Notre Dame Journal of Formal Logic
Volume 23, Number 4, October 1982

Nonstandard Propositional Logics and Their Application to Complexity Theory

MICHAEL EVANGELIST*

1 Introduction and background Let Σ^* be the set of all finite-length strings over some fixed alphabet Σ. Then a *language* (over Σ) is a set $L \subseteq \Sigma^*$. Define $P = \{L \mid L$ is accepted by a deterministic Turing machine (DTM) in a polynomial number of steps$\}$, where the argument to the polynomial function is the length of the input string. *NP* is the analogous family for nondeterministic Turing machines (NDTMs).

The family P is widely considered to represent the class of feasibly solvable computational problems. Representative of this class, in a sense to be defined precisely, is the set S of satisfiable propositional formulas. Cook [3] has shown that S is a member of P if, and only if, $P = NP$. (The proof method is similar to that used by Büchi [1] for establishing the unsolvability of the decision problem for the predicate calculus.)

Cook's result has far-reaching implications for the theory of computational complexity, because many interesting combinatorial problems are in the family *NP* but are not known to be in P. (See Karp [7]) That is, each of these problems can be solved in polynomial time if, and only if, there is a polynomial time decision procedure for S. In addition, as Cook and Reckhow [4] observe, $P = NP$ would also imply an interesting philosophical

*Research was supported in part by NSF grant # MCS-79-08919 and by Colgate Faculty Research Council. The author would like to thank both V. F. Rickey and an anonymous referee on an earlier version of this paper for suggestions leading to improvements in the presentation.

Received June 25, 1979; revised June 15, 1981

consequence for mathematicians. If $P = NP$ then there exists a polynomial p and an algorithm A with the following property: Given any proposition W of set theory and any integer n, A determines within only $p(n)$ steps whether W has a proof of length n, or less in, say, ZF set theory. The problem is in NP because a NDTM can "guess" a proof of W, if one exists, and verify it in polynomial time. S represents the class NP in the sense that every set L in NP is many-one reducible to S by a function f_L computable in deterministic polynomial time. Thus, a polynomial time algorithm for S would imply a polynomial time algorithm for L via f_L.

We write $L_1 \propto L_2$ if there is a polynomial time reduction from set L_1 to L_2. The following theorem is proved in Garey and Johnson [5]:

Theorem 1 *If $L_1 \propto L_2$, then $L_2 \in P$ implies $L_1 \in P$ (and, equivalently, $L_1 \notin P$ implies $L_2 \notin P$).*

A language L is said to be *NP-complete* if L is in NP and, for any language L' in NP, $L' \propto L$. By Theorem 1 a polynomial time decision procedure for any NP-complete language would provide a feasible algorithm for every language in NP. Such a procedure would imply $P = NP$, since, clearly, $P \subseteq NP$. Most complexity theorists believe that $P \neq NP$, but extensive research has failed to settle the question. Cook [3] proved that S is NP-complete, and most other NP-completeness results have used S either directly or indirectly. The set of satisfiable formulas, therefore, is the basic NP-complete problem in the same sense that the halting problem for Turing machines is a basic unsolvable problem. S is representative of NP because, like all NP-complete languages, it is a "hardest" set to recognize.

This discussion of NP-complete languages can easily be recast in terms of decision problems. Simply encode the latter as symbol strings and, thereby, reduce them to questions of set membership. (See Garey and Johnson [5] for details.) It is for this reason that we use the terms "language" and "problem" interchangeably.

Cook and Reckhow [4] studied a question related to the theory of NP-completeness. Let co-NP = $\{\Sigma^* - L | L \text{ is in } NP\}$, and define co-$P$ analogously. Clearly, P = co-P, since a polynomial time DTM that accepts some L in P can be altered to accept $\Sigma^* - L$ by inverting the answer it gives on each input. Therefore to show that $NP \neq$ co-NP is to show that $NP \neq P$. It is possible that NP = co-NP but still $NP \neq P$, although most complexity theorists believe this to be unlikely. (See Garey and Johnson [5], for example.)

