FACULTY
OF MATHEMATICS
AND PHYSICS
Charles University

MASTER THESIS

Daria Bilan

Distinguishing Pairs of Words Using Finite Automata

Department of Applied Mathematics

Supervisor of the master thesis: prof. Mgr. Michal Koucký, Ph.D.
Study programme: Computer Science
Study branch: Discrete Models and Algorithms

Prague 2023

Title: Distinguishing Pairs of Words Using Finite Automata

Author: Daria Bilan

Department: Department of Applied Mathematics

Supervisor: prof. Mgr. Michal Koucký, Ph.D., Computer Science Institute

Abstract: This work considers a fundamental open problem in informatics - distinguishing two words by a deterministic finite automaton with the smallest possible number of states. We review the existing research, where proven lower and upper bounds in terms of words' lengths differ exponentially. Next, we empirically try two approaches not studied previously: analysis of discerning sets and application of randomly generated automata. We show that the first approach does not help improve the bounds for the main problem, while the random automata could be successful for randomly taken word pairs but not for all of them. A combination of a randomly generated automaton with the best performing already known, though, may decrease an average number of states by several orders of magnitude. We propose several topics for further investigation based on the obtained experimental results.

Keywords: finite automata, distinguishing words, complexity, random automata

Contents

Introduction

One can be surprised by the apparent simplicity of some problems in the informatics world, which are still open. Take possibly the simplest one - what are the minimum needed resources to determine if two strings are identical? Considering a computational model of the deterministic finite automaton, the problem was stated in the 1980s and is still open nowadays, while some bounds were obtained in various studies.

The choice of formalism acknowledges the fact that deterministic finite automata is an important and extensively used concept in the computer science field. Automata are used to study the behavior of simple machines with finite memory under mathematical formalism. Their model captures the essential computation features, such as input, output, memory, states, and transitions between them. Although it is an abstract concept, automata may be (and often are) implemented in hardware or software to solve some specific problems, such as pattern matching or lexical analysis. They are used in compilers, artificial intelligence, robotics, and many more fields. Besides their practical use, they are also utilized to theoretically model and analyze the behavior of various discrete systems.

The question addressed in this work is very simple: given two words, what is the minimum size of a deterministic finite automaton that accepts one and rejects the other, thus allowing us to say that those words are different? This formulation is quite original, as we are interested only in distinguishing two given words – automaton's behavior for any other input is not essential. Moreover, the input is known in advance so that we may tailor the automaton to those exact words. The main goal is to provide some bounds for the number of automaton's states depending on the input lengths. Such a significant component as discerning two inputs should not be a restrictive part of any research, so it is important to minimize the considered problem. The examined area is connected, for example, to the topics of data compression [Johnson, 1986] and distinguishing Kolmogorov complexity [Allender et al., 2003].

The research was started at Charles University by Goralčík and Koubek [1986], though authors acknowledged that the question was suggested to them by Christian Choffrut. Later on, Robson [1989] and Chase [2021] gradually improved the suggested upper bound, and Demaine et al. [2011] examined the lower bound for the problem. Obtained bounds, however, are still exponentially far from each other, with the lower being $\Omega(\log n)$ and the upper $O(\sqrt[3]{n})$, where n is the maximum length of the input words. We should also mention Wiedermann [2016], who proposed a significantly different approach to the studied problem, which we describe later.

This thesis discusses existing approaches and presents tools that may help further research. The remainder of this work is organized as follows. We introduce needed preliminaries in Chapter 1, formulate the problem and discuss special cases in Chapter 2, and present related work with proven bounds for the problem in Chapter 3. Then we discuss our experimental results considering the problem.

In Chapter 4, we further develop the ideas of Wiedermann [2016] using a conversion of studied automata to so-called discerning sets. Although we prove that the presented approach is too weak to improve the upper bound for the studied problem, we are convinced that the presented concept and experimental results deserve further research, possibly not in relation to this thesis's subject.

In Chapter 5, we examine the behavior of randomly generated automata with respect to the given topic. We compare the performance of different automata constructions and define a specific subclass of random automata whose elements of constant size perform well on average on randomly generated words without the need to tailor the automaton to a specific input. This allows us to reduce the average needed automaton's size by several orders of magnitude using two simple steps. First, we randomly generate an automaton that will distinguish between the words with a high probability. If it does not declare the words different, we proceed with the deterministic approach according to the best results on the subject to date. Though this approach differs from the problem formulation, it may be successfully used in practice.

Finally, we summarize the results of this work and propose several topics for further research in the Conclusion.

1. Preliminaries

In this chapter, we briefly describe the concept of automata for readers who are not entirely familiar with it. Although presented facts may be considered common knowledge, we base our presentation on the book "Introduction to Automata Theory, Languages, and Computation" by Hopcroft, Motwani, and Ullman [2001]. We start with some basic terminology in Section 1.1, continue with several definitions of deterministic finite automaton in Section 1.2, define a couple of operations over automata in Section 1.3 and finish with defining some specific automata types in Section 1.4.

1.1 Basic definitions

Definition 1 (Alphabet)**.** *An* alphabet Σ *is a non-empty finite set of symbols.* Unary alphabet *is an alphabet containing one symbol,* binary alphabet *is a term used for* $\Sigma = \{0, 1\}$*.*

Definition 2 (Word)**.** *A* word $w \in \Sigma^*$ *is a finite sequence of symbols from the alphabet* Σ*.* $|w|$ *denotes the length of the word* $w \in \Sigma^*$*. Word of zero length is usually denoted as* ε*.* w_i *denotes the* i*-th symbol of the word* w *(counting from* 1*) and* $w_{a\ldots b}$ *the slice of the word* w *from the index* a *(including) to the index* b *(excluding).* w_{-1} *is the last symbol of the word.*

Definition 3 (Words concatenation)**.** *We denote the concatenation of words* u *and* v *by* $u \cdot v$*.*

Remark. In the whole text of this work, log denotes a natural logarithm. Most studies on the topic omit the base of the used logarithms, as it is not essential as long as only the asymptotic complexity of the problem is considered. As we will see in the following text, most of the obtained results in the related work follow from one crucial lemma, which we present in the next section (see Lemma 21). We see that whenever $O(\log n)$ result is obtained using this lemma, the actual bound is $4 \ln n$. We want to present the most precise bounds and avoid asymptotic notation where possible to show that there is no big hidden constant in the obtained complexity. However, we want the text to be consistent with the rest of the studies, so we use log instead of ln.

