Les Cahiers du GERAD ISSN: 0711–2440

A structured approach to model logical constraints

M. Gamache

G–2015–48

May 2015

GERAD HEC Montréal
3000, chemin de la Côte-Sainte-Catherine
Montréal (Québec) Canada H3T 2A7

Tél. : 514 340-6053
Téléc. : 514 340-5665
info@gerad.ca
www.gerad.ca

A structured approach to model logical constraints

Michel Gamache

GERAD & Department of Mathematics and Industrial Engineering, Polytechnique Montréal, Montréal (Québec) Canada, H3C 3A7

michel.gamache@polymtl.ca

May 2015

Les Cahiers du GERAD

G–2015–48

Abstract: This paper presents a structured approach to model logical constraints (expressions that contain propositions and logical operators) in a linear program using binary variables. The approach proceeds in three steps: (1) translating the English sentences into logical compound propositions; (2) transforming these compound propositions in a conjunctive normal form; and (3) creating a linear inequality for each clause of the conjunctive normal form. A detailed example describing the use of this approach is provided.

Key Words: Logical constraints, proposition, clause, conjunctive normal form, linear programming, binary variables.

Résumé : Cet article présente une approche structurée pour modéliser des contraintes logiques (expressions qui contiennent des propositions et des opérateurs logiques) dans un programme linéaire en utilisant des variables binaires. L'approche procède en trois étapes : (1) traduire les phrases en propositions logiques; (2) transformer ces propositions en clauses; et (3) créer une inéquation linéaire pour chaque clause. Un exemple détaillé décrivant l'utilisation de cette approche est donné.

1 Motivations

Building a linear programming model is not an easy task, especially when constraints involve logical expressions, i.e. expressions that contain propositions and logical operators. A *proposition* is a declarative sentence that is either true or false, but not both. It is represented by a *propositional variable*. In linear programming, such propositions occur frequently when it is necessary to know if a machine is on or off, if an event has occurred or not, if a candidate has been selected or not, etc. Propositions are represented using binary variables (also called Boolean or 0-1 variables). Often, *logical operators* or *connectives*, such as AND, OR, and NOT, link these propositions together to create new compound propositions that are called, in this paper, logical expressions. Examples of logical expressions occur in different contexts: a machine M_1 can be used only when machines M_2 and M_3 are already used to their maximal capacity; in open pit mines, a block of material can be extracted only when the k blocks of material immediately above it have been extracted; in scheduling, tasks T_2 and T_3 can start only when task T_1 is finished; if job J is processed, then machines M_1 or M_2 and machines M_3 or M_4 must be in operation, etc.

In the previous examples, translating the restrictions into constraints represented by linear equations is not always trivial. Often, a trial and error approach is used to build the linear inequalities and to make sure that they satisfy all the possible combinations of values of the propositional variables. To the knowledge of the author, no books of operations research dedicated for undergraduate students provides an approach for translating these logical constraints into constraints of a linear programming model. The objective of this paper is to present a structured approach for modelling such logical expressions. The approach proceeds in three steps that consist of : (1) translating English sentences into logical expresions; (2) rewriting these logical expressions in a specific syntax; and (3) converting these expressions in linear inequalities.

The next section provides a reminder of some notions of logic and sets up the elements for the first step of the approach, i.e. the translation from English sentences to logical expressions. The third section establishes the link between logical expressions, when these expressions are written under a specific syntax, and linear inequalities. This syntax will permit the translation of complex logical expressions into linear inequalities. Finally, Section 4 presents the application of this approach to a small academic problem.

2 Reminder on some basic knowledge of logic

In this section, we give the basic definitions of some elements of logics that will help to translate English sentences into logical expressions. We also show how to transform this logical expression in equivalent expressions that will be more useful for the translation of logical expressions into linear constraints.

2.1 Definitions

In the following definitions, let p and q be two propositional variables. The *truth value* of a propositional variable is true, denoted T, if the proposition is true; it is false, denoted F, in the opposite. Four logical operators can be used to form compound propositions: the negation, the conjunction, the disjunction, and the implication.

Proposition NOT p, denoted $\neg p$ or $\overline{p}$ and called the *negation* of p, is a logical operator that is equal to the opposite value of p. A truth table can be used to display the relationships between the truth values of p and $\neg p$. This truth table and those associated with the following operators have been grouped in Table A in the Appendix.