The problem studied by Cook and Reckhow is that of the existence of an efficient proof system for the tautologies of propositional logic. Informally, a proof system is efficient if it provides a polynomially long proof for every tautology. (The length of the proof is given as a function of the length of the formula proved.) We state the following results from Cook and Reckhow [4]:

Theorem 2 (Cook and Reckhow) *NP is closed under complementation if, and only if, the set of tautologies is in NP.*

Theorem 3 (Cook and Reckhow) *A nonempty set L is in NP if, and only if, L has an efficient proof system.*

Therefore, *NP* is closed under complementation if, and only if, there exists an efficient proof system for the tautologies. Efficient proof systems must exist for any L in *NP*, although such formal systems have not generally been studied or even described. The usual practice is to show that L is in *NP* by giving a description of a NDTM that accepts L in polynomial time. This method is generally easier than specifying a system of axioms and inference rules, since the problems of interest are naturally viewed as computational. In the sequel, we examine various formal systems for deriving subsets of the propositional formulas that represent the families *NP* and co-*NP*. It is hoped that these systems will provide a significant alternative means to analyze the structure of these families. The method seems especially suited to *NP*-complete logic sets.

Let us first note that the classical propositional formulas may be partitioned under the usual interpretation into three sets: the tautologies (T), the contingencies (C), and the unsatisfiable formulas (US). From the point of view of the theory of computational complexity, there are six interesting combinations of these subsets:

(1) T
(2) C
(3) US
(4) S (the satisfiable formulas)
(5) NT (the nontautologies)
(6) NC (the noncontingencies).

The sets (2), (4), and (5) are *NP*-complete. This status cannot be claimed for (1), (3), and (6), because it is not known whether these co-*NP* sets are members of *NP*. It is the case, however, that placing any of the latter in P would imply $P = NP$. If either (1), (3), or (6) is a member of *NP*, then $\underline{NP}$ = co-*NP*. For example, if the tautologies are in *NP*, then Theorems 2 and 3 assert that NP = co-*NP*. Analogous results can be shown for the noncontingencies and the unsatisfiable formulas.

Caicedo [2] has provided a formal system for the set NT. (In general, a formal system to derive exactly the strings of set X will be denoted by $\underline{X}$.) NT is a Hilbert system in the sense that each line in the derivation of a nontautology is itself a nontautology.

In Section 2 we extend Caicedo's generalization of the concept of a propositional theorem by giving Hilbert systems for each of the other nonstandard propositional logics listed above. We also present a completeness proof for one of these systems. The primary contribution of this paper is Section 3, where we show that $\underline{C}$, $\underline{S}$, and $\underline{NT}$ are efficient proof systems. Section 4 outlines future work in this area.

*2 **The formal systems*** The only connectives used in the following systems are $\sim$ and $\supset$. We denote atomic formulas by p, q, p_1, p_2, ... The symbols α, β, and γ are used to denote arbitrary formulas. Define the set $\sigma(\alpha) = \{p \mid p \text{ occurs in } \alpha\}$. We say that the condition $**(\alpha, A)$ holds when $A = A_1$ or $A = (A_1 \supset (A_2 \supset \ldots (A_{n-1} \supset A_n) \ldots))$, with $A_i = p_i$ or $A_i = \sim p_i, p_i \neq p_j$ for $i \neq j$, and $\sigma(\alpha) \subseteq \sigma(A)$.

We first give an explicit system for S and then describe the method for obtaining the other nonstandard systems.

Axiom SA $\sim p$ (p atomic)

Rules

SR1(a) $\dfrac{\sim\alpha}{\sim(p \supset \alpha)}$ (p atomic, $p \notin \sigma(\alpha)$)

SR1(b) $\dfrac{\sim\alpha}{\sim(\sim p \supset \alpha)}$ (p atomic, $p \notin \sigma(\alpha)$)

SR2 $\dfrac{\sim(\alpha \supset \beta)}{\sim(\alpha \supset (\alpha \supset \beta))}$

SR3 $\dfrac{\sim(\alpha \supset \beta)}{\sim((\gamma \supset \alpha) \supset \beta)}$

SR4 $\dfrac{\sim(\sim\alpha \supset \beta)}{\sim((\alpha \supset \gamma) \supset \beta)}$

SR5 $\dfrac{\sim(\alpha \supset \beta)}{\alpha}$

SR6 $\dfrac{\sim(\alpha \supset \beta)}{\sim(\sim\sim\alpha \supset \beta)}$

SR7 $\dfrac{\sim(\alpha \supset (\beta \supset \gamma))}{\sim(\beta \supset (\alpha \supset \gamma))}$

SR8 $\dfrac{\sim(\alpha \supset A), \sim(\sim\beta \supset A)}{\sim(\sim(\alpha \supset \beta) \supset A)}$

In SR8 the condition $**(\alpha \supset \beta, A)$ must hold.