1.2 Deterministic finite automata

In this section, we recall the most essential definition of this work: a deterministic finite automaton. Automaton is an abstract computational model which simulates input processing by a system with a limited number of states, where only some information can be remembered. The output of the processing is the state where the system ends after reading the whole input, or, even more simply, a boolean value determining whether or not the input meets some requirements which we are testing.

Let us start with a simple example of an automaton answering the question of whether the number of ones in a given binary input is divisible by 3. In Figure

1.1[1] we see an automaton with three states q_0, q_1 and q_2, the automaton starts in the state q_0 (denoted by the input arrow without any symbol). Then, if the first symbol of the input is 1, the automaton moves to the state q_1; otherwise (if the input symbol is 0), it remains in q_0. Then, the following input symbol is read, and the process continues until we reach the end of the input. The process of moving to another state is always determined by a symbol that is read and guided by the arrow with the corresponding symbol on it. It is easy to see that, after reading the whole input, the automaton will be in q_0 if and only if the number of 1's in the input is divisible by 3. That is why q_0 is also marked as an accepting state (denoted by a double circle around the state). Words finishing in the accepting state q_0 are *accepted* by this automaton; all others are *rejected*. Thus, we claim that this automaton accepts the words with the number of 1's divisible by 3, and only them.

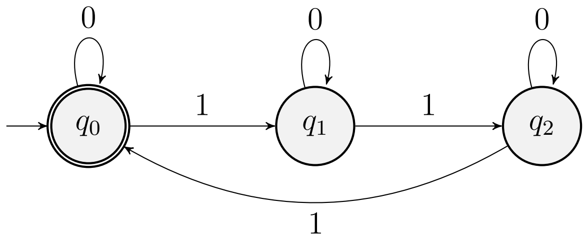


Figure 1.1: Example of an automaton testing if the number of 1's in the input is divisible by 3.

States of the automaton maintain some information about the input read so far. In the preceding example, this information is the remainder of the number of 1's in the input modulo 3. In the example, we asked a true/false question: whether this remainder is zero. However, we may also be interested in a question about what this remainder actually is. Then, it is natural to look at the automaton as a function that outputs the final state for a given input. We may classify all existing words according to the final state: ones that finish in q_0 have the number of 1's divisible by 3, ones in q_1 are those where the number of 1's has a remainder 1 modulo 3, and in q_2 are the ones with the number of 1's having a remainder 2 modulo 3.

As we will be interested in finding an automaton that, for the given two words u and v, accepts one and rejects the other, it is natural to see the automaton M as a function $M(w) : \Sigma^* \to Q$ giving the final state for each word and ask whether $M(u) = M(v)$, omitting the definition of accepting states. First, we will give a classical definition of a deterministic finite automaton for the sake of clarity (see Definition 4), then we will tailor it for our purposes (see Definition 6).

Definition 4 (Deterministic finite automaton)**.** *A* deterministic finite automaton (DFA) *$M = (Q, \Sigma, \delta, q_0, F)$ is a 5-element tuple, where*

- *Q is a finite set of states,*
- *Σ is an input alphabet,*

[1] Automata diagrams in this thesis are generated using **tikz** package as described in [Sikdar, 2017]

- *$\delta : Q \times \Sigma \to Q$ is a transition function which for every state $q \in Q$ and input symbol $a \in \Sigma$ gives a state $\delta(q, a)$ in which automaton gets after reading symbol a while being in the state q,*
- *q_0 is a starting state where the automaton begins the word's processing,*
- *F is a set of* accepting *states, $F \subseteq Q$.*

We define a size *of automaton $|M|$ as the number of its states $|Q|$.*

The word *deterministic* in the definition refers to the fact that for every input symbol and every state, exactly one transition is defined (in contrast with NFA - nondeterministic finite automaton). *Finite* means that we work with a finite number of states in the system.

Definition 5 (Extended transition function)**.** *For a given transition function $\delta : Q \times \Sigma \to Q$ of an automaton $M = (Q, \Sigma, \delta, q_0, F)$, we define* the extended transition function *$\hat{\delta} : Q \times \Sigma^* \to Q$ so that $\hat{\delta}(q, w)$ determines where the automaton M will finish after reading $w \in \Sigma^*$ starting in $q \in Q$. We define its value recursively as follows.*

$$\hat{\delta}(q, w) = \begin{cases} q, & \text{if } w = \varepsilon, \\ \hat{\delta}(q', w_{2\dots|w|}), \text{ where } q' = \delta(q, w_1), & \text{otherwise.} \end{cases}$$

Definition 6 (DFA as a function)**.** *A* deterministic finite automaton *M is a function $\Sigma^* \to Q$ defined as a 4-element tuple (Q, Σ, δ, q_0), where*

- *Q is a finite set of states,*
- *Σ is an input alphabet,*
- *$\delta : Q \times \Sigma \to Q$ is a transition function which for every state $q \in Q$ and input symbol $a \in \Sigma$ gives a state $\delta(q, a)$ in which automaton gets after reading symbol a being in the state q,*
- *q_0 is a starting state where the automaton begins the word's processing.*

We define $M(w)$ as a final state after the automaton M reads the word $w \in \Sigma^*$. It holds that $M(w) = \hat{\delta}(q_0, w)$.

Definition 7. *We say that automaton M* distinguishes *the words $u, v \in \Sigma^*$ if $M(u) \neq M(v)$.*

What happens if we change the starting state of the automaton? Processing of the words will be different, and we may or may not end up in a different state. In the example in Figure 1.1, if we start processing the word in q_1 instead of q_0, words with the number of 1's divisible by 3 will end in q_1 instead of q_0. Naturally, if we want to distinguish two words by a given automaton, it is sufficient that at least one ending state will be different for them if we try all the starting states (clearly, in each trial, we are starting in the same state for both words). Thus, we can furthermore simplify the definition of DFA by omitting the starting state and defining the output for each word as a tuple of final states for each starting state.

