Some Examples of Elaborated Proofs

Finiteness Properties in Partial Orderings *

B.Buchberger
RISC-Linz

August 8, 1991

Abstract

We present a proof of the well-known fact that in partial orderings the following four conditions are equivalent:

- Every ideal has a finite basis.
- Every non-ascending sequence is finite.
- Every infinite sequence has an infinite ascending subsequence.
- Every strictly descending sequence is finite and every set of mutually incomparable elements is finite.

*Solution to the exam for the course "Thinking, Speaking, Writing", 28 June 91

1 Introduction

Finiteness properties in partial orderings are important for proving termination of algorithms. For example, termination of the Gröbner bases algorithm can be proven by applying the lemmas presented in this paper to the divisibility ordering on power products.

The main part of this paper is Section 4, in which various finiteness conditions for ideals and sequences in partial orderings are proven equivalent. In Section 2 the basic notions are defined. Section 3 introduces a graphical representation for the basic notions that will make it easy to understand the intuition behind the proofs. Section 5 is an appendix. It contains some elementary properties of ideals, which are used in the proof of the main theorem in Section 4.

2 Definitions

We use the convention that, if R, P etc. are sets, then r, p etc. are used as typed variables for elements in R, P etc. j, k, l are typed variables ranging over $\mathbf{N}$, the set of natural numbers ≥ 1. We assume that the notion of a P-sequence $\hat{p}$ (a sequence of elements in P) and of the length $|\hat{p}|$ of such a sequence is known.

Definition 2.1 (Relational Domains)

$(R, <)$ is a *relational domain* iff $R \neq \emptyset$ and $< \subseteq R \times R$.

Definition 2.2 (Reflexive and Anti-Reflexive Extensions)

$(R, <, \leq)$ is the *reflexive extension* of the relational domain $(R, <)$ iff
$r_1 \leq r_2$ iff $(r_1 < r_2$ or $r_1 = r_2)$.

$(R, \leq, <)$ is the *anti-reflexive extension* of the relational domain $(R, \leq)$ iff
$r_1 < r_2$ iff $(r_1 \leq r_2$ and $r_1 \neq r_2)$.

Definition 2.3 (Partial Orderings)

$(P, <)$ is a *partial ordering* iff
$(P, <)$ is a relational domain,
if $p_1 < p_2 < p_3$ then $p_1 < p_3$, (transitivity)
not $(p_1 < p_2$ and $p_2 < p_1)$. (antisymmetry)
□

Let now $(P, <)$ range over partial orderings and I and B range over subsets of P.

Definition 2.4 (Ascending and Non-Ascending Sequences)

$\hat{p}$ is an *ascending* $(P, <)$-sequence iff
$\hat{p}$ is a P-sequence and,
for all $j < |\hat{p}|$, $\hat{p}_j \leq \hat{p}_{j+1}$.

$\hat{p}$ is a *non-ascending* $(P, <)$-sequence iff
$\hat{p}$ is a P-sequence and,
for no j, k with $j < k \leq |\hat{p}|$, $\hat{p}_j \leq \hat{p}_k$.

Definition 2.5 (Ideals and Ideal Bases)

I is an *ideal* (of $(P, <)$) iff
$I \neq \emptyset$ and
if $i \in I$ and $i \leq p$ then $p \in I$.

ideal(B) ("the *ideal generated* by B (in $(P, <)$)") =
$= \{p \mid \text{for some } b \in B,\ b \leq p\}$.

B is a *basis* of I iff $I = \text{ideal}(B)$.

I is a *finitely generated ideal* iff,
for some finite $B \neq \emptyset$, $I = \text{ideal}(B)$. □

ideal(p) is an abbreviation for ideal($\{p\}$).

Definition 2.6 (Minimal Elements)

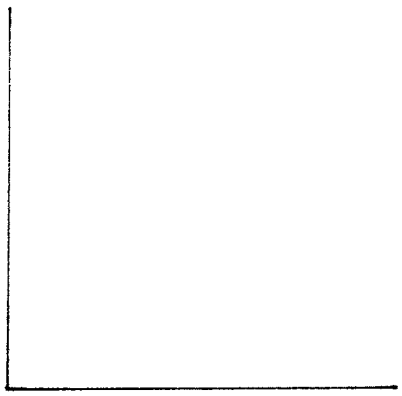
m is *minimal* in I iff
$m \in I$ and,
for no $i \in I$, $i < m$.

$\min(I) := \{m \mid m \text{ is minimal in } I\}$.

3 Graphical Representation

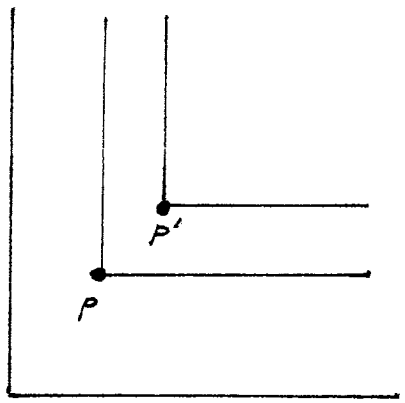
The intuition behind the proofs in the next sections can be illustrated by using the following graphical representation:

The area

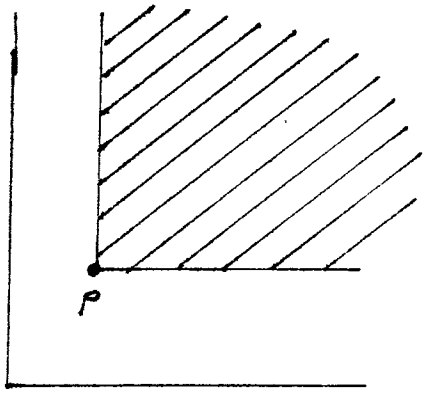


represents the set P.

