ALASDAIR URQUHART

THE COMPLEXITY OF DECISION PROCEDURES IN RELEVANCE LOGIC

1. Introduction

Relevance logic is distinguished among non-classical logics by the richness of its mathematical as well as philosophical structure. This richness is nowhere more evident than in the difficulty of the decision problem for the main relevant propositional logics. These logics are in general undecidable [22]; however, there are important decidable subsystems. The decision procedures for these subsystems convey the impression of great complexity. The present paper gives a mathematically precise form to this impression by showing that the decision problem for $R_{\rightarrow}$, the implication fragment of R, is exponential space hard. This means that any Turing machine which solves this decision problem must use an exponential amount of space (relative to the input size) on infinitely many inputs.

The first-degree fragment common to E, R and most other well-known relevance logics has a simple decision procedure. The details of the Anderson-Belnap decision procedure are to be found in [4, Chapter III]. A four-valued matrix is characteristic for this fragment; since the decision problem for the two-valued propositional calculus can be reduced efficiently to the decision problem for this fragment, the complexity of the decision problem is the same as that for the classical two-valued propositional calculus (that is to say, the problem is co-NP-complete).

The decision problem for the pure implicational fragments of E and R is much harder. It was solved by Saul Kripke in a *tour de force* of combinatorial reasoning, which was published only as an abstract [13]. Belnap and Wallace extended Kripke's decision procedure to the implication-negation fragment of E in [5]; an account of their decision method is to be found in [4, pp. 124–139]. The decision method extends immediately to R. In fact, in the case of R we can go farther; Robert Meyer in his thesis [16] showed how to translate the logic LR, which results from R by omitting the distribution axiom, into $R_{\rightarrow\wedge}$, so that the decision procedure can be extended to all of LR. This decision procedure has been implemented as a program Kripke by Thistlewaite, McRobbie and Meyer [21]. The program is not simply a straightforward implementation of the decision procedure; finite matrices are used extensively to prune invalid nodes from the search tree.

The decision methods of Kripke, Belnap and Wallace are of a truly marvellous complexity. In fact, they are so complex that it is not clear how to compute an upper bound on the number of steps required by the procedures for an input formula of a given length. The key combinatorial lemma of Kripke which forms the basis for all these decision procedures [4, pp. 138–139] simply asserts the finiteness of the search tree without giving an explicit bound on its size.

J. M. Dunn and A. Gupta (eds.), Truth or Consequences, 61–76.

Is this complexity inherent in the logics in question, or could there be a more efficient decision method? The lower bound proved in this paper provides a partial answer to this question. It is proved by adapting a method used by Mayr and Meyer [15] to show that the word problem for finitely presented commutative semigroups is exponential space complete. The method used here is a simplification of Mayr and Meyer 's, and provides an alternative proof of their lower bound.

In the last part of the paper we discuss the complexity of the Kripke decision method, and show, by adapting some ideas of McAloon [14], that the size of the search tree produced by the method is primitive recursive in the Ackermann function. It is not clear whether the large lower bound or the monstrous upper bound is closer to the truth.

2. MODEL THEORY FOR IMPLICATIONAL LOGICS

The lower bound is proved for a fixed logical system, for the sake of convenience; we shall indicate later how the proof can be adapted to a more general class of systems. The logic we discuss is the system $R_{\rightarrow\wedge\circ}$, which contains the fusion connective $\circ$ in addition to $\rightarrow$ and $\wedge$ (the fusion connective is dubbed "co-tenability" by Anderson and Belnap [4, pp. 344–346]). We also include the constant t. Employing the conventions of Church [6] for replacing parentheses by dots, we state the axioms for the system as follows:

1. t
2. $t \rightarrow .A \rightarrow A$
3. $A \rightarrow B \rightarrow .B \rightarrow C \rightarrow .A \rightarrow C$
4. $(A \rightarrow .B \rightarrow C) \rightarrow .B \rightarrow .A \rightarrow C$
5. $(A \rightarrow .B \rightarrow C) \rightarrow .A \rightarrow B \rightarrow .A \rightarrow C$
6. $(A \wedge B) \rightarrow A$
7. $(A \wedge B) \rightarrow B$
8. $(A \rightarrow B) \wedge (A \rightarrow C) \rightarrow .A \rightarrow (B \wedge C)$
9. $A \rightarrow .B \rightarrow .A \circ B$
10. $(A \rightarrow .B \rightarrow C) \rightarrow .(A \circ B) \rightarrow C$

The rules of the system are *modus ponens* and *adjunction*:

From $A \rightarrow B$ and A infer B (*modus ponens*);
From A and B infer $A \wedge B$ (*adjunction*).

It is an easy exercise to prove from these axioms that fusion is associative, commutative and square-increasing, and that t forms the semigroup identity. Thus, using $(A \leftrightarrow B)$ as an abbreviation for $((A \rightarrow B)\wedge(B \rightarrow A))$, the following are theorems:

$$\begin{array}{rcl} ((A \circ B) \circ C) & \leftrightarrow & (A \circ (B \circ C)) \\ (A \circ B) & \leftrightarrow & (B \circ A) \\ A & \rightarrow & (A \circ A) \\ (t \circ A) & \leftrightarrow & A \end{array}$$

We now state the model theory for the system, which is due to Richard Routley and Robert Meyer [19]. A *model structure* is a triple $\langle 0, K, R\rangle$, where K is a set, $0 \in K$, and R is a ternary relation on K satisfying the postulates:

(a) $R0aa$

(b) $Raaa$

(c) $(Rabc \wedge Rcde) \to \exists x(Radx \wedge Rxbe)$

(d) $(R0ax \wedge Rxbc) \to Rabc$

(e) $(Rabc \wedge R0cd) \to Rabd$

If $M = \langle 0, K, R\rangle$ is a model structure, then a *valuation* in M is a function V that assigns to each propositional variable P a set $V(P) \subseteq K$, and which satisfies the condition: if $a \in V(P)$ and $R0ab$ then $b \in V(P)$. A *model* consists of a model structure, together with a valuation.

