic is transformed into linear inequalities involving integer and continuous variables. This allows to arrive at *Mixed Logical Dynamical* (MLD) systems described by linear dynamic equations subject to linear mixed-integer inequalities, i.e. inequalities involving both *continuous* and *binary* (or *logical*, or *0-1*) variables. MLD systems generalize a wide set of models, among which there are constrained linear systems, finite state machines interacting with dynamic systems, some classes of discrete event systems, piece-wise linear systems, systems with discrete inputs.

Several questions of interest when dealing with MLD systems can be suitably formulated as mixed-integer optimization problems. For feedback control purposes, we propose a predictive control scheme which is able to stabilize MLD systems on desired reference trajectories while fulfilling operating constraints, and possibly take into account previous qualitative knowledge in the form of heuristic rules. The dual problem of state estimation can be set up as an optimization problem over a sequence of past estimates.

The aim of the verification problem is to check whether there exist an initial condition and an input sequence, such that a system can be driven to a certain state space region. Typically this region represents a set of dangerous or unsuitable operating conditions. For systems in MLD form this problem can be solved through algorithms based on linear and mixed-integer linear programs. For details and a case study on an automotive suspension system, the reader is referred to [4].

This paper is organized as follows. In Section 2 some basic facts from propositional calculus, Boolean algebra, and mixed-integer linear inequalities are reviewed. An alternative formulation for the translation of purely logical relations into linear inequalities is presented in Section 3. These tools are used in Section 4 to motivate the definition of MLD systems, and in Section 5 to provide examples of systems which can be modeled within this framework. Section 6 deals with the optimal control of MLD systems and shows how heuristics can be taken into account. Section 7 briefly summarizes the techniques to solve the state estimation/fault detection problem. Some considerations about the computational aspects are given in Section 8.

2 Propositional Calculus and Linear Integer Programming

By following standard notation [16, 6, 17], we adopt capital letters X_i to represent statements, e.g. "$x \geq 0$" or "Temperature is hot". X_i is commonly referred to as a *literal*, and has a *truth value* of either "T" (true) or "F" (false). Boolean algebra enables statements to be combined in compound statements by means of *connectives*: "$\wedge$" (and), "$\vee$" (or), "$\sim$" (not), "$\rightarrow$" (implies), "$\leftrightarrow$" (if and only if), "$\oplus$" (exclusive or) (a more comprehensive treatment of Boolean calculus can be found in digital circuit design texts, e.g. [7, 10]. For a rigorous exposition see e.g. [11]). Connectives satisfy several properties (see e.g. [7]), which can be used to transform compound statements into equivalent statements involving different connectives, and simplify complex statements. Correspondingly one can associate with a literal X_i a *logical variable* $\delta_i \in \{0, 1\}$, which has a value of either 1 if X_i =T, or 0 otherwise. Integer programming has been advocated as an efficient inference engine to perform automated deduction [6]. A propositional logic problem, where a statement X_1 must be proved to be true given a set of (compound) statements involving literals X_1, ..., X_n, can in fact be solved by means of a linear integer program, by suitably translating the original compound statements into linear inequalities involving logical variables δ_i. In fact, the propositions and linear constraints reported in Table 1 can easily be seen to be equivalent. Similar ideas originally came up in the early 1960's dealing with the synthesis of linear switching circuits [12, 14]. Threshold logic was introduced to synthesize arbitrary combinational circuits as an alternative to realizations with AND/OR/NOT gates.

We borrow this computational inference technique to model logical parts of processes and heuristic knowledge about plant operation as integer linear inequalities. As we are interested in systems which have both logic and dynamics, we wish to establish a link between the two worlds. In particular, we need to establish how to build statements from operating events concerning physical dynamics. The key idea is to use *mixed-integer linear inequalities*, i.e. linear inequalities involving both *continuous variables* $x \in \mathbb{R}^n$ and logical (*indicator*) variables $\delta \in \{0, 1\}$, as described in Table 1. Consider for instance the statement $X \triangleq [f(x) \leq 0]$ where $f : \mathbb{R}^n \mapsto \mathbb{R}$ is linear, assume that $x \in \mathcal{X}$, where $\mathcal{X}$ is a given bounded set, and define

$$M \triangleq \max_{x \in \mathcal{X}} f(x), \; m \triangleq \min_{x \in \mathcal{X}} f(x) \qquad (1)$$