Two elements p and p' for which $p < p'$ holds are represented by



The ideal(p) is represented by the shaded area



4 Finiteness Properties in Partial Orderings

Before we show the equivalence of certain finiteness properties we establish a lemma that describes the special role of minimal elements in finitely generated ideals. This lemma is interesting in its own right but also contains a crucial part of the equivalence proof.

In the proofs of this section we will, sometimes tacitly, use the elementary properties of ideals compiled in the appendix.

Lemma 4.1 (Minimal Element Bases)

If I is a finitely generated ideal then $\min(I)$ is a finite basis for I.

Proof: Let B be a finite basis for I. By the base property of minimal elements, $\min(I) \subseteq B$. Hence, $\min(I)$ is finite.

We now show that $\text{ideal}(\min(I)) = I$. Of course, $\min(I) \subseteq I$ and, hence, $\text{ideal}(\min(I)) \subseteq \text{ideal}(I) = I$. It remains to show that $I \subseteq \text{ideal}(\min(I))$. For this it suffices to show that $B \subseteq \text{ideal}(\min(I))$ because then

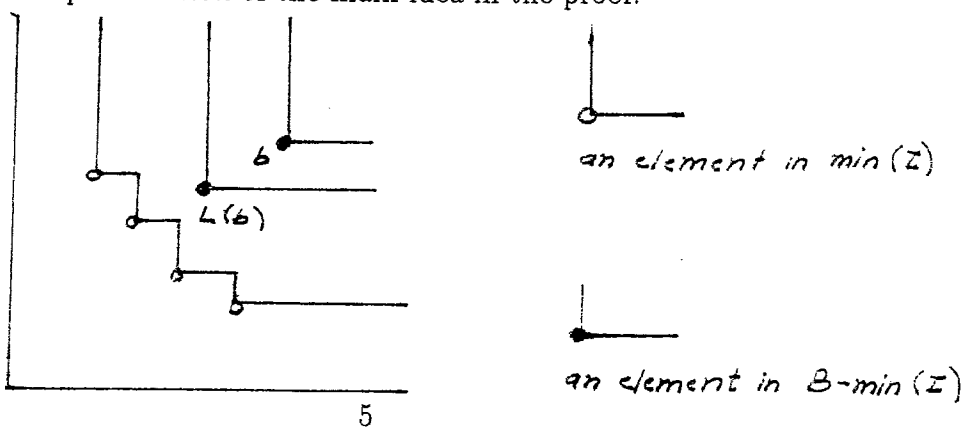
$$I = \text{ideal}(B) \subseteq \text{ideal}(\text{ideal}(\min(I))) = \text{ideal}(\min(I))$$

by ideal monotony and idempodency. For showing $B \subseteq \text{ideal}(\min(I))$ we observe that

for all $b \in B - \min(I)$
there exists an $\mathrm{L}(b) \in B$ such that $\mathrm{L}(b) < b$.

(If $b \in B - \min(I)$ then there exists an $i \in I$ such that $i < b$ because b is not minimal in I. There exists an $\mathrm{L}(b) \in B$ with $\mathrm{L}(b) < i$ because $i \in \text{ideal}(B)$.) Let now $b \in B$ and let j_0 be the smallest index such that $L^{(j_0)}(b) \in \min(I)$. Such a j_0 must exist because otherwise $\{L^{(j)}(b) \mid j \in \mathbf{N}\}$ would be an infinite subset of B. Then, $L^{(j_0)}(b) < b$ and $L^{(j_0)}(b) \in \min(I)$, i.e. $b \in \text{ideal}(\min(I))$.

Graphical Representation of the main idea in the proof:



Corollary 4.2 (Existence of Finite Subbases)

If $I = \text{ideal}(B)$ and I is finitely generated
then, for some finite $B_0 \subseteq B$, $I = \text{ideal}(B_0)$.

Proof: Take $B_0 := \min(I)$. By the base property of minimal elements and the preceding lemma on minimal element bases, B_0 has the required properties. □

We present an alternative proof for the corollary that does not involve the consideration of minimal elements. This proof is easier but less informative.

Alternative Proof: Let $I = \text{ideal}(B) = \text{ideal}(B')$ and B' finite. Since $B' \subseteq \text{ideal}(B)$, we have:

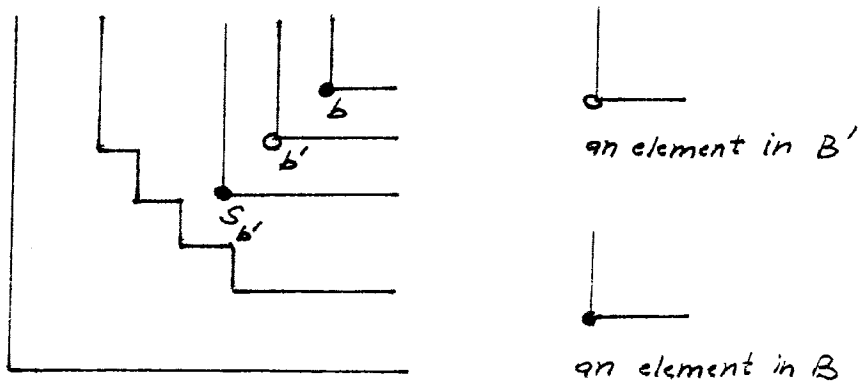
for all $b' \in B'$ there exists an $S_{b'} \in B$ such that $S_{b'} \leq b'$.

Now,

$$B_0 := \{S_{b'} \mid b' \in B'\}$$

is finite. Furthermore, $B_0 \subseteq B$. It remains to be shown that $\text{ideal}(B_0) = \text{ideal}(B)$. Of course, by ideal monotony, $\text{ideal}(B_0) \subseteq \text{ideal}(B)$. On the other hand, $\text{ideal}(B) \subseteq \text{ideal}(B_0)$: If $b \in \text{ideal}(B) = \text{ideal}(B')$ then there exists a $b' \in B'$ such that $b' \leq b$. Now, $S_{b'} \leq b'$ and, hence, $b_0 := S_{b'} \in B_0$ and $b_0 \leq b' \leq b$. This shows that $b \in \text{ideal}(B_0)$.