If $\mathcal{M}$ is a model, we define truth relative to a point in $\mathcal{M}$ by the following recursive definition (writing $x \models A$ for "A is true at point x in $\mathcal{M}$"):

(a) For a variable $P, x \in K,\ x \models P \Leftrightarrow x \in V(P)$;

(b) $x \models t \Leftrightarrow R00x$;

(c) $x \models (A \to B) \Leftrightarrow \forall yz((Rxyz \ \&\ y \models A) \Rightarrow z \models C)$;

(d) $x \models (A \circ B) \Leftrightarrow \exists yz(Ryzx \ \&\ y \models A \ \&\ z \models B)$;

(e) $x \models (A \wedge B) \Leftrightarrow x \models A \ \&\ x \models B$.

A formula A is *valid* if $0 \models A$ in all models $\mathcal{M}$. It is not difficult to verify that all theorems of $R_{\to\wedge\circ}$ are valid (see [19] for details).

3. MODELS CONSTRUCTED FROM SEMI-THUE SYSTEMS

Let Σ be a finite alphabet, Σ^* the set of all finite strings over Σ. We define a *semi-Thue system* to be a finite set of pairs of strings in Σ^*; if $\langle\alpha, \beta\rangle$ is such a pair of strings, we write it in the form $\alpha \longrightarrow \beta$, and refer to it as a *production* of the semi-Thue system.

We say that $\gamma\alpha\delta \Rightarrow \gamma\beta\delta\ (S)$ if γ and δ are words in the alphabet of S, and the production $\alpha \longrightarrow \beta$ belongs to S. The relation $\alpha \overset{*}{\Longrightarrow} \beta\ (S)$ is defined to hold if (1) $\alpha = \beta$ or (2) there is a sequence $\alpha_1, \cdots, \alpha_k$ so that $\alpha = \alpha_1$, $\beta = \alpha_k$ and $\alpha_i \Rightarrow \alpha_{i+1}(S)$ for $1 \leq i < k$.

A semi-Thue system over an alphabet Σ is *commutative* if it includes all productions of the form $xy \longrightarrow yx$, for $x, y \in \Sigma$; it is *contractive* if it includes all productions of the form $xx \longrightarrow x$ for $x \in \Sigma$. If S is a commutative contractive semi-Thue system, we say that a production belonging to S is *proper* if it is not of the form $xy \longrightarrow yx$ or $xx \longrightarrow x$.

We now show how to construct a model for $R_{\to\wedge\circ}$ from a commutative contractive semi-Thue system. The basic idea for the construction is derived from Meyer and Routley [18].

Let S be a semi-Thue system over the alphabet Σ. Define a ternary relation on Σ^* by: $R\alpha\beta\gamma$ if and only if $\alpha\beta \overset{*}{\Rightarrow} \gamma\ (S)$. The relational structure $M(S)$ is defined as $\langle 0, \Sigma^*, R\rangle$, where 0 is the empty string.

Lemma 1. *If S is a commutative, contractive semi-Thue system, then $M(S)$ is a model structure.*

The relation $R0\alpha\alpha$ clearly holds since 0 is defined as the empty string; the relation $R\alpha\alpha\alpha$ holds for all strings α because S is contractive.

To show that the third condition holds, assume that $R\alpha\beta\gamma$ and $R\gamma\delta\epsilon$, that is, $\alpha\beta \overset{\star}{\Longrightarrow} \gamma$ and $\gamma\delta \overset{\star}{\Longrightarrow} \epsilon$. Setting $x = \alpha\delta$, we find: $x\beta \overset{\star}{\Longrightarrow} \alpha\beta\delta \overset{\star}{\Longrightarrow} \gamma\delta \overset{\star}{\Longrightarrow} \epsilon$, so that $R\alpha\delta x$ and $Rx\beta\epsilon$.

Finally, to show that the fourth condition holds, assume that $R0\alpha x$ and $Rx\beta\gamma$; then $\alpha \overset{\star}{\Longrightarrow} x$, so that $\alpha\beta \overset{\star}{\Longrightarrow} x\beta \overset{\star}{\Longrightarrow} \gamma$, that is, $R\alpha\beta\gamma$; the fifth condition is proved similarly. □

If Σ is a finite alphabet, we correlate distinct propositional variables with the elements of Σ. We write $P(\sigma)$ for the propositional variable correlated with the symbol σ.

If α is a string in Σ^*, we write $P(\alpha)$ for the propositional expression corresponding to α, where concatenation is represented by the fusion operator. Parentheses can be omitted in the propositional expression in view of the associative law for fusion. The expression t is correlated with the empty string.

Let S be a commutative, contractive semi-Thue system. The canonical model $\mathcal{M}(S)$ associated with S is the model defined on the model structure $M(S)$ by the valuation: $V(P(\sigma)) = \{\alpha \in \Sigma^* : \sigma \overset{\star}{\Longrightarrow} \alpha\ (S)\}$, and $V(P) = \emptyset$ for uncorrelated variables.

Lemma 2. *Let S be a commutative, contractive semi-Thue system, and γ, δ words of S. Then $\gamma \models P(\delta)$ in the canonical model $\mathcal{M}(S)$ if and only if $\delta \overset{\star}{\Longrightarrow} \gamma$.*

Proof. If δ is a single symbol, then the lemma holds by definition. If δ is the empty word, then $\gamma \models P(\delta)$ if and only if $R00\gamma$, that is, $\delta \overset{\star}{\Longrightarrow} \gamma$.

Assume that it holds for the words δ_1 and δ_2. Then $\gamma \models P(\delta_1\delta_2)$ if and only if there exist α and β so that $\alpha\beta \overset{\star}{\Longrightarrow} \gamma$, $\delta_1 \overset{\star}{\Longrightarrow} \alpha$ and $\delta_2 \overset{\star}{\Longrightarrow} \beta$. This last condition is easily seen to be equivalent to $\delta_1\delta_2 \overset{\star}{\Longrightarrow} \gamma$. □

Let S be a semi-Thue system, γ and δ words belonging to S. The formula $\Psi(S, \gamma, \delta)$, is defined as:

$$[P(\beta_1) \to P(\alpha_1)\ \wedge \ldots \wedge\ P(\beta_n) \to P(\alpha_n)\ \wedge t] \to .P(\delta) \to P(\gamma)$$

where $\alpha_1 \longrightarrow \beta_1, \ldots, \alpha_n \longrightarrow \beta_n$ are the proper productions of S.

