Constructing Truth-Tables in Propositional Multi-Valued Logics with DERIVE 4[*]

Eugenio Roanes Lozano
Dept. Algebra
Facultad de Educación
Universidad Complutense de Madrid
e-mail: eroanes@eucmos.sim.ucm.es

Abstract

A classical approach to multi-valued Logics, based on the use of truth-tables, can be easily implemented in a CAS (DERIVE 4 in this case). Truth-tables are very laborious but many intuitive ideas come from working with them. We presented at the 1st International Derive Conference (Plymouth, 1994) an approach to the Boolean case in DERIVE 2, that was published in The International DERIVE Journal [3]. But the next DERIVE 3 included a built-in function for the same purpose.

An obvious extension is to construct truth-tables in multi-valued logics. This programme deals with Kleene's style (min/max) p-valued Logic with modal connectives (for any given prime number p), and it can:

i) Build the truth-table corresponding to two given propositions.

ii) Check if a given proposition is a tautology.

iii) Check if the second of two given propositions is a tautological consequence of the first one.

This implementation will be included within DERIVE 5. Some exercises and some notes about our experience in the classroom are also included.

[*]Partially supported by the DGES research project PB96-0098-C04-03 (Spain).

Introduction

We have developed models (implemented in Maple and CoCoA) to study inference in propositional algebras and consistency of Knowledge Based Systems (KBSs) based on Boolean and multi-valued modal Logics [6, 7, 8, 10] that follow the line opened by Kapur and Narendran, Hsiang, Chazarain et al. [5, 4, 1]. These treatments are straightforward and very powerful, but involve the calculation of Gröbner Bases of polynomial ideals over fields of finite characteristic (what is not intuitive!) [2, 16]. A similar approach has been used to develop a topology-independent railway interlocking system [11].

So we think that producing a specific purpose little package that could construct truth-tables and check tautologies and tautological consequences using only truth-tables-derived manipulations could be useful in teaching. An implementation in Maple is available [9] and an implementation in Macsyma is included within the latest versions of this CAS. The implementation presented here is written in DERIVE 4. A classic language (Pascal, Logo, C,...) [14] could also be used, but using a CAS makes it more simple and straightforward.

1 Classic Propositional Bivalued Logic

The classic propositional bivalued Logic is $(\mathcal{C}, \vee, \wedge, \neg)$, where

i) $\mathcal{C}$ is the set of propositions

ii) $\vee, \wedge$ are the (basic) logical binary connectives "and" and "or"

iii) $\neg$ is the logical unary connective "negation".

In a constructive way, if the propositional variables are $X_1, X_2, ..., X_m$ then $\mathcal{C}$ is the set of well-constructed formulae using $\vee, \wedge, \neg$ and $X_1, X_2, ..., X_m$.

Although connectives are used to build $\mathcal{C}$, they are usually identified with functions $\mathcal{C} \times \mathcal{C} \to \mathcal{C}$ (if binary) or $\mathcal{C} \to \mathcal{C}$ (if unary).

1.1 Defining the Binary Connectives

Let P, Q be any propositions. Each logical binary connective can be given as a truth-table (i.e., as a mapping $\{0,1\} \times \{0,1\} \to \{0,1\}$)

P	Q	$P \vee Q$	$P \wedge Q$	$P \to Q$	$P \leftrightarrow Q$
0	0	0	0	1	1
0	1	1	0	1	0
1	0	1	0	0	0
1	1	1	1	1	1

(where 0 represents *False* and 1 represents *True*).

The function can also be given as a (discrete) set of points. For $P \vee Q$: $\{(0,0,0), (0,1,1), (1,0,1), (1,1,1)\}$.