Graphical Representation of the main idea in the proof:



Theorem 4.3 (Equivalent Finiteness Properties)

Let $(P, <)$ be a partial ordering. The following conditions are equivalent:

Every ideal in $(P, <)$ is finitely generated.

Every non-ascending $(P, <)$-sequence is finite.

Every infinite P-sequence has an infinite ascending $(P, <)$-subsequence.

Every strictly descending $(P, <)$-sequence is finite and every set of mutually incomparable elements is finite.

Proof:

(Ideals finitely generated) $\Longrightarrow$ *(non-ascending sequences finite)*: Let $\hat{p}$ be an infinite non-ascending $(P, <)$-sequence. Let $\hat{P} := \{\hat{p}_j \mid j \in \mathbf{N}\}$ and assume that

$$I := \text{ ideal}(\hat{P})$$

is finitely generated. By the above corollary there exists a finite basis B_0 for I that satisfies $B_0 \subseteq \hat{P}$. The set $\{j \mid \hat{p}_j \in B_0\}$ is finite because B_0 is finite and all elements $\hat{p}_j$ are distinct. Hence, we can define

$$k := \max\{j \mid \hat{p}_j \in B_0\} + 1.$$

Now, $\hat{p}_k \in I = \text{ideal}(B_0)$. Hence there exists a $j < k$ such that $\hat{p}_j \leq \hat{p}_k$, which contradicts the assumption that $\hat{p}$ is non-ascending.

(Non-ascending sequences finite) $\Longrightarrow$ *(ideals finitely generated)*: Let I be an ideal without a finite basis. Then

for all finite sets $B \subseteq I$,
there exists an $S(B) \in I -$ ideal(B).

We construct an infinite non-ascending $(P, <)$-sequence $\hat{p}$. Since $I \neq \emptyset$ we can choose a

$$\hat{p}_1 \in I$$

and then we can inductively define,

$$\hat{p}_{k+1} := S(\{\hat{p}_j \mid j \leq k\}), \text{ for all } k \geq 1.$$

By construction it is clear that,

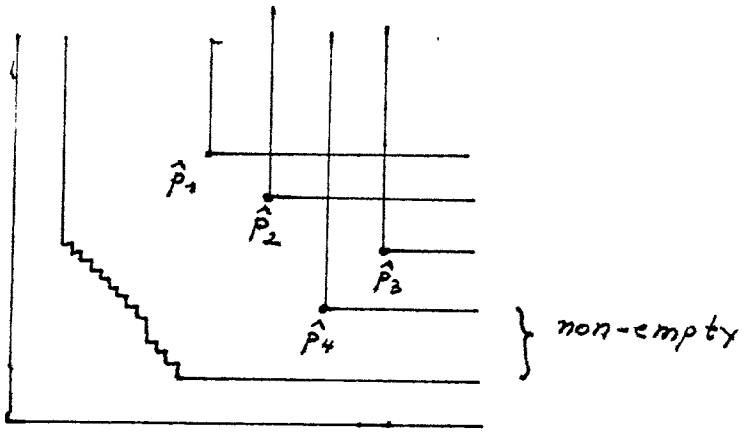
$$\hat{p}_k \in I - \text{ ideal}(\{\hat{p}_j \mid j < k\}), \text{ for all } k > 1.$$

Hence,

for all j, k,
if $j < k$ then $\hat{p}_j \not\leq \hat{p}_k$,

i.e. $\hat{p}$ is an infinite non-ascending $(P, <)$-sequence.

Graphical Representation of the main idea in the proof:



(Non-ascending sequences finite) $\Longrightarrow$ *(infinite ascending subsequences)*: See lecture notes.

(Infinite ascending subsequences) $\Longrightarrow$ *(non-ascending sequences finite)*: An infinite non-ascending sequence $\hat{p}$ would have an infinite ascending subsequence $\hat{p}_{i_1}, \hat{p}_{i_2}, \ldots$ with $i_1 < i_2 < \ldots$ and $\hat{p}_{i_1} \leq \hat{p}_{i_2} \leq \ldots$ Hence, $\hat{p}$ would not be non-ascending because, for example, $i_1 < i_2$ and $\hat{p}_{i_1} \leq \hat{p}_{i_2}$. This is a contradiction.

(Descending sequences finite) $\Longleftrightarrow$ *(ideals finitely generated)*: Not part of the exam.

5 Appendix: Some Elementary Properties of Ideals

Lemma 5.1 (Elementary Ideal Properties)

$B \subseteq \text{ideal}(B)$. (basis inclusion)

If $B \neq \emptyset$ then $\text{ideal}(B)$ is an ideal. (ideal generation)

If $B_1 \subseteq B_2$ then $\text{ideal}(B_1) \subseteq \text{ideal}(B_2)$. (ideal monotony)

If I is an ideal then $\text{ideal}(I) = I$. (ideal closure)

$\text{ideal}(B) = \text{ideal}(\text{ideal}(B))$. (ideal idempotency)

If $I = \text{ideal}(B)$ then $\min(I) \subseteq B$. (base property of minimal elements)

Proof:

(Basis inclusion): Let $b \in B$. Of course, $b \leq b$. Hence, $b \in \text{ideal}(B)$.

(Ideal generation): $\text{ideal}(B)$ is non-empty because B is non-empty and $B \subseteq \text{ideal}(B)$. Let now $i \in \text{ideal}(B)$ and $i \leq p$. Then $b \leq i$ for some $b \in B$. Hence, $b \leq i \leq p$ and therefore, by transitivity, $b \leq p$. This shows that $p \in \text{ideal}(B)$.