Theoretically, an over[under]-estimate of M [m] suffices for our purpose. However, more realistic estimates

Table 1: Basic conversion of logic relations into mixed-integer inequalities. Relations involving the form $[\delta = 0]$ can be obtained by substituting $(1 - \delta)$ for δ in the corresponding inequalities.

	relation	logic	mixed integer (in)equalities
P1	AND ($\wedge$)	$[\delta_1 = 1] \wedge [\delta_2 = 1]$	$\delta_1 = 1$ $\delta_2 = 1$
P2		$[\delta_3 = 1] \leftrightarrow$ $[\delta_1 = 1] \wedge [\delta_2 = 1]$	$-\delta_1 + \delta_3 \leq 0$ $-\delta_2 + \delta_3 \leq 0$ $\delta_1 + \delta_2 - \delta_3 \leq 1$
P4	OR ($\vee$)	$[\delta_1 = 1] \vee [\delta_2 = 1]$	$\delta_1 + \delta_2 \geq 1$
P5		$[\delta_3 = 1] \leftrightarrow$ $[\delta_1 = 1] \vee [\delta_2 = 1]$	$\delta_1 - \delta_3 \leq 0$ $\delta_2 - \delta_3 \leq 0$ $-\delta_1 - \delta_2 + \delta_3 \leq 0$
P6	NOT ($\sim$)	$\sim [\delta_1 = 1]$	$\delta_1 = 0$
P7	XOR ($\oplus$)	$[\delta_1 = 1] \oplus [\delta_2 = 1]$	$\delta_1 + \delta_2 = 1$
P8		$[\delta_3 = 1] \leftrightarrow$ $[\delta_1 = 1] \oplus [\delta_2 = 1]$	$-\delta_1 - \delta_2 + \delta_3 \leq 0$ $-\delta_1 + \delta_2 - \delta_3 \leq 0$ $\delta_1 - \delta_2 - \delta_3 \leq 0$ $\delta_1 + \delta_2 + \delta_3 \leq 2$
P9	IMPLY ($\rightarrow$)	$[\delta_1 = 1] \rightarrow [\delta_2 = 1]$	$\delta_1 - \delta_2 \leq 0$
P10		$[f(x) \leq 0] \rightarrow [\delta = 1]$	$f(x) \geq \epsilon + (m - \epsilon)\delta$
P11		$[\delta = 1] \rightarrow [f(x) \leq 0]$	$f(x) \leq M - M\delta$
P12	IFF ($\leftrightarrow$)	$[\delta_1 = 1] \leftrightarrow [\delta_2 = 1]$	$\delta_1 - \delta_2 = 0$
P13		$[f(x) \leq 0] \leftrightarrow [\delta = 1]$	$f(x) \leq M - M\delta$ $f(x) \geq \epsilon + (m - \epsilon)\delta$
P14	Product	$z = \delta \cdot f(x)$	$z \leq M \delta$ $-z \leq -m \delta$ $z \leq f(x) - m(1 - \delta)$ $-z \leq -f(x) + M(1 - \delta)$

provide computational benefits [17, p. 171]. By associating the binary variable δ with the literal X, one can transform $X \triangleq [f(x) \leq 0]$ into mixed integer inequalities as described in P13, Table 1, where ϵ is a small tolerance (typically the machine precision), beyond which the constraint is regarded as violated. Note that sometimes this translation requires the introduction of *auxiliary variables* [17, p. 178], for instance according to P14 a product between logic and continuous quantities requires the introduction of a real variable z.

3 Automatic Translation of Truth Tables into Linear Integer Inequalities

Any combinational relation of logical variables can be represented in *conjunctive normal form (CNF)*. Using CNF the derivation of linear inequalities describing a given relation requires the introduction of auxiliary binary variables prior to the usage of the rules of table 1. However, this procedure increases the number of binary variables, which is undesirable.