And the function can also be given directly in a functional way: P and Q can be valued in $\{0,1\}$ and

$$P \vee Q = max(P, Q)$$

$$P \wedge Q = min(P, Q)$$

or in polynomial form:

$$P \vee Q = P + Q - P \cdot Q$$

$$P \wedge Q = P \cdot Q$$

1.2 Defining the Unary Connectives

Let P be any proposition. $\neg P$ can be given as a truth-table

P	$\neg P$
1	0
0	1

$\neg P$ can also be given as the (discrete) set of points

$$\{(1,0), (0,1)\} \ .$$

And $\neg P$ can also be given in a functional or polynomial way: P can be valued in $\{0,1\}$ and

$$\neg P = 1 - P \ .$$

Let us observe that now that $\neg$ has been introduced, $\rightarrow$ and $\leftrightarrow$ can also be defined as follows (Kleene's Logic):

$$P \rightarrow Q \ \text{ iff } \ \neg P \vee Q$$

$$P \leftrightarrow Q \ \text{ iff } \ (P \rightarrow Q) \wedge (Q \rightarrow P)$$

1.3 A Formal Definition

"To give values to the propositional variables that appear in a proposition" is a bit vague. Moreover, the introduction above used an abuse in the notation as connectives were applied both to propositions and truth-values.

They can be defined formally distinguishing between a function $\tilde{F}$ applied to numbers and the logical connective F and using valuations for propositional variables that are extended to valuations of propositions. We shall not give details here (see [9] for instance).

1.4 To Be a Tautological Consequence and To Be a Tautology in Classic Bivalued Logic

1.4.1 Definition.- *P is a tautological consequence of Q $(P \models Q)$* iff *whenever P is True, then Q is also True.*

1.4.2 Example.- *Prove that $P \wedge Q \models P \vee Q$*

P	Q	$P \wedge Q$	$P \vee Q$
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	1

1.4.3 Definition.- *P is a tautology $(\models P)$* iff *P is always True.*

1.4.4 Example.- *Prove that $\models ((P \wedge Q) \rightarrow (P \vee Q))$*

P	Q	$P \wedge Q$	$\neg(P \wedge Q)$	$P \vee Q$	$(P \wedge Q) \rightarrow (P \vee Q)$
0	0	0	1	0	1
0	1	0	1	1	1
1	0	0	1	1	1
1	1	1	0	1	1

1.4.5 Theorem.- *It can be proven that, in classic bivalued Logic, for any propositions P and Q,*

$$\models (P \to Q) \text{ iff } P \models Q \ .$$

2 Three-Valued Logics

An introduction to multivalued Logics can be found in [15].

Observe that DERIVE's `IF` uses a three-valued Logic, as its fourth (optional) argument is what to do in case it can not be evaluated whether the condition is true or false.

2.1 Kleene's Three-Valued Logic

In this Logic there are three truth-values: "True", "False" and "Undecided" that we shall represent by $1, 0$ and $\frac{1}{2}$ respectively (what has the advantage that the greater the value the greater the certainty, what is very intuitive).

There are now two more unary connectives: "necessary" ($\Box$) and "possible" ($\Diamond$).

P	$\neg P$	$\Box P$	$\Diamond P$
0	1	0	0
$\frac{1}{2}$	$\frac{1}{2}$	0	1
1	0	1	1

P	Q	$P \vee Q$	$P \wedge Q$	$P \to Q$	$P \leftrightarrow Q$
0	0	0	0	1	1
0	$\frac{1}{2}$	$\frac{1}{2}$	0	1	$\frac{1}{2}$
0	1	1	0	1	0
$\frac{1}{2}$	0	$\frac{1}{2}$	0	$\frac{1}{2}$	$\frac{1}{2}$
$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$
$\frac{1}{2}$	1	1	$\frac{1}{2}$	1	$\frac{1}{2}$
1	0	1	0	0	0
1	$\frac{1}{2}$	1	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$
1	1	1	1	1	1

Observe that the polynomial forms change depending on the number of truth-values of the Logic.

"Implies" can also be defined as in classic bivalued Logic

$$(P \to Q) \text{ iff } (\neg P \vee Q) \ .$$

2.2 Lukasiewicz's Three-Valued Logic

Let us represent "True" by 1, "False" by 0 and "Indeterminate" by $\frac{1}{2}$.