(Ideal monotony): Let $i \in \text{ideal}(B_1)$. Then there exists $b_1 \in B_1$ such that $b_1 \leq i$. Now b_1 is also in B_2. Hence, $i \in \text{ideal}(B_2)$.

(Ideal closure): By basis inclusion we know that $I \subseteq \text{ideal}(I)$. Let now $i' \in \text{ideal}(I)$. Then $i \leq i'$ for some $i \in I$. Hence, $i' \in I$ because I is an ideal.

(Ideal idempotency): $\text{ideal}(B)$ is an ideal. Hence, by ideal closure, $\text{ideal}(B) = \text{ideal}(\text{ideal}(B))$.

(Base property of minimal elements): Let $m \in \min(I)$. There exists a $b \in B$ such that $b \leq m$ because $m \in I = \text{ideal}(B)$. We can exclude $b < m$ because $b \in I$ by base inclusion and m is minimal in I. Hence $m = b \in B$.

Finite Ideal Bases, Descending Sequences and Mutually Incomparable Elements in Partial Orderings *

B.Buchberger
RISC-Linz

February 12, 1992

Abstract

We present a proof of the well-known fact that in partial orderings the following two conditions are equivalent:

- Every ideal has a finite basis.
- Every strictly descending sequence is finite and every set of mutually incomparable elements is finite.

*Solution to the exam for the course "Thinking, Speaking, Writing", 31 January 92

1 Notation and Definitions

In this paper we use the notation and definitions of [1]. In addition, we need the following two definitions.

Definition 1.1 (Strictly Descending Sequences)

$\hat{p}$ is a *strictly descending* $(P, <)$-sequence iff
$\hat{p}$ is a P-sequence and
for all $j < |\hat{p}|$, $\hat{p}_j > \hat{p}_{j+1}$.

Definition 1.2 (Sets of Incomparable Elements)

S is a *$(P, <)$-set of mutually incomparable elements* iff
for all $s_1, s_2 \in S$, $s_1 \not< s_2$.

2 Theorem and Proof

In [1], some finiteness properties in partial orderings have been proven equivalent. Here, we show an additional equivalence. In the proof we use some of the lemmata established in [1].

Theorem 2.1 *Let $(P, <)$ be a partial ordering. Then the following conditions are equivalent:*

Every ideal in $(P, <)$ is finitely generated.
Every strictly descending $(P, <)$-sequence is finite and
every $(P, <)$-set of mutually incomparable elements is finite.

Proof:

(Ideals finitely generated) $\Longrightarrow$ (strictly descending sequences finite) and (mutually incomparable sets finite):

Let $\hat{p}$ be a strictly descending sequence. Then $\hat{p}$ is non-ascending. Hence, $\hat{p}$ is finite by Theorem 4.3 in [1].

Let now S be a set of mutually incomparable elements and let

$$I := \text{ideal}(S).$$

By Lemma 4.1 of [1], $\min(I)$ is a finite basis of I. Now, every element $s \in S$ is minimal in I. (Assume, for some $i \in I$, $i < s$. Then, for some $s' \in S$, $s' \leq i$ because S is a basis for I. Hence, $s' < s$, a contradiction to the fact that S is

a set of mutually incomparable elements). Hence, $S \subseteq \min(I)$ and, therefore, S is finite.

(Strictly descending sequences finite) and (mutually incomparable sets finite) $\Longrightarrow$ *(ideals finitely generated)*:

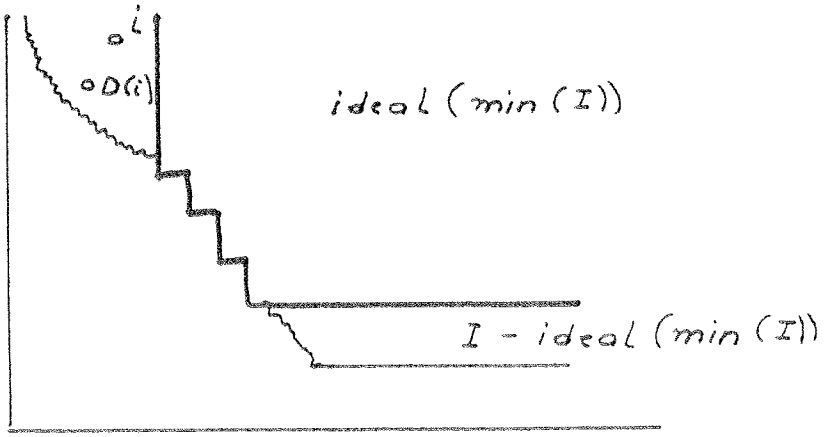
We assume that all sets of mutually incomparable elements are finite and consider an ideal I that is not finitely generated. We have to construct an infinite strictly descending sequence $\hat{p}$.

First, we observe that $\min(I)$ is a set of mutually incomparable elements. Hence, $\min(I)$ is finite by the assumption.

Now we prove that

for all $i \in I - \text{ideal}(\min(I))$
there exists a $D(i) \in I - \text{ideal}(\min(I))$ with $D(i) < i$.

[Graphically,



]

In fact, if $i \in I - \text{ideal}(\min(I))$, there exists a $D(i) \in I$ with $D(i) < i$ because otherwise i would be in $\min(I)$ and, hence, in $\text{ideal}(\min(I))$. Furthermore, $D(i) \notin \text{ideal}(\min(I))$ because otherwise $i > D(i) \geq i_0$ for some $i_0 \in \min(I)$, i.e. $i \in \text{ideal}(\min(I))$.

Now we define

$\hat{p}_1 :=$ an element in $I - \text{ideal}(\min(I))$
(such an element exists because I is not finitely generated) and
$\hat{p}_{k+1} := D(\hat{p}_k)$, for all $k \geq 1$.

Now it is easy to show by induction that $\hat{p}$ is an infinite strictly descending sequence.

References

[1] B. Buchberger: *Finiteness Properties in Partial Orderings*. Solution to the exam for the course TSW, 1991.