We describe a method for translating *any* logical relation between Boolean literals, given in the form of a logical proposition or truth table, into a set of linear integer inequalities. This set is also proved to provide the smallest domain after relaxation. Contrary to other methods which perform the translation through CNF, the proposed approach does not introduce any additional Boolean variable. This feature, and the minimality of the relaxed set, are particularly appealing when the model is used to set up mixed-integer programs. The results of applying the method can be seen as a generalization of the rules of Table 1 for relations involving an arbitrary number of exclusively discrete variables combined by arbitrary connectives.

Consider the following problems:

- **PB1** Impose that the proposition $F(X_1, \dots, X_n)$ is true
- **PB2** Define $X_n = f(X_1, \dots, X_{n-1})$

The two problems can be given in terms of truth tables or using connectives "$\wedge$" (and), "$\vee$" (or), "$\sim$" (not), "$\rightarrow$" (implies), "$\leftrightarrow$" (if and only if), "$\oplus$" (exclusive or). As it is immediate to transform statements given in terms of connectives into truth tables, without loss of generality we assume that F, f are defined via a truth table. It is also clear that problem PB2 is included in PB1.

Consider the unit hypercube $\mathcal{H} \triangleq [0,1]^n$, and let H denote the set of its vertices. Let $\mathrm{conv}(S)$ be the convex hull of a set $S \subseteq H$, and $C_H(S)$ the complementary set in H of S, i.e. $C_H(S) \bigcup S = H$, $C_H(S) \bigcap S = \emptyset$. The following Lemma 1 proves that the subsets S of H can be "wrapped" inside a polytope which does not contain any vector of the complementary set $C_H(S)$.

Lemma 1 *Let $S \subseteq H$. Then* $\mathrm{conv}(S) \bigcap C_H(S) = \emptyset$.

Proof: All vectors $h \in H$ are extreme points of $\mathcal{H}$. Therefore, they cannot be written as nontrivial convex linear combinations of other vectors in $\mathcal{H}$. In particular, the vectors $h_c \in C_H(S)$ cannot be written as convex combinations of vectors $h_s \in S$. This proves that $h_c \notin \mathrm{conv}(S)$, $\forall h_c \in C_H(S)$. □

Consider now the truth table $\mathcal{T}$ expressing the combinations of $\delta_1, \dots, \delta_n$ associated with problems PB1/PB2. Let m be the number of rows in $\mathcal{T}$, and let $T = \{R_1, \dots, R_m\}$ the set of the rows R_i of $\mathcal{T}$, considered as vectors in $\mathbb{R}^n$, i.e. the collection of all *true* combinations of F. Each component of a row R_i is either 0 or 1, and therefore $R_i \in H$, i.e. R_i is a vertex of the hypercube $\mathcal{H}$.

Theorem 1 *Let $\mathcal{T}$ be a truth table associated to a statement F on literals $X_1, \dots, X_n$. Then $F(X_1, \dots, X_n)$ is true if and only if the vector $\delta \triangleq [\delta_1, \dots, \delta_n] \in \{0,1\}^n$ satisfies the inequalities*

$$A\delta \leq B$$

where the polytope $P \triangleq \{\delta : A\delta \leq B\} = \mathrm{conv}(T)$. Moreover, each integer translation $\bar{P} = \{\delta : \bar{A}\delta \leq \bar{B}\}$ of F is such that $P \subseteq \bar{P}$, i.e. P is minimal.

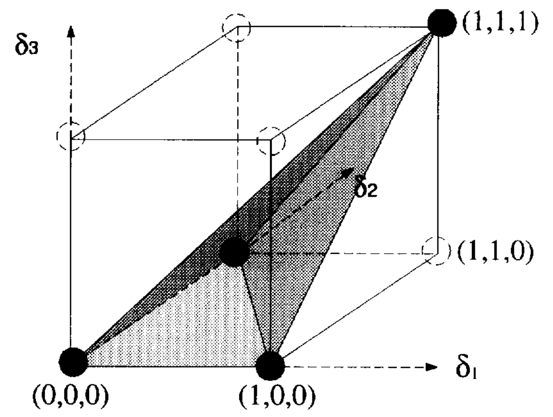

Figure 1: Convex hull of the rows of the truth table of $X_3 = X_1 \wedge X_2$.