Caicedo's [2] system for NT can be shortened by using the single axiom p, rather than his two axioms. The system $\underline{C}$ is similar to $\underline{NT}$, but care must be taken with several rules. For example, one rule of $\underline{NT}$ is

$$\frac{\alpha \supset \beta}{(\gamma \supset \alpha) \supset \beta}.$$

This rule is invalid in $\underline{C}$, because $\gamma \equiv \alpha$ and $\beta \in US$ is possible. Thus, β must be added to the list of hypotheses for the rule.

The axioms for the system $\underline{US}$ are easily derived from any complete set of axioms for the tautologies by negating each of the latter. The inference rules would then be:

$$\frac{\sim(\alpha \supset \beta), \sim\alpha}{\sim\beta} \text{ and } \frac{\sim\sim\alpha}{\alpha}.$$

Similarly, each axiom of $\underline{NC}$ has the form $p \supset \alpha$, where α is an axiom of $\underline{T}$, and p is an atomic propositional symbol not occurring in α. A modified form of modus ponens is obtained analogously by adding the antecedent p to both hypotheses and conclusion. In addition, the rules

$$\frac{p \supset \beta}{\beta} \text{ and } \frac{\sim\beta}{\beta}$$

are required.

Examples: We use the symbol $\vdash_{\underline{X}}\alpha$ to assert that α is derivable within the system $\underline{X}$. When $\underline{X}$ is clear from the context, we abbreviate $\vdash_{\underline{X}}$ to $\vdash$.

S1 $\vdash_{\underline{S}} \sim(p \supset \sim p)$

1. $\sim q$	SA
2. $\sim(p \supset \sim q)$	SR1(a)
3. $\sim(p \supset (p \supset \sim q))$	SR2
4. $\sim(\sim\sim p \supset (p \supset \sim q))$	SR6
5. $\sim(\sim(p \supset \sim p) \supset (p \supset \sim q))$	3,4 SR8
6. $\sim(p \supset \sim p)$	SR5

S2 $\vdash_{\underline{S}} \sim(\sim p \supset p)$

1. $\sim q$	SA
2. $\sim(\sim p \supset \sim q)$	SR1(b)
3. $\sim(\sim p \supset (\sim p \supset \sim q))$	SR2
4. $\sim(\sim(\sim p \supset p) \supset (\sim p \supset \sim q))$	3,3 SR8
5. $\sim(\sim p \supset p)$	SR5

S3 $\vdash_{\underline{S}} \sim\sim p$

1. $\sim q$	SA
2. $\sim(p \supset q)$	SR1(a)
3. $\sim(\sim\sim p \supset q)$	SR6
4. $\sim\sim p$	SR5

Remark: The axioms for $\underline{T}$, $\underline{US}$, and $\underline{NC}$ are actually axiom schemata. Those for $\underline{C}$, $\underline{S}$, and $\underline{NT}$ are concrete axioms. In the latter case, schemata cannot be used exclusively, nor can a substitution rule be allowed. The properties characterizing the latter sets are not hereditary with respect to substitution.

We present a completeness theorem for the system $\underline{S}$ to facilitate the analysis of proof lengths in the next section. Completeness theorems for the other formal systems discussed in this paper are similarly obtained.

Theorem 4

A. *If* $\vdash_{\underline{S}}\alpha$ *then* $\alpha \in S$.
B. *If* $\alpha \in S$ *then* $\vdash_{\underline{S}}\alpha$.

Proof: Since $\alpha \in S$ implies $\sim\alpha \in NT$, $\underline{S}$ is essentially $\underline{NT}$ with the appropriate negation signs. The following proof resembles Caicedo's [2] proof for $\underline{NT}$, again with the appropriate negation signs.

A. Each rule of NT takes either the form $\frac{\alpha}{\beta}$ or $\frac{\alpha_1,\alpha_2}{\beta}$. Since $\alpha \in NT$, we have $\sim\alpha \in S$, and $\beta \in NT$ implies $\sim\beta \in S$. Thus, $\frac{\sim\alpha}{\sim\beta}$ and $\frac{\sim\alpha_1,\sim\alpha_2}{\sim\beta}$ are valid for $\underline{S}$,

if $\frac{\alpha}{\beta}$ and $\frac{\alpha_1,\alpha_2}{\beta}$ are valid for $\underline{NT}$. These observations establish part A for SA and each rule except SR5. In the latter case, note that $\sim(\alpha \supset \beta) \in S$ implies $\alpha \in S$.