An Interpolation Method for Finding Boolean Expressions that Represent Boolean Functions *

B.Buchberger
RISC-Linz

December 30, 1991

Abstract

We describe and verify an interpolation method for finding boolean expressions that represent boolean functions. We present the method both in a variant that proceeds by recursion over the number of variables and in a distributive variant that handles all variables at once. The method is important, for example, as a basic step in the design of switching circuits.

*Exercise in the the course "Thinking, Speaking, Writing"

1 Introduction

"Switches" are devices that receive simple input, e. g. "T" ("true") and "F" ("false"), from a finite number of input lines and produce a simple output ("F" or "T") as a deterministic answer. The most common types of elementary switches are the "negation", the "conjunction", the "disjunction", and some variants thereof. More complex switches ("switching circuits") can be built from elementary switches by interconnecting their input and output lines in a way that does not lead to "cycles" in the circuits. For an introduction to switching circuits see [].

The *representation problem of switching circuit theory* is the problem of finding a switching circuit for *any* given "finite behavior", i. e. for any given function with finite domain and range. This is the most fundamental problem of switching circuit theory. In this paper, we will present a proof of the surprising fact that switching circuits built from conjunctions, disjunctions, and negations are powerful enough to represent any finite function. (In fact, disjunctions and negations alone would suffice. Also, some other combinations of elementary switches are "complete" in the sense that they can represent any finite function.)

It should be clear that the elements of a finite set can be "coded" by tuples of "F" and "T". Hence, arbitrary finite functions can be coded by "boolean" functions, i. e. functions whose input and output is taken from the set $\{F, T\}$. Furthermore, in a natural way, the structure of switching circuits can be described by the syntactical structure of "boolean expressions" (or "boolean terms"), see for example []. Therefore, summarizing, the representation problem of switching circuit theory can be formulated as the *boolean representation problem*, i. e. the problem of representing arbitrary boolean functions by boolean expressions.

In this paper we prove that the boolean representation problem has an algorithmic solution (see the Boolean Representation Theorem in Section 4, whose proof is constructive, i. e. exhibits an algorithm). Our presentation of the necessary terminology and the proof is so detailed that an implementation of the algorithm is immediately possible. In fact, we will provide two proofs of the representation theorem. Both proofs are based on the concept of "interpolation", i. e. a general approach by which function representations are derived from function values. However, the first proof proceeds by *recursion* over the number of variables while the second one handles all variables at once ("*distributive*" approach).

Before we formally specify the boolean representation problem and introduce terminology in Section 3, we give an easy example in Section 2 that illustrates the basic idea of the representation method (in the distributive va-

riant). Section 5 contains some formal details that are used in the proof of the Boolean Representation Theorem in Section 4.

2 An Example

We consider the ternary boolean function f defined by the following table:

t_1	t_2	t_3	$f(t_1, t_2, t_3)$
T	T	T	T
T	T	F	T
T	F	T	F
T	F	F	F
F	T	T	T
F	T	F	F
F	F	T	F
F	F	F	T

We want to construct a boolean expression e, i. e. an expression composed from variables and the symbols $\neg$ ("not" or "negation"), $\wedge$ ("and" or "conjuction"), and $\vee$ ("or" or "disjunction") with the property that e represents f. For this we first determine all input triples for which f has the value T, namely

$$(\mathrm{T},\mathrm{T},\mathrm{T}),(\mathrm{T},\mathrm{T},\mathrm{F}),(\mathrm{F},\mathrm{T},\mathrm{T}), \text{ and } (\mathrm{F},\mathrm{F},\mathrm{F}).$$

Now, a suitable e can be obtained as a disjunction of conjunctions, where each of the conjunctions characterizes one of the above triples:

$$\begin{aligned} e := {} & (v_1 \wedge v_2 \wedge v_3) \vee \\ & (v_1 \wedge v_2 \wedge \neg v_3) \vee \\ & (\neg v_1 \wedge v_2 \wedge v_3) \vee \\ & (\neg v_1 \wedge \neg v_2 \wedge \neg v_3). \end{aligned}$$

3 The Boolean Representation Problem

The following problem is called the *boolean representation problem*:

Given: f and n, where f is an n-ary boolean function.

Find: e, an n-ary boolean expression,
such that the n-ary boolean function represented by e
is equal to f.

In the sequel we define the notions occuring in this problem specification:

- the notion of "boolean function",
- the notion of "boolean expression",
- the notion of "the boolean function represented by a boolean expression".

We use the variables i, j, k, l, m, n as typed variables ranging over $\mathbb{N}$, the set of natural numbers ≥ 1. Furthermore, we use the convention that, if X, $\mathbb{X}$ etc. are sets, then x, t etc. are used as typed variables for elements in X, $\mathbb{X}$ etc. and $\bar{x}$, $\bar{t}$ are used as typed variables for tuples of elements in X, $\mathbb{X}$ etc. For understanding the notation in some of the subsequent formulae, it is important to note that we consider n-tuples as being built up recursively from pairs in the following way

$S^1 = S$, and

$S^{n+1} = \{(\bar{s}, s) \mid \bar{s} \in S^n, s \in S\}$, if $n \in \mathbb{N}$.

As usual, if f is a function, we sometimes write $f(s_1, s_2)$ instead of $f((s_1, s_2))$.

Let now T ("true") and F ("false") be two distinct fixed objects.

Definition 3.1 (Boolean Functions)

$\mathbb{T}$ ("the set of *truth values*") $:= \{\mathrm{T}, \mathrm{F}\}$.

$\mathbb{F}_n$ ("the set of *n-ary boolean functions*") $:= \{f \mid \mathbb{T}^n \to \mathbb{T}\}$.

$\mathbb{F} := \bigcup_n \mathbb{F}_n$. □

We will use t and f as typed variables ranging over $\mathbb{T}$ and $\mathbb{F}$, respectively.