Lemma 3. *If S is a commutative contractive semi-Thue system, and γ, δ words belonging to S, then $R_{\to\wedge\circ} \vdash \Psi(S, \gamma, \delta)$ if and only if $\gamma \overset{\star}{\Longrightarrow} \delta\ (S)$.*

Proof. First, we show that if $\gamma \overset{\star}{\Longrightarrow} \delta\ (S)$, then $\vdash \Psi(S,\gamma,\delta)$. If $\gamma \Rightarrow \delta\ (S)$, then the formula is provable by the axioms for conjunction, the theorem

$$(A \rightarrow B) \rightarrow .(A \circ C) \rightarrow (B \circ C),$$

and the associative, commutative and square increasing theorems for $\circ$. The general case then follows by Axiom 2, and transitivity of implication.

For the converse, let us assume that $\vdash \Psi(S,\gamma,\delta)$. Let $\alpha_k \longrightarrow \beta_k$ be a proper production of S. If $\gamma \models P(\beta_k)$ in the canonical model, then by Lemma 2, we have $\beta_k \overset{\star}{\Longrightarrow} \gamma$, hence $\alpha_k \overset{\star}{\Longrightarrow} \gamma$ so that $\gamma \models P(\alpha_k)$. Thus in the canonical model we have $0 \models P(\beta_k) \rightarrow P(\alpha_k)$. We have thus shown that the antecedent of $\Psi(S,\gamma,\delta)$ is true at 0 in the canonical model. It follows that the consequent $P(\delta) \rightarrow P(\gamma)$ is also true at 0. Hence, $\gamma \overset{\star}{\Longrightarrow} \delta\ (S)$ by Lemma 2. □

4. COMPLEXITY THEORY

In this section, we review the basic notions of complexity theory required in the sequel. For a more complete treatment see [10].

An *off-line Turing machine* is defined to be a multi-tape Turing machine with a read-only input tape, a write-only output tape on which the head never moves left and a read-write work tape. Let Σ_1, Σ_2 be finite alphabets. A function $f : \Sigma_1^* \rightarrow \Sigma_2^*$ *reduces* a set $A \subseteq \Sigma_1^*$ to a set $B \subseteq \Sigma_2^*$ provided that $\alpha \in A \Leftrightarrow f(\alpha) \in B$ for all $\alpha \in \Sigma_1^*$.

If f reduces A to B, and in addition f is computable by a Turing machine which visits at most $log_2\, n$ work tape squares during its computation on any word $\alpha \in \Sigma_1^*$ of length $n > 1$, then A is said to be *log-space reducible* to B; if in addition the length of $f(\alpha)$ is $O(length(\alpha))$, then A is *log-lin reducible* to B.

If A is log-lin reducible to B, then any procedure for deciding B can be converted, by means of the reduction, to a procedure for deciding A which uses space not much greater than that used by the first procedure. In particular, there is a $k > 0$ such that given a Turing machine which decides B in space c^n, one can find a Turing machine deciding A in space c^{kn}.

A set $A \subseteq \Sigma^*$ is said to be *decidable in space* $g : N \rightarrow N$ if there is a Turing machine which accepts A and visits at most $g(n)$ work tape squares during its computation on any word $\beta \in \Sigma^*$ of length n. A is decidable in *exponential space* if it is decidable in space g where $g(n) \leq c^n$ for some $c > 1$.

A set A of words is *exponential space hard with respect to log-lin reducibility* if every set which is decidable in exponential space is log-lin reducible to A. If A itself is decidable in exponential space, then A is *exponential space complete with respect to log-lin reducibility.*

LEMMA 4. *If B is exponential space hard with respect to log-lin reducibility then there is a constant $c > 1$ such that any Turing machine accepting B uses space c^n for infinitely many inputs of length n.*

Proof. By the deterministic space hierarchy theorem [10, pp. 297–298], there is a set A decidable in space 3^n, but not 2^n. Since A is decidable in exponential space, it is log-lin reducible to B. Since A cannot be decided in space 2^n, it follows that B cannot be decided in space $2^{n/k}$. □

The preceding lemma shows that to prove an exponential space lower bound for decision procedures for $R_{\rightarrow\wedge\circ}$, it is sufficient to show that $R_{\rightarrow\wedge\circ}$ is exponential space hard with respect to log-lin reducibility. Since log-lin reducibility is transitive, this will follow provided we can show that a set known to be exponential space hard is log-lin reducible to $R_{\rightarrow\wedge\circ}$. In the following section, we exhibit such an exponential space hard set.

5. SPACE BOUNDED COUNTER MACHINES

A counter machine models a computer having a finite number of registers each of which can contain an arbitrary integer. The basic operations of the machine consist of adding 0, -1 or 1 to the current contents of a register, and branching to new states conditional upon a given register being empty or not. We now give a formal definition of this model.

A *3-counter machine* C consists of a finite set of states Q containing a pair of distinguished states q_0 (the initial state) and q_a (the accepting state), and a transition function

$$\delta : (Q \setminus \{q_a\}) \rightarrow (Q \times \{0, \pm 1\} \times I_3) \cup (Q \times Q \times I_3),$$

where $I_k = \{1, \ldots, k\}$. The *computation* of C is the (possibly infinite) sequence $c^0, c^1, \ldots$ of *instantaneous descriptions* $c^i \in Q \times Z^3$ where

(a) $c^0 = (q_0, 0, 0, 0)$;

(b) If $i \in N$, $c^i = (q, z_1, z_2, z_3)$ and $\delta(q) = (q', d, k) \in Q \times \{0, \pm 1\} \times I_3$, then

$$c^{i+1} = (q', z_1', z_2', z_3'), \text{where } z_i' = \begin{cases} z_i + d & \text{if } i = k \\ z_i & \text{otherwise;} \end{cases}$$

(c) If $i \in N$, $c^i = (q, z_1, z_2, z_3)$ and $\delta(q) = (q', q'', k) \in Q \times Q \times I_3$, then

$$c^{i+1} = \begin{cases} (q', z_1, z_2, z_3) & \text{if } z_k = 0 \\ (q'', z_1, z_2, z_3) & \text{otherwise.} \end{cases}$$

The vector entry c^i_{k+1} is referred to as the contents of the k-th *counter* after i steps, for $k \in I_3$.

The machine C is said to *terminate* iff its computation contains the instantaneous description $(q_a, 0, 0, 0)$. This quadruple, if it occurs, is necessarily the last element in the computation. The *size* of a machine is the number of its states. The computation of a 3-counter machine is *bounded by* n if after any step in the computation the contents of all three counters are ≥ 0 and $\leq n$.