Proposition p AND q, denoted $p \wedge q$ or $p \cdot q$ and called the *conjunction* of p and q, is true when both p and q are true and is false otherwise.

Proposition p OR q, denoted $p \vee q$ and called the *disjunction* of p and q, is false when both p and q are false and is true otherwise. A disjunction is either *inclusive* or *exclusive*. An *inclusive disjunction* is true when at least one proposition is true. The inclusive disjunction is the one that has just been defined. An *exclusive disjunction*, denoted $p \oplus q$ is true when exactly one of proposition p and proposition q is true. It

is false when both propositions are false and when both are true. In the following text, all conjunctions are inclusive.

Proposition p implies q, denoted $p \rightarrow q$ and called the *implication*, is a proposition that is false when p is true and q is false, and true otherwise. Proposition p is called the *hypothesis* and q the *consequence*.

The *biconditionnal*, denoted $A \leftrightarrow B$, is a proposition that is true when A and B have the same truth values, and is false otherwise.

2.2 Translating statements into logical expressions

The translation of statements into logical expressions is difficult. Particularly, much difficulty arises from compound propositions involving implications; more precisely, the identification of the hypothesis and the consequence. Most often, the hypothesis will be associated with the keyword *if* and the consequence with the keyword *then*. Figure 2.1 illustrates the translation of an implication.

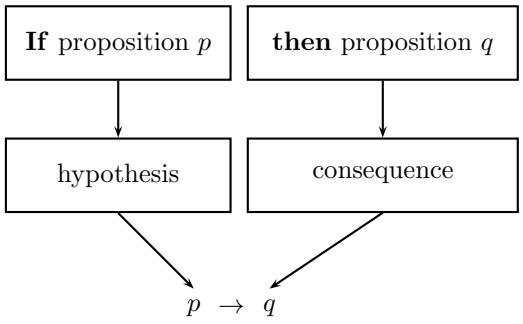

Figure 2.1: Identification of the hypothesis and the consequence

Books on discrete mathematics offer some tricks in order to translate some English statements into logical expressions (see Rosen (2003), Dossey et al. (2005)). For example, Rosen (2003) shows 12 different ways to write an English statement for the logical expression $p \rightarrow q$. Some examples will be given in Section 4.

Such translations necessitate a good knowledge of the language into which the statement has been written. Since each language has its own characteristics, it makes difficult to develop a general concept that will be language independent. It is not the intention of the author to develop more on this issue. Readers are invited to consult the above references to develop their skills for writing logical expressions.

2.3 Equivalences

Compound propositions that have the same truth values in all possible cases are called logically equivalent. An equivalence allows substituting a compound proposition with another compound proposition having the same truth value. Table 2.1 presents some well established equivalences between logical expressions. These equivalences will be essential in Section 3 for rewriting compound proposition under a specific syntax that will ease their translation into linear equations.

3 Translating compound propositions into linear equations

Some compound propositions can be easily translated into linear inequalities. This is the case for the disjunction of two or more propositions and the negation of a proposition. Using logical equivalences and knowing how to translate these compound propositions into linear inequalities, it will be shown that some complex logical expressions can be translated quite easily into constraints of a linear programming model.

Table 2.1: Equivalences between logical expressions

Name	Equivalence	Identificator
Commutative laws	$p \vee q \equiv q \vee p$	eqv01
	$p \wedge q \equiv q \wedge p$	eqv02
Associative laws	$(p \vee q) \vee r \equiv p \vee (q \vee r) \equiv p \vee q \vee r$	eqv03
	$(p \wedge q) \wedge r \equiv p \wedge (q \wedge r) \equiv p \wedge q \wedge r$	eqv04
Distributive laws	$p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$	eqv05
	$p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$	eqv06
De Morgan's law	$\neg(p \wedge q) \equiv \neg p \vee \neg q$	eqv07
	$\neg(p \vee q) \equiv \neg p \wedge \neg q$	eqv08
Negation laws	$p \vee \neg p \equiv T$	eqv09
	$p \wedge \neg p \equiv F$	eqv10
Implication	$(p \rightarrow q) \equiv \neg p \vee q$	eqv11
Biconditionnal	$(p \leftrightarrow q) \equiv (p \rightarrow q) \wedge (q \rightarrow p)$	eqv12

In the proposed approach, the compound proposition representing the logical expression should be expressed as a clause or a conjunctive normal form. A *clause*, denoted $\mathcal{C}$, is an inclusive disjunction of propositional variables or negations of these propositional variables, while a *conjunctive normal form* or a *clausal normal form* is a conjunction of clauses. For example, $A \vee B \vee \neg C \vee F$ is a clause, while $(\neg A \vee B) \wedge (C)$ is in a conjunctive normal form. In the latter, $(\neg A \vee B)$ and (C) are two clauses.