Definition 8 (Simplified DFA). *A simlified deterministic finite automaton $\hat{M}$ is a function $\Sigma^* \to Q^{|Q|}$ defined as a triple (Q, Σ, δ), where*

- *Q is a finite set of states,*
- *Σ is an input alphabet,*
- *$\delta : Q \times \Sigma \to Q$ is a transition function which for every state $q \in Q$ and input symbol $a \in \Sigma$ gives a state $\delta(q, a)$ in which automaton gets after reading symbol a being in the state q,*

$$\hat{M}(w) = \big(M_q(w)\big)_{q \in Q},$$

where $M_q(w)$ is the output of an automaton $M_q = (Q, \Sigma, \delta, q)$ defined as above.

Definition 9. *We say that a simplified DFA $\hat{M}$* distinguishes *the words $u, v \in \Sigma^*$ if $\hat{M}(u) \neq \hat{M}(v)$ (i.e., the tuples corresponding to both words differ in at least one element).*

Definition 10 (Automaton size). *We define the* size *of a (simplified) DFA $M = (Q, \dots)$ as the number of its states. We denote it by $|M| = |Q|$.*

1.3 Automata operations

There are multiple ways to combine two automata into one. The choice of a particular method depends on the desired behavior of the resulting automata. In this section, we define a *concatenation* of two automata, which will be useful in further chapters. We define it for a classic automata definition, where the automaton is a quintuple with a set of accepting states. If the input automata are defined as a function, i.e., as a quadruple without the accepting states, the accepting states of the resulting automaton can be trivially omitted.

Concatenation of the two automata produces an automaton that starts with simulating the work of the first given automaton and then switches to the second one when the first automaton has nothing left to do (we define this as the first automaton getting to the *terminal state*, from which he cannot exit, i.e., all transitions from this state lead to the same state).

Definition 11 (Terminal state). *State $t \in Q$ is called* terminal *in an automaton $M = (Q, \Sigma, \delta, q_0, F)$ if $\delta(t, a) = t$ for any $a \in \Sigma$. Usually, terminal states are also the accepting ones.*

Definition 12 (Automata concatenation). *Given two automata M_1 and M_2, where $M_1 = (Q_1, \Sigma, \delta_1, q_0^1, F_1)$, $M_2 = (Q_2, \Sigma, \delta_2, q_0^2, F_2)$, $Q_1 \cap Q_2 = \emptyset$ and M_1 contains exactly one terminal state $f \in Q_1$, we define their concatenation as an automaton $M_1 \circ M_2 = M = (Q_1 \cup Q_2 \setminus \{q_0^2\}, \Sigma, \delta, q_0^1, F_2)$, where transition function δ is defined as follows.*

$$\delta(q, a) = \begin{cases} \delta_1(q, a), & \text{if } q \in Q_1 \setminus \{f\}, \\ \delta_2(q_0^2, a), & \text{if } q = f, \\ f, & \text{if } q \in Q_2 \text{ and } \delta_2(q, a) = q_0^2, \\ \delta_2(q, a), & \text{otherwise.} \end{cases}$$

1.4 Automata types

In this section, we define *a permutation automaton*, which is mentioned plenty of times in this thesis as well as in the related work. Furthermore, we define *random* automata, to whom Chapter 5 is dedicated.

Definition 13 (Permutation automaton)**.** *An automaton $M = (Q, \Sigma, \delta, q_0)$ is a* permutation automaton *if all partial transition functions $\delta_a(q) = \delta(q, a)$ form a permutation of the states Q. Equivalently, the definition states that every state $q \in Q$ has exactly one input arrow with each symbol. I.e., if we define $I(q, a) = \{q' \in Q \mid \delta(q', a) = q\}$, $|I(q, a)| = 1$ for each $q \in Q$, $a \in \Sigma$.*

In literature, one can encounter different names for the same concept. A permutation automaton is also often called *reversible* [Pin, 1992] or *injective* [Hall, 1984].

Definition 14 (Random automaton)**.** *We define a construction of a* random automaton *$M = (Q, \Sigma, \delta, q_0)$, where Q, Σ and q_0 are given, in a way that for every $q \in Q$ and $s \in \Sigma$, we choose a value for $\delta(q, s)$ uniformly at random from Q.*

We want to point out that one can often find a slightly different definition of this term in various sources: *random automaton* of size n is a DFA taken uniformly at random from the collection of all $n^{|\Sigma|n}$ deterministic finite automata of size n over alphabet Σ [Nicaud, 2014, Chapuy and Perarnau, 2023]. This definition is similar to ours, as by the described construction, we obtain one of those automata, each one with the same probability.

Finally, we present a combination of two definitions above - a *random permutation* automaton.

Definition 15. *We define a construction of a* random permutation automaton *$M = (Q, \{0, 1\}, \delta, q_0)$, where Q and q_0 are given, in a way that we choose two permutations π_0, π_1 of the set of states Q uniformly at random and assign $\delta(q, 0) = \pi_0(q)$ and $\delta(q, 1) = \pi_1(q)$. It is clear that this construction yields a permutation automaton as introduced in Definition 13.*

Now we have all we need to proceed with the definition of this thesis's main problem in the next chapter.

2. Problem statement

In this chapter, we formulate the main problem of this thesis and discuss some special, easily solvable cases. We define a precise formulation in Section 2.1, discuss several dual problems in Section 2.2, present the solutions for some trivial cases in Section 2.3, and discuss what is remaining besides those in Section 2.4.

2.1 Formulation

Let us now formulate the problem this thesis is all about. As mentioned earlier, our goal is to distinguish between two words using the smallest possible automata. Given two words u, v of limited length, we are looking for an automaton that will distinguish between them in terms of accepting one and rejecting the other (or, equivalently, finishing in the different states after reading those words). We want this automaton to have as few states as possible. We stick to the definitions as presented in [Goralčík and Koubek, 1986].