The following lemma can be derived from the results of [8]:

LEMMA 5. *The set ESC* $= \{C : C$ *is a terminating 3-counter machine whose computation is bounded by* $2^{2^{size(C)}}\}$ *is exponential space complete under log-lin reducibility.*

To prove an exponential space lower bound for the decision problem for $R_{\to\wedge\circ}$ we shall show how to construct from a given 3-counter machine a commutative contractive semi-Thue system so that we can reduce the problem ESC efficiently to the word problem for commutative contractive semi-Thue systems. In view of Lemma 3, this will prove an exponential space lower bound for the logical problem.

6. SUCCINCT SEMIGROUP PRESENTATIONS

Before giving the construction of the semi-Thue system, we first present a method used by Mayr and Meyer [15] to give succinct presentations of semigroups. A semigroup presentation is a Thue system, that is, a semi-Thue system in which the inverse of every production is included. We write $\alpha \equiv \beta$ for the two productions $\alpha \longrightarrow \beta$ and $\beta \longrightarrow \alpha$; we use the notation $\alpha \overset{\star}{\equiv} \beta$ as an abbreviation for $\alpha \overset{\star}{\Longrightarrow} \beta$ and $\beta \overset{\star}{\Longrightarrow} \alpha$. We also employ the abbreviation e_n for 2^{2^n}.

For each $n \in N$, we define a commutative Thue system $\mathcal{P}_n$. The system $\mathcal{P}_n$ and its alphabet A_n are defined by induction on n. We define:

$$\begin{aligned} A_0 &= \{s, f, c_1, c_2, c_3, c_4, b_1, b_2, b_3, b_4\} \\ \mathcal{P}_0 &= \{sc_i \equiv fc_ib_i^2 : i \in I_4\}. \end{aligned}$$

For $m > 0$, let $S, F, C_1, C_2, C_3, C_4, B_1, B_2, B_3, B_4$ be distinct symbols not in A_{m-1}. Then the alphabet of $\mathcal{P}_m$ is defined as:

$$A_m = A_{m-1} \cup \{S, F, C_1, C_2, C_3, C_4, B_1, B_2, B_3, B_4\}.$$

The elements of A_0 are of *level* 0, the elements of $A_m \setminus A_{m-1}$ of *level* m. As a notational convention, we use upper case letters $S, \ldots, B_4$ for the symbols of level m, lower case letters $s, \ldots, b_4$ for the corresponding symbols of level $m - 1$.

The Thue system $\mathcal{P}_m$ is defined as the system whose proper productions are those of $\mathcal{P}_{m-1}$ and the following equivalences:

$$\begin{aligned} S &\equiv sc_1 && (a) \\ fc_1b_1 &\equiv sc_2 && (b) \\ fc_2 &\equiv fc_3 && (c) \end{aligned}$$

$$\begin{array}{rlll} & sc_3b_1 & \equiv sc_2b_4 & \textit{(d)} \\ & sc_3 & \equiv fc_4b_4 & \textit{(e)} \\ & sc_4 & \equiv F & \textit{(f)} \\ \text{and for } i \in I_4, & C_ifb_2 & \equiv C_iB_ifb_3 & \textit{(g)–(j)} \end{array}$$

The definition of $\mathcal{P}_m$ is identical with that of Mayr and Meyer, except that for every level greater than 0, they include symbols $Q_1, \ldots, Q_4$. These symbols have been omitted because (as will be clear subsequently) they are redundant.

The Thue system $\mathcal{P}_n$ is designed to generate e_n B_i symbols from a single symbol C_i. This is accomplished by repeated squaring; each level represents a set of rules which squares the number of symbols given it as input from the previous level. Since $e_{n+1} = (e_n)^2$, this results in a set of rules of size $O(n)$ which generates e_n symbols.

Lemma 6. *Let S, F, C_i, B_i, for $i \in I_4$ be of level n. Then*

$$SC_i \overset{*}{\equiv} FC_iB_i^{e_n}\ (\mathcal{P}_n), \text{ for } i \in I_4.$$

Proof. The lemma can be proved in a straightforward way by induction on n (for a full proof, see [15, Lemma 6]). □

7. Semi-Thue Systems from Counter Machines

In the present section, we give the basic construction on which the lower bound is based. Let C = (Q, δ) be a 3-counter machine, where $n = |Q|$ is the size of C. In defining the semi-Thue system $\mathcal{S}_C$associated with C, we shall leave the commuting and contracting rules tacit, giving only the proper rules of $\mathcal{S}_C$explicitly.

The alphabet of $\mathcal{S}_C$consists of the alphabet A_n of $\mathcal{P}_n$, together with the following added symbols:

(a) Symbols h_1, h_2, h_3 (h_1, h_2 and h_3 serve as tokens indicating the contents of the counters in the machine);

(b) For $q \in Q$, a symbol q in the semi-Thue alphabet;

(c) For $q \in Q$ with $\delta(q) = (q', q'', k) \in Q \times Q \times I_3$, two additional symbols q_r and q_e;

(d) For the initial state q_0, three additional symbols q_{02}, q_{03}, q_{04};

(e) For the accepting state q_a, three additional symbols q_{a2}, q_{a3}, q_{a4}.

All of the symbols of $\mathcal{S}_C$which are added to $\mathcal{P}_n$ are defined to be of level n. The symbols introduced by clauses 2 to 5 will be referred to as q *symbols*. In clause 2, we identify the symbols q with the corresponding states of the machine, so that we can write $q \in Q$, for example. In the last three cases, where new q symbols are added to the alphabet of $\mathcal{S}_C$, we associate the new q symbols with states in Q as follows: the symbol q_r has associated with it the state q, the symbol q_e the state q', the symbols q_{0i} the state q_0, and the symbols q_{ai} the state q_a.