Definition 3.2 (Elementary Boolean Functions)

$f_\neg$ ("the *negation function*") is the unary boolean function defined by:
$f_\neg(t) = \mathrm{T}$ iff $t = \mathrm{F}$.

$f_\wedge$ ("the *conjunction function*") is the binary boolean function defined by:
$f_\wedge(t_1, t_2) = \mathrm{T}$ iff ($t_1 = \mathrm{T}$ and $t_2 = \mathrm{T}$).

$f_\vee$ ("the *disjunction function*") is the binary boolean function defined by:
$f_\vee(t_1, t_2) = \mathrm{T}$ iff ($t_1 = \mathrm{T}$ or $t_2 = \mathrm{T}$). □

Let now $\mathbb{V}$ be a countable set ("the set of *boolean variables*") and let v be a bijective enumeration of $\mathbb{V}$, i. e. $\mathrm{v}\colon \mathbb{N} \xrightarrow{1-1} \mathbb{V}$ so that $\mathbb{V} = \{\mathrm{v}_1, \mathrm{v}_2, \ldots\}$. Let "$\neg$" (negation symbol), "$\wedge$" (conjunction symbol), and "$\vee$" (disjunction symbol) be three distinct objects that do not occur in $\mathbb{V}$.

Definition 3.3 (Boolean Expressions) The set $\mathbb{E}_n$ ("the set of *n-ary boolean expressions*") is the set of strings over the alphabet $\mathbb{V} \cup \{\neg, \wedge, \vee\}$ that is recursively defined as follows:

if $1 \leq i \leq n$ then $\mathrm{v}_i \in \mathbb{E}_n$,

if $e \in \mathbb{E}_n$ then $\neg e \in \mathbb{E}_n$,

if $e_1, e_2 \in \mathbb{E}_n$ then $\wedge e_1 e_2 \in \mathbb{E}_n$,

if $e_1, e_2 \in \mathbb{E}_n$ then $\vee e_1 e_2 \in \mathbb{E}_n$.

Furthermore,

$$\mathbb{E} := \bigcup \mathbb{E}_n. \qquad \square$$

Note that the particular syntax we use here for boolean expressions is "prefix" and does not need any parentheses. However, in the example above, we used "infix" notation and parentheses for the sake of easy readability. We will use v and e as typed variables ranging over $\mathbb{V}$ and $\mathbb{E}$, respectively.

Definition 3.4 (Boolean Representation) The function $[]_n$ on $\mathbb{E}_n$ ("the n-ary *boolean representation function*") is recursively defined as follows:

if $1 \leq i \leq n$ then $[\mathrm{v}_i]_n(\bar{t}) = \bar{t}_i$,

$[\neg e]_n(\bar{t}) = f_\neg([e]_n(\bar{t}))$,

$[\wedge e_1, e_2]_n(\bar{t}) = f_\wedge([e_1]_n(\bar{t}), [e_2]_n(\bar{t}))$,

$[\vee e_1, e_2]_n(\bar{t}) = f_\vee([e_1]_n(\bar{t}), [e_2]_n(\bar{t}))$.

(For $[e]_n$ read: "*the boolean function represented by e*".)

4 The Boolean Representation Theorem

Theorem 4.1 (The Boolean Representation Theorem)

For all n-ary boolean functions f
there exists an n-ary boolean expression e such that $[e]_n = f$.

Recursive Proof:

The theorem is true for $n = 1$: It can be easily verified that the four unary boolean functions can be represented by the boolean expressions v_1, $\neg\mathrm{v}_1$, $\wedge\mathrm{v}_1\neg\mathrm{v}_1$, and $\vee\mathrm{v}_1\neg\mathrm{v}_1$.

Let now n be fixed. As induction hypothesis, we assume that all n-ary boolean functions can be represented by boolean expressions. Let f be an $(n+1)$-ary boolean function.

It is easy to check that (for $t \in \mathbb{T}$ and $\bar{t} \in \mathbb{T}^n$)

$$f(\bar{t}, t) = f_\vee(f_\wedge(t, f(\bar{t}, \mathrm{T})), f_\wedge(f_\neg(t), f(\bar{t}, \mathrm{F}))). \qquad \text{(recursive interpolation)}$$

(Hint: consider the two cases $t = \mathrm{T}$ and $t = \mathrm{F}$.)

Now define

$$f_\mathbf{T}(\bar{t}) := f(\bar{t}, \mathrm{T}) \text{ and}$$

$$f_\mathbf{F}(\bar{t}) := f(\bar{t}, \mathrm{F}).$$

$f_\mathbf{T}$ and $f_\mathbf{F}$ are n-ary boolean functions and therefore, by induction hypothesis, there exist boolean expressions $e_\mathbf{T}$ and $e_\mathbf{F}$ in $\mathbb{E}_n$ such that

$$[e_\mathbf{T}]_n = f_\mathbf{T} \text{ and}$$

$$[e_\mathbf{F}]_n = f_\mathbf{F}.$$

Since, for arbitrary $e \in \Xi_n$, $[e]_{n+1}(\bar{t}, t) = [e]_n(\bar{t})$, this implies that

$$[e_\mathbf{T}]_{n+1}(\bar{t}, t) = f(\bar{t}, \mathrm{T}) \text{ and} \qquad \text{(shifting true)}$$

$$[e_\mathbf{F}]_{n+1}(\bar{t}, t) = f(\bar{t}, \mathrm{F}). \qquad \text{(shifting false)}$$

We finally define

$$e := \vee \wedge \mathrm{v}_1 e_\mathbf{T} \wedge \neg\mathrm{v}_1 e_\mathbf{F}$$

and verify

$$[e]_{n+1}(\bar{t}, t) = \qquad \text{(by the definition of boolean representation)}$$

$$= f_\vee(f_\wedge(t, [e_\mathbf{T}]_{n+1}(\bar{t}, t)), f_\wedge(f_\neg(t), [e_\mathbf{F}]_{n+1}(\bar{t}, t))) =$$