B. We use induction on the complexity of α to prove

(*) If $**(\alpha, A)$ holds, then $\sim(\alpha \supset A) \in S$ implies $\vdash_{\underline{S}} \sim(\alpha \supset A)$.

Case I. $\alpha = p_j$. Since there exists a v such that $\mathsf{v}(\sim(p_j \supset A)) = \mathsf{T}$, we have $\mathsf{v}(p_j) = \mathsf{T}$ and $\mathsf{v}(A) = \mathsf{F}$. Thus, $\mathsf{v}(A_i) = \mathsf{T}$, $1 \leqslant i < n$, and $\mathsf{v}(A_n) = \mathsf{F}$.

Subcase Ia. $j < n$. Since $\mathsf{v}(A_j) = \mathsf{v}(p_j) = \mathsf{T}$, we have $A_j = p_j$, and $A = A_1 \supset (A_2 \supset \ldots (p_j \supset \ldots . \supset A_n) \ldots)$. To derive $\sim(p_j \supset A)$:

	$\sim A_n$	(either SA or example S3)
SR1	$\sim(A_{n-1} \supset A_n)$	
SR1	$\sim(A_{n-2} \supset (A_{n-1} \supset A_n))$	
	.	
	.	(SR1)
	.	
	$\sim(p_j \supset (A_{j+1} \supset \ldots \supset A_n) \ldots)$	
SR2	$\sim(p_j \supset (p_j \supset (A_{j+1} \supset \ldots \supset A_n) \ldots)$	
SR1	$\sim(A_{j-1} \supset (p_j \supset (p_j \supset \ldots \supset A_n) \ldots)$	
SR7	$\sim(p_j \supset (A_{j-1} \supset (p_j \supset \ldots \supset A_n) \ldots)$	
	.	
	.	(SR1 & SR7)
	.	
	$\sim(p_j \supset (A_1 \supset \ldots \supset A_n) \ldots)$	
	$\vdash_{\underline{S}} \sim(p_j \supset A)$	

Subcase Ib. $j = n$. Since $\mathsf{v}(p_n) = \mathsf{v}(p_j) = \mathsf{T}$ and $\mathsf{v}(A_n) = \mathsf{F}$, we must have $A_n = \sim p_n$. Thus,

S1	$\sim(p_n \supset \sim p_n)$	
SR1	$\sim(A_{n-1} \supset (p_n \supset \sim p_n))$	
SR7	$\sim(p_n \supset (A_{n-1} \supset \sim p_n))$	
	.	
	.	(SR1 & SR7)
	.	
	$\sim(p_n \supset (A_1 \supset \ldots \supset \sim p_n) \ldots)$	
	$\vdash_{\underline{S}} \sim(p_n \supset A)$	

Case II. (inductive step) $\alpha = \sim\beta$.

Subcase IIa. $\beta = p_j$. Similar to Case I.

Subcase IIb. $\beta = \sim\gamma$. Since $\mathsf{v}(\sim(\sim\sim\gamma \supset A)) = \mathsf{T}$, we have $\mathsf{v}(\sim(\gamma \supset A)) = \mathsf{T}$. By the induction hypothesis $\vdash \sim(\gamma \supset A)$. By SR6: $\vdash_{\underline{S}} \sim(\sim\sim\gamma \supset A)$.

Subcase IIc. $\beta = (\gamma \supset \gamma')$. Thus, $\mathsf{v}(A) = \mathsf{F}$, $\mathsf{v}(\gamma) = \mathsf{T}$, and $\mathsf{v}(\gamma') = \mathsf{F}$. By the induction hypothesis, $\vdash \sim(\gamma \supset A)$ and $\vdash \sim(\sim\gamma' \supset A)$. By SR8: $\vdash_{\underline{S}} \sim(\sim(\gamma \supset \gamma') \supset A)$.

