# $\mathcal{P}$ is not equal to $\mathcal{NP}$.

Sten-Åke Tärnlund*†‡

November 23, 2018

**Abstract**

*SAT* $\notin \mathcal{P}$ is true, and provable in a simply consistent extension $\mathbf{B}'$ of a first order theory $\mathbf{B}$ of computing, with a single finite axiom $B$ characterizing a universal Turing machine. Therefore $\mathcal{P} \neq \mathcal{NP}$ is true, and provable in a simply consistent extension $\mathbf{B}''$ of $\mathbf{B}$.

## 1 Introduction

*SAT* $\notin \mathcal{P}$ is true, theorem 1, and provable, corollary 9, in a simply consistent extension $\mathbf{B}'$ of a first order theory $\mathbf{B}$ of computing, with a single finite axiom $B$ characterizing a universal Turing machine.[1] Therefore, by the Cook-Levin theorem,[2] $\mathcal{P} \neq \mathcal{NP}$ is true, theorem 2, and provable, corollary 10,[3] in a simply consistent extension $\mathbf{B}''$ of theory $\mathbf{B}$.

*SAT* is the set of satisfiable propositional formulas. $\mathcal{P}$ and $\mathcal{NP}$ are sets of classes of problems of polynomial time complexity for deterministic and nondeterministic Turing machines respectively.[4]

The chief idea of the proof of theorem 1 is a relationship between computing time,[5] and proof complexity,[6] in fact, the principal lemma 3, which essentially is developed from axiom $B$. First, there is a relationship between a formal proof in theory $\mathbf{B}$, and a formula in propositional logic.[7] However, lemma 3 gives more i.e., the number of tapehead moves of a deterministic Turing machine computing unsatisfiability of $\neg F$ expressed as a simple formula in propositional logic, from

*gmail name: stenake.



‡These results were presented August 6, 2008 at a seminar in the Department of Information Science at Uppsala University Sweden, www. uu. se.

[1] Following Tärnlund 2008 [24], cf. Turing 1936 [25] and Kleene 1967 [14].

[2] *SAT* $\in \mathcal{P} \equiv \mathcal{P} = \mathcal{NP}$, Cook 1971 [4] and Levin 1973 [16].

[3] $\mathcal{P}$ vs $\mathcal{NP}$ cf. Smale 1998 [23], Cook 2003 [5], and Cook *www.claymath.org/millennium.*

[4] cf. Karp 1972 [13], and for a textbook cf. Sipser 2005 [22].

[5] The number of tapehead moves of a Turing machine with an input computing an output. Historically, Hartmanis and Stearns 1965 [10], and for a textbook cf. Sipser 2005 [22]. In theory $\mathbf{B}$, definition 6.

[6] The size of a formal deduction (proof) in propositional logic. Historically, Cook and Reckhow 1979 [6]. In theory $\mathbf{B}$, definition 21.

[7] cf. Gentzen's Hauptsatz 1935 [8], Herbrand's theorem 1930 [11], and Kleene 1967 [14].

which the tautology $F$ is deducible. More precisely: if there is a deterministic Turing machine $i \in \mathcal{U}$ [8] that computes unsatisfiability of $\neg F$ in computing time $n$ polynomial in the size of $F$ then $Y(i, F, n)$, for a sufficiently large tautology $F$.[36]

More to the point, by definition 20, $Y(i, F, n)$ is equivalent to:

$$(Q_0 \supset F) \wedge Q_n \wedge \bigwedge_{1 \leq k \leq n} (Q_k \supset Q_{k-1}) \quad \text{some single letters } Q_0, \ldots, Q_n \quad (1)$$

In propositional logic there is a formal deduction of the tautology $F$ in polynomial time in the size of $F$ if $Y(i, F, n)$ is true. There is now an indirect proof of theorem 1.

Assume that

$$SAT \in \mathcal{P}. \quad (2)$$

By lemma 3, and a sufficiently large pigeonhole formula $PF_m$,[9]

$$Y(i, PF_m, n) \text{ and } n \leq c \cdot |PF_m|^q \text{ some } c\, q \in N \; i \in \mathcal{U} \text{ a sufficiently large } m. \quad (3)$$

Thus, by (1),

$$(Q_0 \supset PF_m) \wedge Q_n \wedge \bigwedge_{1 \leq k \leq n} (Q_k \supset Q_{k-1}) \text{ some single letters } Q_0, \ldots, Q_n. \quad (4)$$

By Robinson's resolution,[10] and (3) - (4), no more than $c \cdot |PF_m|^q$ resolvents are constructed in the deduction of $PF_m$ from (4), and the size of the formal deduction is polynomial in the size of $PF_m$. Therefore

$$\vdash_R PF_m, \text{ with a polynomial size of the deduction in the size of } PF_m. \quad (5)$$

Hence, a contradiction between (5), Haken's theorem,[11] [12] and the fact that theory **B** is simply consistent i.e., has no contradiction, for the subset of the deterministic Turing machines that compute satisfiability or unsatisfiability of propositional formulas,[8] corollary 2.[13] [14]

[8] The set $\mathcal{U}$ is defined in definition 14.

[9] A propositional pigeonhole formula, $PF_n$ : $n$ pigeons in $n+1$ holes have an empty hole, is a tautology $\models PF_n$ any $n \in N$, by Cook and Reckhow 1979 [6]. Generally, a pigeonhole principle: there is no injective function with a smaller co-domain than domain for finite sets.

[10] Robinson 1965 [21].

[11] Every resolution proof of $PF_n$ contains at least $c^n$ different clauses for $c > 1$ some $c \in R$ any sufficiently large $n \in N$, Haken 1985 [9].

[12] cf, bounded depth Frege systems, Ajtai 1988 [1], Bellantoni, Pitassi, and Urquhart 1992 [3], Beame, Impagliazzo, Krajíček, Pitassi, Pudlá, and Woods 1992 [2], Pitassi, Beame, and Impagliazzo 1993 [18], Krajíček, Pudlák, and Woods 1995 [15].

[13] Simple consistency, cf. Kleene 1967 [14].

[14] However, simple consistency of theory **B** is not provable for the set of all Turing machines. In contrast, theory **B** is simply consistent if and only if simple consistency of **B** is not provable in **B**, Tärnlund 2008 [24].

Thus

$$SAT \notin \mathcal{P}. \tag{6}$$

It remains to lay down the first order theory **B** of computing, with axiom $B$, and to prove the principal lemma 3. Then a proof of theorem 1 follows in the proof (meta) theory of **B**, as outlined in (2) - (6).

## 2 A theory of computing

Before applying the axiomatic method[15] to computing complexity, a first order theory **B** of computing, with a single finite axiom $B$ characterizing a universal Turing machine, is presented.[1] [16]

Syntactically, there are two predicate symbols of **B** written $T(i, a, u)$, and $U(x, s, z, q, j, i, u)$. In addition, one function symbol $.$ in infix notation $x\,.\,y$.

A Turing machine has a finite supply of arbitrary constant symbols, for example, the alphabetic symbols $A, a, B, b, \ldots$, the natural numbers,[17] and the symbols of propositional logic. For convenience, there is at least a subset

$$\{\emptyset, 0, 1, \sqcup\} \subseteq K, \tag{7}$$

where $K$ is a finite set of constant symbols, and $\sqcup$ a blank symbol.

There are six sets in **B**.

A finite set

$$Q \subset N \text{ of states}, \tag{8}$$

where 0 is the halt state and 1 the start state.

A finite set $S$ of symbols,

$$S \text{ for } \{\, u : u \in K \vee (u = r\,.\,\emptyset \vee u = \emptyset\,.\,r) \wedge r \in K\}. \tag{9}$$

A set $D$ of moves of the tapehead of a Turing machine,

$$D \text{ for } \{0, 1\}, \tag{10}$$

where 0 is a move to the left and 1 a move to the right.

There is a finite arbitrary large two-way tape, with a left and right tape having an element between them at the tapehead.[18] Initially, the two-way tape has an empty left tape, the input on the right tape, and the element between them has the symbol $\emptyset$. When a computation starts the tapehead reads the

[15] Hilbert and Bernays 1934 [12], cf. Kleene 1967 [14].

[16] The postulates of predicate calculus are employed in theory **B**. Frequently, $G4$ of Kleene 1967 [14], a Gentzen-type system 1934-5 [8] is used. The notation can simply be changed to a Hilbert-type system by Gentzen's theorem Kleene 1967 [14], and to a resolution system by Robinson's theorem 1965 [21]. Similarly, for systems in Quine 1974 § 37 [20].

[17] Writing $N$ for the set of the natural numbers.