The semi-Thue system $\mathcal{S}_C$contains the following proper productions:

1. All equivalences of $\mathcal{P}_n$;
2. For $q \in Q$, with $\delta(q) = (q', d, k) \in Q \times \{0, \pm 1\} \times I_3$:
   - (a) $q \longrightarrow q'$ if $d = 0$
   - (b) $qB_k \longrightarrow q'h_k$ if $d = 1$
   - (c) $qh_k \longrightarrow q'B_k$ if $d = -1$
3. For $q \in Q$ with $\delta(q) = (q', q'', k) \in Q \times Q \times I_3$:
   - (a) $qh_k \longrightarrow q''h_k$
   - (b) $q \longrightarrow q_rFC_k$
   - (c) $q_rSC_k \longrightarrow q_eSC_k$
   - (d) $q_eFC_k \longrightarrow q'$
4. For q_0, the initial state of C,
   - (a) $q_{02}FC_1 \longrightarrow q_{03}SC_2$
   - (b) $q_{03}FC_2 \longrightarrow q_{04}SC_3$
   - (c) $q_{04}FC_3 \longrightarrow q_0$
5. For q_a, the accepting state of C,
   - (a) $q_a \longrightarrow q_{a4}FC_3$
   - (b) $q_{a4}SC_3 \longrightarrow q_{a3}FC_2$
   - (c) $q_{a3}SC_2 \longrightarrow q_{a2}FC_1$.

The symbols belonging to $\mathcal{P}_n$ in the foregoing productions are all of level n. This completes the description of $\mathcal{S}_C$.

8. Basic Lemmas

The present section contains some unavoidable calculations. The purpose of these calculations is simple: to ensure that the productions corresponding to the zero-test capability of the 3-counter machine C work as intended, and that the number of occurrences of h_k, $k \in I_3$, never exceeds e_n. To accomplish this, it is necessary to define a number of numerical invariants.

So far, we have followed the development of Mayr and Meyer in [15] more or less literally. However, in proving the basic lemmas, we are compelled to diverge from their approach. Lemmas 7 and 8 of [15], which correspond to the lemmas of the present section, are syntactical in nature. However, this syntactical approach is not easy to use in the presence of the contraction rule.

To gain control over the problem, we need to define functions which assign values to the symbols in the words of $\mathcal{S}_C$. Symbols of different levels are assigned different values, because a single symbol of a higher level can be converted by the rules into a very large number of symbols of a lower level. We shall use the notation $\Phi(\alpha, \sigma)$ for the number of occurrences of symbol σ in the word α.

First, we define a function V^i_m on words of $\mathcal{S}_C$, where $-1 \leq m \leq n$ and $i \in I_4$. The function is defined by induction on m. $V^i_{-1}(\alpha) = 0$ for all $i \in I_4$. For $m \geq 0$, if the level m symbol $C_i \notin \alpha$ then $V^i_m(\alpha) = \Phi(\alpha, B_i)$; if the level m symbol $C_i \in \alpha$ then

$$V^i_m(\alpha) = \Phi(\alpha, B_i) + e_m \cdot \Phi(\alpha, S) + e_{m-1} \cdot V^1_{m-1}(\alpha) + V^2_{m-1}(\alpha),$$

where B_i and S are of level m.

Define the *height* $h(\alpha)$ of a word α of $\mathcal{S}_C$as:

$$h(\alpha) = min\{m \in N : \Phi(\alpha, \sigma) > 0 \text{ for some } \sigma \text{ of level } m\}.$$

A word α of $\mathcal{S}_C$is defined to be a *machine word* if it contains exactly one q symbol, which belongs to Q, $h(\alpha) = n$, and α contains no occurrences of F, S or C_i, $i \in I_4$. A word α of $\mathcal{S}_C$is defined to be *regular* if it contains exactly one q symbol, and is either a machine word, or satisfies the conditions:

(1) The q symbol is not in Q;

(2) $\sum_{i=1}^4 \Phi(\alpha, C_i) = \begin{cases} 1 & \text{if } C_1, \ldots, C_4 \text{ are of level } m, \\ & h(\alpha) \leq m \leq n, \\ 0 & \text{otherwise;} \end{cases}$

(3) $\Phi(\alpha, s) + \Phi(\alpha, f) = \begin{cases} 1 & \text{if } s, f \text{ are of level } h(\alpha) \\ 0 & \text{otherwise;} \end{cases}$

(4) For $m < n$,

$$e_m \cdot V^1_m(\alpha) + V^2_m(\alpha) + V^3_m(\alpha) + e_m \cdot V^4_m(\alpha) \leq e_{m+1}.$$

Lemma 7. *If* $\alpha \overset{*}{\Longrightarrow} \beta$ *(*$\mathcal{S}_C$*), and* α *is regular, then* β *is regular.*

Proof. We begin by considering the equivalences in $\mathcal{P}_n$. If α is a machine word, then none of the productions in $\mathcal{P}_n$ can be applied, so we may assume that α satisfies the conditions (1) to (4). Since the rules of $\mathcal{P}_n$ leave the q symbol invariant, condition (1) is preserved.

In proving invariance for conditions (2) to (4), we shall consider the application of a production to a regular word α, which contains a symbol C_i of height $k = h(\alpha)$; the result of the production is β. If the production applied belongs to $\mathcal{P}_m$, and $V^i_m(\alpha) = V^i_m(\beta)$, for $i \in I_4$, then $V^i_j(\alpha) = V^i_j(\beta)$ for all j, $m \leq j \leq n$, since the production does not alter symbols of level above m.

For the rules in $\mathcal{P}_0$, it is easy to see that $V^i_0(\alpha) = V^i_0(\beta)$, $i \in I_4$, hence $V^i_m(\alpha) = V^i_m(\beta)$, for all m, $0 \leq m \leq n$.

We now consider the case where β is derived from α by a rule in $\mathcal{P}_m$, where $m > 0$.

If the rule applied is (a) $S \longrightarrow sc_1$, then by (3), α contains no symbols of level less than k, so that $V^1_{k-1}(\beta) = e_{k-1}$, hence $V^i_k(\alpha) = V^i_k(\beta)$, so that $V^j_m(\alpha) = V^j_m(\beta)$, for all $j \in I_4$, $m \geq k$. Hence, the conditions (2) – (4) are preserved.

If the rule applied is the converse production $sc_1 \longrightarrow S$, then $V^1_k(\alpha) \geq e_k$, hence by (4), $V^2_k(\alpha) = V^3_k(\alpha) = V^4_k(\alpha) = 0$. Thus, s and c_1 are the only symbols of level $h(\alpha)$, showing that (2) and (3) are preserved. Since $V^i_{k+1}(\alpha) = V^i_{k+1}(\beta)$, condition (4) is also preserved.