Definition 16 (Discernibility)**.** *Let $u, v \in \Sigma^*$ be distinct words over finite alphabet Σ. We say that u and v are* d-discernible *if there exists an automaton with d states which accepts u and rejects v.*

Definition 17. *Let $u, v \in \Sigma^*$ be again distinct words over finite alphabet Σ. We say that $D(u, v) = d$ if u and v are d-discernible, and u and v are not d'-discernible for any $d' < d$.*

Definition 18 (Worst-case discernibility function)**.** *We define a function $f(n)$, so-called* worst-case discernibility function*, as a minimum value for each $n \in \mathbb{N}$ such that $D(u, v) \leq f(n)$ for any two distinct words u, v such that $|u|, |v| \leq n$.*

In Section 2.3.3, we will see that $f(n)$ does not depend on a specific alphabet Σ or its size as long as it contains more than one symbol. It is also easy to see that $f(n) \leq n + 1$, as one can trivially construct an automaton with $n + 1$ states which accepts the given word of length n and only that. While there exist proven lower and upper bounds for $f(n)$, they are asymptotically far from each other and should be improved. We discuss them in Chapter 3. It is believed that $f(n)$ actually behaves logarithmically, which corresponds to the proven lower bound, while the proven upper bound nowadays is as far as $O(\sqrt[3]{n})$.

Conjecture 19 (Discerning problem)**.**

$$f(n) \in \Theta(\log n)$$

This thesis aims to examine $f(n)$ behavior: present existing proofs, study some special cases, and experiment with approaches not studied before.

2.2 Dual problem

There are several possibilities to reverse the discerning problem, which we briefly describe in this section.

2.2.1 Discernible automata

First, a natural inversion of the studied question can be stated as follows. Given the two automata M_1 and M_2, what is the shortest word distinguishing between them (i.e., one automaton accepts the word, and another one rejects it)? This question arose very early in the studies of the automata theory and is solved routinely by construction and automata minimization techniques with a tight bound for the word length to be $\leq |M_1| + |M_2| - 2$ [Shallit, 2009, Theorems 3.10.5 and 3.10.6].

The relationship of this question to the main problem of this thesis is unclear, but it presents an interesting insight into a related problem.

2.2.2 Indiscernible words

The second possible dual question is, given an automaton M, to find two words that it can not distinguish. It makes sense to generalize this approach and consider a simplified DFA $\hat{M}$ instead of a normal one. Thus, we are looking for such words u, v that $\hat{M}(u) = \hat{M}(v)$. This is clearly a stronger question than words non-distinguishable by a classic 5-tuple DFA, as we are not restricted by a starting state, and a given automaton should not distinguish the words regardless of the state it starts in. This problem statement was not studied as it is, as far as we are concerned. However, there is an area whose results are useful in answering the question - *synchronizing words.*

Definition 20. *The word w is called* synchronizing *for a simplified DFA $\hat{M}$ if vector $\hat{M}(w)$ is constant (i.e., contains one and only state).*

That is to say, if any word contains a synchronizing word as a suffix, the final state after reading it is uniquely determined and does not depend on the starting one. Thus, if we have a synchronizing word for a given automaton, we can construct a pair of words it can not distinguish by adding any prefixes to a synchronizing one. However, the question of synchronizing words remains unsolved in general: first, not every DFA has such a word (e.g., the example automaton from Figure 1.1 clearly does not have one), and second, the bound for the word length is not proven. There exists an automaton that has the shortest synchronizing word of length $(|M|-1)^2$, which is also an upper bound conjectured by Černy [1964], but the proven upper bound is $O(n^3)$[Shitov, 2019]. Thus, the length of a synchronizing word is (if exists) at least quadratic in terms of the number of automaton states.

But are those considerations applicable to the primal problem? If we are able to find the word of a given length, which will be synchronizing for a lot of small automata, then it will be harder to provide a discerning automaton of a small size, as any inputs containing the word as a suffix will be indistinguishable. However, it is not clear if words that are synchronizing for some automata possess some unique qualities. I.e., even with a sufficiently large word, it is hard to say if it will be synchronizing for some automaton. Moreover, a synchronizing word is clearly doing much more than we need to ensure that two words are indistinguishable. For those reasons, this approach is not applicable to the main problem of this thesis.

Another way to look at this question is to count all possible values of a mapping $\hat{M}(u)$. Clearly, if $\hat{M}$ represents an automaton with m states, there are m^m possible values for $\hat{M}(u)$. Let us assume that we are working with words from the binary alphabet. Then, there are $2^{n+1} - 1$ words of length up to n, thus, if $2^{n+1} - 1 > m^m$, there necessarily exists a pair of words indistinguishable by $\hat{M}$. Omitting negligible 1's, we have that $n \geq m \log_2 m$. That is to say, for any simplified automaton of size m, there necessarily exists a pair of indistinguishable words of length $m \log_2 m$, which is less than quadratic and thus gives a stronger result than with synchronizing words. Gimadeev and Vyalyi [2010] show that for a permutation automaton (see Definition 13), this bound can be stated as $n \geq \sqrt{m} \log m$, i.e., for a simplified permutation automaton of size m there necessarily exists an indistinguishable pair of words of length $\sqrt{m} \log m$. This bound is more powerful than the previous one, as the length of the words is smaller than the number of automaton's states.

This problem can be related to the main one in terms of the existence of an automaton distinguishing between all the words of a given length. If we fix the word length n, presented results give us a lower bound of the size of such an automaton - if an automaton is too small with respect to the length of the word, it would not be capable of distinguishing between all the words due to the limited number of states it has. However, it turns out that already obtained results for the main problem ($m \in O(\sqrt[3]{n})$, see Chapter 3) are stronger than any of the suggested bounds. Thus, we may surely say that there does not exist a sufficiently small automaton distinguishing between all the words of length up to n simultaneously, and the solution to the main problem should be tailored to the two input words.

2.3 Trivial cases

This section describes several special cases where estimating bounds for $f(n)$ is easy or can be reduced to a simpler problem. All presented lemmas are rather trivial and could be found in more or less every work considering this subject, containing different levels of details in proofs. Thus, we collect and present them with our own proofs, citing specific sources only in non-obvious parts. In all the examined cases, it is possible to prove that $f(n) \in O(\log n)$ or even $f(n) \in O(1)$. We show, however, that those special cases, although presenting interesting ideas, cover only an insignificant part of all examined words. Thus, the main part of the problem remains unsolved. In Chapter 3, we will see that, in general, $f(n) \in \Omega(\log n)$. We start with a helpful lemma about the existence of a separating modulo, which we formulate in two slightly different ways to utilize whichever is better suitable for the situation.

Lemma 21. *For given $a, b, n \in \mathbb{N}$, $a, b \leq n$, $a \neq b$ there exists $m \in \mathbb{N}$ such that $a \not\equiv b \mod m$ and $m < 4 \log n$.*

Proof. Without loss of generality, we assume that $a > b$. Let $p_1, \ldots, p_k$ denote first k prime numbers. Choose k so that $\prod_{i=1}^{k-1} p_i \leq a < \prod_{i=1}^{k} p_i$.