A conjunctive normal form $\mathcal{C}_1 \wedge \mathcal{C}_2 \wedge \cdots \wedge \mathcal{C}_n$ is true only when all the clauses $\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_n$ are true. Based on the assumption that we can find a linear inequality for each clause that composes the conjunction, this logical expression can be translated into a set of n constraints, one for each clause.

3.1 Translating a clause into a linear inequality

In linear programming, a logical expression is a constraint that must be satisfied by any feasible solution. The word "satisfy" means that when we use the value of the solution in the constraint, the relation expressed by this constraint is "true".

Let the clause $\mathcal{C} \equiv p_1 \vee p_2 \vee \cdots \vee p_n$. The clause $\mathcal{C}$ is true when at least one propositional variable among $p_1, p_2, \ldots, p_n$ is true. Let x_i be a binary variable associated with a propositional variable p_i such as

$$x_i = \begin{cases} 1 & \text{if the propositional variable } p_i \text{ is true;} \\ 0 & \text{otherwise.} \end{cases}$$

If $x_1, x_2, \ldots, x_n$ are associated with propositional variables $p_1, p_2, \ldots, p_n$ respectively, then it means that at least one variable must be equal to 1. This sentence can be translated into the following linear inequality:

$$p_1 \vee p_2 \vee \cdots \vee p_n \quad \equiv \quad x_1 + x_2 + \cdots + x_n \geq 1 \tag{3.1}$$

Table 3.1 compares the clause and its equivalent inequality in linear algebra in order to highlight the equivalence between both expressions.

3.2 Negation

Let x_p be a binary variable that represents the propositional variable p and $\overline{x}_p$ be a binary variable that represents $\neg p$. From the previous definition, we know that $x_p = 1$ when p is true; otherwise $x_p = 0$. From the definition of x_p and $\neg p$, it becomes obvious that $\overline{x}_p = 0$ when p is true, i.e. when $x_p = 1$. Similarly, $\overline{x}_p = 1$ when p is false, i.e. $x_p = 0$.

Table 3.1: Equivalence between the logical expression and the inequality

p	q	$p \vee q$	x_p	x_q	$x_p + x_q \geq 1$
T	T	T	1	1	$2 \geq 1 \equiv T$
T	F	T	1	0	$1 \geq 1 \equiv T$
F	T	T	0	1	$1 \geq 1 \equiv T$
F	F	F	0	0	$0 \geq 1 \equiv F$

From these observations, we can deduce the following equation:

$$\neg p \quad \equiv \quad \overline{x}_p = 1 - x_p \tag{3.2}$$

Table 3.2 compares the logical expression and its equivalent equation in linear algebra in order to highlight the equivalence between both expressions.

Table 3.2: Equivalence between the compound proposition and the inequality

p	$\neg p$	x_p	$1 - x_p$
T	F	1	$0 \equiv F$
F	T	0	$1 \equiv T$

3.3 Implication

The logical expression *If p is true, then q must be true* is equivalent to the implication, i.e. $p \rightarrow q$. From (eqv11) in Table 2.1, that implication is equivalent to

$$p \rightarrow q \equiv \neg p \vee q$$

which is a clause. From equations (3.1) and (3.2) , one can deduce the following equivalences:

$$\begin{aligned} p \rightarrow q \quad &\equiv \quad \neg p \vee q \\ &\equiv \quad (1 - x_p) + x_q \geq 1 \\ &\equiv \quad x_p \leq x_q \end{aligned}$$

Table 3.3 compares the logical expression and its equivalent equation in linear algebra in order to highlight the equivalence between both expressions.