Proof: If δ is a true combination, then δ is a row R_i of the truth table, i.e. $\delta \in T$, and therefore $\delta \in \mathrm{conv}(T)$. On the contrary, let $\delta \in \mathrm{conv}(T)$. If δ were a *false* combination, i.e. $\delta \in C_H(T)$, Lemma 1 would be violated. Moreover, if $\bar{P}$ is an integer representation of F, then $R_i \in \bar{P}$, $\forall i = 1, \dots, m$. Since $\bar{P}$ is convex, $P = \mathrm{conv}(\{R_1, \dots, R_m\}) \subseteq \bar{P}$. □

Several packages exist for transforming a polyhedron P from the form $P = \{x : x = \sum_{i=1}^{m} \lambda_i x_i + \sum_{i=1}^{p} \mu_i r_i\}$, where x_i, r_i are the extreme points and extreme directions of P respectively, $0 \leq \lambda_i \leq 1$, $\sum \lambda_i = 1, \mu_i \geq 0$, to the form $P = \{x : Ax \leq B\}$. For a detailed survey of these packages, the reader is referred to `http://www.geom.umn.edu/software/cglist/ch.html`.

Example 3.1

Consider the relation $X_3 = X_1 \wedge X_2$, whose truth table is

X_1	X_2	X_3
F	F	F
F	T	F
T	F	F
T	T	T

$\Leftrightarrow$

δ_1	δ_2	δ_3
0	0	0
0	1	0
1	0	0
1	1	1

The rows of the truth table are represented as points in $\mathbb{R}^3$ in Fig. 1. Their convex hull, computed by using the package `lrs` written by David Avis, McGill University (`ftp://mutt.cs.mcgill.ca/pub/C`), coincides with the linear inequalities in P2 (Table 1). □

4 Mixed Logical Dynamic (MLD) Systems

In the previous Sections we have provided some tools to transform logical facts involving continuous variables into linear inequalities. These tools can be used to express relations describing the evolution of systems where physical laws, logic rules, and operating constraints are interdependent. The general form of the system class obtained are the Mixed Logical Dynamic (MLD) systems [3] given by the following linear relations

$$x(t+1) = Ax(t) + B_1u(t) + B_2\delta(t) + B_3z(t) \quad (2a)$$
$$y(t) = Cx(t) + D_1u(t) + D_2\delta(t) + D_3z(t) \quad (2b)$$
$$E_2\delta(t) + E_3z(t) \leq E_1u(t) + E_4x(t) + E_5 \quad (2c)$$

Each of the vectors x (state), u (input), and y (output) is partitioned into discrete and continuous components, e.g.,

$$x = \begin{bmatrix} x_c \\ x_\ell \end{bmatrix}, \; x_c \in \mathbb{R}^{n_c}, \; x_\ell \in \{0,1\}^{n_\ell}, \; n \triangleq n_c + n_\ell$$

In principle, the inequality in expression (2c) might be satisfied for many values of $\delta(t)$ and/or $z(t)$. On the other hand, in order to define *trajectories* in the x and y-space for system (2), we require that $x(t+1)$ and $y(t)$ are uniquely determined by $x(t)$ and $u(t)$, i.e., that the system is *well posed* (for a formal definition see [3]). Typically, when the model derives from a real system, there is no need to check for well-posedness. However, a simple numerical test for checking this property is reported in [3]. We will denote by $x(t, t_0, x_0, u_{t_0}^{t-1})$ the trajectory generated in accordance with (2) by applying the command inputs $u(t_0)$, $u(t_0+1)$, ..., $u(t-1)$ from the initial state $x(t_0) = x_0$.

Note that input/state constraints of the form $Fx + Gu \leq H$ have the form (2c). Therefore, operating constraints can be included in (2c).

5 Model Components Transformable in MLD Form

The class of MLD systems includes the following important classes of systems:

- Linear Hybrid Systems;
- Sequential logical systems (Finite State Machines, Automata) ($n_c = 0$);
- Nonlinear dynamic systems, where the nonlinearity can be expressed through combinational logic ($n_\ell = 0$);
- Some classes of discrete event systems;
- Linear systems, possibly subject to constraints

where the terms "combinational" and "sequential" are borrowed from digital circuit design jargon.