First, we prove by contradiction that there exists $1 \leq i \leq k$ such that $a \not\equiv b$ mod p_i. Assume this is false and $a \equiv b \mod p_i$ for all $1 \leq i \leq k$. Then, $p_i | a - b$ for any $1 \leq i \leq k$ and $\prod_{i=1}^{k} p_i \mid a - b$ as $p_1, \ldots, p_k$ are co-prime. However, this is possible only if $a - b = 0$, as $a - b < a < \prod_{i=1}^{k} p_i$, which leads to a contradiction with the fact that $a \neq b$. Assign $m = p_i$ with a minimum i such that $a \not\equiv b \mod p_i$.

It remains to prove that m is sufficiently small. There are several ways to do this, we utilize one from Wiedermann [2016]. We use the fact stated in Rosser and Schoenfeld [1962, formula (3.16)], that for any k such that $p_k \geq 41$

$$\log \prod_{i=1}^{k} p_i > p_k(1 - \frac{1}{\log p_k}) > p_k/2. \tag{2.1}$$

We also use Bertrand's postulate [Ramanujan, 1919], which states that for any $j > 1$ there exists a prime p such that $j < p < 2j$.

We have $\log a \geq \log \prod_{i=1}^{k-1} p_i > p_{k-1}/2$ (by Equation 2.1). Using the Bertrand's postulate we state that $p_{k-1} < p_k < 2p_{k-1}$. Thus, $m \leq p_k < 2p_{k-1} < 4 \log a \leq 4 \log n$. □

Lemma 22. *For given $n \in \mathbb{N}$ there exists a prime $p \in \mathbb{N}$ such that $p \nmid n$ and $p < 4 \log n$.*

Proof. The statement immediately follows from Lemma 21 after setting $a = n, b = 0$ and noticing that any $m < 4 \log n$ has a prime divisor $p \leq m < 4 \log n$ for which the claim also holds. □

From now on, we deal with some special cases defined as restrictions on the set of examined words, where $f(n)$ behaves "nicely".

Definition 23 (Restricted worst-case discernibility function)**.** *We call a set of word pairs that possess some unique quality* a case. *For example, the case of words with different lengths can be defined as $C = \{(u, v) \in \Sigma^* \mid |u| \neq |v|\}$. We define* a restriction *of function $f(n)$ to a case C to be*

$$f_C(n) = \min(\{D(u, v) \mid (u, v) \in C,\ |u|, |v| \leq n\}).$$

Clearly, if we are able to separate all existing word pairs into different cases and bound $f_C(n)$ for each of them, we obtain a bound for $f(n)$. Even if we are not able to do this, restricting the set of analyzed words by excluding some belonging to "simple" cases may help with the examination of $f(n)$, especially in the case of experiments where one wants to analyze all possible word pairs.

2.3.1 Different length

Lemma 24. *If we restrict ourselves to the case $C = \{(u, v) \in \Sigma^* \mid |u| \neq |v|\}$ of the words with different lengths, $f_C(n) \leq 4 \log n$.*

Proof. To prove this lemma, we create a *modulo automaton*, which distinguishes the words whose lengths have different remainders by some modulo. Let m be a minimum modulo such that $|u| \not\equiv |v| \mod m$, by Lemma 21 it exists and $m \leq 4 \log \max(|u|, |v|)$. We create an automaton $M = (\{q_0, \ldots, q_{m-1}\}, \Sigma, \delta, q_0)$ with m states, where $\delta(q_i, a) = q_{i+1 \bmod m}$ for any $a \in \Sigma$. An example of such an automaton for $m = 5$ is shown in Figure 2.1. It is easy to see that any word of length k will end up in the state $q_{k \bmod m}$, thus, as $|u| \not\equiv |v| \mod m$, modulo automaton with m states will distinguish u and v. It follows that $f_C(n) \leq m \leq 4 \log n$. □

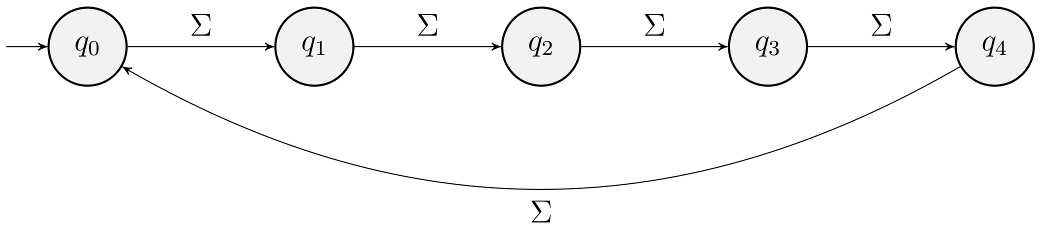


Figure 2.1: Modulo automaton for $m = 5$.

2.3.2 Unary alphabet

Lemma 25. *If we restrict ourselves to the case $C = \{(u, v) \in \Sigma^*\}$, $|\Sigma| = 1$ of words over unary alphabet, $f_C(n) \leq 4 \log n$.*

Proof. We require the words u and v to be distinct, and words from the unary alphabet could be distinct only if they have different lengths. Thus, the statement clearly follows from Lemma 24. □

2.3.3 Alphabet size

Lemma 26. *Bounds for $f(n)$ do not depend on the size of the input alphabet as long as its size is bigger than 1.*

Proof. Let us denote by $f_d(n)$ a restriction of $f(n)$ to the case C_d containing words over an alphabet of size d. We claim that $f_d(n) = f_2(n)$ for any $d > 1$.

- $f_d(n) \geq f_2(n)$

  Let $\Sigma = \{a, b, \dots\}$. Clearly, words from $\{a, b\}^*$ form a subset of the words from Σ^*. Thus, their bound could not be bigger than the bound for all words.