Table 3.3: Equivalence between the compound proposition and the inequality

p	q	$p \rightarrow q$	x_p	x_q	$x_p \leq x_q$
T	T	T	1	1	$1 \leq 1 \equiv T$
F	T	T	0	1	$0 \leq 1 \equiv T$
T	F	F	1	0	$1 \leq 0 \equiv F$
F	F	T	0	0	$0 \leq 1 \equiv T$

The proposed approach can be summarized in three steps:

Step 1 Translate English sentences into compound propositions;

Step 2 Use the logical equivalences to transform the compound propositions into a clause or a conjunctive normal form;

Step 3 Use equations (3.1) and (3.2) to translate in each clause in a linear inequality.

Section 4 presents an academic example that illustrates the use of this technique to translate logical expressions into linear equations.

4 Example

Suppose that a project director wants to constitute a group of 8 scientists that will have to work together for one year in a laboratory emulating the conditions of the biosphere, without getting in contact with the external world. Among all the candidates who applied for a job, twelve of them (denoted A to L) remain in competition to obtain a place in the biosphere. To compose the best team of scientists, the project director must take into account different characteristics of the candidates, such as their domain of research, the languages they speak and their psychological profile in order to evaluate their affinity with the other candidates, etc. After analyzing the candidates, a score between 0 and 20 is given for each candidate, where a score of 20 represents the best candidate and 0 the worst. Table 4.1 gives the score of each candidate.

Table 4.1: Score of the candidates

Candidate	Score	Candidate	Score
A	10	G	13
B	12	H	15
C	14	I	18
D	11	J	11
E	14	K	10
F	15	L	18

The problem consists of maximizing the score of the team while also taking into account the following restrictions while composing the team:

(a) Candidates A, E and H are chemists, and the team must have at least one chemist;
(b) Candidate A is epileptic and he can be selected only if F and G are selected since both are doctors;
(c) Candidate B speaks Russian only, she cannot be selected unless candidate H or I is selected since these two candidates are the only one who can speak Russian;
(d) Due to psychological incompatibility, if candidate E is selected, than candidate F or candidate G must be rejected;
(e) Candidates I and J cannot be selected, if candidates C and D are selected;
(f) If I or H is selected, then J or K must be selected;
(g) If K is selected, then L must be selected, and vice-versa since they are husband and wife;
(h) If candidate A is chosen, then candidate B or candidate C must be chosen, but not candidate D.

To elaborate the linear program for this problem, let x_i be a binary variable such that :

$$x_i = \begin{cases} 1 & \text{if candidate } i \text{ is selected;} \\ 0 & \text{otherwise.} \end{cases}$$

The objective function, which consists of maximizing the score of the team, can be written as follows:

$$\max Z = 10x_A + 12x_B + \cdots + 10x_K + 18x_L$$

Let us now see how these logical expressions are translated into linear inequalities.

(a) Candidates A, E and H are chemists, and the team must have at least one chemist;
This sentence is translated into a logical expression:

$$A \vee E \vee H$$

Since this logical expression is represented by a unique clause, this clause is simply translated into the following linear equation:

$$x_A + x_E + x_H \geq 1$$

(b) Candidate A is epileptic and he can be selected only if F and G are selected since both are doctors;

This sentence is equivalent to: *If candidate A is selected, then candidates F and G must be selected.*

Step 1: Translate the English sentence into a compound proposition

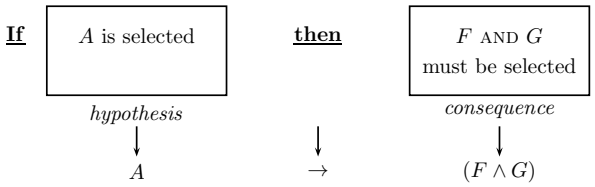

Figure 4.1: Illustration of Step 1 for question (b)

Step 2: Transform the compound proposition in a conjunctive normal form

$$\begin{aligned} A \rightarrow (F \wedge G) &\equiv \neg A \vee (F \wedge G) && (eqv11) \\ &\equiv (\neg A \vee F) \wedge (\neg A \vee G) && (eqv06) \end{aligned}$$

Step 3: Write the linear inequality associated with each clause

Clause 1: $(\neg A \vee F)$

$$\begin{aligned} (\neg A \vee F) &\equiv (1 - x_A) + x_F \geq 1 \\ &\equiv x_A \leq x_F \end{aligned}$$

Clause 2: $(\neg A \vee G)$

$$\begin{aligned} (\neg A \vee G) &\equiv (1 - x_A) + x_G \geq 1 \\ &\equiv x_A \leq x_G \end{aligned}$$