Lukasiewicz's connectives and Kleene's differs only in the conditional and biconditional (check the fith row)

P	Q	$P \to_{Lu} Q$	$P \leftrightarrow_{Lu} Q$
0	0	1	1
0	$\frac{1}{2}$	1	$\frac{1}{2}$
0	1	1	0
$\frac{1}{2}$	0	$\frac{1}{2}$	$\frac{1}{2}$
$\frac{1}{2}$	$\frac{1}{2}$	1	1
$\frac{1}{2}$	1	1	$\frac{1}{2}$
1	0	0	0
1	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$
1	1	1	1

A polynomial form of Lukasiewicz's $\to_{Lu}$ is

$$P \to_{Lu} Q = 4P^2Q^2 - 4P^2Q - 4PQ^2 + 5PQ - P + 1$$

and it is clear that

$$P \leftrightarrow_{Lu} Q = (P \to_{Lu} Q) \wedge (Q \to_{Lu} P) \ .$$

An example of the use of this Logic is [12], where an algebraic model for a certain set of Medical Appropriateness Criteria (including uncertainty) is constructed.

3 Multi-Valued Propositional Logics

3.1 Introducing Multi-Valued Propositional Logics

Let p be a prime number. A p-valued propositional Logic is $(\mathcal{C}, F_1, ..., F_n)$, where

i) $\mathcal{C}$ is the set of propositions

ii) $F_1, ..., F_n$ are the logical connectives (unary or binary).

If the propositional variables are $X_1, X_2, ..., X_m$, then $\mathcal{C}$ is the set of well-constructed formulas using $F_1, ..., F_n$ and $X_1, X_2, ..., X_m$.

3.2 Defining the Connectives

We shall consider the truth-values to be $\{0, \frac{1}{p-1}, \frac{2}{p-1}, ..., \frac{p-2}{p-1}, 1\}$ for 0 representing False, 1 representing True and intermediate values representing intermediate degrees of certainty.

The logical connectives can be then defined

$$P \vee Q = max(P, Q)$$

$$P \wedge Q = min(P, Q)$$

$$\neg P = 1 - P \ .$$

($\neg$ tries to translate the idea of reversing the “probability” of something to happen).

The two modal connectives, necessary ($\Box$) and possible ($\Diamond$) are valued, respectively, 0 unless the proposition is True and 1 unless the proposition is false. Therefore, they can be expressed as follows

$$\Box P = floor(P)$$

$$\Diamond P = 1 - floor(1 - P) \ .$$

It is clear that this model is ready to be implemented very easily.

3.3 To be a Tautological Consequence and To Be a Tautology in Multi-Valued Logics

3.3.1 Theorem.- *In the general case of p-valued Logics ($p > 2$) it can only be proven that, for any propositions P and Q,*

$$\models (P \to Q) \quad \Rightarrow \quad P \models Q \ .$$

Adding certain conditions it can be proven that,

$$\models (\Box P \to Q) \quad \Leftrightarrow \quad P \models Q$$

(see [10] and the examples in §5).

Therefore, the behaviour is no longer intuitive. We think it would be a great help to have available a "Logic calculator" to explore these Logics.

4 Implementation in DERIVE 4

Connectives are defined below as DERIVE functions. They begin with an "M" (standing for "multivalued"), not to interfere with the built-in boolean connectives.

```
MNEG(a_) := 1-a_
MPOS(a_) := 1-FLOOR(1-a_)
MNEC(a_) := FLOOR(a_)
MOR(a_,b_) := MAX(a_,b_)
MAND(a_,b_) := MIN(a_,b_)
```

Conditional and biconditional (Kleene-style) can be defined using the previous connectives:

```
MIMP(a_,b_) := MOR( MNEG(a_) , b_ )
MIFF(a_,b_) := MAND( MIMP(a_,b_) , MIMP(b_,a_) )
```

Tautology and contradiction are respectively denoted `t` and `c` and are respectively assigned to the constants 1 and 0:

```
t:=1
c:=0
```

4.1 Constructing Truth-Tables

The number of truth-values is assigned to the global variable `w`, e.g.:

```
w:=3
```

A truth-table corresponding to propositions including m propositional variables in a w-valued Logic will have w^m rows.