Case III. (inductive step) $\alpha = (\gamma \supset \gamma')$. If $v(\sim((\gamma \supset \gamma') \supset A)) = T$, then $v(A) = F$ and either $v(\gamma) = F$ or $v(\gamma') = T$. In the first case, $v(\sim(\sim\gamma \supset A)) = T$. By the induction hypothesis: $\vdash \sim(\sim\gamma \supset A)$, and by SR4: $\vdash_{\underline{S}} \sim((\gamma \supset \gamma') \supset A)$. In the latter case, $v(\sim(\gamma' \supset A)) = T$. By the induction hypothesis: $\vdash \sim(\gamma' \supset A)$, and by SR3: $\vdash_{\underline{S}} \sim((\gamma \supset \gamma') \supset A)$.

We have therefore established (*). Now let $v(\alpha) = T$, $\sigma(\alpha) = \{p_1, p_2, \ldots, p_n\}$, and define $p_i^v = p_i$ if $v(p_i) = T$ and $p_i^v = \sim p_i$, otherwise. Construct $A = p_1^v \supset (p_2^v \supset \ldots (p_{n-1}^v \supset \sim p_n^v) \ldots)$. Then, $v(A) = F$, and $v(\sim(\alpha \supset A)) = T$, and by (*) we have $\vdash \sim(\alpha \supset A)$. By SR5: $\vdash_{\underline{S}} \alpha$.

*3 **Proof lengths*** By Theorem 3, each of *C*, *NT*, and *S* has an efficient proof system. The next theorem states that the formal systems given in Section 2 provide polynomially long proofs for members of the respective sets. We take the following definitions from Cook and Reckhow [4]:

Definitions We denote by $\mathcal{L}$ the set of functions $f: \Sigma_1^* \to \Sigma_2^*$, where Σ_1 and Σ_2 are any finite alphabets, such that f can be computed by a DTM in time bounded by a polynomial in the length of the input. If $L \subseteq \Sigma^*$, a *proof system* for L is a function $f: \Sigma_1^* \to L$ for some alphabet Σ_1 and f in $\mathcal{L}$ such that f is onto. A proof system is *polynomially bounded* if, and only if, there is a polynomial $p(n)$ such that for all y in L there exists x in Σ_1^* such that $y = f(x)$ and $|x| \leqslant p(|y|)$, where $|z|$ denotes the length of z. If $y = f(x)$, we say that x is a *proof* of y.

It should be clear that each of the formal systems discussed in Section 2 denotes a proof system in the sense defined. A crucial open question is whether $\underline{T}$, $\underline{US}$, and $\underline{NC}$ provide polynomially bounded proof systems. Although this problem is not addressed here, we point out that standard completeness theorems construct exponentially long proofs. For example, Kalmár's [6] completeness theorem for $\underline{T}$ exploits the semantics of tautologies. There is a line in every proof constructed for each of the 2^n truth-table rows that satisfy the tautology. Our completeness proof for $\underline{S}$ proceeded similarly, but at most two rows of the truth-table were reflected in the formal proof constructed.

Theorem 5 *The proof systems $\underline{C}$, $\underline{NT}$, and $\underline{S}$ are polynomially bounded.*

Proof: We will examine only the system $\underline{S}$. The proofs constructed by $\underline{C}$ and $\underline{NT}$ are analogously investigated.

Let α be a satisfiable formula, where $\sigma(\alpha) \subseteq \{p_1, p_2, \ldots, p_n\}$. We use induction on the complexity of α to show that α has a proof $\Pi = B_1, B_2, \ldots, B_k$ in $\underline{S}$ such that $k \leqslant r|\alpha|$, where $r > 0$ is some integer constant. Since each B_i, $1 \leqslant i \leqslant k$, will be no longer than $s|\alpha|$, $s > 0$, we will conclude that $|\Pi| \leqslant t|\alpha|^2$, $t > 0$.

Let v be a valuation such that $v(\alpha) = T$. Construct the formula $A = p_1^v \supset (p_2^v \supset \ldots (p_{n-1}^v \supset \sim p_n^v) \ldots)$ specified at the end of the proof for Theorem 4. Clearly, there exists a constant $s > 0$ such that $|A| \leqslant s|\alpha|$. Since the initial portion of the proof of α is a proof of $\sim(\alpha \supset A)$, we first use induction on the complexity of α to show that $\sim(\alpha \supset A)$ has a proof of length no more than $r'|\alpha|^2$, for some $r' > 0$. To do so we break the analysis into the same cases as the completeness proof, to which the reader should refer.

Case I. $\alpha = p_j$. In Subcase Ia we obtain $\vdash_{\underline{S}} \sim(\alpha \supset A)$ in at most $4 + (n - j) + 1 + 2(j - 1) = n + j + 3$ lines. Since j is bounded by $n - 1$, we conclude that no more than $3n$ lines are needed.