These two equations can be combined into one equation:

$$2x_A \leq x_F + x_G$$

(c) Candidate B speaks Russian only, she cannot be selected unless candidate H or I is selected since these two candidates are the only one who can speak Russian;

This sentence is equivalent to: *If candidate B is selected, then candidates H or I must be selected.*

Step 1: Translate the English sentence into a compound proposition

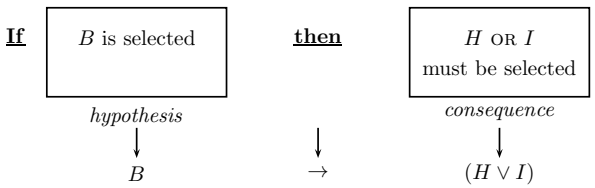

Figure 4.2: Illustration of Step 1 for question (c)

Step 2: Transform the compound proposition in a conjunctive normal form

$$\begin{aligned} B \rightarrow (H \vee I) &\equiv \neg(B) \vee (H \vee I) && (eqv11) \\ &\equiv (\neg B \vee H \vee I) && (eqv03) \end{aligned}$$

Step 3: Write the linear inequality associated with each clause

$$\begin{aligned} (\neg B \vee H \vee I) &\equiv (1 - x_B) + x_H + x_I \geq 1 \\ &\equiv x_B \leq x_H + x_I \end{aligned}$$

(d) Due to psychological incompatibility, if candidate E is selected, than candidate F or candidate G must be rejected;

This sentence is equivalent to: *If candidate E is selected, then candidates F is not selected or G is not selected.*

Step 1: Translate the English sentence into a compound proposition

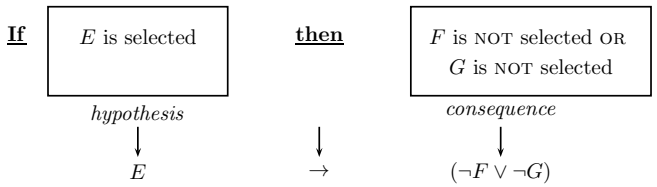

Figure 4.3: Illustration of Step 1 for question (d)

Step 2: Transform the compound proposition in a conjunctive normal form

$$\begin{aligned} E \rightarrow (\neg F \vee \neg G) &\equiv \neg E \vee (\neg F \vee \neg G) && (eqv11) \\ &\equiv \neg E \vee \neg F \vee \neg G && (eqv03) \end{aligned}$$

Step 3: Write the linear inequality associated with each clause

Clause 1: $\neg E \vee \neg F \vee \neg G$

$$\begin{aligned} \neg E \vee \neg F \vee \neg G &\equiv (1 - x_E) + (1 - x_F) + (1 - x_G) \geq 1 \\ &\equiv x_E + x_F + x_G \leq 2 \end{aligned}$$

(e) Candidates I and J cannot be selected, if candidates C and D are selected;
This sentence is equivalent to: *If candidate C and D are selected, then candidates I and J are not selected.*

Step 1: Translate the English sentence into a compound proposition

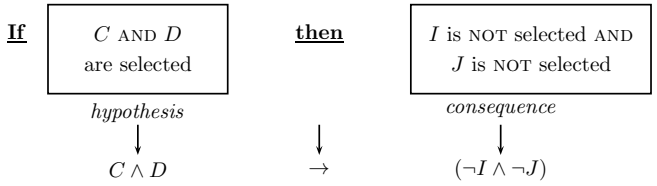

Figure 4.4: Illustration of Step 1 for question (e)

Step 2: Transform the compound proposition in a conjunctive normal form

$$\begin{aligned} (C \wedge D) \rightarrow (\neg I \wedge \neg J) &\equiv \neg(C \wedge D) \vee (\neg I \wedge \neg J) && (eqv11) \\ &\equiv (\neg C \vee \neg D) \vee (\neg I \wedge \neg J) && (eqv08) \\ &\equiv (\neg C \vee \neg D \vee \neg I) \wedge (\neg C \vee \neg D \vee \neg J) && (eqv06) \end{aligned}$$