The elements in the truth-tables corresponding to the propositional variables can be organized as follows ($w = 3$, $m = 2$):

$$\begin{vmatrix} 0 & 0 & ... \\ \frac{1}{2} & 0 & ... \\ 1 & 0 & ... \\ 0 & \frac{1}{2} & ... \\ \frac{1}{2} & \frac{1}{2} & ... \\ 1 & \frac{1}{2} & ... \\ 0 & 1 & ... \\ \frac{1}{2} & 1 & ... \\ 1 & 1 & ... \end{vmatrix}$$

Let us suppose that the propositional variables are the first m in `vp:=[p, q,r,s,u,v]`. In the general case of a truth-table corresponding to propositions including m propositional variables in a w-valued Logic will have w^m rows. The values given to the first propositional variables can be loops of:

$$0, \tfrac{1}{w}, \tfrac{2}{w}, ..., \tfrac{w-1}{w}, 1$$

those given to the second propositional variables can be loops of:

$$0, 0, \overbrace{\cdots}^{w}, 0, \tfrac{1}{w}, \tfrac{1}{w}, \overbrace{\cdots}^{w}, \tfrac{1}{w}, \tfrac{2}{w}, \tfrac{2}{w}, \overbrace{\cdots}^{w}, \tfrac{2}{w}, ...,$$
$$\tfrac{w-1}{w}, \tfrac{w-1}{w}, \overbrace{\cdots}^{w}, \tfrac{w-1}{w}, 1, 1, \overbrace{\cdots}^{w}, 1$$

those given to the third propositional variables can be loops of:

$$0, 0, \overbrace{\cdots}^{w^2}, 0, \tfrac{1}{w}, \tfrac{1}{w}, \overbrace{\cdots}^{w^2}, \tfrac{1}{w}, \tfrac{2}{w}, \tfrac{2}{w}, \overbrace{\cdots}^{w^2}, \tfrac{2}{w}, ...,$$
$$\tfrac{w-1}{w}, \tfrac{w-1}{w}, \overbrace{\cdots}^{w^2}, \tfrac{w-1}{w}, 1, 1, \overbrace{\cdots}^{w^2}, 1$$

...

If `j_` is the number of the row, this can be obtained in DERIVE the following way:

```
p:=(1/(w-1))*MOD(FLOOR(j_/w^0),w)
q:=(1/(w-1))*MOD(FLOOR(j_/w^1),w)
```

```
r:=(1/(w-1))*MOD(FLOOR(j_/w^2),w)
s:=(1/(w-1))*MOD(FLOOR(j_/w^3),w)
u:=(1/(w-1))*MOD(FLOOR(j_/w^4),w)
v:=(1/(w-1))*MOD(FLOOR(j_/w^5),w)
...
```

Then the main procedure `TT` constructs the truth-table:

```
TT(m_,a_,b_):=
VECTOR( APPEND(VECTOR(vp SUB i_,i_,1,m_),[a_,b_]), j_,0,w^m_-1)
```

which entries are:

i) `m_` is the number of different propositional variables that appear in `a_` together with `b_` (that must be the first `m_` in `vp`).

ii) `a_` and `b_` are the formulae which truth-table has to be calculated.

Observe that the

```
APPEND(VECTOR(vp SUB i_,i_,1,m_),[a_,b_])
```

just constructs the (row) vector of the `m_` first propositional variables in `vp` followed by `a_` and `b_`. Then a matrix is constructed by the "`VECTOR`" outside by stacking this row vectors upon each others when giving to `j` the values 0 to $w^m - 1$. Observe that the elements in `vp` are functions in the only variable `j` and `a_` and `b_` are functions of the variables in `vp`, so this is a numerical matrix.