If the rule applied is one of the equivalences (b) – (e) or (g–j) then the level is invariant, and it is clear that conditions (2) and (3) are preserved. Since the rules in question alter only symbols of level $h(\alpha)$ or $h(\alpha) + 1$, it is sufficient to show that in each case the sum

$$e_j \cdot V^1_j(\alpha) + V^2_j(\alpha) + V^3_j(\alpha) + e_j \cdot V^4_j(\alpha)$$

for $j = k, k+1$ is left fixed by these productions. This verification is an easy exercise, and is omitted.

If the production applied is (f) $sc_4 \longrightarrow F$, then, arguing as for the rule (a), the left-hand side of the production contains the only symbols of level $h(\alpha)$, so (2) and (3) are preserved, and clearly (4) is also preserved. If the production is the converse $F \longrightarrow sc_4$, then (2) and (3) are preserved, since F must be of level $h(\alpha)$, and condition (4) is also preserved. This completes the proof for the rules of $\mathcal{P}_n$.

In the case of the remaining productions 2(a) – 5(a), α must be either a machine word, or a word of level n, and it is easy to see from the form of the rules that regularity is preserved. □

LEMMA 8. *If α is regular and $\alpha \overset{*}{\Longrightarrow} \beta$ $(\mathcal{P}_m)$, then $V^i_m(\alpha) = V^i_m(\beta)$, for $i \in I_4, m \leq n$.*

Proof. The rules of $\mathcal{P}_m$ alter only symbols of level m or lower, so that to prove the conclusion of the lemma it is sufficient to show that $V^i_m(\alpha) = V^i_m(\beta)$, for all $i \in I_4$. The lemma is proved by induction on m.

For $m = 0$, the lemma follows immediately by the definition of V^i_0. Let $m > 0$. By induction hypothesis, we may assume that the production applied belongs to $\mathcal{P}_m \setminus \mathcal{P}_{m-1}$. If the production applied is (a) $S \longrightarrow sc_1$, then since α is regular, $h(\alpha) = m$. The equality $V^i_m(\alpha) = V^i_m(\beta)$ follows by definition. If the converse production $sc_1 \longrightarrow S$ is used, then by Lemma 7, β is regular, hence $h(\beta) = m$, so the equality again follows. If the production used is (f), then the word containing F has height m, and the value of V^i_m is not altered by the application of the production.

In the remaining cases, $h(\alpha) = h(\beta) = m - 1$, by regularity; the equation $V^i_m(\alpha) = V^i_m(\beta)$ is readily checked from the definition. □

We now define functions W^i, for $i \in I_3$:

$W^1(\alpha) = V^1_n(\alpha) + \Phi(\alpha, h_1)$,
$W^2(\alpha) = V^2_n(\alpha) + \Phi(\alpha, h_2) + e_n(\Phi(\alpha, q_{02}) + \Phi(\alpha, q_{a2}))$,
$W^3(\alpha) = V^3_n(\alpha) + \Phi(\alpha, h_3) + e_n(\Phi(\alpha, q_{02}) + \Phi(\alpha, q_{03}) + \Phi(\alpha, q_{a2}) + \Phi(\alpha, q_{a3}))$.

Lemma 9. *If $\alpha \overset{*}{\Longrightarrow} \beta$ $(\mathcal{S}_C)$, and α is regular, then $W^i(\alpha) \geq W^i(\beta)$, for $i \in I_3$.*

Proof. For the rules of $\mathcal{P}_n$, this follows from the previous lemma. For the rules which correspond to the rules of the counter machine C, the lemma is easy to check. Finally, since the value of W^i is monotone in the number of occurrences of symbols in a word, it is clear that the contraction rule can never increase the value $W^i(\alpha)$. □

Let α be a regular word in $\mathcal{S}_C$. Define the *state associated with* α, $state(\alpha)$, to be the state associated with the q symbol occurring in α. We associate with α an instantaneous description $D(\alpha)$, defined as follows:

$$D(\alpha) = (state(\alpha), \Phi(\alpha, h_1), \Phi(\alpha, h_2), \Phi(\alpha, h_3)).$$

Lemma 10. *If $q_{02}SC_1 = \alpha_0 \Rightarrow \cdots \Rightarrow \alpha_k = q_{a2}SC_1$ is a derivation in $\mathcal{S}_C$:*

(1) *For every i, $0 \leq i < k$, either $D(\alpha_i) = D(\alpha_{i+1})$ or $D(\alpha_{i+1})$ is derived from $D(\alpha_i)$ by a transition of C;*

(2) *$\Phi(\alpha_i, h_j) \leq e_n$ for all $j \in I_3$.*

Proof. First, observe that for $i \in I_3$, $W^i(q_{02}SC_1) = W^i(q_{a2}SC_1) = e_n$. It follows by Lemma 9 that for $i \in I_3$, $0 \leq j \leq k$, $W^i(\alpha_j) = e_n$, so that (2) follows immediately.

It is not hard to see that (1) holds for the rules of $\mathcal{S}_C$, with the possible exception of the productions corresponding to the zero-test capability of the counter machine, that is, the four productions 3(a) to 3(d) in the definition of $\mathcal{S}_C$. The crucial case is the production 3(c). If $\alpha_i \Rightarrow \alpha_{i+1}$ by this production, then $V_n^k(\alpha_i) = V_n^k(\alpha_{i+1}) = e_n$, hence $\Phi(\alpha_i, h_k) = \Phi(\alpha_{i+1}, h_k) = 0$. Thus condition (1) holds in this case. □

Theorem 1. *For a 3-counter machine C,*

$$C \in ESC \textit{ if and only if } q_{02}SC_1 \overset{*}{\Longrightarrow} q_{a2}SC_1 \ (\mathcal{S}_C)$$

Proof. The implication holds left to right by the construction of $\mathcal{S}_C$, as can be easily verified. The implication holds right to left by Lemma 10. □

Theorem 2. (1) *The decision problem for $R_{\rightarrow\wedge\circ}$ is exponential space hard with respect to log-lin reducibility;* (2) *There is a constant $c > 1$ so that any Turing machine solving the decision problem for $R_{\rightarrow\wedge\circ}$ must use space c^n for infinitely many inputs of length n.*

Proof. The construction of the formula $\Psi(\mathcal{S}_C, q_{02}SC_1, q_{a2}SC_1)$ can be carried out in log-space, and its size is $O(size(C))$. The first part of the theorem now follows immediately from Lemma 5, Lemma 3 and Theorem 1. The second part follows from Lemma 4. □

The preceding theorem shows that the decision problem for $R_{\rightarrow\wedge\circ}$ is very intractable. This intractability remains even if unlimited parallel processing is available as a computational resource. Let us represent the characteristic function of the set of theorems of $R_{\rightarrow\wedge\circ}$ as a family of Boolean functions, by encoding the formulas of $R_{\rightarrow\wedge\circ}$ as binary strings, and defining f_n to be the characteristic function of the set of theorems restricted to formulas having encodings of length n. Then a simple diagonal argument (see Wegener [23, pp. 140–141]) establishes that for infinitely many n, the number of logic gates in the smallest circuit computing f_n is bounded below by a function which is exponential in n.