In Subcase Ib only $1 + 2(n - 1) = 2n - 1$ lines are generated. The number of lines is bounded, therefore, by $2n$.

Case II. $\alpha = \sim\beta$. In Subcase IIa the analysis is similar to the atomic case. In Subcase IIb the induction hypothesis states that $\sim(\alpha \supset A)$ has a proof in $\underline{S}$ of length less than or equal to $r'|\gamma|^2$ for some constant $r' > 0$. Since $\sim(\sim\sim\gamma \supset A)$ is then deduced in one step, we see that this formula has a proof no longer than $r''|\sim\sim\gamma|^2 = r''|\alpha|^2$, for some $r'' > 0$. The induction hypothesis tells us, in Subcase IIc, that $\sim(\gamma \supset A)$ and $\sim(\sim\gamma' \supset A)$ have proofs no longer than $r'|\gamma|^2$ and $r''|\sim\gamma'|^2$, respectively, for $r', r'' > 0$. We conclude that there is a proof of $\sim(\alpha \supset A)$ of length less than or equal to $r'''|\alpha|^2$, where $r''' > r' + r'' + 1$.

Case III. $\alpha = (\gamma \supset \gamma')$. The analysis is similar to Case II.

In Case I no more than $3n$ lines of proof are needed, each of which is no longer than $r'|\alpha|$, for some $r' > 0$. Since we may assume $n \leqslant |\alpha|$, it is clear that the proof is no longer than $r|\alpha|^2$, for some $r > 0$. The same holds for Cases II and III. To derive α from $\sim(\alpha \supset A)$ requires only one more step. We conclude that there is a proof in $\underline{S}$ of α of length less than or equal to $t|\alpha|^2$, for some $t > 0$.

Therefore by Theorem 5 each formal system for the *NP* sets is as efficient as needed for studying the complexity of these sets.

4 Observations and future research As noted in Section 1, T is in *NP* if and only if T has an efficient proof system. Although the NDTMs given for *NP* sets do constitute such proof systems, they provide little insight into the syntactic structure of these sets. The Hilbert systems discussed in Section 2 surmount this difficulty to the extent that they proceed syntactically and are more "natural" as proof systems than Turing machine computations.

Numerous questions are suggested by the research described in this paper. The distinction between *NP* and co-*NP*, if *NP* $\neq$ co-*NP*, seems to be captured in the difference between the efficiencies of proof systems for the *NP* subsets of the propositional formulas and the co-*NP* subsets. The more natural formal systems for these sets may lead to a better understanding of these differences. For example, the fact that substitution into axioms is allowable for the co-*NP* systems but not for the *NP* systems argues for a careful study of the proof complexity introduced by a substitution rule. A similar phenomenon warranting investigation is the fact that schematic inference rules in the *NP* systems must have hypotheses, whereas such rules in the co-*NP* systems need not.

REFERENCES

[1] Büchi, J. R., "Turing machines and the *Entscheidungsproblem*," *Mathematische Annalen*, vol. 148 (1962), pp. 201-213.

[2] Caicedo, X., "A formal system for the non-theorems of the propositional calculus," *Notre Dame Journal of Formal Logic*, vol. 19 (1978), pp. 147-151.

[3] Cook, S. A., "The complexity of theorem proving procedures," pp. 151-158 in *Proceedings of the Third Annual ACM Symposium on the Theory of Computing, May 1971*.

[4] Cook, S. A. and R. A. Reckhow, "The relative efficiency of propositional proof systems," *The Journal of Symbolic Logic*, vol. 44 (1979), pp. 36-50.

[5] Garey, M. R. and D. S. Johnson, *Computers and Intractibility: A Guide to the Theory of NP-Completeness*, W. H. Freeman, San Francisco, 1979.

[6] Kalmár, L., "Zuruckführung des Entscheidungsproblems auf den Fall von Formeln mit einer einzigen binären Funktionsvariablen," *Compositio Mathematica*, vol. 4 (1936), pp. 137-144.

[7] Karp, R. M., "Reducibility among combinatorial problems," pp. 85-103 in *Complexity of Computer Computations*, eds., R. E. Miller and J. W. Thatcher, Plenum Press, New York, 1972.

Computer and Information Studies
Colgate University
Hamilton, New York 13346