4.1.1 Example.-

```
w:=3
TT( 3, MAND(P,MAND(Q,R)), MAND(MAND(P,Q),R) )=
```

$$\begin{vmatrix} 0 & 0 & 0 & 0 & 0 \\ \frac{1}{2} & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 \\ 1 & \frac{1}{2} & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ \frac{1}{2} & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & 0 & 0 \\ \frac{1}{2} & 0 & \frac{1}{2} & 0 & 0 \\ 1 & 0 & \frac{1}{2} & 0 & 0 \\ 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \\ 1 & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \\ \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \end{vmatrix}$$

4.2 Checking Tautologies

Instead of constructing the truth-table, `ISTAUT` calculates the product of all the truth-values of the column corresponding to `a_` and checks if it is 1 or not, answering 1 (YES) or 0 (NO):

```
ISTAUT(m_,a_):= if( PRODUCT(a_,j_,0,w^m_-1) = 1 , 1 , 0 )
```

4.2.1 Example.-

```
w:=5
ISTAUT(1, MIFF( MNEG(MPOS(Q)) , MNEC(MNEG(Q)) ) =
1
```

4.3 Checking Tautological Consequences

`ISCONSTAUT` checks weather `b_` is a tautological consequence of `a_` or not. It is similar to `ISTAUT` but each new truth-value of `b_` is multiplied iff the truth-value corresponding to the first one is 1.

```
ISCONSTAUT(m_,a_,b_):=
if( PRODUCT(IF(a_=1,b_,1),j_,0,w^m_-1) = 1 , 1 , 0 )
```

4.3.1 Example.-

```
w:=3
ISCONSTAUT(2, MAND(P,Q) , MOR(P,Q) ) =
1
```

5 Using the Implementation

A few exercises that show some of the possibilities of this implementation (they try to help students in understanding the behaviour of these Logics):

1) For classic bivalued Logic: Construct the correspondent truth tables and check weather classic bivalued Logic is a lattice or not. Remember that the properties that have to be tested are
   - Associative of $\vee$ and $\wedge$.
   - Commutative of $\vee$ and $\wedge$.
   - Cancelative of $\vee$ w.r.t. $\wedge$ and of $\wedge$ w.r.t. $\vee$

   (as a consequence idempotency of $\vee$ and $\wedge$ holds).

2) Repeat the previous exercise for Kleene's three-valued Logic.

3) Is classic bivalued Logic a Boolean algebra? (a Boolean-algebra is a distributive and complementary lattice). Hint: the complement of a proposition is its negation.

4) As a consequence, the De Morgan laws hold. Check it.

5) Repeat exercise 3) for Kleene's three-valued Logic. Try also with `NEG(POS )` instead of `NEG` as complement. Try to find a complement.

6) Check if in classic bivalued Logic $(P \wedge Q) \models (P \vee Q)$ and $\models (P \wedge Q) \to (P \vee Q)$.

7) Repeat the previous exercise for Kleene's three-valued Logic. Test also if $P \models P$ and $\models P \to P$. Try to explain what happens.

8) Consider $\models \Box(P \wedge Q) \to (P \vee Q)$ instead.

9) Repeat exercises 2), 5), 7) and 8) for Kleene's style min/max five-valued Logic to confirm that the same happens.

6 Educational Experience. Conclusions

I teach both at a Teacher-Training Center (School of Education) and the School of Mathematics of the "Universidad Complutense de Madrid". Since 1987 we have developed two builders of truth-tables for classic bivalued Logic in Logo [14] and Derive [3] and two for multivalued Logic with modal connectives in Maple and Macsyma.

They first two were used in the classroom with undergraduates studying the subjects "Mathematics III" (where there was a lesson devoted to Lattices and Boolean Algebras) and "Computers and Mathematics" (that introduces programming and some educational programmes to students). The attitude of the students was very positive, but according to the subject two attitudes were usual. For the first group, the main attraction was the novelty of working with a computer and the joy of obtaining in a fast and automatic way the result. The other group was interested in knowing "how" the implementation was done and to compare it with their own ideas.

The generalization to multi-valued Logics was proposed by professor Luis M. Laita (of the School of Computer Science of the "Universidad Politécnica de Madrid", and leader of the research group of the author) to be used with his students of the subject "Logic".

We always use a white-box/black-box approach:

1) Theory and hand-made examples are presented first.

2) Work is done with the computer afterwards, using it as a tool that allows to increase the number and complexity of the examples solved.

