Chapter 3

LOGIC-BASED MODELING

John N Hooker
Graduate School of Industrial Administration
Carnegie Mellon University, Pittsburgh, PA 15213
USA
jh38@andrew.cmu.edu

Abstract
Logic-based modeling can result in decision models that are more natural and easier to debug. The addition of logical constraints to mixed integer programming need not sacrifice computational speed and can even enhance it if the constraints are processed correctly. They should be written or automatically reformulated so as to be as nearly consistent or hyperarc consistent as possible. They should also be provided with a tight continuous relaxation. This chapter shows how to accomplish these goals for a number of logic-based constraints: formulas of propositional logic, cardinality formulas, 0-1 linear inequalities (viewed as logical formulas), cardinality rules, and mixed logical/linear constraints. It does the same for three global constraints that are popular in constraint programming systems: the all-different, element and cumulative constraints.

Clarity is as important as computational tractability when building scientific models. In the broadest sense, models are descriptions or graphic representations of some phenomenon. They are typically written in a formal or quasi-formal language for a dual purpose: partly to permit computation of the mathematical or logical consequences, but equally to elucidate the conceptual structure of the phenomenon by describing it in a precise and limited vocabulary. The classical transportation model, for example, allows fast solution with the transportation simplex method but also displays the problem as a network that is easy to understand.

Optimization modeling has generally emphasized ease of computation more heavily than the clarity and explanatory value of the model. (The transportation model is a happy exception that is strong on both counts.) This is due in part to the fact that optimization, at least in the context of operations research, is often more interested in prescription than description. Practitioners who model
a manufacturing plant, for example, typically want a solution that tells them how the plant should be run. Yet a succinct and natural model offers several advantages: it is easier to construct, easier to debug, and more conducive to understanding how the plant works.

This chapter explores the option of enriching mixed integer programming (MILP) models with logic-based constraints, in order to provide more natural and succinct expression of logical conditions. Due to formulation and solution techniques developed over the last several years, a modeling enrichment of this sort need not sacrifice computational tractability and can even enhance it.

One historical reason for the de-emphasis of perspicuous models in operations research has been the enormous influence of linear programming. Even though it uses a very small number of primitive terms, such as linear inequalities and equations, a linear programming model can formulate a remarkably wide range of problems. The linear format almost always allows fast solution, unless the model is truly huge. It also provides such analysis tools as reduced costs, shadow prices and sensitivity ranges. There is therefore a substantial reward for reducing a problem to linear inequalities and equations, even when this obscures the structure of the problem.

When one moves beyond linear models, however, there are less compelling reasons for sacrificing clarity in order to express a problem in a language with a small number of primitives. There is no framework for discrete or discrete/continuous models, for example, that offers the advantages of linear programming. The mathematical programming community has long used MILP for this purpose, but MILP solvers are not nearly as robust as linear programming solvers, as one would expect because MILP formulates NP-hard problems. Relatively small and innocent-looking problems can exceed the capabilities of any existing solver, such as the market sharing problems identified by Williams [35] and studied by Comuejols and Dawande [16]. Even tractable problems may be soluble only when carefully formulated to obtain a tight linear relaxation or an effective branching scheme.

In addition MILP often forces logical conditions to be expressed in an unnatural way, perhaps using big-$M$ constraints and the like. The formulation may be even less natural if one is to obtain a tight linear relaxation. Current solution technology requires that the traveling salesman problem, for example, be written with exponentially many constraints in order to represent a simple all-different condition. MILP may provide no practical formulation at all for important problem classes, including some resource-constrained scheduling problems.

It is true that MILP has the advantage of a unified solution approach, since a single branch-and-cut solver can be applied to any MILP model one might write. Yet the introduction of logic-based and other higher-level constraints no longer sacrifices this advantage.