(by shifting true and false)

$$= f_\vee(f_\wedge(t, f(\bar{t}, \mathrm{T})), f_\wedge(f_\neg(t), f(\bar{t}, t))) =$$

(by recursive interpolation)

$$= f(\bar{t}, t).$$

Distributive Proof:

Let f be an n-ary boolean function. Define now

$e := \bigvee_{f(\bar{t})=\mathrm{T}} \mathrm{conjunction}(\bar{t})$, where (distributive interpolation)

$\mathrm{conjunction}(\bar{t}) = \bigwedge_{1 \le i \le n} \mathrm{literal}(i, \bar{t})$ and

$$\mathrm{literal}(i, \bar{t}) = \begin{cases} v_i & \text{if } \bar{t}_i = \mathrm{T}, \\ \neg v_i & \text{if } \bar{t}_i = \mathrm{F}. \end{cases}$$

Then $[e]_n = f$ as can easily be verified: First we note that

for all $1 \le i \le n$ we have $[\mathrm{literal}(i, \bar{t})]_n(\bar{t}) = \mathrm{T}$ and

if $\bar{t}' \neq \bar{t}$ then
for some $1 \le i \le n$ we have $[\mathrm{literal}(i, \bar{t}')]_n(\bar{t}) = \mathrm{F}$.

Hence, by Lemma 5.3 on boolean set operations,

$[\mathrm{conjunction}(\bar{t})]_n(\bar{t}) = \mathrm{T}$ and (equal truth tuples)

if $\bar{t}' \neq \bar{t}$ then $[\mathrm{conjunction}(\bar{t}')]_n(\bar{t}) = \mathrm{F}$. (unequal truth tuples)

Let now $\bar{t}$ be fixed. We want to show $[e]_n(\bar{t}) = f(\bar{t})$. For this we consider two cases:

Case $f(\bar{t}) = \mathrm{T}$: In this case there exists a $\bar{t}'$, namely $\bar{t}' := \bar{t}$, with $f(\bar{t}') = \mathrm{T}$ such that $[\mathrm{conjunction}(\bar{t}')]_n(\bar{t}) = \mathrm{T}$ (use (equal truth tuples)). Hence, by Lemma 5.3 on boolean set operations,

$$[e]_n(\bar{t}) = [\bigvee_{f(\bar{t}')=\mathrm{T}} \mathrm{conjunction}(\bar{t}')]_n(\bar{t}) = \mathrm{T} = f(\bar{t}).$$

Case $f(\bar{t}) = \mathrm{F}$: In this case there does not exist a $\bar{t}'$ such that $f(\bar{t}') = \mathrm{T}$ and $[\mathrm{conjunction}(\bar{t}')]_n(\bar{t}) = \mathrm{T}$ because otherwise by (unequal truth tuples) $\bar{t} = \bar{t}'$ and, hence, $f(\bar{t}) = f(\bar{t}') = \mathrm{T}$ in contradiction to the case assumption. Hence, by Lemma 5.3 on boolean set operations,

$$[e]_n(\bar{t}) = [\bigvee_{f(\bar{t}')=\mathrm{T}} \mathrm{conjunction}(\bar{t}')]_n(\bar{t}) = \mathrm{F} = f(\bar{t}).$$

5 A Technicality: Boolean Quantifiers

In the distributive proof of the Boolean Representation Theorem, we need the "*boolean quantifiers*" $\bigwedge$ and $\bigvee$ for working with boolean expressions. They are declared as follows: If $\tau(x)$ is a term and $\phi(x)$ is a formula with free variable x then

$$\bigvee_{\phi(x)} \tau(x) \text{ and } \bigwedge_{\phi(x)} \tau(x) \qquad \text{(boolean quantification)}$$

are terms that abbreviate the terms

$$\bigvee(\{\tau(x) \mid \phi(x)\}) \text{ and } \bigwedge(\{\tau(x) \mid \phi(x)\}), \qquad \text{(boolean set operations)}$$

respectively. In boolean quantification, the variable x becomes bounded. In fact, one would have to indicate the bounded variable explicitly. However, we assume that the bounded variable is always clear from the context.

In (boolean quantification) the symbols $\bigvee$ and $\bigwedge$ are quantifiers. However, in (boolean set operations), the same symbols denote functions:

Definition 5.1 (Boolean Set Operations) On the set of finite subsets of Ξ the functions $\bigvee$ ("*set disjunction*") and $\bigwedge$ ("*set conjunction*") are recursively defined as follows:

$$\bigvee(\emptyset) = \bigwedge \mathbf{v}_1 \neg \mathbf{v}_1,$$

$$\bigvee(\{e\}) = e,$$

if E is a finite subset of Ξ and $|E| > 1$
then $\bigvee(E) = \bigvee \text{choice}(E) \bigvee(E - \{\text{choice}(E)\})$.

$$\bigwedge(\emptyset) = \bigvee \mathbf{v}_1 \neg \mathbf{v}_1,$$

$$\bigwedge(\{e\}) = e,$$

if E is a finite subset of Ξ and $|E| > 1$
then $\bigwedge(E) = \bigwedge \text{choice}(E) \bigwedge(E - \{\text{choice}(E)\})$. —

In this definition we need some function "choice" on the set of finite subsets of Ξ that satisfies the following axiom:

Axiom 5.2 (Choice Function)

If $E \neq \emptyset$ then $\text{choice}(E) \in E$. □

It is easy to see that the following lemma holds:

Lemma 5.3 (Boolean Set Operations)

Let $E \subseteq \Xi_n$ be finite. Then

$[\bigvee(E)]_n(\bar{t}) = \mathrm{T}$ iff, for some $e \in \Xi_n$, $[e]_n(\bar{t}) = \mathrm{T}$ and

$[\bigwedge(E)]_n(\bar{t}) = \mathrm{T}$ iff, for all $e \in \Xi_n$, $[e]_n(\bar{t}) = \mathrm{T}$.