References

[1] **J. Chazarain, A. Riscos, J.A. Alonso, E. Briales**: *Multivalued Logic and Gröbner Bases with Applications to Modal Logic*. Journal of Symbolic Computation 11, 1991 (pages 181-194).

[2] **D. Cox, J. Little, D. O'Shea**: *Ideals, Varieties, and Algorithms.* Springer-Verlag, 1992.

[3] **M. Garbayo, E. Roanes M., E. Roanes L.**: *Automating Logic with Derive* The International Derive Journal, Vol. 1, No. 3, 1994 (pages 73-80).

[4] **J. Hsiang**: *Refutational Theorem Proving using Term-Rewriting Systems.* Artificial Intelligence, vol. 25, 1985 (pages 255-300).

[5] **D. Kapur, P. Narendran**: *An Equational Approach to Theorem Proving in First-Order Predicate Calculus.* 84CRD296 General Electric Corporate Research and Development Report, Schenectady, NY, March 1984, rev. Dec. 1984. Also in: *Proceedings of the 9th International Joint Conference on Artificial Intelligence (IJCAI-85)*, vol. 2 (pages 1146-1153).

[6] **L. M. Laita, L. de Ledesma, E. Roanes L., E . Roanes M.**: *An Interpretation of the Propositional Boolean Algebra as a k-algebra. Effective Calculus.* In: **J. Campbell, J. Calmet** (editors): *Proceedings of the Second International Workshop/Conference on Artificial Intelligence and Symbolic Mathematical Computing (AISMC-2).* LNCS 958. Springer-Verlag, 1995 (pages 255-263).

[7] **E. Roanes L., L. M. Laita, E. Roanes M.**: *Maple V in A.I.: The Boolean Algebra Associated to a KBS.* CAN Nieuwsbrief num. 14, April 1995 (pages 65-70).

[8] **E. Roanes L., L. M. Laita, E . Roanes M.**: *An Inference Engine for Propositional Two-valued Logic Based on the Radical Membership Problem.* In: **J. Campbell, J. Calmet, J. Pfalzgraf** (editors): *Proceedings of the Third International Workshop/Conference on Artificial Intelligence and Symbolic Mathematical Computing (AISMC-3).* LNCS 1138. Springer-Verlag, 1996 (pages 71-86).

[9] **E. Roanes L.**: *Introducing Propositional Multi-Valued Logics with the Help of a CAS.* Proceedings of the International Society for Analysis, Applications and Computing (ISAAC) Conference 1997. Newark, June 1997 (to appear).

[10] **E. Roanes L., L.M. Laita, E. Roanes M.**: *A Polynomial Model for Multi-Valued Logics with a Touch of Algebraic Geometry and Computer Algebra.* In: E. Roanes L., Stanly Steinberg, Hoon Hong (editors): Special Volume of Computers and Mathematics and Simulation "Non Standard Applications of Computer Algebra", vol. 45 (1) 1998 (pages 83-99).

[11] **E. Roanes-L., L.M. Laita, E. Roanes-M.**: *An Application of an AI Methodology to Railway Interlocking Systems using Computer Algebra.* In: **A. Pasqual del Pobil, J. Mira, M. Ali** (editors): *Tasks and Methods in Applied Artificial Intelligence, Procs. IEA-98-AIE* (Vol. II). LNAI 1416. Springer, 1998 (pages 687-696).

[12] **L.M. Laita, E. Roanes-L., V. Maojo**: (1998) *Inference and Verification in Medical Appropriateness Criteria.* In: **J. Calmet, J. Plaza** (editors): *Artificial Intelligence and Symbolic Computation, Procs. AISC'98.* LNAI 1476. Springer-Verlag, 1998 (pages 183-194).

[13] **A. Rich, J. Rich, D. Stoutemyer**: *DERIVE User Manual.* Soft Warehouse, 1992.

[14] **E. Roanes M., E. Roanes L.**: *MACO. Matemática con Ordenador.* Sintesis, 1988.

[15] **R. Turner**: *Logics for Artificial Intelligence.* Ellis Horwood, 1984.

[16] **F. Winkler**: *Polynomial Algorithms in Computer Algebra.* Springer-Verlag, 1996.