9. OTHER LOGICS

In proving the lower bound, we made use of the constant t and the fusion operator $\circ$. However, the lower bound proved above also holds for the system $R_{\rightarrow\wedge}$. This system contains only the implication and conjunction connectives, and is axiomatized by omitting the axiom schemes 9 and 10 from $R_{\rightarrow\wedge\circ}$, and replacing the first two axiom schemes by the scheme $A \rightarrow A$.

To extend the lower bound to $R_{\rightarrow\wedge}$, we need to define an embedding of $R_{\rightarrow\wedge\circ}$ into $R_{\rightarrow\wedge}$ which reduces the first decision problem to the second. More precisely, we shall define a translation from the formulas of $R_{\rightarrow\wedge\circ}$ into the formulas of $R_{\rightarrow\wedge}$ which constitutes a log-lin reduction of the decision problem for $R_{\rightarrow\wedge\circ}$ to the decision problem for $R_{\rightarrow\wedge}$.

Let A be a formula of $R_{\rightarrow\wedge\circ}$, and let P_t and P_f be propositional variables not occurring in A. Define $\neg B$ as $(B \rightarrow P_f)$ and $(B \circ' C)$ as $\neg(B \rightarrow \neg C)$. Let A' be the result of replacing t by P_t and the operator $\circ$ by the defined operator $\circ'$. Then define $Trans(A)$ to be:

$$\bigwedge\{(\neg\neg P \rightarrow P) \wedge (P_t \rightarrow .P \rightarrow P) : P \text{ a variable in } A'\} \wedge P_t. \rightarrow .A'.$$

LEMMA 11. *If A is a formula of $R_{\rightarrow\wedge\circ}$, then $R_{\rightarrow\wedge\circ}\vdash A$ if and only if $R_{\rightarrow\wedge}\vdash Trans(A)$.*

Proof. Assume that $R_{\rightarrow\wedge\circ}\vdash A$. Then A has a derivation in $R_{\rightarrow\wedge\circ}$ that contains only variables which occur in A. We can show by induction on the length of this derivation that if B is a formula in the derivation, then $Trans(B)$ is provable in $R_{\rightarrow\wedge}$. This is a straightforward exercise in the axiomatics of relevance logics; the details are left to the reader.

Conversely, assume that $R_{\rightarrow\wedge}\vdash Trans(A)$. Then $Trans(A)$ is provable in the full system R, with $\circ$, t and f as primitive symbols. Consider the formula which results from $Trans(A)$ by substituting t for P_t and f for P_f throughout. The antecedent of this formula is provable in R, hence the consequent is also provable in R. But the consequent is provably equivalent to A in R, since $(B \circ C)$ and $(B \rightarrow .C \rightarrow f) \rightarrow f$ are equivalent in R. Hence A is provable in R, and so $R_{\rightarrow\wedge\circ}\vdash A$, since R is a conservative extension of $R_{\rightarrow\wedge\circ}$. □

With a little more work, we can extend the lower bound to $R_{\rightarrow}$, the pure implicational fragment of R. The only occurrences of conjunction in the formula $\Psi(S, \gamma, \delta)$ defined above are those displayed in the antecedent. If F is a set of formulas of $R_{\rightarrow}$, then $\bigwedge F \rightarrow A$ is provable in $R_{\rightarrow\wedge}$if and only if a formula of the form $A_1 \rightarrow .A_2 \rightarrow \cdots A_k \rightarrow A$ is provable in $R_{\rightarrow}$, where $\{A_1, \cdots, A_k\} \subseteq F$. Using this observation, we can reduce the problem ESC to the decision problem for $R_{\rightarrow}$. In this case, the reduction does not belong to log-space, because there are exponentially many formulas to check for provability in general. However, the formulas are all of size $O(C)$, where C is the given 3-counter machine, so that the lower bound extends to $R_{\rightarrow}$as well.

It is possible by means of appropriate translations to extend the lower bound to fragments of other relevance logics. R. K. Meyer and S. Giambrone [17] define an efficient embedding from the positive fragment of R into the positive fragment of the system of ticket entailment T ([4, p. 41f]). An examination of the embedding shows it to be a log-lin reduction of the decision problem for $R_{\rightarrow\wedge}$to the decision problem for the implication-conjunction fragment of T. Since this constitutes a reduction to a proper subsystem of $R_{\rightarrow\wedge}$, we can state a general result.

THEOREM 3. *The decision problem for the implication-conjunction fragment of any logic intermediate between T and R is exponential space hard under log-lin reducibility.*

10. THE STRICT λ-CALCULUS

It is well known that valid schemes of implicational logic can be identified with the types of certain combinators ([7, pp. 312–315]). We can use this correspondence to transfer the lower bound to the appropriate form of the λ-calculus.

We consider the typed λI-calculus, formulated with infinitely many ground types $O_1, \cdots, O_n, \cdots$ (see Hindley and Seldin [9] for details). The restriction to λI-terms means that a term $\lambda x.F$ is well formed only if F contains a free occurrence of the variable x. If we correlate the variables of $R_{\rightarrow}$bijectively with the ground types, then any implicational formula can be identified with a type, by interpreting $(A \rightarrow B)$ as the family of all functions from A into B.

THEOREM 4. *The problem of determining of an arbitrary type whether it is the type of a closed term of the λI-calculus is exponential space hard.*

Proof. It follows by the correspondence described by Curry and Feys [7] (see also Howard [11]) that for any type A, there is a closed term of type A if and only if A is provable in $R_{\rightarrow}$. □

This result may be compared with the corresponding problem for the typed λ-calculus; this decision problem is polynomial-space complete (Statman [20]).

11. AN UPPER BOUND

In this final section, we discuss briefly the problem of providing an upper bound on the complexity of the decision problem for $R_{\rightarrow\wedge\circ}$. The large gap between the upper and lower bounds is evidence of how far we are from a real understanding of the true complexity of these problems.