In [3] some examples are described of basic systems that can be expressed as MLD systems, such as linear systems with output nonlinearities, discrete inputs, qualitative outputs, bilinear systems, piece-wise linear systems, and automata driven by events on continuous dynamics. We refer to [3] for a detailed discussion of these model types.

6 Control

For an MLD system of the form (2), consider the following problem: Given an initial state x_0 and a final time T, find (if it exists) the control sequence $u_0^{T-1} \triangleq \{u(0), u(1), \ldots, u(T-1)\}$ which transfers the state from x_0 to x_f and minimizes the *performance index*

$$J(u_0^{T-1}, x_0) \triangleq \sum_{t=0}^{T-1} \|u(t) - u_f\|_{Q_1}^2 + \|\delta(t, x_0, u_0^t) - \delta_f\|_{Q_2}^2 + \|z(t, x_0, u_0^t) - z_f\|_{Q_3}^2 + \|x(t, x_0, u_0^{t-1}) - x_f\|_{Q_4}^2 + \|y(t, x_0, u_0^{t-1}) - y_f\|_{Q_5}^2 \quad (3)$$

subject to the terminal state constraint

$$x(T, x_0, u_0^{T-1}) = x_f \quad (4)$$

and the MLD system dynamics (2), where $\|x\|_Q^2 \triangleq x'Qx$, $Q_i = Q_i' \geq 0$, $i = 1, \ldots, 5$, are given weight matrices, and x_f, u_f, δ_f, z_f, y_f satisfy (2) in steady state for $x(t+1) = x(t) = x_f$.

This optimal control problem can be solved as a *Mixed-Integer Quadratic Program* (MIQP). In fact, let $x(t)$ be a compact notation for $x(t, x_0, u_0^{t-1})$, and use the same convention for $\delta(t)$, $z(t)$. From (2a), we have the formula

$$\begin{aligned} x(t) \; &= \; A^t x_0 + \sum_{i=0}^{t-1} A^i \left[B_1 u(t-1-i) + B_2\delta(t-1-i) + \right. \\ & \quad \left. B_3 z(t-1-i)\right] \end{aligned} \quad (5)$$

where the relation between $x(t)$ and x_0, u_0^{t-1} is only apparently linear, because $\delta(i)$, $z(i)$ hide a nonlinear dependence on x_0 and u_0^{t-1}, as observed earlier. By plugging (5) into (2c) and (3), the optimization problem is an MIQP in the variables $\{u(0), \ldots, u(T-1), \delta(0), \ldots, \delta(T-1), z(0), \ldots, z(T-1)\}$.

In order to track a desired reference $r(t)$, the optimization problem (3)-(4) is solved on-line in a *receding horizon* fashion leading to a model predictive controller [3]. In particular, at each time t, $J(u_t^{t+T-1}, x(t))$ is minimized for $y_f \triangleq r(t)$ (and x_f, δ_f, z_f defined correspondently), and only the first optimal move $u(t)$ is applied to the system.

The framework of MLD systems allows to include information about the process in form of heuristic rules and logic statements. Prioritization of contraints can also be naturally incorporated when solving the optimal control problem. For details we defer to [3], where a stability result for the controller above is also given.

7 State Estimation and Fault Detection

In the last Section we pointed out that the MLD framework (2) can be used for the synthesis of a model predictive controller [3]. The dual problem, i.e. the *moving horizon estimation* problem [15] can also be formulated in terms of the iterative solution of MIQPs. The aims of such an estimation can be of various nature, like state estimation, fault detection or disturbance estimation. The common feature in all these problems is the minimization of a quadratic cost function involving the quantities to be estimated. Contrary to the control problem, the estimation horizon extends backwards in time, allowing at time t to estimate the quantities of interest at times prior to t. For further details about an application of fault detection and state estimation to MLD systems see [2].