[18] Turing 1936 [25] has a one-way tape, for a two-way tape cf. Post 1947 [19].

symbol $\emptyset$. The arbitrary long but finite two-way tape is represented as two lists.[19]

The list on the right tape grows to the right, if the symbol $r \,.\, \emptyset \in S$ substitutes $\emptyset$. The list on the left tape grows to the left, if the symbol $\emptyset \,.\, r \in S$ substitutes $\emptyset$. Therefore the size of the two-way tape grows one element at the time controlled by the Turing machine.

There are two sets $L$ and $L'$ of lists, representing the right tape, and left tape respectively. These sets are as follows.

First, [20] [21]

$$L(\emptyset) \wedge \forall\, (s \in S \wedge L(z) \supset L(s \,.\, z)), \text{ and} \tag{11}$$

$$L'(\emptyset) \wedge \forall\, (s \in S \wedge L'(z) \supset L'(z \,.\, s)) \tag{12}$$

Then,

$$L \text{ for } \{u :\ L(u)\,\}. \tag{13}$$

$$L' \text{ for } \{u :\ L'(u)\,\}. \tag{14}$$

A set $M \subset L$ of codes of Turing machines.[22] But, first the code of a Turing machine,

$$M(\emptyset) \wedge \forall\, (p\ q \in Q \wedge\ r\ s \in S \wedge d \in D \wedge M(z) \supset M(p \,.\, s \,.\, q \,.\, r \,.\, d \,.\, z)). \tag{15}$$

Then the set of codes of Turing machines,

$$M \text{ for } \{u :\ M(u)\,\}. \tag{16}$$

The formulas of theory **B** are as follows, first the propositional atoms.

**Definition 1** *The set PROP of atomic propositional formulas of theory* **B** *has the elements* $U(x, s, z, q, j, i, u)$*, and* $T(i, a, u)$ $i\ j \in M\ \ a\ \ z \in L\ \ x\ \ u \in L'\ \ q \in Q\ \ s \in S$*.*

Then, with a countable infinity of variables in theory **B**, the atomic formulas of theory **B**.

**Definition 2** *The set ATOM of atomic formulas of theory* **B** *has as members the elements in PROP, and an atomic formula constructed by substituting a variable for a term in an atomic propositional formula in PROP.*

Finally, all the formulas of theory **B** are next.

[19] Historically, Turing 1936 [25] and Kleene 1967 [14] have a potentially infinite tape. In contrast, Davis 1958 [7] and Minsky 1967 [17] grow the arbitrary large finite tape in the computation.

[20] Operators: $\supset, \equiv, \vee, \wedge, \neg, =, \leq, \forall, \exists, \vdash, \models, \rightarrow$, are ranked decreasingly to get simpler expressions. Thus, $\vdash B \rightarrow P \wedge F$ shall mean $(\vdash (B \rightarrow P)) \wedge F$. Moreover, the operators are used autonymously Kleene 1967 [14].

[21] $\forall F$ for a free variable is universally quantified over the entire formula $F$.

[22] The code of a Turing machine is a list of quintuples, cf. Turing 1936 [25].

**Definition 3** *The set of formulas of theory* **B** *has as members the elements in ATOM, and an element constructed by the formation rules of predicate calculus from elements in ATOM.*

Semantically, the infix function symbol and the predicate symbols denote two functions[23] and two relations[24]. Of course, there is an intended interpretation of the function symbol and the predicates in theory **B**.

$T(i, a, u)$ shall mean Turing machine $i$ with input $a$ computes an output $u$ and then halts for $i \in M\ a \in L\ u \in L'$.[25]

$U(x, s, z, q, j, i, u)$ shall mean Turing machine $i$ computes $u$ and then halts, where $x \,.\, s \,.\, z$ is the two-way tape of $i$,[18] $s$ is a symbol (at the tapehead), $x$ is the left tape, $z$ is the right tape, $i$ is in state $q$, and has an auxiliary code $j$, for $i\ j \in M\ s \in S\ z \in L\ q \in Q\ x\ u \in L'$.

For instance, if $T(i, A \,.\, \emptyset, \emptyset \,.\, \emptyset \,.\, A \,.\, B)$ then $U(\emptyset, \emptyset, A \,.\, \emptyset \,.\, \emptyset, 1, i, i, \emptyset \,.\, \emptyset \,.\, A \,.\, B)$ i.e., a Turing machine $i$ computes the output $\emptyset \,.\, \emptyset \,.\, A \,.\, B$. It starts in state 1, reads the symbol $\emptyset$ (by its tapehead), and has the two-way tape $\emptyset, \emptyset, A \,.\, \emptyset \,.\, \emptyset$ initially. Here, the left tape is the empty list $\emptyset$, and the right tape is the list $A \,.\, \emptyset \,.\, \emptyset$, where $A \,.\, \emptyset$ is the input list.[26] Thus, the output has added one element $B$ to the input list.

The axiom, with name $B$, of theory **B** is now presented. Then there is an informal explanation, and some examples are given in the appendix.

**Axiom 1** *B for*

$$\forall\ T(i,\ a,\ u) \supset U(\emptyset,\ \emptyset,\ a \,.\, \emptyset,\ 1,\ i,\ i,\ u) \quad i \in M\ a \in L\ u \in L'. \tag{17}$$

$$\forall\ U(\emptyset,\ \emptyset,\ a \,.\, \emptyset,\ 1,\ i,\ i,\ u) \supset T(i,\ a,\ u) \quad a \in L\ i \in M\ u \in L'. \tag{18}$$

$$\forall\ U(x,\ s,\ z,\ 0,\ i,\ i,\ x) \quad x \in L'\ s \in S\ \ z \in L\ i \in M. \tag{19}$$

$$\forall\ U(x,\ v,\ r \,.\, z,\ p,\ i,\ i,\ u) \supset U(x \,.\, v,\ s,\ z,\ q,\ q \,.\, s \,.\, p \,.\, r \,.\, 0 \,.\, j,\ i,\ u) \tag{20}$$
$$x\ u \in L'\ v\ r\ s \in S\ \ z \in L\ p\ q \in Q\ \ i\ j \in M.$$

$$\forall\ U(x \,.\, r,\ v,\ z,\ p,\ i,\ i,\ u) \supset U(x,\ s,\ v \,.\, z,\ q,\ q \,.\, s \,.\, p \,.\, r \,.\, 1 \,.\, j,\ i,\ u) \tag{21}$$
$$x\ u \in L'\ r\ v\ s \in S\ \ z \in L\ p\ q \in Q\ \ i\ j \in M.$$

$$\forall\ U(x,\ s,\ z,\ q,\ j,\ i,\ u) \supset U(x,\ s,\ z,\ q,\ q' \,.\, s' \,.\, p \,.\, r \,.\, d \,.\, j,\ i,\ u) \tag{22}$$
$$x\ u \in L'\ s\ s'\ r \in S\ \ z \in L\ q\ q'\ p \in Q\ \ j\ i \in M\ d \in D.$$

*Here,* $\emptyset$, 0 *and* 1 *are constant symbols, and* . *a function symbol.*[23]

[23] There are two functions with similar syntax. (i) $.(r, z) \mapsto r \,.\, z$, where $r \in S$ and $z \in L$. (ii) $.(x, r) \mapsto x \,.\, r$, where $r \in S$ and $x \in L'$. They are distinguished by their appearance, function (i) on the right tape and (ii) on the left tape.

[24] $T \subseteq M \times L \times L$, and $U \subseteq L \times S \times L \times Q \times M \times M \times L$.

[25] cf. Kleene p. 243 cf. footnote 167 Kleene 1967 [14].

[26] Note that the input list $A \,.\, \emptyset$ is represented as $A \,.\, \emptyset \,.\, \emptyset$ on the right tape of the two-way tape i.e., a list on a list. Thus, the beginning and end of the input list are marked with the symbol $\emptyset$. The beginning and end of the two-way tape itself are also marked with $\emptyset$.

In sentences (17) - (18), there is an equivalence between a Turing machine $i$ with an input $a$ that computes output $u$, and $i$ (with the concrete representation of $i$ the two-way tape, state, symbol, and auxiliary code) with input $a$ that computes output $u$.

There is a halt condition in sentence (19), thus for state 0 the left tape $x$ is the output of the computation, which then ends.

In sentence (20), if there is a new configuration (the tapehead of a Turing machine $i$ has printed $r$ on the tape, moved to the left, and entered state $p$) then in the previous configuration $i$ is in state $q$ reading symbol $s$ and has the quintuple $q\,.\,s\,.\,p\,.\,r\,.\,0\,.$

Similarly in sentence (21), if there is a new configuration (the tapehead of a Turing machine $i$ has printed $r$ on the tape, moved to the right, and entered state $p$) then in the previous configuration $i$ is in state $q$ reading symbol $s$ and has the quintuple $q\,.\,s\,.\,p\,.\,r\,.\,1\,.$

In sentence (22), a Turing machine $i$ in state $q$ reading symbol $s$ is searching for a quintuple $q\,.\,s\,.\,p\,.\,r\,.\,d\,.$

The deterministic Turing machines are a proper subset of Turing machines.

**Definition 4** $\mathcal{D} = \{i :$ *i has no quintuples* $q\,.\,s\,.\,p\,.\,r\,.\,d$ *and* $q\,.\,s\,.\,p'\,.\,r'\,.\,d'$ *and* $\neg(p = p' \wedge r = r' \wedge d = d')$ *for* $i \in M\}$.

If $\exists u T(i,a,u)$ is true then $\exists u T(i,a,u)$ is provable in theory **B**, by axiom 1 and induction on the number of moves of the tapehead in (20) - (21) in a proof.[27] [28]

**Lemma 1** $\exists u T(i,a,u) \supset \vdash B \rightarrow \exists u T(i,a,u)$ *any* $i \in M$ $a \in L$.

The next definition is justified by the notion of a Turing machine computation, a formal proof in theory **B**, axiom 1, and lemma 1.[29]

**Definition 5** *A Turing machine computation for a formal proof in theory* **B**.

# 3 Computing time

There exists a relation,[30] which computes the computing time[5] of a Turing machine with an input computing an output. Thus there is a formal proof (computation) in theory **B**, on which this relation counts the number of tapehead moves, definition 5.[31] [29]

[27] The quantifiers some and any in the proof (meta) theory of **B** are applied to the entire sentence in front of them.

[28] There is a proof of lemma 1 in Tärnlund 2008 [24].

[29] A formal proof in a first order theory, cf. Kleene 1967 [14]. For instance, a formal deduction (proof) in propositional logic, cf. (45).

[30] $H \subseteq E \times N$. $E$ is the set of proofs in theory **B**.[17]

[31] A move of the tapehead of a Turing machine with an input computing an output is specified in (20) - (21) of axiom 1, thus independent of a system of predicate calculus, cf. footnote 16.

**Definition 6** *$H(\vdash B \rightarrow \exists u T(i,a,u), n)$ for there is a formal proof of the sequent $B \rightarrow \exists u T(i,a,u)$ in G4 with n moves of the tapehead of a Turing machine $i$ with an input $a$ computing an output, any $i \in M$ $a \in L$ $n \in N$.*

A function that exists, and computes the size of an (input) list.[32]

**Definition 7** *$|a|$ for the number of symbols of $a \in L$.*

A polynomial upper bound in the size of the input is written as follows.

**Definition 8** *$p(a)$ for $c \cdot |a|^q$ some $c$ $q \in N$ any $a \in L$.*

Then a polynomial upper bound of the computing time is written next.

**Definition 9** *$\vdash B \rightarrow \exists u T(i,a,u)$ in $p(a)$ for $H(\vdash B \rightarrow \exists u T(i,a,u), n) \wedge n \leq c \cdot |a|^q$ any $i \in M$ some $c$ $q \in N$ any $a \in L$ some $n \in N$.*

**Definition 10** *$\mathcal{F}$ for the set of formulas of propositional logic.*

The set of satisfiable propositional formulas is written $SAT$.

**Definition 11** *$SAT$ for $\{F : F \in \mathcal{F} \wedge \not\models \neg F\}$.*

The set of tautologies of propositional formulas is written $TAUT$.

**Definition 12** *$TAUT$ for $\{F : F \in \mathcal{F} \wedge \models F\}$.*

Introducing a name $s$ of a Turing machine that exists, and computes whether a propositional formula is satisfiable, with output $\emptyset \,.\, 0$, or unsatisfiable ($\emptyset \,.\, 1$).[33]

**Definition 13** *$T(s,a,u)$ for $a = F \,.\, \emptyset \wedge ((u = \emptyset \,.\, 0 \wedge \not\models \neg F) \vee (u = \emptyset \,.\, 1 \wedge \models \neg F))$ any $a \in L$ $F \in \mathcal{F}$ $u \in L$.*

The Turing machine $s$ specifies a set of correct deterministic Turing machines computing the same output for an input as $s$, by definitions 4 and 13.

**Definition 14** *$\mathcal{U}$ for $\{ i : \exists u (T(i,a,u) \wedge T(s,a,u)) \wedge i \in \mathcal{D} \wedge$ any $a \in L \}$.*

The deterministic Turing machines in $\mathcal{U}$ compute unsatisfiability i.e., $\emptyset \,.\, 1$, for a negation of a propositional formula if and only if the formula is valid.

**Corollary 1** *$T(i, \neg F \,.\, \emptyset, \emptyset \,.\, 1) \equiv F$ any $i \in \mathcal{U}$ any $F \in TAUT$.*

Theory **B** is simply consistent in $\mathcal{U}$ i.e., there is no contradiction, by axiom 1, lemma 1, and definitions 13 - 14.[13] [14]

**Corollary 2** *$\vdash B \rightarrow \exists u\, T(i,\, a,\, u) \wedge \neg \exists u\, T(i,\, a,\, u)$ no $i \in \mathcal{U}$ $a \in L$.*

[32] $|| : L \rightarrow N$.

[33] Of course, Turing machine $s$ could compute satisfiable (unsatisfiable) rather than $\emptyset \,.\, 0$ ($\emptyset \,.\, 1$), respectively. For reasons of space, the shorter version is used.

A special case,[14] $\exists u\, T(i,\, a,\, u)$ is true if and only if $\exists u\, T(i,\, a,\, u)$ is provable from axiom $B$ for $i \in \mathcal{U}$ $a \in L$. By axiom 1, corollary 2, lemma 1, and definitions 13 - 14.

**Corollary 3** $\vdash B \rightarrow \exists u\, T(i,\, a,\, u) \equiv \exists u\, T(i,\, a,\, u)$ *any* $i \in \mathcal{U}$ $a \in L$.

Introducing the formula $SAT \in \mathcal{P}$ in a simply consistent extension $\mathbf{B}'$ of theory $\mathbf{B}$, by corollary 2.

$SAT \in \mathcal{P}$ if and only if satisfiability or unsatisfiability is computable in polynomial time by some Turing machine $i \in \mathcal{U}$ for any propositional formula, cf. footnotes 2 - 4.

**Definition 15** $SAT \in \mathcal{P}$ *for* $\vdash B \rightarrow \exists u\, T(i,\, F\,.\,\emptyset,\, u)$ *in* $p(F)$ *some* $i \in \mathcal{U}$ *any* $F \in \mathcal{F}$.

Unsatisfiability $(\emptyset\,.\,1)$ of $\neg F$ is provable from axiom $B$ for any deterministic Turing machine in $\mathcal{U}$ if and only if $F$ is a tautology, any propositional formula $F$. By corollary 3, and definition 14.

**Corollary 4** $\vdash B \rightarrow T(i,\, \neg F\,.\,\emptyset,\, \emptyset\,.\,1) \equiv \models \rightarrow F$ *any* $i \in \mathcal{U}$ $F \in \mathcal{F}$.

If $SAT \in \mathcal{P}$ then $T(i,\, \neg F\,.\,\emptyset,\, \emptyset\,.\,1)$ is provable from axiom $B$ in polynomial time for some deterministic Turing machine $i$ in $\mathcal{U}$ any tautology $F$. By definitions 12 and 15, and corollary 4.

**Corollary 5** *If* $SAT \in \mathcal{P}$ *then* $\vdash B \rightarrow T(i,\, \neg F\,.\,\emptyset,\, \emptyset\,.\,1)$ *in* $p(F)$ *some* $i \in \mathcal{U}$ *any* $F \in TAUT$.

# 4 Computing time and proof complexity

A relationship between computing time[5] and proof complexity[6] is established in the principal lemma 3, using lemma 2, and axiom $B$. Indeed, there is a relation between the notion of a formal proof in the first order theory $\mathbf{B}$, and the notion of a formal proof in propositional logic.[7] However, lemma 3 gives more i.e., the number of headmoves in a deterministic computation of unsatisfiability for $\neg F$, expressed as a simple formula in propositional logic, from which a (sufficiently large) tautology $F$ is deducible.

This idea is introduced by an example.

**Example 1** *If* $T(i,\, \neg F\,.\,\emptyset,\, \emptyset\,.\,1)$ *then the sequent* $B \rightarrow T(i,\, \neg F\,.\,\emptyset,\, \emptyset\,.\,1)$ *is provable in G4,* $i \in \mathcal{U}$ $F \in TAUT$, *by lemma 1.*

*Such a proof can be constructed, first, by successive applications of the rule* $\forall \rightarrow$ *in Kleene's G4 getting the propositional conjunctions(31) to (23) from axiom* $B$. *Second, using successive applications of the rule* $\supset \rightarrow$ *in G4.*

*A Turing machine in $\mathcal{U}$ is deterministic, so for a tautology $F$ the conjunctions (23) - (31) are unique.*

*Writing $A(i,\, F,\, n)$ for*

$$U(\emptyset\,.\,1, s_n, z_n, 0, i, i, \emptyset\,.\,1)\ \wedge \tag{23}$$

$$[U(\emptyset\,.\,1, s_n, z_n, 0, i, i, \emptyset\,.\,1)\ \supset \tag{24}$$

$$U(x_{n-1}, s_{n-1}, z_{n-1}, q_{n-1}, j_{n-1}, i, \emptyset\,.\,1)]\ \wedge \cdots \wedge$$

$$[U(x_{m+3}, s_{m+3}, z_{m+3}, q_{m+3}, i, i, \emptyset\,.\,1)\ \supset \tag{25}$$

$$U(x_{m+2}, s_{m+2}, z_{m+2}, q_{m+2}, j_{m+2}, i, \emptyset\,.\,1)]\ \wedge$$

$$[U(x_{m+2}, s_{m+2}, z_{m+2}, q_{m+2}, j_{m+2}, i, \emptyset\,.\,1)\ \supset \tag{26}$$

$$U(x_{m+2}, s_{m+2}, z_{m+2}, q_{m+2}, i, i, \emptyset\,.\,1)]\ \wedge \cdots \wedge$$

$$[U(x_2, s_2, z_2, q_2, i, i, \emptyset\,.\,1) \supset U(x_1, s_1, z_1, q_1, j_1, i, \emptyset\,.\,1)]\ \wedge \tag{27}$$

$$[U(x_1, s_1, z_1, q_1, j_1, i, \emptyset\,.\,1) \supset U(x_1, s_1, z_1, q_1, i, i, \emptyset\,.\,1)]\ \wedge \tag{28}$$

$$[U(x_1, s_1, z_1, q_1, i, i, \emptyset\,.\,1) \supset U(\emptyset,\, \emptyset,\, \neg F\,.\,\emptyset\,.\,\emptyset, 1, j_0, i, \emptyset\,.\,1)]\ \wedge \tag{29}$$

$$[U(\emptyset,\, \emptyset,\, \neg F\,.\,\emptyset\,.\,\emptyset, 1, j_0, i, \emptyset\,.\,1) \supset U(\emptyset,\, \emptyset,\, \neg F\,.\,\emptyset\,.\,\emptyset, 1, i, i, \emptyset\,.\,1)]\ \wedge \tag{30}$$

$$[U(\emptyset,\, \emptyset,\, \neg F\,.\,\emptyset\,.\,\emptyset, 1, i, i, \emptyset\,.\,1) \supset T(i,\, \neg F\,.\,\emptyset,\, \emptyset\,.\,1)]\ \textit{any}\ i \in \mathcal{U}\ F \in TAUT \tag{31}$$

$$n \in N\ \textit{some}\ \ m \in N\ \ j_m\ \ j_{m+2}\ \ j_{n-1} \in \mathcal{D}\ \ x_m\ \ x_{m+2}\ \ x_{m+3}\ \ x_{n-1} \in L'$$

$$z_m\ \ z_{m+2}\ \ z_{m+3}\ \ z_{n-1} \in L\ \ s_m\ \ s_{m+2}\ \ s_{m+3} \in S\ \ q_m\ \ q_{m+2}\ \ q_{m+3} \in Q.$$

*Then,*

$$\vdash A(i,\, F,\, n) \rightarrow T(i,\, \neg F\,.\,\emptyset, \emptyset\,.\,1)\ \textit{any}\ i \in \mathcal{U}\ \ F \in TAUT\ \ n \in N. \tag{32}$$

*But also, by axiom B, lemma 1, and corollary 2.*

$$T(i,\, \neg F\,.\,\emptyset, \emptyset\,.\,1) \supset A(i,\, F,\, n)\ \textit{any}\ i \in \mathcal{U}\ \ F \in TAUT\ \textit{some}\ n \in N. \tag{33}$$

*Thus, for a deterministic Turing machine $i$ in $\mathcal{U}$ and a tautology $F$,*

$$\vdash A(i,\, F,\, n) \rightarrow T(i,\, \neg F\,.\,\emptyset, \emptyset\,.\,1)\ \equiv\ T(i,\, \neg F\,.\,\emptyset, \emptyset\,.\,1)\ \textit{some}\ n \in N, \tag{34}$$

*where $n$ is the computing time*[5] *of $i$ to compute unsatisfiability $(\emptyset\,.\,1)$ of $\neg F$.*

*Hence, there is a relationship in propositional logic between the computing time,*[5] *and proof complexity,* [6] *by a proof of the sequent,*

$$A(i,\, F,\, n) \rightarrow T(i,\, \neg F\,.\,\emptyset, \emptyset\,.\,1)\ \textit{in G4, any}\ \ i \in \mathcal{U}\ F \in TAUT\ \textit{some}\ n \in N.$$

Thus, introducing a relationship between the computing time,[5] and proof complexity, as in example 1, is justified.[34]

[34] $V \subseteq \mathcal{U} \times TAUT \times N$.

**Definition 16** $V(i,F,n)$ *for*

$$(U(x_0,s_0,z_0,q_0,i,i,\emptyset\,.\,1) \supset T(i,\neg F\,.\,\emptyset,\emptyset\,.\,1)) \wedge \tag{35}$$

$$U(\emptyset\,.\,1,s_n,z_n,0,i,i,\emptyset\,.\,1) \quad \wedge \tag{36}$$

$$\bigwedge_{1\leq m\leq n}((U(x_m,s_m,z_m,q_m,i,i,\emptyset\,.\,1) \supset \tag{37}$$

$$U(x_{m-1},s_{m-1},z_{m-1},q_{m-1},j_{m-1},i,\emptyset\,.\,1)\,) \wedge$$

$$(U(x_{m-1},s_{m-1},z_{m-1},q_{m-1},j_{m-1},i,\emptyset\,.\,1) \supset$$

$$U(x_{m-1},s_{m-1},z_{m-1},q_{m-1},i,i,\emptyset\,.\,1)))$$

$$any\ i \in \mathcal{U}\ \ F \in TAUT\ n \in N\ some\ m \in N\ x_m\ x_{m-1} \in L'$$

$$z_n\ z_m\ z_{m-1} \in L\ s_n\ s_m\ s_{m-1} \in S\ q_m\ q_{m-1} \in Q\ j_{m-1} \in \mathcal{D}.$$

If the computing time [5] is $n$ for a Turing machine $i$ to compute unsatisfiability $(\emptyset\,.\,1)$ of $\neg F$ then $V(i,F,n)$, any $i \in \mathcal{U}\ F \in TAUT\ n \in N$.

**Lemma 2** $H(\vdash\ B\ \rightarrow\ T(i,\neg F\,.\,\emptyset,\emptyset\,.\,1),n)\ \supset\ V(i,F,n)$ *any* $i \in \mathcal{U}\ F \in TAUT\ n \in N$.

Proof.[35]

$$\star \qquad H(\vdash B \rightarrow T(i,\neg F\,.\,\emptyset,\emptyset\,.\,1),n) \quad i \in \mathcal{U}\ F \in TAUT\ n \in N \tag{38}$$

$$\star \qquad \vdash B \rightarrow T(i,\neg F\,.\,\emptyset,\emptyset\,.\,1),\ \text{definition 6} \tag{39}$$

$$\star \qquad V(i,F,n),\ \text{axiom 1,\ corollary 2,\ and definition 16} \tag{40}$$

$H(\vdash B \rightarrow T(i,\neg F\,.\,\emptyset,\emptyset\,.\,1),n)\ \supset\ V(i,F,n)$ any $i \in \mathcal{U}\ F \in TAUT\ n \in N$

Large propositional formulas are not only inconvenient, they may lead to large proof complexity. Such problems are handled for sufficiently large input, by introducing auxiliary single propositional letters ("propositional variables"), and their truth values. This also gives simpler propositional formulas in a standard syntax of propositional logic.

**Definition 17** $Q_m$ *for* $U(x_m,\ s_m,\ z_m,\ q_m,\ i,\ i,\ \emptyset\,.\,1)$ *any* $i \in \mathcal{U}\ \ x_m \in L'\ \ z_m \in L\ s_m \in S\ \ q_m \in Q\ \ m \in N$ *some single letters* $Q_m \in \mathcal{F}$.

**Definition 18** $R_{m+1}$ *for* $U(x_m,\ s_m,\ z_m,\ q_m,\ j_m,\ i,\ \emptyset\,.\,1)$ *any* $i \in \mathcal{U}\ \ j_m \in \mathcal{D}$ $x_m \in L'\ \ z_m \in L\ \ s_m \in S\ \ q_m \in Q\ \ m \in N$ *some single letters* $R_{m+1} \in \mathcal{F}$.

Writing $W(i,F,n)$ for $V(i,F,n)$ any $i \in \mathcal{U}\ \ F \in TAUT\ \ n \in N$, using the auxiliary single letters in definitions 17 - 18.

**Definition 19** $W(i,F,n)$ *for* $Q_n \wedge \bigwedge_{1\leq m\leq n}(R_m \supset Q_{m-1}) \wedge (Q_m \supset R_m) \wedge (Q_0 \supset T(i,\ \neg F\,.\,\emptyset,\emptyset\,.\,1))$ *any* $i \in \mathcal{U}\ \ F \in TAUT\ n \in N$ *some* $m \in N$ *some single letters* $Q_0,\ldots,Q_n,\ R_1,\ldots,R_n \in \mathcal{F}$.

[35] Introduction of a star shall mean introduction of an assumption, and elimination of a star shall mean that an assumption is discharged. cf. Quine 1974 § 37 [20].

Then $V(i,F,n)$ implies $W(i,F,n)$ for any $i \in \mathcal{U}$ $F \in TAUT$ $n \in N$, by definitions 16 - 19.

**Corollary 6** *$V(i,F,n) \supset W(i,F,n)$ any $i \in \mathcal{U}$ $F \in TAUT$ $n \in N$.*

Writing $Y(i,F,n)$ for a simplified $W(i,F,n)$.

**Definition 20** *$Y(i,F,n)$ for $(Q_0 \supset F) \wedge Q_n \wedge \bigwedge\limits_{1 \leq m \leq n} (Q_m \supset Q_{m-1})$ any $i \in \mathcal{U}$ $F \in TAUT$ $n \in N$ some $m \in N$ some single letters $Q_0, \ldots, Q_n \in \mathcal{F}$.*

Thus $W(i,F,n)$ implies $Y(i,F,n)$ for any $i \in \mathcal{U}$ $F \in TAUT$ $n \in N$, by corollary 1 and definitions 19 - 20.

**Corollary 7** *$W(i,F,n) \supset Y(i,F,n)$ any $i \in \mathcal{U}$ $F \in TAUT$ $n \in N$.*

If the computing time[5] is polynomial for a Turing machine $i$ to compute unsatisfiability of $\neg F$ then the truth values of the single letters in $W(i,F,n)$, and $V(i,F,n)$ are computable in polynomial time, any $i \in \mathcal{U}$ a sufficiently large $F \in TAUT$ some $n \in N$, by axiom 1 definitions 16 - 20 and lemma 2.[36] [37]

**Corollary 8** *If $\vdash B \rightarrow T(i, \neg F \,.\, \emptyset, \emptyset \,.\, 1)$ in $p(F)$ then the truth values of $Q_0, \ldots, Q_n$, $R_1, \ldots, R_n$ in $W(i,F,n)$, and $Y(i,F,n)$, where $n \leq c \cdot |F|^q$, are computable in polynomial time $n$ some $c$ $q \in N$ any $i \in \mathcal{U}$ any sufficiently large $F \in TAUT$ some $n \in N$ some single letters $Q_0, \ldots, Q_n$, $R_1, \ldots, R_n \in \mathcal{F}$.*

If the computing time[5] is polynomial for a Turing machine $i$ to compute unsatisfiability $(\emptyset \,.\, 1)$ of $\neg F$ then $Y(i,F,n)$, where $n \leq c \cdot |F|^q$, some $c$ $q \in N$ any $i \in \mathcal{U}$ a sufficiently large $F \in TAUT$ some $n \in N$.[36] [37]

**Lemma 3** *$\vdash B \rightarrow T(i, \neg F \,.\, \emptyset, \emptyset \,.\, 1)$ in $p(F) \supset Y(i,F,n) \wedge n \leq c \cdot |F|^q$ some $c$ $q \in N$ any $i \in \mathcal{U}$ any sufficiently large $F \in TAUT$ some $n \in N$.*

Proof.

$$\star \quad \vdash B \rightarrow T(i, \neg F \,.\, \emptyset, \emptyset \,.\, 1) \text{ in } p(F) \quad i \in \mathcal{U} \text{ sufficiently large } F \in TAUT \tag{41}$$

$$\star \quad H(\vdash B \rightarrow T(i, \neg F \,.\, \emptyset, \emptyset \,.\, 1), n) \wedge n \leq c \cdot |F|^q \text{ some } c \; q \in N \tag{42}$$

$$\star \quad Y(i,F,n), \text{ lemma 2 and corollaries } 6-8 \tag{43}$$

$$\vdash B \rightarrow T(i, \neg F \,.\, \emptyset, \emptyset \,.\, 1) \text{ in } p(F) \supset Y(i,F,n) \wedge n \leq c \cdot |F|^q \tag{44}$$

$$\text{some } c \; q \in N \text{ any } i \in \mathcal{U} \text{ sufficiently large } F \in TAUT \text{ some } n \in N$$

Briefly, for a sufficiently large tautology there is a propositional formula giving a proof of the tautology, and measuring its computing time.[5] This formula also measures the proof complexity of this tautology, as shall be seen next.

[36] A sentence $C$ is true for sufficiently large natural numbers if there exists a $n' \in N$ such that the sentence $C(n)$ is true for all $n \geq n'$ $n \in N$.

[37] $\lim\limits_{\substack{|F| \rightarrow \infty \\ F \in TAUT}} \frac{|U(x_m, s_m, z_m, q_m, i, i, \emptyset \,.\, 1)|}{|F|} = 1$ and $\lim\limits_{\substack{|F| \rightarrow \infty \\ F \in TAUT}} \frac{|U(x_m, s_m, z_m, q_m, j, i, \emptyset \,.\, 1)|}{|F|} = 1$ for $x_m \in L'$ $s_m \in S$ $z_m \in L$ $q_m \in Q$ $i \in \mathcal{U}$ $j \in \mathcal{D}$ $m \in N$, is used, cf. definition 16.

## 4.1 Proof complexity

In a Hilbert system of propositional logic, a formal deduction is a finite list of formulas $F_1, \ldots, F_n$, where $F_k$ is either an assumption formula $A_1, \ldots, A_m$, an axiom, $1 \leq k \leq n$, or follows from $F_i$ and $F_j$ by modus ponens for $1 \leq i\ j < k$. Such a formal deduction is a deduction of its last formula.[38] Writing,

$$A_1, \ldots, A_m \vdash F_n \text{ for there is a formal deduction } F_1, \ldots, F_n. \tag{45}$$

If there is no assumption, $m < 1$, then there is a formal proof of $F_n$, in (45).

There exists a relation,[39] which computes the size of a formal deduction i.e., the number of symbols in the deduction in (45).[40]

**Definition 21** *$G(A_1, \ldots, A_m \vdash F_n, u)$ for $A_1, \ldots, A_m \vdash F_n \wedge u = \sum_{1 \leq k \leq n} |F_k|$.*

For the purpose of proving the main result, interesting propositional systems have a greater than polynomial lower bound on the size of the deductions of tautologies of the pigeonhole principle. For example, Robinson's resolution system[10][11] and bounded depth Frege systems.[12]

Robinson's resolution principle is complete and consistent.[41] A deduction in resolution is an indirect deduction, the formulas in the deduction are clauses, and the last formula is the empty clause.[42] Thus, similar to (45).

**Definition 22** *$A \vdash_R F$ for there is a deduction of a DNF formula $F$ from a CNF formula $A$ by resolution (including propositional logic).*

Similarly to a Hilbert system, the size of a resolution deduction is the number of symbols in the deduction.

**Definition 23** *$G(A \vdash_R F, u)$ for there is a resolution deduction of a DNF formula $F$ from a CNF formula $A$ and $u$ is the number of symbols in the deduction, $A\ F \in \mathcal{F}$.*

A polynomial upper bound of the size of a resolution deduction is written as follows.

**Definition 24** *$A \vdash_R F$ in $p(F)$ for $G(A \vdash_R F, u) \wedge u \leq c \cdot |F|^q$ some $c\ q \in N$ any $A\ F \in \mathcal{F}$ some $u \in N$.*

Next there is an indirect proof in the proof (meta) theory of theory **B**, by lemma 3 (the relationship between computing time [5] and pigeonhole formulas,[9] Haken's theorem,[11] and simply consistency of theory **B** for the subset of Turing machines in $\mathcal{U}$, corollary 2. [43] [36]

[38] Following Kleene 1967 [14].

[39] $G \subseteq K \times N$. $K$ is the set of formal deductions, and $N$ the set of the natural numbers.

[40] $|F|$ for the number of symbols of $F \in \mathcal{F}$, cf. footnote 32 and definition 7.

[41] $A \models F$ if and only if $A \vdash_R F$, by Robinson's theorem 1965 [21].

[42] A CNF formula is a conjunction of disjunctions of atoms, and negated atoms. A DNF formula is a disjunction of conjunctions of atoms, and negated atoms. cf. Kleene 1967 [14].

[43] A bounded depth Frege system, with a greater than polynomial lower bound for the pigeonhole principle, gives a proof of theorem 1 too.[12]

**Theorem 1** *$SAT \notin \mathcal{P}$ is true in a simply consistent extension $\mathbf{B}'$ of theory $\mathbf{B}$.*

Proof.

$$\star \quad SAT \in \mathcal{P} \text{ in a simply consistent extension } \mathbf{B}' \text{ of } \mathbf{B}, \text{ definition 15} \tag{46}$$

$$\star \quad \vdash B \rightarrow T(i, \neg F \,.\, \emptyset, \emptyset \,.\, 1) \text{ in } p(F) \text{ some } i \in \mathcal{U} \text{ any } F \in TAUT, \text{ corollary 5} \tag{47}$$

$$\star \quad \vdash B \rightarrow T(i, \neg PF_m \,.\, \emptyset, \emptyset \,.\, 1) \text{ in } p(PF_m) \text{ any sufficiently large } m \,^{9} \tag{48}$$

$$\star \quad Y(i, PF_m, n) \wedge n \leq c \cdot |PF_m|^q \text{ some } c\ q \in N \text{ sufficiently large } m, \text{ lemma 3} \tag{49}$$

$$\star \quad (Q_0 \supset PF_m) \wedge Q_n \wedge \bigwedge_{1 \leq k \leq n} (Q_k \supset Q_{k-1}) \text{ some single letters } Q_0, \ldots, Q_n \in \mathcal{F} \tag{50}$$

$$\star \quad \vdash_{R} PF_m \text{ in } p(PF_m) \quad \text{sufficiently large } m, \text{ by (50) and (49)} \tag{51}$$

$$\star \quad \neg(\vdash_{R} PF_m \text{ in } p(PF_m)) \quad \text{sufficiently large } m,^{11} \text{ contradiction} \tag{52}$$

$$SAT \notin \mathcal{P} \text{ in a simply consistent extension } \mathbf{B}' \text{ of } \mathbf{B}, \text{ corollary 2} \tag{53}$$

Then, by the Cook-Levin theorem,[2] and theorem 1.

**Theorem 2** *$\mathcal{P} \neq \mathcal{NP}$ is true in a simply consistent extension $\mathbf{B}''$ of theory $\mathbf{B}$.*

By lemma 1, definitions 14 - 15, and theorem 1.

**Corollary 9** *$SAT \notin \mathcal{P}$ is true, and provable in a simply consistent extension $\mathbf{B}'$ of theory $\mathbf{B}$.*

By lemma 1, theorem 2, and corollary 9.

**Corollary 10** *$\mathcal{P} \neq \mathcal{NP}$ is true, and provable in a simply consistent extension $\mathbf{B}''$ of theory $\mathbf{B}$.*

## Acknowledgment

Hanna-Nina and Niklas Ekelund, Andreas Hamfelt, Kaj Børge Hansen, Sophie Maisnier-Patin, Erik Nerep, Jørgen Fischer Nilsson, Torsten Palm, Alan Robinson, Thomas Sjöland, and Carl-Anton Tärnlund thank you.

## Appendix

**Example 2** *(Growing the tape one element) Turing machine $i_g$ for $1 \,.\, \emptyset \,.\, 1 \,.\, \emptyset \,.\, 1 \,.\, 1 \,.\, A \,.\, 1 \,.\, A \,.\, 1 \,.\, 1 \,.\, \emptyset \,.\, 2 \,.\, B \,.\, \emptyset \,.\, 0 \,.\, 2 \,.\, A \,.\, 2 \,.\, A \,.\, 1 \,.\, 2 \,.\, B \,.\, 0 \,.\, B \,.\, 1$. It is a list of 5 quintuples, and is deterministic i.e., no more than one quintuple applies for a state and symbol. It growths the input list $A \,.\, \emptyset$ with one symbol $B$. For example, the Turing machine $i_g$ computes the output (left tape) $\emptyset \,.\, \emptyset \,.\, A \,.\, B$ with the list $\emptyset \,.\, A \,.\, B$, where $B$ is the top element (reading the list from the right).*

**Example 3** *(A computation is a proof in theory $\mathbf{B}$) In theory $\mathbf{B}$ the Turing machine $i_g$ for the input list $A \,.\, \emptyset$ computes the left tape $\emptyset \,.\, \emptyset \,.\, A \,.\, B$ with the list $\emptyset \,.\, A \,.\, B$. Thus, the output list has grown with one symbol. There is a proof of*

$T(i_g, A\,.\,\emptyset, \emptyset\,.\,\emptyset\,.\,A\,.\,B)$ *from axiom* $B$. *The computing time i.e., the number of moves of the head of Turing machine* $i_g$ *for* $A\,.\,\emptyset$ *is 5 (the moves of the tapehead in (20) and (21) of axiom* $B$*).*

*Thus,* $\vdash B \rightarrow T(i_g,\ A\,.\,\emptyset, \emptyset\,.\,\emptyset\,.\,A\,.\,B)$.

*Proof.*

*Writing* $(22)^*$ *for a finite number of applications of* $(22)$ *of axiom* $B$.

$$B \rightarrow T(i_g, A\,.\,\emptyset, u) \tag{54}$$

$$B \rightarrow U(\emptyset, \emptyset,\ A\,.\,\emptyset\,.\,\emptyset, 1, i_g, i_g, u),\ \textit{by (18) of axiom } B \tag{55}$$

$$B \rightarrow U(\emptyset\,.\,\emptyset, A,\ \emptyset\,.\,\emptyset, 1, i_g, i_g, u),\ \textit{by (21)} \tag{56}$$

$$B \rightarrow U(\emptyset\,.\,\emptyset, A,\ \emptyset\,.\,\emptyset, 1, 1\,.\,A\,.\,1\,.\,A\,.\,1\,.\,j, i_g, u),\ \textit{by } (22)^* \tag{57}$$

$$B \rightarrow U(\emptyset\,.\,\emptyset\,.\,A,\ \emptyset,\ \emptyset, 1, i_g, i_g, u),\ \textit{by (21)} \tag{58}$$

$$\vdots$$

$$B \rightarrow U(\emptyset\,.\,\emptyset\,.\,A,\ \emptyset,\ \emptyset, 1, 1\,.\,\emptyset\,.\,2\,.\,B\,.\,\emptyset\,.\,0\,.\,j, i_g, u),\ \textit{by } (22)^* \tag{59}$$

$$B \rightarrow U(\emptyset\,.\,\emptyset,\ A, B\,.\,\emptyset\,.\,\emptyset, 2, i_g, i_g, u),\ \textit{by (21)} \tag{60}$$

$$\vdots$$

$$B \rightarrow U(\emptyset\,.\,\emptyset, A, B\,.\,\emptyset\,.\,\emptyset, 2, 2\,.\,A\,.\,2\,.\,A\,.\,1\,.\,j, i_g, u),\ \textit{by } (22)^* \tag{61}$$

$$B \rightarrow U(\emptyset\,.\,\emptyset\,.\,A,\ B,\ \emptyset\,.\,\emptyset, 2, i_g, i_g, u),\ \textit{by (20)} \tag{62}$$

$$\vdots$$

$$B \rightarrow U(\emptyset\,.\,\emptyset\,.\,A,\ B,\ \emptyset\,.\,\emptyset, 2, 2\,.\,B\,.\,0\,.\,B\,.\,1\,.\,j, i_g, u),\ \textit{by } (22)^* \tag{63}$$

$$B \rightarrow U(\emptyset\,.\,\emptyset\,.\,A\,.\,B,\ \emptyset,\ \emptyset, 0, i_g, i_g, u),\ \textit{by (21)} \tag{64}$$

$$B, U(\emptyset\,.\,\emptyset\,.\,A\,.\,B,\ \emptyset,\ \emptyset, 0, i_g, i_g, \emptyset\,.\,\emptyset\,.\,A\,.\,B) \overset{\times}{\rightarrow} U(\emptyset\,.\,\emptyset\,.\,A\,.\,B,\ \emptyset,\ \emptyset, 0, i_g, i_g, \emptyset\,.\,\emptyset\,.\,A\,.\,B) \quad \textit{by (19)} \tag{65}$$

**Example 4** *(A parenthesis checker)*[44] *Turing machine* $i_p$ *for* $1\,.\,\emptyset\,.\,q_0\,.\,\emptyset\,.\,1\,.\,q_0\,.\,]\,.\,q_1\,.\,X\,.\,0\,.\,q_0\,.\,[\,.\,q_0\,.\,[\,.\,1\,.\,q_0\,.\,\emptyset\,.\,q_2\,.\,\emptyset\,.\,0\,.\,q_0\,.\,X\,.\,q_0\,.\,X\,.\,1\,.\,q_1\,.\,]\,.\,q_1\,.\,]\,.\,0\,.\,q_1\,.\,[\,.\,q_0\,.\,X\,.\,1\,.\,q_1\,.\,\emptyset\,.\,0\,.\,0\,.\,1\,.\,q_1\,.\,X\,.\,q_1\,.\,X\,.\,0\,.\,q_2\,.\,[\,.\,0\,.\,0\,.\,1\,.\,q_2\,.\,\emptyset\,.\,0\,.\,1\,.\,1\,.\,q_2\,.\,X\,.\,q_2\,.\,X\,.\,0\,.\,\emptyset$. *It is a list of 12 quintuples, and is deterministic i.e., no more than one quintuple applies for a state and symbol. It checks whether its input is of well-formed parenthesis. For example,* $a_1$ *for the list* $[\,.\,[\,.\,]\,.\,]\,.\,\emptyset$ *is well-formed, but* $a_2$ *for* $[\,.\,[\,.\,]\,.\,]\,.\,]\,.\,\emptyset$ *is not. Turing machine* $i_p$ *computes an output* $u\,.\,1$ *if its input is well-formed and* $u\,.\,0$ *if the input is not, some* $u \in L$.

**Example 5** *(A computation is a proof in theory* **B***) In theory* **B** *Turing machine* $i_p$ *for input* $a_1$ *computes the output list* $\emptyset\,.\,1$. *Thus, the input list is well-formed. There is a proof of* $T(i_p, a_1, \emptyset\,.\,1)$ *from axiom* $B$. *The computing time i.e., the number of moves of the head of Turing machine* $i_p$ *for* $a_1$ *is 19 (the head moves in (20) and (21) of axiom* $B$*).*

*Hence,* $\vdash B \rightarrow T(i_p,\ a_1,\ \emptyset\,.\,1)$.

[44] c.f. Minsky 1967 [17].

*Proof.*
*Writing (22)* for a finite number of applications of (22) of axiom B.*

$$B \to T(i_p, a_1, u) \tag{66}$$
$$B \to U(\emptyset, \emptyset, [\,.\,[\,.\,]\,.\,]\,.\,\emptyset\,.\,\emptyset, 1, i_p, i_p, u),\ \textit{by (18) of axiom B} \tag{67}$$
$$B \to U(\emptyset\,.\,\emptyset, [, [\,.\,]\,.\,]\,.\,\emptyset\,.\,\emptyset, q_0, i_p, i_p, u),\ \textit{by (21)} \tag{68}$$
$$B \to U(\emptyset\,.\,\emptyset,\ [,\ [\,.\,]\,.\,]\,.\,\emptyset\,.\,\emptyset, q_0, q_0\,.\,[\,.\,q_0\,.\,[\,.\,1\,.\,j, i_p, u),\ \textit{by (22)}^* \tag{69}$$
$$B \to U(\emptyset\,.\,\emptyset\,.\,[, [, ]\,.\,]\,.\,\emptyset\,.\,\emptyset, q_0, i_p, i_p, u),\ \textit{by (21)} \tag{70}$$
$$\vdots$$
$$B \to U(\emptyset\,.\,\emptyset\,.\,[, [, ]\,.\,]\,.\,\emptyset\,.\,\emptyset, q_0, q_0\,.\,[\,.\,q_0\,.\,[\,.\,1\,.\,j, i_p, u),\ \textit{by (22)}^* \tag{71}$$
$$B \to U(\emptyset\,.\,\emptyset\,.\,[\,.\,[, ], ]\,.\,\emptyset\,.\,\emptyset, q_0, i_p, i_p, u),\ \textit{by (21)} \tag{72}$$
$$\vdots$$
$$B \to U(\emptyset\,.\,\emptyset\,.\,[\,.\,[, ], ]\,.\,\emptyset\,.\,\emptyset, q_0, q_0\,.\,]\,.\,q_1\,.\,X\,.\,0\,.\,j, i_p, u),\ \textit{by (22)}^* \tag{73}$$
$$B \to U(\emptyset\,.\,\emptyset\,.\,[,\ [, X\,.\,]\,.\,\emptyset\,.\,\emptyset, q_1, i_p, i_p, u),\ \textit{by (20)} \tag{74}$$
$$\vdots$$
$$B \to U(\emptyset\,.\,\emptyset\,.\,[,\ [, X\,.\,]\,.\,\emptyset\,.\,\emptyset, q_1, q_1\,.\,[\,.\,q_0\,.\,X\,.\,1\,.\,j, i_p, u),\ \textit{by (22)}^* \tag{75}$$
$$B \to U(\emptyset\,.\,\emptyset\,.\,[\,.\,X,\ X, ]\,.\,\emptyset\,.\,\emptyset, q_0, i_p, i_p, u),\ \textit{by (21)} \tag{76}$$
$$\vdots$$
$$B \to U(\emptyset\,.\,\emptyset\,.\,[\,.\,X,\ X, ]\,.\,\emptyset\,.\,\emptyset, q_0, q_0\,.\,X\,.\,q_0\,.\,X\,.\,1\,.\,j, i_p, u),\ \textit{by (22)}^* \tag{77}$$
$$B \to U(\emptyset\,.\,\emptyset\,.\,[\,.\,X\,.\,X, ], \emptyset\,.\,\emptyset, q_0, i_p, i_p, u),\ \textit{by (21)} \tag{78}$$
$$\vdots$$
$$B \to U(\emptyset\,.\,\emptyset\,.\,[\,.\,X\,.\,X, ], \emptyset\,.\,\emptyset, q_0, q_0\,.\,]\,.\,q_1\,.\,X\,.\,0\,.\,j, i_p, u),\ \textit{by (22)}^* \tag{79}$$
$$B \to U(\emptyset\,.\,\emptyset\,.\,[\,.\,X,\ X, X\,.\,\emptyset\,.\,\emptyset, q_1, i_p, i_p, u),\ \textit{by (20)} \tag{80}$$
$$\vdots$$
$$B \to U(\emptyset\,.\,\emptyset\,.\,[\,.\,X,\ X, X\,.\,\emptyset\,.\,\emptyset, q_1, q_1\,.\,X\,.\,q_1\,.\,X\,.\,0\,.\,j, i_p, u),\ \textit{by (22)}^* \tag{81}$$
$$B \to U(\emptyset\,.\,\emptyset\,.\,[,\ X, X\,.\,X\,.\,\emptyset\,.\,\emptyset, q_1, i_p, i_p, u),\ \textit{by (20)} \tag{82}$$
$$\vdots$$
$$B \to U(\emptyset\,.\,\emptyset\,.\,[,\ X, X\,.\,X\,.\,\emptyset\,.\,\emptyset, q_1, q_1\,.\,X\,.\,q_1\,.\,X\,.\,0\,.\,j, i_p, u),\ \textit{by (22)}^* \tag{83}$$
$$B \to U(\emptyset\,.\,\emptyset,\ [, X\,.\,X\,.\,X\,.\,\emptyset\,.\,\emptyset, q_1, i_p, i_p, u),\ \textit{by (20)} \tag{84}$$
$$\vdots$$
$$B \to U(\,\emptyset\,.\,\emptyset,\ [, X\,.\,X\,.\,X\,.\,\emptyset\,.\,\emptyset, q_1, q_1\,.\,[\,.\,q_0\,.\,X\,.\,1\,.\,j, i_p, u),\ \textit{by (22)}^* \tag{85}$$
$$B \to U(\emptyset\,.\,\emptyset\,.\,X,\ X, X\,.\,X\,.\,\emptyset\,.\,\emptyset, q_0, i_p, i_p, u),\ \textit{by (21)} \tag{86}$$
$$\vdots$$

$$B \to U(\emptyset \,.\, \emptyset \,.\, X,\ X, X \,.\, X \,.\, \emptyset \,.\, \emptyset, q_0, q_0 \,.\, X \,.\, q_0 \,.\, X \,.\, 1 \,.\, j, i_p, u),\ \textit{by (22)}^* \quad (87)$$
$$B \to U(\emptyset \,.\, \emptyset \,.\, X \,.\, X,\ X, X \,.\, \emptyset \,.\, \emptyset, q_0, i_p, i_p, u),\ \textit{by (21)} \quad (88)$$
$$\vdots$$
$$B \to U(\emptyset \,.\, \emptyset \,.\, X \,.\, X,\ X, X \,.\, \emptyset \,.\, \emptyset, q_0, q_0 \,.\, X \,.\, q_0 \,.\, X \,.\, 1 \,.\, j, i_p, u),\ \textit{by (22)}^* \quad (89)$$
$$B \to U(\emptyset \,.\, \emptyset \,.\, X \,.\, X \,.\, X,\ X, \emptyset \,.\, \emptyset, q_0, i_p, i_p, u),\ \textit{by (21)} \quad (90)$$
$$\vdots$$
$$B \to U(\emptyset \,.\, \emptyset \,.\, X \,.\, X \,.\, X,\ X, \emptyset \,.\, \emptyset, q_0, q_0 \,.\, X \,.\, q_0 \,.\, X \,.\, 1 \,.\, j, i_p, u),\ \textit{by (22)}^* \quad (91)$$
$$B \to U(\emptyset \,.\, \emptyset \,.\, X \,.\, X \,.\, X \,.\, X,\ \emptyset, \emptyset, q_0, i_p, i_p, u),\ \textit{by (21)} \quad (92)$$
$$\vdots$$
$$B \to U(\emptyset \,.\, \emptyset \,.\, X \,.\, X \,.\, X \,.\, X,\ \emptyset, \emptyset, q_0, q_0 \,.\, \emptyset \,.\, q_2 \,.\, \emptyset \,.\, 0 \,.\, j, i_p, u),\ \textit{by (22)}^* \quad (93)$$
$$B \to U(\emptyset \,.\, \emptyset \,.\, X \,.\, X \,.\, X,\ X, \emptyset \,.\, \emptyset, q_2, i_p, i_p, u),\ \textit{by (20)} \quad (94)$$
$$\vdots$$
$$B \to U(\emptyset \,.\, \emptyset \,.\, X \,.\, X \,.\, X,\ X, \emptyset \,.\, \emptyset, q_2, q_2 \,.\, X \,.\, q_2 \,.\, X \,.\, 0 \,.\, j, i_p, u),\ \textit{by (22)}^* \quad (95)$$
$$B \to U(\emptyset \,.\, \emptyset \,.\, X \,.\, X,\ X, X \,.\, \emptyset \,.\, \emptyset, q_2, i_p, i_p, u),\ \textit{by (20)} \quad (96)$$
$$\vdots$$
$$B \to U(\emptyset \,.\, \emptyset \,.\, X \,.\, X,\ X, X \,.\, \emptyset \,.\, \emptyset, q_2, q_2 \,.\, X \,.\, q_2 \,.\, X \,.\, 0 \,.\, j, i_p, u),\ \textit{by (22)}^* \quad (97)$$
$$B \to U(\emptyset \,.\, \emptyset \,.\, X, X, X \,.\, X \,.\, \emptyset \,.\, \emptyset, q_2, i_p, i_p, u),\ \textit{by (20)} \quad (98)$$
$$\vdots$$
$$B \to U(\emptyset \,.\, \emptyset \,.\, X,\ X, X \,.\, X \,.\, \emptyset \,.\, \emptyset, q_2, q_2 \,.\, X \,.\, q_2 \,.\, X \,.\, 0 \,.\, j, i_p, u),\ \textit{by (22)}^* \quad (99)$$
$$B \to U(\emptyset \,.\, \emptyset,\ X, X \,.\, X \,.\, X \,.\, \emptyset \,.\, \emptyset, q_2, i_p, i_p, u),\ \textit{by (20)} \quad (100)$$
$$\vdots$$
$$B \to U(\emptyset \,.\, \emptyset,\ X, X \,.\, X \,.\, X \,.\, \emptyset \,.\, \emptyset, q_2, q_2 \,.\, X \,.\, q_2 \,.\, X \,.\, 0 \,.\, j, i_p, u),\ \textit{by (22)}^* \quad (101)$$
$$B \to U(\emptyset,\ \emptyset, X \,.\, X \,.\, X \,.\, X \,.\, \emptyset \,.\, \emptyset, q_2, i_p, i_p, u),\ \textit{by (20)} \quad (102)$$
$$\vdots$$
$$B \to U(\emptyset,\ \emptyset, X \,.\, X \,.\, X \,.\, X \,.\, \emptyset \,.\, \emptyset, q_2, q_2 \,.\, \emptyset \,.\, 0 \,.\, 1 \,.\, 1 \,.\, j, i_p, u),\ \textit{by (22)}^* \quad (103)$$
$$B \to U(\emptyset \,.\, 1,\ X, X \,.\, X \,.\, X \,.\, \emptyset \,.\, \emptyset, 0, i_p, i_p, u),\ \textit{by (20)} \quad (104)$$
$$B, U(\emptyset \,.\, 1, X, X \,.\, X \,.\, X \,.\, \emptyset \,.\, \emptyset, 0, i_p, i_p, \emptyset \,.\, 1) \xrightarrow{\times} \quad (105)$$
$$U(\emptyset \,.\, 1, X, X \,.\, X \,.\, X \,.\, \emptyset \,.\, \emptyset, 0, i_p, i_p, \emptyset \,.\, 1) \qquad \textit{by (19)}$$

# References


[1] Miklòs Ajtai. The complexity of the pigeonhole principle. In *Proceedings of the 29th Annual IEEE Symposium on the Foundations of Computer Sci-*

*ence*, pages 346–355, White Plains, New York USA, 1988. IEEE.

[2] Paul Beame, Russell Impagliazzo, Jan Krajíček, Toniann Pitassi, Pavel Pudlák, and Alan Woods. Exponential lower bounds for the pigeonhole principle. In *STOC '92: Proceedings of the twenty-fourth annual ACM symposium on Theory of computing*, pages 200–220, New York, NY, USA, 1992. ACM.

[3] Stephen Bellantoni, Toniann Pitassi, and Alasdair Urquhart. Approximation and Small-Depth Frege Proofs. *SIAM Journal on Computing*, 21(6):1161–1179, 1992.

[4] Stephen Cook. The complexity of theorem-proving procedures. In *Third Annual ACM Symposium on Theory of Computing*, pages 151–158, New York, NY, USA, 1971. ACM Press.

[5] Stephen A. Cook. The Importance of the P versus NP Question. *Journal of the ACM*, 50(1):27–29, 2003.

[6] Stephen A. Cook and Robert A. Reckhow. The relative efficiency of propositional proof systems. *Journal of Symbolic Logic*, 44(1):36–50, 1979.

[7] Martin Davis. *Computability and Unsolvability*. McGraw–Hill, NY, USA, 1958.

[8] Gerhard Gentzen. Untersuchungen über das Logische Schliessen. *Mathematische Zeitschrift*, 39:176–210 and 405–431, 1935.

[9] Armin Haken. The intractability of resolution (complexity). *Theoretical Computer Science*, 39:297–308, 1985. Also Ph D thesis University of Illinois at Urbana-Champaign 1984.

[10] J. Hartmanis and R. E. Stearns. On the computational complexity of algorithms. *Transactions of the American Mathematical Society*, 117:285–306, 1965.

[11] Jacques Herbrand. *Recherches sur la thèorie de la dèmonstration*. Thesis, University of Paris, France, 1930.

[12] David Hilbert and Paul Bernays. *Grundlagen der Mathematik*, volume 1. Springer-Verlag, Berlin, 1934. Volume 2, 1939.

[13] Richard M. Karp. Reducibility among combinatorial problems. In R. E. Miller and J. W. Thatcher, editors, *Complexity of Computer Computations*, pages 85–103. Plenum Press, New York, NY, USA, 1972.

[14] Stephen C. Kleene. *Mathematical Logic*. John Wiley and Sons, New York, USA, 1967. First corrected printing, March, 1968.

[15] Jan Krajíček, Pavel Pudlák, and Alan Woods. An exponential lower bound to the size of bounded depth Frege proofs of the pigeonhole principle. *Random Structures and Algorithms*, 7(1):15–39, 1995.

[16] L. Levin. Universal search problems (in Russian). *Problemy Peredachi Informatsii*, 9(3):115–116, 1973.

[17] Marvin Minsky. *Computation Finite and Infinite Machines*. Prentice-Hall, Inc., Englewood Cliffs NJ, USA, 1967.

[18] Toniann Pitassi, Paul Beame, and Russell Impagliazzo. Exponential lower bounds for the pigeonhole principle. *Computational Complexity*, 3(2):97–140, 1993.

[19] Emil L. Post. Recursive Unsolvability of a Problem of Thue. *Journal of Symbolic Logic*, 12(3):1–11, 1947.

[20] W. V. Quine. *Methods of Logic*, volume 3. Routledge & Keagan Paul, London, UK, 1974.

[21] John Alan Robinson. A Machine-Oriented Logic Based on the Resolution Principle. *Journal of the ACM*, 12(1):23–41, 1965.

[22] Michael Sipser. *Introduction to the Theory of Computation*. Thomson Course Technology, 2005. Second Edition International Edition.

[23] Steve Smale. Mathematical Problems for the Next Century. *Mathematical Intelligencer*, 20:7–15, 1998.

[24] Sten-Åke Tärnlund. Computing. Updated 2008, Nov 2004.

[25] Alan M. Turing. On Computable Numbers with an Application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society*, 2/46:230–265, 1936.