Ackermann's function is the function defined by Ackermann in 1928 [1] which eventually dominates any primitive recursive function.

THEOREM 5. *Kripke's decision procedure for $R_{\rightarrow\wedge\circ}$is primitive recursive in the Ackermann function.*

Proof. The decision procedure for $R_{\rightarrow\wedge\circ}$is described in [21, pp. 30–39](see also [4, pp. 136–139]). The fundamental strategy is to construct a proof search tree for a Gentzen-style formulation of $R_{\rightarrow\wedge\circ}$, and then to show that in each case, the tree is finite.

The key lemma of Kripke which shows the tree to be finite ([4, pp. 138–139]) is essentially the same as the argument used in 1968 by Karp and Miller [12] to show that the reachability tree of a vector addition system (equivalently, Petri net or commutative semi-Thue system) is finite. McAloon in [14] analyses the lemma using non-standard models of arithmetic, and thereby provides an upper bound on the size of the reachability tree of a vector addition system. This bound is primitive recursive in the Ackermann function.

Because of the essential identity of the Kripke lemma and the Karp/Miller lemma, McAloon's arguments adapt directly to show that the size of the proof search tree is also primitive recursive in the Ackermann function. The procedure which consists in examining the proof search tree to see if it contains a proof of the given formula is primitive recursive in the size of the tree, so the bound holds for the entire procedure. □

Although the lower bound proved above certainly shows the decision problem for $R_{\rightarrow\wedge\circ}$to be very intractable, the huge gap between upper and lower bounds is far from satisfactory. Which bound is closer to the truth?

Kripke in a letter of 1981 to Michael McRobbie is quoted ([21, p. 40]) as conjecturing that the decidability proof is unprovable in elementary recursive arithmetic. If this is so, then the upper bound may be closer to the truth than the lower bound; this seems quite appropriate, since it was Ackermann who not only invented the famous function, but also in his basic 1956 paper [2] founded the theory of entailment.

ACKNOWLEDGMENTS

I first discussed these problems in 1982 with Paul Thistlewaite, to whom I wish to express my thanks for his very stimulating conversation. It was Alan Ross Anderson and Nuel Belnap who taught me how to do logic. Thank you, Alan! Thank you, Nuel!

BIBLIOGRAPHY

[1] Ackermann, W. (1928): 'Zum Hilbertschen Aufbau der reellen Zahlen', *Mathematische Annalen* 99, pp. 118–133.

[2] Ackermann, W. (1956): 'Begründung einer strengen Implikation', *Journal of Symbolic Logic* 21, pp. 113–128.

[3] Anderson, A.R. and Belnap, N.D. Jr. (1958): 'A modication of Ackermann's 'rigorous implication.' *The journal of symbolic logic* 23, pp. 457–458. Abstract.

[4] Anderson, A.R. and Belnap, N.D. Jr. (1975): *Entailment: The Logic of Relevance and Necessity, Vol. 1*. Princeton University Press, New Jersey.

[5] Belnap, N.D. Jr. and Wallace, J.R. (1965): 'A Decision Procedure for the System $\mathbf{E}_{\mathrm{I}}$ of Entailment with Negation' *Zeitschrift für mathematische Logik und Grundlagen der Mathematik* 11, pp. 277–289.

[6] Church, A. (1956): *Introduction to Mathematical Logic*. Princeton University Press, New Jersey.

[7] Curry, H.B. and Feys, R. (1968): *Combinatory Logic, Vol. 1*. North-Holland Publishing Company, Amsterdam.

[8] Fischer, P.C., Meyer, A.R., Rosenberg, A.L. (1968): 'Counter Machines and Counter Languages', *Mathematical Systems Theory* 2, pp. 265–283.

[9] Hindley, J.R. and Seldin, J.P. (1986): *Introduction to Combinators and λ-calculus*. Cambridge University Press, Cambridge.

[10] Hopcroft, J.E. and Ullmann, J.D. (1979): *Introduction to Automata Theory, Languages and Computation* Addison-Wesley Publishing Company, Reading, Massachusetts.

[11] Howard, W. 'The formulae-as-types notion of construction' in *To H. B. Curry, Essays on Combinatory Logic, Lambda Calculus and Formalism*, ed. by Hindley and Seldin, Academic Press, New York and London, pp. 479–490.

[12] Karp, R.M. and Miller, R. E. (1969): 'Parallel program schemata' *Journal of Computer and System Sciences* 3, pp. 147–195.

[13] Kripke, S. A. (1959): 'The Problem of Entailment', *Journal of Symbolic Logic* 24, p. 324. Abstract.

[14] McAloon, K. (1984): 'Petri nets and large finite sets', *Theoretical Computer Science* 32, pp. 173–183.

[15] Mayr, E. and Meyer, A. (1982): 'The Complexity of the Word Problems for Commutative Semigroups and Polynomial Ideals', *Advances in Mathematics* 46, pp. 305–329.

[16] Meyer, R.K. (1966): *Topics in Modal and Many-Valued Logic*. PhD thesis, University of Pittsburgh, Pennsylvania.

[17] Meyer, R.K. and Giambrone, S. (1980): 'R_+ is contained in T_+', *Bulletin of the Section of Logic, Polish Academy of Sciences* 9 (March 1980), pp. 30–32.

[18] Meyer, R.K. and Routley, R. (1973): 'An Undecidable Relevant Logic' *Zeitschrift für mathematische Logik und Grundlagen der Mathematik* 19, pp. 389–397.

[19] Routley, R. and Meyer, R.K. (1973): 'The Semantics of Entailment I', in Leblanc, H. (editor), *Truth, Syntax and Modality*. North Holland, Amsterdam.

[20] Statman, R. (1979): 'Intuitionistic propositional logic is polynomial-space complete' *Theoretical Computer Science* 9, pp. 67–72.

[21] Thistlewaite, P.B., McRobbie, M.A. and Meyer, R.K. (1988): *Automated Theorem-Proving in Non-Classical Logics*. Pitman, London; Wiley, New York and Toronto.

[22] Urquhart, A. (1984): 'The undecidability of entailment and relevant implication', *Journal of Symbolic Logic* 49, pp. 1059–1073.

[23] Wegener, I. (1987): *The Complexity of Boolean Functions*. B.G. Teubner, Stuttgart; John Wiley and Sons, Chichester.