8 Computational Aspects

One drawback of the methods described in this paper lies in the complexity of the MILPs and MIQPs that must be solved. These types of optimization problems exhibit an exponential increase of the worst case complexity with an increasing number of binary variables. However, this does not necessarily preclude the application of the methods. For instance, if we use branch and bound methods (considered widely to be best for these types of problems [8]) the solution time can vary considerably according to the tree exploring strategy and the branching variable selection rule.

We have experimented successfully with a tree exploring strategy which assumes that the binary variables change only infrequently over the considered time horizon [1]. This assumption is particularly good, when the binary variables represent faults that do not occur very often and that are usually not recoverable.

Also note that for control purposes it is not critical to find the global optimum to guarantee stability, a feasible suboptimal solution suffices.

9 Conclusions

Motivated by the key idea of transforming propositional logic into linear mixed-integer inequalities, and by the availability of techniques for solving mixed-integer quadratic programs, this paper presented a framework for modeling and controlling systems described by both dynamics and logic, and subject to operating constraints, denoted as *Mixed Logical Dynamical* (MLD) systems. The MLD structure is also suitable for formal verification, state estimation, and fault detection.

References

[1] A. Bemporad, D. Mignone, and M. Morari. An Efficient Branch and Bound Algorithm for State Estimation and Control of Hybrid Systems. *European Control Conference*, 1999.

[2] A. Bemporad, D. Mignone, and M. Morari. Moving Horizon Estimation for Hybrid Systems and Fault Detection. *Proceedings of the American Control Conference*, 1999.

[3] A. Bemporad and M. Morari. Control of systems integrating logic, dynamics, and constraints. *Automatica*, Special issue on hybrid systems, to appear. `ftp://control.ethz.ch/pub/reports/postscript/AUT98-04.ps`, 1999.

[4] A. Bemporad and M. Morari. Verification of Hybrid Systems via Mathematical Programming. In *Hybrid Systems: Computation and Control; Second International Workshop, Nijmegen, The Netherlands*, 1999.

[5] M.S. Branicky, V.S. Borkar, and S.K. Mitter. A unified framework for hybrid control: model and optimal control theory. *IEEE Transactions on Automatic Control*, 43(1):31–45, 1998.

[6] T.M. Cavalier, P.M. Pardalos, and A.L. Soyster. Modeling and integer programming techniques applied to propositional calculus. *Computers Opns Res.*, 17(6):561–570, 1990.

[7] D. Christiansen. *Electronics Engineers' Handbook, 4th edition.* IEEE Press/ McGraw Hill, Inc., 1997.

[8] R. Fletcher and S. Leyffer. Numerical experience with lower bounds for MIQP branch-and-bound. Technical report, University of Dundee, Dept. of Mathematics, Scotland, U.K., 1995. submitted to SIAM Journal on Optimization, `http://www.mcs.dundee.ac.uk:8080/~sleyffer/miqp_art.ps.Z`.

[9] R.L. Grossmann and H. Rischel, editors. *Hybrid Systems, no. 736 Lecture Notes in Computer Science.* Springer Verlag, New York, 1993.

[10] J.P. Hayes. *Introduction to Digital Logic Design.* Addison-Wesley Publishing Company, Inc., 1993.

[11] E. Mendelson. *Introduction to mathematical logic.* Van Nostrand, 1964.

[12] R. C. Minnick. Linear-Input Logic. *IRE Transactions on Electronic Computers*, pages 6–16, March 1961.

[13] R. Raman and I.E. Grossmann. Integration of logic and heuristic knowledge in MINLP optimization for process synthesis. *Computers and Chemical Engineering*, 16(3):155–171, 1992.

[14] O. B. Stram. Arbitrary Boolean Functions of N Variables Realizable in Terms of Threshold Devices. *Proceedings of the IRE*, 49:210–220, January 1961.

[15] M. Tyler and M. Morari. Stability of Constrained Moving Horizon Estimation Schemes. *Automatica*, 1999.

[16] H.P. Williams. Logical problems and integer programming. *Bulletin of the Institute of Mathematics and Its Applications*, 13:18–20, 1977.

[17] H.P. Williams. *Model Building in Mathematical Programming.* John Wiley & Sons, Third Edition, 1993.