Step 3: Write the linear inequality associated with each clause

Clause 1: $(\neg C \vee \neg D \vee \neg I)$

$$\begin{aligned} (\neg C \vee \neg D \vee \neg I) &\equiv (1 - x_C) + (1 - x_D) + (1 - x_I) \geq 1 \\ &\equiv x_C + x_D + x_I \leq 2 \end{aligned}$$

Clause 2: $(\neg C \vee \neg D \vee \neg J)$

$$\begin{aligned} (\neg C \vee \neg D \vee \neg J) &\equiv (1 - x_C) + (1 - x_D) + (1 - x_J) \geq 1 \\ & \quad x_C + x_D + x_J \leq 2 \end{aligned}$$

These two equations can be combined into the following equation:

$$2(x_C + x_D) + x_I + x_J \leq 4$$

(f) If I or H is selected, then J or K must be selected;
This sentence is equivalent to: *If candidate I or H is selected, then candidates J or K is selected.*

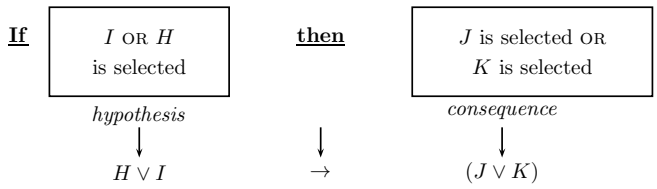

Figure 4.5: Illustration of Step 1 for question (f)

Step 1: Translate the English sentence into a compound proposition

Step 2: Transform the compound proposition in a conjunctive normal form

$$\begin{array}{rcll}(I \vee H) \rightarrow (J \vee K) & \equiv & \neg(I \vee H) \vee (J \vee K) & (eqv11)\\ & \equiv & (\neg I \wedge \neg H) \vee (J \vee K) & (eqv08)\\ & \equiv & (\neg I \vee J \vee K) \wedge (\neg H \vee J \vee K) & (eqv05)\end{array}$$

Step 3: Write the linear inequality associated with each clause

Clause 1: $(\neg I \vee J \vee K)$

$$\begin{array}{rcl}(\neg I \vee J \vee K) & \equiv & (1 - x_I) + x_J + x_K \geq 1\\ & \equiv & x_J + x_K \leq x_I\end{array}$$

Clause 2: $(\neg H \vee J \vee K)$

$$\begin{array}{rcl}(\neg H \vee J \vee K) & \equiv & (1 - x_H) + x_J + x_K \geq 1\\ & & x_J + x_K \leq x_H\end{array}$$

These two equations can be combined into the following equation:

$$2(x_J + x_K) \leq x_I + x_H$$

(g) If K is selected, then L must be selected, and vice-versa since they are husband and wife
This sentence is equivalent to: *If candidate K is selected, then candidate L is selected, and if K is not selected, then L is not selected.*

Step 1: Translate the English sentence into a compound proposition

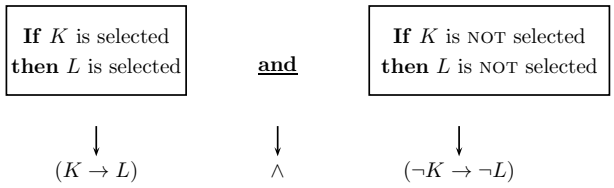

Figure 4.6: Representation of the biconditional : question (g)

Step 2: Transform the compound proposition in a conjunctive normal form

$$\begin{array}{rcll}K \leftrightarrow L & \equiv & (K \rightarrow L) \wedge (L \rightarrow K) & (eqv12)\\ & \equiv & (\neg K \vee L) \wedge (\neg L \vee K) & (eqv11)\end{array}$$

Step 3: Write the linear inequality associated with each clause

Clause 1: $(\neg K \vee L)$

$$\begin{array}{rcl}(\neg K \vee L) & \equiv & (1 - x_K) + x_L \geq 1\\ & \equiv & x_K \leq x_L\end{array}$$

Clause 2: $(\neg L \vee K)$

$$\begin{aligned}(\neg L \vee K) &\equiv (1 - x_L) + x_K \geq 1 \\ &\quad\; x_L \leq x_K\end{aligned}$$

These two equations are equivalent to the following one:

$$x_K = x_L$$

(h) If candidate A is chosen, then candidate B or candidate C must be chosen, but not candidate D.