6 Conclusion

We have given two constructive proofs (a recursive and a distributive one) of the Boolean Representation Theorem that show that arbitrary boolean functions can be represented by boolean expressions. The two proofs are both based on the general idea of "interpolation". In fact, by expanding the recursive proof for the case of 2, 3, ... variables one immediately sees that the resulting booelan expression, after some easy simplification, is exactly the expression described in the distributive proof.

Also it should be noted that the formulae for the desired boolean expressions in the recursive and distributive proof could be used immediately as an algorithm, for example in MATHEMATICA, for obtaining the boolean expressions.

The booelan representation problem treated in this paper is intimately connected with two other basic problems in switching circuit theory:

- the *assignment problem*, i. e. the problem of coding elements in a finite set in such a way that subsequent representation of a given function on the finite set by a switching circuit (boolean expression) becomes as economic as possible,
- the *optimization problem*, i. e. the problem of finding the optimal boolean expression for a given boolean function.

These two problems are much more difficult than the representation problem and are beyond the scope of this paper.

König's Lemma: Finitely Branching Trees Without Infinite Paths are Finite *

B.Buchberger
RISC-Linz

November 29, 1991

Abstract

We present a proof of König's lemma that guarantees that finitely branching trees without infinite paths are finite. The lemma is useful in various branches of computer science in which trees are models for derivations, computations, reductions, simplifications etc., for example, in automated theorem proving.

*Exercise in the the course "Thinking, Speaking, Writing"

1 The Role of König's Lemma

König's lemma is important in various areas of computer science. For example, in automated theorem proving, the proof of the famous Herbrand theorem is based on the fact that certain "semantical trees" whose nodes are labeled by clauses are finitely branching and do not possess infinite paths and, hence, by König's lemma must be finite. Herbrand's theorem enables to reduce validity in predicate logic to an iterative test of validity in propositional logic and, hence, is crucial for the automation of predicate logic.

We will first give the precise definition of the notions involved in the formulation of König's lemma (Section 2) and then state and prove the lemma (Section 3).

2 Terminology

We use the convention that, if G, T etc. are sets, then g, t etc. are used as typed variables for elements in G, T etc. Furthermore, j, k, l, m, n are typed variables ranging over $\mathbf{N}$, the set of natural numbers ≥ 1. We assume that the notion of a (finite or infinite) G-sequence $\hat{g}$ (a sequence of elements in G) and the notion of the length $|\hat{g}|$ of such a sequence is known.

Definition 2.1 (Directed Graphs)

$(G, \rightarrow)$ is a *directed graph* iff $G \neq \emptyset$ and $\rightarrow \subseteq G \times G$.

Let now $(G, \rightarrow)$ be a directed graph.

Definition 2.2 (Auxiliary Notions for Directed Graphs)

$\hat{g}$ is a *path* iff
$\hat{g}$ is a G-sequence and
for all $i < |\hat{g}|, \hat{g}_i \rightarrow \hat{g}_{i+1}$.

$g \rightarrow^* g'$ iff there exists a finite path $\hat{g}$ such that
$\hat{g}_1 = g$ and $\hat{g}_n = g'$, where $n = |\hat{g}|$.

S(g) ("the set of successors of g") $:= \{g' \mid g \rightarrow^* g'\}$ □

If $(G, \rightarrow)$ is a directed graph then the elements of G are called the "nodes" of the graph and a pair (g, g') such that $g \rightarrow g'$ is also called an "edge" of the graph. If $g \rightarrow g'$ then we also say that g is an "immediate predecessor" of g' and that g' is an "immediate successor" of g. If $g \rightarrow^* g'$ then we say that g is a "predecessor" of g' and g' is a "successor" of g.

Definition 2.3 (Trees)

A directed graph $(T, \rightarrow, \dot{t})$ is a (directed) *tree* iff
$\dot{t} \in T$ and $\dot{t}$ has no immediate predecessor,
each $t \neq \dot{t}$ has exactly one immediate predecessor and
$S(\dot{t}) = T$.

A tree $(T, \rightarrow, \dot{t})$ is called "finitely branching" iff
each $t \in T$ has only finitely many immediate successors. □

If $(T, \rightarrow, \dot{t})$ is a tree then $\dot{t}$ is also called the "root" of the tree.

3 König's Lemma

Lemma 3.1 (König's Lemma)

If $(T, \rightarrow, \dot{t})$ is a finitely branching tree without infinite paths
then T is finite.

Proof: We assume that

$(T, \rightarrow, \dot{t})$ is a finitely branching tree and
T is infinite.

We have to construct an infinite path.
It is clear that for all $t \in T$

$\mathrm{S}(t) = \{t\} \cup \bigcup_{t'} \text{ with } {}_{t \rightarrow t'} \mathrm{S}(t')$.

From this we see that, since $(T, \rightarrow, \dot{t})$ is *finitely* branching,

for all $t \in T$
there exists a $C_t \in T$ such that
if $\mathrm{S}(t)$ is infinite then $t \rightarrow C_t$ and $\mathrm{S}(C_t)$ is infinite.

Now we can construct the following infinite sequence $\hat{t}$:

$\hat{t}_1 := \dot{t}$,
$\hat{t}_{i+1} := C_{\hat{t}_i}$.

We can show by induction that

for all $i \in \mathbf{N}$, $\hat{t}_i \rightarrow \hat{t}_{i+1}$ and $\mathrm{S}(\hat{t}_{i+1})$ is infinite.

(For $i = 1$ note that $T = \mathrm{S}(\hat{t}_1)$, i. e. $\mathrm{S}(\hat{t}_1)$ is infinite.) Hence, $\hat{t}$ is an infinite path. □

We can summarize the proof in the following graphics: