Julius-Maximilians-Universität Würzburg
Institut für Informatik
Lehrstuhl für Informatik IV
Theoretische Informatik

Bachelor Thesis

Simulation of Proof Systems

Nils Wisiol

submitted on May 11, 2012

supervisor:
Dr. Christian Glaßer

Acht Jahre nachdem die P-NP-Frage von Cook gestellt wurde [Coo71], war es auch Cook zusammen mit Reckhow, der den engen Zusammenhang dieser Frage mit Beweissystemen feststellte [CR79].

Die Bedeutung der bis heute ungeklärten P-NP-Frage ist immens. Die Gleichheit dieser Mengen würde bedeuten, dass man zu jedem Problem, dessen Lösung sich leicht überprüfen lässt, auch leicht eine Lösung finden kann. Es wäre beispielsweise genau so schwer, einen Beweis zu verifzieren, wie einen Beweis zu finden. Es würde kaum einen Unterschied machen, ob man die PIN zu einer EC-Karte herausfinden möchte, oder überprüfen möchte, ob die eingegebene Nummer die korrekte ist. Aufgrund dieser unwirklich erscheinenden Implikationen glauben die meisten Komplexitätstheoretiker, dass $P \neq NP$.

Der Zusammenhang der P-NP-Frage mit Beweissystemen lässt sich wie folgt formulieren. Existiert kein polynomiell beschränktes Beweissystem für TAUT, dann folgt $P \neq NP$. Diesen Satz werden wir im Laufe dieser Arbeit beweisen, und anschließend noch etwas tiefer in die Theorie der Beweisssysteme einsteigen.

Zunächst wird der Leser in die wichtigsten Defintionen und Begriffe aus dem Gebiet der Beweissysteme eingeführt. Anschließend werden in Kapitel 3 einige wichtige Ergebnisse zusammengefasst. In Kapitel 4 wird bewiesen, dass in bestimmten superpolynomiellen Komplexitätsklassen Sprachen existieren, die keine optimalen Beweissysteme besitzen. Abschließend werden die Ergebnisse der Arbeit zusammengefasst und ein Ausblick auf weitere interessante und offene Fragen gegeben.

Contents

1 Introduction

After Cook stated the P versus NP question in 1971 [Coo71], he gave the main motivation to study proof systems in 1979. Cook and Reckhow showed in their article the close relation between the separation of complexity classes and the existence of polynomially bounded proof systems [CR79].

Despite its importance, the P versus NP question remains still open. Informally speaking, P = NP means that for every problem that has an efficiently verifiable solution, we can find that solution efficiently as well. While most theoreticians assume that P $\neq$ NP, their equality would have heavy implications. One consequence affects the security of communication, as public-key cryptography depends on the existence of certain problems in NP that are not efficiently to decide. If there are no problems with solutions fast to verify but hard to find—as P = NP would imply—the small lock beside the URL in one's browser could not indicate a secure communication anymore [For09].

P = NP would also have fundamental implications on mathematics: As mathematical proofs must by definition be efficiently verifiable, P = NP would imply that they are efficiently to find [CR79]. This argument to believe that P $\neq$ NP is further illustrated in the following quotation, taken from Aaronsons Blog[1].

> "If P = NP, then the world would be a profoundly different place than we usually assume it to be. There would be no special value in 'creative leaps', no fundamental gap between solving a problem and recognizing the solution once it's found. Everyone who could appreciate a symphony would be Mozart; everyone who could follow a step-by-step argument would be Gauss; everyone who could recognize a good investment strategy would be Warren Buffett."

Closely related to the P versus NP problem is the question if NP = co-NP. If P = NP, then NP = co-NP, since P is closed under complement. In return, if one can separate NP from co-NP, then P $\neq$ NP. To connect the field of proof systems with the P versus NP questions, we state

Proposition 1 ([KMT03], [CR79])**.** *NP = co-NP if and only if a polynomially bounded proof system for TAUT exists.*

In order to introduce the reader into the field of proof systems, we will first define the important notions used related to it. Subsequently, we will prove proposition 1 and give an overview of important results about proof systems in chapter 3. These results will connect proof systems with the P versus NP question as mentioned above and will give a basis for later proofs. In chapter 4 we will proof the main theorem of this thesis, showing that there are languages without optimal proof systems in all super-polynomial complexity classes. This chapter features also an analysis of the the inner structure of these languages that do not possess an optimal proof system.

Finally, we will give a conclusion and look forward to currently unresolved problems and further questions.

[1] http://www.scottaaronson.com/blog/

2 Preliminaries

As mentioned before, we will first introduce important symbols and definitions. Although some familiarity with standard notions of complexity theory is assumed, we will here define most of the notions used in this thesis. For the most important ones, we will provide a short discussion.

Let $\Sigma = \{0,1\}$ denote the *alphabet*. The *output* of a Turing transducer M on input $x \in \Sigma^*$ is denoted by $M(x)$. If the transducer M does not accept or runs forever on input x, we define $M(x) = \bot$. We say a Turing transducer *calculates* a partial function f, if $M(x) = f(x)$ for all $x \in \Sigma^*$. We further define $\text{time}_M(x)$ as the number of steps the transducer M runs on input $x \in \Sigma^*$. With $\mathcal{FP}$ we denote the set of all partial functions f for which a Turing transducer M calculating f exists such that $\text{time}_M(x) \leq p(|x|)$ for a polynomial p.

Definition 1. *A function $h \in \mathcal{FP}$ is called* proof system *for a language L if the range of h is L. A string w with $h(w) = x$ is called an h-*proof *for x. h is called an* polynomially bounded *proof system, if there is a polynomial p such that for every $x \in L$, there is a h-proof w with $|w| \leq p(|x|)$.*

With this definition, a proof system for L is basically a polynomial-time bounded function that enumerates L. To give an example, let h be defined by

$$sat(x) = \begin{cases} \varphi & (x = \langle a, \varphi \rangle \text{ and } \alpha \text{ is an satisfying assignment for } \varphi), \\ \bot & (\text{otherwise}). \end{cases}$$

Then h is a proof system for SAT.

Notice, in spite of its time bound against the input, the shortest proof of a string $w \in L$ can be be very long. There may be various proof systems for a language L. In order to make them comparable, we define the notion of *simulation* of proof systems. It turns out that the notion of simulation corresponds in a certain way with the notion of many-one reducibility [KM00]. The following definitions are constructed in a way that will keep this correspondence.

Definition 2. *Let h and h' be proof systems for a language L. If there is a polynomial p and a function f such that for all $w \in \Sigma^*$*

$$h(f(w)) = h'(w)$$

and $|f(w)| \leq p(|w|)$, then h simulates *h'.*

Informally speaking, f translates h-proofs into polynomial length bounded h'-proofs. In the given definition, f could be hard or even impossible to calculate. Hence we define a stronger version of this notion, demanding $f \in \mathcal{FP}$.

Definition 3. *Again, let h and h' be proof systems for a language L. If h simulates h' with a function f and additionally $f \in \mathcal{FP}$, h* p-simulates *h'.*

Notice, if f is a function that can be calculated in polynomial time p, then we obtain $|f(w)| \leq p(|w|)$ as required in the definition of simulation. With a proof system p-simulating another, we can translate proofs as described above in polynomial short time. The notion of simulation of proof systems allows us to compare different proof systems for a language L. With respect to these notions, we will define a notion of the best proof system as follows.

Definition 4. *A proof system h for L is called* optimal*, if it simulates every proof system for L. It is called* p-optimal*, if it p-simulates every proof system for L.*

Like noted above, this definition corresponds with the definition of complete problems in respect of many-one-reducibility [KMT03]. We will investigate further on the connection between complete problems and optimal proof systems in lemma 2.

It is an open question whether *sat* is p-optimal. Köbler and Messner showed that this question is equivalent to a variety of well studied complexity theoretic assumptions [KM00].

The existence of optimal proof systems for a arbitrary language L is an important question in complexity theory. For languages in P and NP, there is always an optimal proof system, as we will see in lemma 4. For super-polynomial time complexity classes, there are languages without an optimal proof system, as we will show in chapter 4. For that reason, we will define a complexity class containing all languages possessing an optimal proof system.

Definition 5. *Let* OPT *be the complexity class of all languages that have an optimal proof system.*

Observe that for OPT we use the weaker notion of simulation. As noted above, we can easily state a proof system for languages in P or NP, therefore we obtain NP $\subseteq$ OPT. It is an open question whether OPT $\subseteq$ NP.

With these notions, we will take a look at important results in the field of optimal proof systems in the next chapter. For notions not defined in this thesis, refer to a standard work of computational complexity like the one by Papadimitriou [Pap94].

3 A Brief Overview of Proof Systems

After defining the important notions for this thesis, we will give a brief overview of some important results in the field of optimal proof systems.

3.1 Languages Possessing Optimal Proof Systems

One basic lemma that is widely used formalizes a part of the connection between optimal proof systems and polynomial many-one-reducibility. Later in this thesis, we will use it to proof corollary 12. The following proof is mainly taken from Köbler et al. We will omit the proof for optimal proof systems, as it easily follows from the proof for p-optimal ones.

Lemma 2 ([KMT03])**.** *If A has a (p-)optimal proof system and if $B \leq_m^p A$, then B has a (p-) optimal proof system, too.*

Proof. Let h be a p-optimal proof system for A and let B many-one reduce to A via $f \in \mathcal{FP}$, that is $x \in B \Leftrightarrow f(x) \in A$. Then h' defined by

$$h'(\langle x, w\rangle) = \begin{cases} x & (h(w) = f(x)), \\ \bot & (\text{otherwise}), \end{cases}$$

is a proof system for B, as $h(w) = f(x) \in A$ is equivalent to $x \in B$. To show h' is optimal, let g' be a proof system for B. In order to obtain a proof system for A, let g be

$$g(bw) = \begin{cases} h(w) & (b = 0), \\ f(g'(w)) & (b = 1). \end{cases}$$

Since both $h(w)$ and $f(g'(w))$ are in A and h is a proof system for A, g is also a proof system for A. As h is p-optimal, there is a function $t \in \mathcal{FP}$ translating g-proofs to h-proofs implying that

$$h(t(1w)) = g(1w) = f(g'(w)).$$

This implies $h'(\langle g'(w), t(1w)\rangle) = g'(w)$. Hence, h' p-simulates g'. □

In contraposition to this, we can state that for $B \leq_m^p A$, if B has no (p-)optimal proof system, then A has not either.

Using this lemma, for a language $L \in \mathrm{OPT}$, we can construct further languages possessing optimal proof systems. We call two languages A and B *polynomial time equivalent*, if $A \leq_m^p B$ and $B \leq_m^p A$. With this notion, we can easily state the following.

Corollary 3. *For polynomial time equivalent languages A and B, $A \in OPT$ if and only if $B \in OPT$. Especially, if $L \notin OPT$, then for $b \in \Sigma$, $bL := \{bw : w \in L\} \notin OPT$ as well as $0L \cup 1L \notin OPT$.*

Proof. From $A \leq_m^p B$ we obtain $B \in \text{OPT} \Rightarrow A \in \text{OPT}$, using lemma 2. Similar, we obtain $A \in \text{OPT} \Rightarrow B \in \text{OPT}$ from $B \leq_m^p A$.

We now construct polynomial time functions in order to show that L, bL and $0L \cup 1L$ are polynomial time equivalent. With the function discarding the first bit of any word, $bw \mapsto w$, we obtain $bL \leq_m^p L$. With the function adding b to the start of any word, $w \mapsto bw$, we obtain $L \leq_m^p bL$. With $x \mapsto 0x$, we obtain $L \leq_m^p 0L \cup 1L$, as $x \in L \Leftrightarrow 0x \in 0L \cup 1L$. Using $bw \mapsto w$, we further obtain $0L \cup 1L \leq_m^p L$. □

The following lemma gives a partial answer to the basic question, which languages actually have optimal proof systems. Although Koebler et al. omitted the proof of the following theorem due to its straightforwardness, we will give a proof for the purpose of this thesis.

Lemma 4 ([KMT03])**.** *(i) Every language in P has a p-optimal proof system.*

(ii) Every language in NP has an optimal proof system.

Proof. (i) Let $L \in \text{P}$. Then there is a function $f \in \mathcal{FP}$ with $f(w) = 1 \Leftrightarrow w \in L$. To show there is a proof system, let h be defined by

$$h(w) = \begin{cases} w & (f(w) = 1), \\ \bot & (\text{otherwise}). \end{cases}$$

Then h is a proof system for L. To show h is optimal, let h' be an arbitrary proof system for L. Then $h' \in \mathcal{FP}$ by definition and we can translate h'-proofs with h' in polynomial time into h-proofs, in formulas

$$h(h'(w)) = h'(w).$$

Therefore, h p-simulates every proof system h'.

(ii) Let $L \in \text{NP}$. Then there is a nondeterministic Turing transducer M deciding L in polynomial time. Let $f_i(x) \in \mathcal{FP}$ be the function calculating the i-th path of the nondeterministic calculation of M. Finally, let h be defined by

$$h(\langle i, w \rangle) = \begin{cases} w & (f_i(w) \text{ accepts}), \\ \bot & (\text{otherwise}). \end{cases}$$

Then $h \in \mathcal{FP}$ is a proof system for L. To show h is optimal, let h' be an arbitrary proof system for L. Let g be a function that maps an $w \in L$ to an i such that $f_i(w)$ accepts in polynomial time. Notice that g may be not in $\mathcal{FP}$. With these definitions, we obtain

$$h(\langle g(h'(w)), h'(w) \rangle) = h'(w).$$

Therefore, h simulates every proof system h' via the translating function

$$w \mapsto \langle g(h'(w)), h'(w) \rangle.$$

□

3.2 Proof Systems and the P Versus NP Question

As we have seen, $\text{NP} \subseteq \text{OPT}$. It is an open question whether $\text{NP} \supseteq \text{OPT}$. Lemma 4 connects to the P-versus-NP-question by stating different properties for P and NP: if one would find a set in NP without an p-optimal proof system, one would have separated P from NP.

Corollary 5. *If there is no p-optimal proof system for SAT, then $P \neq NP$.*

In order to prove proposition 1, we will show two lemmata of which the first is taken from the work of Cook and Reckhow [CR79]. It gives an equivalent formulation of the question whether NP = co-NP.

Lemma 6. *NP = co-NP if and only if $TAUT \in NP$.*

Proof. Assume TAUT $\in$ NP, then $\overline{\text{SAT}}$ is in co-NP. As SAT is NP complete, is follows that co-NP = NP. Assume TAUT $\notin$ NP, then $\overline{\text{TAUT}} \notin$ co-NP. As $\overline{\text{TAUT}} \in$ NP, we obtain NP $\neq$ co-NP. □

The second lemma connects with the theory of proof systems by formulating a necessary and sufficient condition for a language being in NP [CR79].

Lemma 7. *A set $L \neq \emptyset$ is in NP if and only if L has a polynomially bounded proof system.*

Proof. Assume $L \in$ NP, then some nondeterministic Turing transducer M accepts L in polynomial time. Let $f_i(x) \in \mathcal{FP}$ be the function calculating the i-th path of the nondeterministic calculation of M. We define f by

$$f(\langle i, x\rangle) = \begin{cases} x & (f_i(x) \text{ accepts}), \\ \bot & (\text{otherwise}). \end{cases}$$

Then f is a polynomially bounded proof system for L.

Conversely, assume f is a polynomially bounded proof system for L. Then a nondeterministic Turing transducer on input y can guess a proof x and verify $f(x) = y$. □

Putting lemma 6 and 7 together, we obtain proposition 1,

$$\text{NP} = \text{co-NP} \Leftrightarrow \text{ there is a polynomially bounded proof system for TAUT.}$$

Using this theorem, one tried to separate NP from co-NP by studying more and more powerful proof systems, showing that they are not polynomially bounded [KMT03]. As mentioned before, there was no success in answering this questions yet. To take the notion of optimal proof systems into account, one could ask if there is an optimal or even p-optimal proof system for TAUT. If that were the case, the existence of one specific proof system that is not polynomially bounded would suffice to proof that NP $\neq$ co-NP and hence P $\neq$ NP [KMT03].

Krajícek and Pudlák proved a sufficient condition for the existence of optimal proof systems for TAUT [KP89].

Theorem 8. *If NE = co-NE then optimal proof systems for TAUT exist. If E = NE then p-optimal proof systems for TAUT exists.*

We will omit their proof in this thesis, since it uses many notions of formal logics and a huge equivalence theorem not introduced here.

Corollary 9. *As a conclusion, we can state the following implication diagram.*

$$\begin{array}{ccccccc} & & & & & & E = NE \\ & & & & & & \Downarrow \\ P = NP & \Rightarrow & NP = co\text{-}NP & \Leftrightarrow & TAUT \in NP & \Leftrightarrow & TAUT \in OPT \\ & & & & & & \Uparrow \\ & & & & & & NE = co\text{-}NE \end{array}$$

4 A set in co-NEXP $\setminus$ OPT

In the main theorem of this thesis, we will show that in certain complexity classes above of NP there are always languages that possess no optimal proof system. To formalize this, we say a function is *time-constructible*, if there is a Turing transducer that stops on input n after $t(n)$ steps.

4.1 Basic Construction

Before stating and proofing the main theorem of this thesis, we will have a look at universal Turing transducers. In contrast to usual transducers, an universal transducer U is able to simulate any other given transducer M_α. Given a program α and an input y, it calculates $M_\alpha(y) = U(\alpha, y)$. Doing so, the runtime of U is quadratic slower than the runtime of M. We will need the propositions proven here later to proof theorem 11.

Lemma 10. *(i) There is an enumeration of all $\mathcal{FP}$-functions f_i such that for each function it holds* $\text{time}(f_i) \leq n^i + i$.

(ii) Let $\alpha \in \Sigma^*$ be an arbitrary program, and M_α the associated Turing transducer. Then there is an universal Turing transducer U that can simulate M_α on any input in $O(n^2)$ steps, where n is the number of steps M_α runs on the same input.

(iii) There is an universal Turing transducer U that simulates for every $i \in \mathbb{N}$ and every input $y \in \Sigma^*$ the function $f_i(y)$ in $O(n^{2i})$ steps, where $n = |y|$.

Proof. (i) Let $M_1, M_2, ...$ be Gödel's enumeration of all Turing transducers. We modify this enumeration by applying a clock to each transducer M_i that rejects the given input y after the $(|y|^i + |y|)$-th step, obtaining an enumeration $M_1', M_2', ...$ of machines runtime bounded by $n^i + i$. Now let $f_1, f_2, ...$ be the functions calculated by the transducers M_i'. As the runtime maximum is unbounded for $n \to \infty$, this enumeration contains all polynomial time functions.

(ii) Based on a similar proof in the book of Arora and Barak [AB09]. Without loss of generality, we can assume that M_α has only one work tape and runs on alphabet $\{\bigstar, \square, 0, 1\}$. This is possible, as if M_α runs on more tapes or uses a greater alphabet, we can construct a transducer M' that fulfills our requirements and uses $O(n^2)$ steps on input y, where n is the number of steps needed for the calculation of $M_\alpha(y)$.

Let now U be a Turing transducer with five tapes, an input tape, an output tape, a tape to store the description of M_α, a tape to store the current status in the simulation of M_α and finally a tape that simulates the working tape of M_α.

In order to simulate one computational step of M, U scans the transition table of M_α to obtain the new status, a symbol to write and the head movement. It then simulates the actions that would be done by M_α. As we can see, the simulation of one step of M_α takes c steps of U, where c is a number depending only on the size of the transition's function table of M.

(iii) By construction of the enumeration $f_1, f_2, ...$, for each f_i there is a transducer M_i with $\text{time}_M(y) \leq |y|^i + |y|$. As proven before, there is an universal transducer that can simulate M_i' with quadratic slowdown. It follows that $\text{time}_U(i, y) \leq (|y|^i + |y|)^2$, and therefore $\text{time}_U(i, y) \in O(|y|^{2i})$.

□

Theorem 11. *Let $t : \mathbb{N} \to \mathbb{N}$ be a time-constructible function such that for every polynomial p there is a number n with $p(n) \leq t(n)$. Then there is a language $L \in$ co-NTIME$(t(n))$ that has no optimal proof system.*

Notice that, if for a function f there is an $n \in \mathbb{N}$ with $f(n) \geq p(n)$ for every polynomial p, then for every polynomial p there are infinitely many $n \in \mathbb{N}$ for which $f(n) \geq p(n)$ is satisfied. Otherwise, we could construct a polynomial q that is greater than f for every $n \in \mathbb{N}$. To see this, let p be a polynomial for which $f(n) \geq p(n)$ for only finitely many $n \in \mathbb{N}$. Then we can determine the greatest difference between f and p by taking the maximum of the set $\eta = \max\{f(n) - p(n) : f(n) \geq p(n)\}$. Then for the polynomial $p'(n) = \eta + p(n) + 1$ there is no n with $f(n) \geq p'(n)$, which contradicts the assumption that for every polynomial p there is a natural number such that $f(n) \geq p(n)$.

Messner showed that under the same presumptions as in our theorem, there is a language $L \in$ co-NTIME$(t(n))$ without an optimal acceptor [Mes99]. He also proved that the existence of an optimal acceptor is equivalent to the existence of an optimal proof system for every p-cylinder L. We will here give a proof that is based on the work of Messner, but stays in the notion of proof systems.

Before formally proofing theorem 11, let us briefly discuss the main idea of the proof. As proven before, there is an enumeration of all $\mathcal{FP}$-functions f_i such that $\text{time}(f_i) \leq n^i + i$. We construct a language L by uniting languages L_i', which we construct in a way such that there are only long f_i-proofs for all strings in L_i'. Afterwards we assume there is an optimal proof system f_i for L. As for every $\mathcal{FP}$-function there is a subset that has only long proofs, there is such a subset L_i' for f_i. But as it turns out, we can state a proof system for L_i' that has short proofs, we can combine both proof systems to a system that runs polynomial equally fast on nearly every string in L, but way faster on strings in L_i'. Hence, f_i cannot be an optimal proof system.

Formal proof of theorem 11. Let $f_1, f_2, ...$ be a enumeration of all $\mathcal{FP}$-functions with $\text{time}(f_i) \leq n^i + i$. For any $i > 0$, let L_i be the regular language described by the expression $0^i 10^*$. Define

$$L_i' = \{x \in L_i | \forall_{y \in \Sigma^*} |y|^{2i} \leq t(|x|) \implies f_i(y) \neq x\}.$$

That is, as long as we put strings of length $|y|^{2i} \leq t(|x|)$ into f_i, we will not obtain x. Let $L = \bigcup_{i>0} L_i'$.

First, we prove $L \in$ co-NTIME$(t(n))$. To show this, one considers

$$L \in \text{co-NTIME}(t(n)) \Leftrightarrow \overline{L} = \overline{\bigcup_{i>0} L_i'} = \bigcap_{i>0} \overline{L_i'} \in \text{NTIME}(t(n)).$$

By negating the condition for L_i', we obtain

$$\overline{L_i'} = \{x \in \Sigma^* | x \notin L_i \vee \big(\exists_{y \in \Sigma^*} |y|^{2i} \leq t(|x|) \wedge f_i(y) = x\big)\}.$$

For any given x, we can decide in polynomial time whether it is in any L_i or not. If it is not, then x is in $\overline{L_i'}$ for all $i > 0$ and therefore $x \in \overline{L}$, so we are done. If it is in any L_i, it is in exactly one L_i. Let i^* be the set with $x \in L_{i^*}$. Using the abilities of the nondeterministic time bound, we now simulate the Turing transducer M_{i^*} that corresponds to f_{i^*} with an universal transducer

on every input $y \in \Sigma^*$ with $|y|^{2i} \leq t(|x|)$. As proven before in lemma 10, this is possible with quadratic slowdown. Hence the calculation takes $O((\text{time}_{M_{i^*}}(y))^2) = O(|y|^{2i})$ steps to complete. As $|y|^{2i} < t(|x|)$, this is within our time bound. We then can decide whether $x \in \overline{L}$. If, and only if, there is a path with $f_{i^*}(y) = x$, then $x \in \overline{L}$. In both cases, we obtain $\overline{L} \in \text{NTIME}(t(n))$.

For a proof system f_i with $f_i(\Sigma^*) = L$, we observe that $L'_i = L_i$. Assume there is an $x = 0^i 1z \in L_i$ that is not in L'_i. Then there is an y with $|y|^{2i} \leq t(|x|)$ and $f_i(y) = x$. Since f_i is a proof system for L, this yields $x = 0^i 1z \in L$ and so $x \in L'_i$, which contradicts the assumption. Therefore, for any y with $f_i(y) = x \in L_i$ we know that $|y|^{2i} > t(|x|)$. Speaking informally, every proof system f_i for L has long proofs on $L'_i \subset L$.

Assume now, for contradiction, that f_i is an optimal proof system for L. Let g be a function defined as

$$g(bx) = \begin{cases} f_i(x) & (b = 0), \\ x & (b = 1 \text{ and } x = 0^i 10^* \in L_i = L'_i). \end{cases}$$

g is a proof system for L with polynomial length-bounded proofs for all $x \in L_i$. As f_i is optimal, there is a function f^* such that for all $x \in L'_i$, $f_i(f^*(x)) = g(x)$ and $|f^*(x)| \leq p(|x|)$ for a polynomial p. Let q be the polynomial $q(n) = p(n)^{2i}$. Then $p(|x|) \leq p(|x|)^{2i}$. As there is an n with $q(n) \leq t(n)$, there is an x in L_i such that $|f^*(x)| \leq p(|x|) \leq q(|x|) = p(|x|)^{2i} \leq t(|x|)$. According to the definition of L'_i, this yields $f_i(f^*(x)) \neq x$. Therefore, f_i is not optimal on L'_i, which contradicts the assumption that f_i is optimal on L. □

Using the relation to many-one-hard reducible sets proven in lemma 2, we obtain

Corollary 12. *No set $\leq^p_m$-hard for co-NE has an optimal proof system.*

Proof. With $t(n) = 2^n$, we can get an $L \in$ co-NE that has no optimal proof system. Any $\leq^p_m$-hard set A for co-NE is $L \leq^p_m A$. Together with lemma 2 we obtain, that A cannot have an optimal proof system. □

4.2 Making the Language Tally

Now, let us take a closer look at this set L that has no optimal proof system. One first observation is that L is sparse. As every L'_i only contains strings that are of the form $0^i 10^*$, L is a subset of the regular language $L_R = 0^*10^*$. Therefore, the density of L_R is an upper bound for the density of L. In L_R, there are exactly n words of length n. As a consequence, $\text{dens}_{L_R}(n) = \sum_{i=1}^{n} i = \frac{n^2+n}{2}$. Hence, L_R and L are both sparse. But we can construct even less dense languages by constructing tally sets without optimal proof systems.

Theorem 13. *If $L \subseteq 0^*10^*$ has no optimal proof system, then there is a $T \in$ TALLY, that has no optimal proof system and is polynomial time equivalent to L.*

Proof. Using the Cantor pairing function, we can construct a bijective function $t \in \mathcal{FP}$ thats maps each element of L one-to-one to an element of $T \subseteq$ TALLY. Since all strings in L are almost tally—namely every string in L contains only one 1—t is a polynomial time function.

Let g be an optimal proof system for $T = \{t(x), x \in L\}$. Then $t^{-1} \circ g$ is a proof system for L. Let h be an arbitrary proof system for L. Then $t \circ h$ is a proof system for T. As g is optimal, there is a polynomial bounded f with $g(f(x)) = t(h(x))$. It follows that $t^{-1}(g(f(x))) = h(x)$ for all proof systems h for L. Hence, $t^{-1} \circ g$ is an optimal proof system for L, which contradicts the assumption that L has no optimal proof system. Using t we obtain $L \leq^p_m T$ and $T \leq^p_m L$. □

Using this theorem, we can further improve our results on the density of languages not possessing an optimal proof system. For any tally language $p(n) = n$ is an upper bound of its density.

Corollary 14. *There is a language L with $\mathrm{dens}_L(n) \leq n$ that possesses no optimal proof system.*

4.3 Super-Sparse Sets without an Optimal Proof System

By adapting the proof of our main theorem 11, we can show that there are even arbitrary sparse sets in co-NTIME($t(n)$) that do not possess an optimal proof system. To formalize the notion of arbitrary sparse, we introduce a polynomial-time computable, strictly increasing function $u : \mathbb{N} \to \mathbb{N}$. Notice, as u is strictly increasing, we have $u(n) \geq n$ for all $n \in \mathbb{N}$.

We will use u to constrain the length of any string in L. Therefore, we define

$$\widetilde{L}'_i = \{x \in L'_i : \exists_{k \in \mathbb{N}} |x| = u(k)\}.$$

By definition we obtain $\widetilde{L}'_i \subseteq L_i$. We further define $\widetilde{L} = \bigcup_{i>0} \widetilde{L}_i$. By doing so, in $\widetilde{L}$ will only be strings whose length is in the range of u. Informally speaking, the faster u is growing, the more sparse becomes $\widetilde{L}$. By choosing fast-growing u, we can obtain arbitrary sparse sets in $\overline{\mathrm{OPT}}$.

In order to adapt the proof of theorem 11 to this new circumstances, we have to discuss one issue. It occurs in the proof that $\widetilde{L}$ is in co-NTIME($t(n)$). In order to decide whether an given string x is in the complement of $\widetilde{L}$, we have to additionally check in polynomial time whether there is an $k \in \mathbb{N}$ such that $|x| = u(k)$. Since $u \in \mathcal{FP}$ and is strictly increasing, this is possible. As for any $k > |x|$, it follows $u(k) \geq |x|$, we have to calculate at most $|x|$ times the value of $u(k)$ in order to check if there is an k with $u(k) = |x|$. If there is no such k, then the given x is for sure in the complement of $\widetilde{L}$ and we are done. If there is such a k, we proceed with the algorithm as described before.

Adapting the proof as described yields

Corollary 15. *Let $t : \mathbb{N} \to \mathbb{N}$ be a time-constructible function such that for every polynomial p there is a number n with $p(n) \leq t(n)$. Let $u : \mathbb{N} \to \mathbb{N}$ be a polynomial-time computable, strictly increasing function. Then there is a language $L \in$ co-NTIME($t(n)$) that has no optimal proof system and possesses only strings whose length is in $u(\mathbb{N})$.*

4.4 Adding Redundancy

Given a language L not possessing an optimal proof system, we can show that there is even a mitotic language that has no optimal proof system. To show this, we mainly rely on results proven in chapter 3.

Theorem 16. *If L possesses no optimal proof system, there is an polynomial time equivalent, many-one-mitotic language $L' = 0L \cup 1L$ not possessing an optimal proof system, too.*

Proof. As L has no optimal proof system, corollary 3 states that $L' = 0L \cup 1L$ is polynomial time equivalent and does not possess an optimal proof system.

To show that L' is many-one-mitotic, let A be the language $0\Sigma^*$, then $\overline{A} = 1\Sigma^* \cup \{\varepsilon\}$. We obtain $L' \cap A = 0L$ and $L' \cap \overline{A} = 1L$. As shown in corollary 3, L and bL are polynomial time equivalent. We obtain $0L$ and L polynomial time equivalent, as well as L and $1L$. Hence, L' is many-one-mitotic. □

4.5 Putting the Results Together

In this section, we will show that we can combine all results into one language, that is arbitrary sparse, tally, has no optimal proof system and possesses redundancy.

Corollary 17. *Let $t : \mathbb{N} \to \mathbb{N}$ be a time-constructible function such that for every polynomial p there is a number n with $p(n) \leq t(n)$. Then there is a language $T \subseteq \Sigma^*$ that is polynomial time equivalent to a language of co-NTIME($t(n)$), possesses no optimal proof system and is*

(i) arbitrary sparse,

(ii) many-one-mitotic,

(iii) and tally.

Proof. Let u be a polynomial-time computable, strictly increasing function $u : \mathbb{N} \to \mathbb{N}$. As we have seen in corollary 15, we can obtain a language $L \in$ co-NTIME($t(n)$) that has no optimal proof system and possesses only strings whose length is in the range of u. As there is no upper bound for growth of u, L is arbitrary sparse.

By using theorem 16, we obtain the language $0L \cup 1L$ that is polynomial time-equivalent to L and possesses no optimal proof system. Is contains only strings of length $u(\mathbb{N}) + 1$; is density is given by $\mathrm{dens}_{0L \cup 1L}(n) = 2\,\mathrm{dens}_L(n)$.

Finally, we can use theorem 13 to construct a tally set $T \notin$ OPT that is polynomial time-equivalent to L. As we use a strictly increasing pairing function, T is even more sparse than $0L \cup 1L$. Is follows that $\mathrm{dens}_T(n) \leq 2\,\mathrm{dens}_L(n)$. Hence, T is arbitrary sparse. □

By choosing $t(n) = k^n$ respectively $t(n) = 2^{n^k}$, we can obtain all results for the complexity classes co-NE respectively co-NEXP.

5 Conclusion and Future Work

5.1 Conclusion

In this thesis, we presented some fundamental insights into proof systems and the notion of the optimality of proof systems. The main motivation to study proof systems was given by some implications that propositions on proof systems could have on the P versus NP question and the separation of NP and co-NP.

In the first part, we proved some basic lemmata and showed which relatively easy complexity classes contain languages that possess optimal proof systems. The proof that every language L in P or NP has an optimal proof system was straightforward and did not depend on results proven earlier. Differently, we needed more work to prove the proposition establishing the connection between proof systems and the P versus NP problem.

Subsequently, we moved on to complexity classes above the polynomial ones. We showed that by leaving the polynomial area, we can construct a language in co-NTIME($t(n)$) that does not possess an optimal proof system. By further research on L, we could state that we even obtain sparse, tally or redundant languages that do not have an optimal proof system. In conclusion, we succeeded on combining these properties in one language, uniting all previous stated attributes.

5.2 Future Work

As mentioned before, the question whether NP = OPT remains still open. Although we could easily prove NP $\subseteq$ OPT, we do not know if every set that possesses an optimal proof system is in NP.

In order to get some insight into this question, we could define variations of the simulation notion for proof systems. Let h, h' be proof systems for L. One could define h *simulates* h' f*-bounded*, if there is a function g such that for every string $w \in \Sigma^*$

$$h(g(w)) = h'(w)$$

and $|g(w)| \leq f(|w|)$. By doing so, we obtain f*-optimal* proof systems in the same way as defined before and are able to define complexity classes OPT_f. One could investigate whether $\mathrm{OPT}_f \supseteq \mathrm{NP}$ for an arbitrary f, like $f(n) = c \cdot n$ or $f(n) = n + c$.

Bibliography

[AB09] Sanjeev Arora and Boaz Barak, *Computational complexity: A modern approach*, 1st ed., Cambridge University Press, New York, NY, USA, 2009.

[Coo71] Stephen A. Cook, *The complexity of theorem-proving procedures*, Proceedings of the third annual ACM symposium on Theory of computing (New York, NY, USA), STOC '71, ACM, 1971, pp. 151–158.

[CR79] Stephen A. Cook and Robert A. Reckhow, *The relative efficiency of propositional proof systems*, Journal of Symbolic Logic **44** (1979), 36–50.

[For09] Lance Fortnow, *The status of the P versus NP problem*, Commun. ACM **52** (2009), no. 9, 78–86.

[KM00] Johannes Köbler and Jochen Messner, *Is the standard proof system for SAT p-optimal?*, Proceedings of the 20th Conference on Foundations of Software Technology and Theoretical Computer Science (London, UK, UK), FST TCS 2000, Springer-Verlag, 2000, pp. 361–372.

[KMT03] Johannes Köbler, Jochen Messner, and Jacobo Torán, *Optimal proof systems imply complete sets for promise classes*, Inf. Comput. **184** (2003), no. 1, 71–92.

[KP89] Jan Krajícek and Pavel Pudlák, *Propositional proof systems, the consistency of first order theories and the complexity of computations*, J. Symb. Log. **54** (1989), no. 3, 1063–1079.

[Mes99] Jochen Messner, *On optimal algorithms and optimal proof systems*, Proceedings of the 16th annual conference on Theoretical aspects of computer science (Berlin, Heidelberg), STACS'99, Springer-Verlag, 1999, pp. 541–550.

[Pap94] Christos H. Papadimitriou, *Computational complexity*, Addison-Wesley, 1994.

Hiermit versichere ich, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen Hilfsmittel und Quellen als die angegebenen benutzt habe. Weiterhin versichere ich, die Arbeit weder bisher noch gleichzeitig einer anderen Prüfungsbehörde vorgelegt zu haben.

Würzburg, den _____, _____

(Nils Wisiol)