This sentence is equivalent to: *If candidate A is selected, then candidates B or C is selected, but not candidate D.*

Step 1: Translate the English sentence into a compound proposition

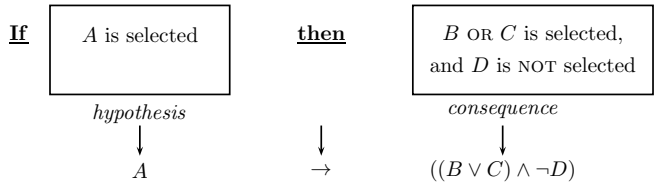

Figure 4.7: Illustration of Step 1 for question (h)

Step 2: Transform the compound proposition in a conjunctive normal form

$$\begin{aligned}A \rightarrow ((B \vee C) \wedge \neg D) &\equiv \neg A \vee ((B \vee C) \wedge \neg D) && (eqv11) \\ &\equiv (\neg A \vee B \vee C) \wedge (\neg A \vee \neg D) && (eqv06)\end{aligned}$$

Step 3: Write the linear inequality associated with each clause

Clause 1: $(\neg A \vee B \vee C)$

$$\begin{aligned}(\neg A \vee B \vee C) &\equiv (1 - x_A) + x_B + x_C \geq 1 \\ &\equiv x_B + x_C \geq x_A\end{aligned}$$

Clause 2: $(\neg A \vee \neg D)$

$$\begin{aligned}(\neg A \vee \neg D) &\equiv (1 - x_A) + (1 - x_D) \geq 1 \\ &\quad\; x_A + x_D \leq 1\end{aligned}$$

Finally, a constraint must be added to make sure that 8 candidates are selected among the 12 candidates.

$$\sum_{i=A}^{L} x_i = 8.$$

The linear program is summarized as follows:

$$\begin{aligned}&\max\; Z = 10x_A + 12x_B + \cdots + 10x_K + 18x_L \\ &\text{subject to:} \\ &\qquad x_A + x_E + x_H \geq 1 \\ &\qquad 2x_A - x_F - x_G \leq 0 \\ &\qquad x_B - x_H - x_I \leq 0 \\ &\qquad x_E + x_F + x_G \leq 2 \\ &\qquad 2(x_C + x_D) + x_I + x_J \leq 4 \\ &\qquad 2(x_J + x_K) - x_I - x_H \leq 0 \\ &\qquad x_K - x_L = 0 \\ &\qquad x_B + x_C - x_A \geq 0 \\ &\qquad x_A + x_D \leq 1 \\ &\qquad \textstyle\sum_{i=A}^{L} x_i = 8 \\ &\qquad x_i \in \{0,1\} \quad i = A, B, \ldots, L\end{aligned}$$

The optimal solution to this problem is to select candidates B, C, E, F, H, I, K, and L which provides a score of 116 for the team.

5 Conclusion

For the last ten years, this approach has been presented in an undergraduate course of operations research in an engineering school. First, the students must develop a linear model to solve the biosphere problem. The modelling of logical constraints into linear inequalities represents a major challenge for the majority of the students. Most of them use a trial and error approach to develop those equations. Most of the time, especially when implication proposals are involved, the resulting inequalities from their approach are valid when the binary variable representing the hypothesis is equal to 1, but are invalid or too restrictive compared to the statement of the problem when this variable is equal to 0. With this exercise, students realize that modelling such logical constraints is not an easy task.

The approach based on logic is then presented. The students retry the formulation of a linear program for the biosphere problem using this approach. The major difficulty is to translate each constraint into a logical expression. However, when this translation is well done, finding equivalent logical expressions, dividing the logical expression into clauses and translating these clauses into inequalities do not pose any problem.

Since the approach based on logic provides a systemic method that is easy to use, it gives the students confidence and considerably improves their ability to model such difficult constraints.

Appendix A Truth Table

p	q	$\neg p$	$p \wedge q$	$p \vee q$	$p \oplus q$	$p \rightarrow q$	$p \leftrightarrow q$
T	T	F	T	T	F	T	T
T	F	F	F	T	T	F	F
F	T	T	F	T	T	T	F
F	F	T	F	F	F	T	T

References

[1] K. H. Rosen, Discrete Mathematics and its Applications, Fifth Edition, McGraw Hill. 2003.

[2] Dossey, John A., Otto, Albert D., Spence, Lawrence E., Vanden Eynden, C., Discrete Mathematics, Fifth Edition, Pearson. 2005.