- $f_d(n) \leq f_2(n)$

  Consider any two distinct words u, v of length n over the alphabet $\Sigma = \{a, b, \dots\}$ with more than two symbols. There exists an index i where they differ for the first time, having $u_i = a$ and $v_i = b$. Create $u', v' \in \{0, 1\}^*$ so that all symbols a in the u, v are replaced by 0 and all other symbols from $\Sigma \backslash \{a\}$ are replaced by 1. It is easy to see that we have two distinct words over the alphabet $\{0, 1\}$ of size 2. There exists an m-state DFA M', $m \leq f_2(n)$, which distinguishes u' and v'. It follows that there exists an m-state DFA M which distinguishes u and v, created as follows:

  $$\begin{aligned} &\text{Let } M' = (Q, \{0, 1\}, \delta', q_0), \\ &\text{We create } M = (Q, \Sigma, \delta, q_0), \text{ where} \\ &\delta(q, s) = \begin{cases} \delta'(q, 0), & \text{if } s = a, \\ \delta'(q, 1), & \text{otherwise.} \end{cases} \end{aligned}$$

  Thus, $D(u, v) \leq m \leq f_2(n)$ for any words u, v and $f_d(n) \leq f_2(n)$.

□

This lemma allows us to restrict the main problem to $\Sigma = \{0, 1\}$ from now on.

2.3.4 Parity

Definition 27. *Let us denote the number of zeros and ones in a given word u by $|u|_0$ and $|u|_1$, respectively.*

Lemma 28. *If we restrict $f(n)$ to the case $C = \{(u, v) \mid |u|_1 \not\equiv |v|_1 \mod 2\}$ of the words with a number of 1's in each having different parity, then $f_C(n) \leq 2$.*

Proof. To prove this lemma, we create a *parity automaton*, as shown in Figure 2.2. It is easy to see, that all the words with an even number of 1's will finish in the state q_0, and all the words with an odd number of 1's will finish in the state q_1. Thus, any u, v from the lemma are distinguishable by an automaton of the size 2, and $f_C(n) \leq 2$. □

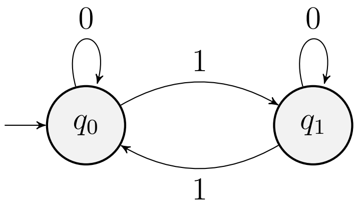


Figure 2.2: Parity automaton.

2.3.5 Different composition

Now, let us present a slightly more complicated lemma, which unites the ones described in Subsections 2.3.1 and 2.3.4. The idea of the union is similar: if the words have a different number of ones (or, equivalently, zeros) in them, we should be able to detect the difference using Lemma 21.

Lemma 29. *If we restrict the problem to the case $C = \{(u, v) \mid |u|_1 \neq |v|_1\}$ of the words with a different number of 1's, then $f_C(n) \leq 4 \log n$.*

Proof. Let us construct an improved m-state modulo automaton, which counts the number of ones in the input modulo m. Each of its m states $\{q_0, \dots, q_{m-1}\} = Q$ represents one reminder. We describe the automaton as $M = (Q, \{0, 1\}, \delta, q_0)$, $Q = \{q_0, \dots q_{m-1}\}$, where $\delta(q, 0) = q$ for any $q \in Q$ and $\delta(q_i, 1) = q_{i+1 \mod m}$ for any $0 \leq i < m$. Figure 2.3 shows an example of such an automaton with $m = 5$ states.

It is easy to see that words with k ones will end up in the state $q_{k \mod m}$. Let $k_u = |u|_1$ and $k_v = |v|_1$, $k_u \neq k_v$. Then, by the Lemma 21, there exists an m such that $k_u \not\equiv k_v \mod m$ and thus m-state automaton as described above will distinguish u and v. By the same lemma, $m \leq 4 \log \max(k_u, k_v) \leq 4 \log n$. □

2.3.6 Limited differences

In this subsection, we present several lemmas about pairs of words where the position of the first/last difference or the number of differences is small (constant). While the Lemmas 30 and 31 about the differences near the start/end of the words can be considered common knowledge, the last Lemma 32 about the pairs with a small number of differences requires a slightly more involved argument and was presented in [Demaine et al., 2011].

Lemma 30. *If we restrict ourselves to the case C with words u, v where the first difference occurs at the index $\leq d \in \mathbb{N}$, then $f_C(n) \leq d + 2$.*

Proof. By construction. Construct a difference automaton $D_d = (Q, \{0, 1\}, \delta, q_0)$ with $d + 2$ states $Q = \{q_0, \dots, q_{d-1}, t_1, t_2\}$ such that

$$\begin{aligned}
&\delta(q_i, s) = q_{i+1}, \quad \forall s \in \{0, 1\}, 0 \leq i < d - 1,\\
&\delta(q_{d-1}, 0) = t_1,\\
&\delta(q_{d-1}, 1) = t_2,\\
&\delta(t, s) = t, \quad \forall s \in \{0, 1\},\ \forall t \in \{t_1, t_2\}.
\end{aligned}$$

It is clear that after reading $d - 1$ symbols, the automaton is in state q_{d-1} for both words. After that, as the symbols on the d-th position differ, the automaton will proceed to the different states t_1 and t_2. Those are terminal (all transitions go into them), so the automaton will remain in them and distinguish the words. Example of a difference automaton D_3 can be seen in Figure 2.4. □

Lemma 31. *If we restrict $f(n)$ to a case C of the words u, v where the last difference occurs at the index $> n - d$, $d \in \mathbb{N}$, then $f_C(n) \leq d + 1$.*

Proof. Given the string s of length d, it is easy to construct an automaton that accepts the words ending with s and rejects all others (i.e., all the words ending with s will finish in one particular state, while all others in different ones). Having it, we can distinguish the words u, v, which differ at the index $n - d + 1$, so $u_{-d\dots} \neq v_{-d\dots}$, by creating a "string difference" automaton S_s for a string $s = u_{-d\dots}$ of length d. S_s has exactly $d + 1$ states.

We construct a "string difference" automaton $S_s = (\{q_0, \dots, q_d\}, \{0, 1\}, \delta, q_0)$ that maintains the invariant "word w finishes in the state q_i where i is a maximum index such that $w_{-i\dots-1} = s_{1\dots i}$ (i.e., w ends with first i symbols of s). To do this, we define a transition function δ as follows:

$$\delta(q_i, a) = \begin{cases} q_{i+1}, & \text{if } i < d \text{ and } s_{i+1} = a,\\ q_j, & \text{where } j = \max(\{2 \leq j \leq i \mid s_{(i-j+2)\dots i} \cdot a = s_{1\dots j}\}) \text{ if exists,}\\ q_1, & \text{if } a = s_1,\\ q_0, & \text{otherwise.} \end{cases}$$

We present an example of such an automaton for a string 1101 in Figure 2.5. □

The next lemma considers a constant Hamming distance $H(u, v)$ between the words and was presented in [Demaine et al., 2011]. Clearly, for the words u and v of the same lengths, $H(u, v) = d$ means that u and v differ in exactly d positions. Although it does not mean that the numbers of ones and zeros in the two words have different parity and we can not apply Lemma 28 directly, we can prove that there exists a subset of indices that is easily detectable by DFA and where parity differs. The proof uses the same idea as the one we further develop in Chapter 4, but with much easier assumptions.

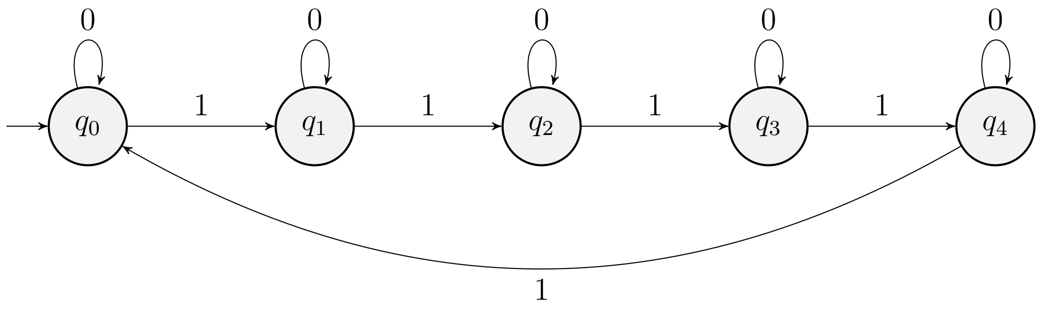


Figure 2.3: Improved modulo automaton for $m = 5$ counting a remainder of the number of 1's in the input modulo m.

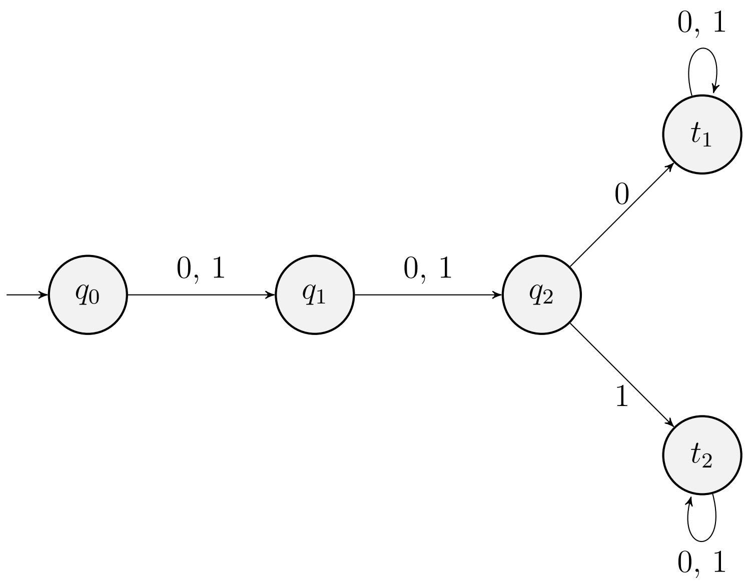


Figure 2.4: Example of a difference automaton D_3 which distinguishes the words which differ in the third symbol.

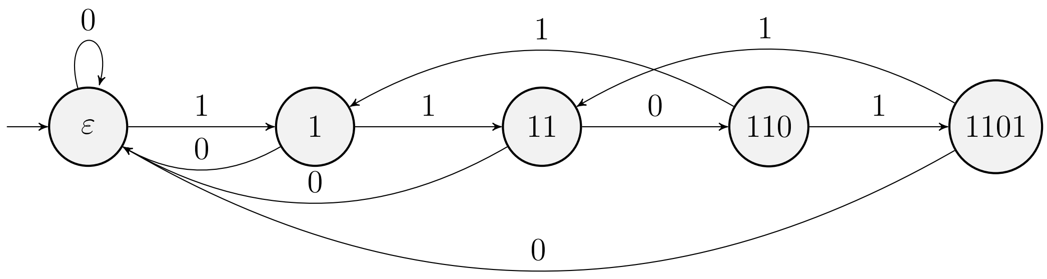


Figure 2.5: Example of a string difference automaton S_{1101}.

Lemma 32 ([Demaine et al., 2011]). *If we limit ourselves to the words u, v with at most $d \in \mathbb{N}$ differences, then $f(n) \in O(d \log n)$.*

Proof. Let the words u, v have the differences on positions $x_1, \dots, x_d$. Assign $N = \prod_{i=2}^{k}(x_i - x_1)$. By Lemma 22 there exists a prime p such that N is not divisible by p and $p \in O(\log N) = O(d \log n)$, as $N < n^{d-1}$. Consider the values

$$d_{p,x_1}(u) = \left(\sum_{j \equiv x_1 \mod p} u_j \right) \mod 2.$$

We claim that $d_{p,x_1}(u) \neq d_{p,x_1}(v)$, as p does not divide N and thus p does not divide any $x_i - x_1$ for $2 \leq i \leq k$. Thus, on indices $\{j \mid j \equiv x_1 \mod p\}$ words u and v contain exactly one difference (on index x_1). Now we construct a Hamming distance automaton $H_{p,x_1} = (Q, \{0,1\}, \delta, q_0)$ with $2p \in O(d \log n)$ states which computes $d_{p,x_1}(u)$. To do this, we construct two "circles" of p states, where all states in the first one indicate that $d_{p,x_1}(u) = 0$, and in the second one, conversely, $d_{p,x_1}(u) = 1$. To do this, we assign $Q = \{a_1, \dots, a_p, b_1, \dots, b_p\}$, $q_0 = a_p$ and

$$\delta(a_i, s) = \begin{cases} b_{i+1 \mod p}, & \text{if } s = 1 \text{ and } i \equiv x_1 \mod p, \\ a_{i+1 \mod p}, & \text{otherwise,} \end{cases},$$

$$\delta(b_i, s) = \begin{cases} a_{i+1 \mod p}, & \text{if } s = 1 \text{ and } i \equiv x_1 \mod p, \\ b_{i+1 \mod p}, & \text{otherwise.} \end{cases}$$

It is clear that the word with $d_{p,x_1}(u) = 1$ finishes in some of $\{b_1, \dots b_p\}$ states and the word with $d_{p,x_1}(u) = 0$ finishes in some of $\{a_1, \dots a_p\}$ states. Thus, u and v are distinguished by H_{p,x_1}. Figure 2.6 shows an example of $H_{3,1}$. □

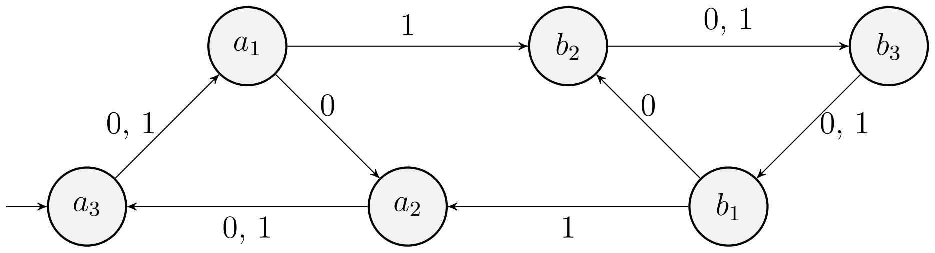


Figure 2.6: Example of a hamming distance automaton $H_{3,1}$.

2.4 Unsolved cases

As can be seen in the preceding section, a large number of word pairs are easily distinguishable by a small automaton of logarithmic or even constant size. As a logarithmic lower bound is already known, it makes sense to restrict the analysis to the word pairs that are not distinguishable in the logarithmic case. Any restriction of the word pairs set is useful, especially for some experimental solutions where all possible pairs undergo some inspection. However, we see that the ratio of those pairs belonging to easily-solvable cases is mostly negligible. Let us summarize what restrictions to the main problem are available using all the lemmas stated in this chapter and describe the word pairs we should still deal with. To summarize, words that do not belong to easily-solvable cases...

- ...come from a binary alphabet $\{0, 1\}$,

  This restriction clearly allows us to simplify the studied problem greatly, as the values of $f(n)$ do not depend on the size or composition of the input alphabet.

- ...have the same length,

  This restriction again helps with the reduction of the studied set. Clearly, if we pair any word of length n with one of the length up to n, we have $2^n(2^0 + 2^1 + \cdots + 2^n - 1) = 2^n(2^{n+1} - 2)$ total pairs. However, if we count the pairs where both of the words have length n, we have only $2^n(2^n - 1)$ possible pairs, which is roughly half that number.

  Unfortunately, all other stated restrictions do not really help. We show that the ratio of restricted pairs tends toward zero in each case. Let us denote the number of all pairs of distinct words of length n over the binary alphabet by A_n. Clearly, $A_n = 2^n(2^n - 1) = 4^n - 2^n$.

- ...have the same number of zeros (and ones, conversely),

  Let us count the number of pairs having the same composition (number of ones and zeros) and denote it by B_n. We could compute it as a sum per possible number of ones: there are $\binom{n}{k}$ words with exactly k ones. Thus, the total number of pairs with the same number of ones is

$$B_n = \sum_{k=0}^{n} \binom{n}{k}\left(\binom{n}{k} - 1\right) = \sum_{k=0}^{n} \binom{n}{k}^2 - \sum_{k=0}^{n} \binom{n}{k} = \binom{2n}{n} - 2^n.$$

  We want to show that $\lim_{n\to\infty} \frac{B_n}{A_n} = 0$. Clearly, as $A_n > B_n$,

$$\frac{B_n}{A_n} = \frac{\binom{2n}{n} - 2^n}{4^n - 2^n} < \frac{\binom{2n}{n}}{4^n}.$$

  Using Stirling's approximation [Namias, 1986], we get that

$$\binom{2n}{n} \sim \frac{4^n}{\sqrt{\pi n}}.$$

  Thus,

$$\lim_{n\to\infty} \frac{\binom{2n}{n}}{4^n} = \lim_{n\to\infty} \frac{1}{\sqrt{\pi n}} = 0.$$

  From that we derive that $\lim_{n\to\infty} \frac{B_n}{A_n} = 0$, as it is clearly non-negative and bounded by 0 from above.

- ...do not differ in the first and last $\omega(\log n)$ positions, Let C_n denote the number of pairs of words that differs only among the first $d \in O(\log n)$ positions (not necessarily in all of them). Clearly, the number of those words can be expressed as

$$C_n = 2^d(2^d - 1)2^{n-d} = 2^n(2^d - 1).$$

It is easy to see that $\lim_{n\to\infty} \frac{C_n}{A_n} = \lim_{n\to\infty} \frac{2^d-1}{2^n-1} \leq \lim_{n\to\infty} 2^{d-n} = 0$. Thus, $\lim_{n\to\infty} \frac{C_n}{A_n} = 0$, as all the numbers are non-negative.

The case when the differences between the words occur only in the last d positions is perfectly similar and thus asymptotically negligible.

- ...have $\omega(1)$ number of differences.

  Here, we denote the number of word pairs with at most d differences by D_n. We may see that $D_n \leq \binom{n}{d} 2^d(2^d - 1)2^{n-d} = \binom{n}{d} 2^n(2^d - 1)$. Again, we are interested in the limit

$$\lim_{n\to\infty} \frac{D_n}{A_n} = \lim_{n\to\infty} \binom{n}{d} \frac{2^d - 1}{2^n - 1} \leq \lim_{n\to\infty} \frac{\binom{n}{d}}{2^{n-d}} =$$
$$= \lim_{n\to\infty} \frac{n(n-1)\cdots(n-d+1)}{2^{n-d}d!} \leq \lim_{n\to\infty} \frac{n^d}{2^{n-d}d!} = 0.$$

  As d is a small constant, we may ignore the $\frac{2^d}{d!}$ part. Thus, as values of D_n and A_n are non-negative, $\lim_{n\to\infty} \frac{D_n}{A_n} = 0$.

We may see that the main contribution to the studied problem consists of the independence of $f(n)$ values on the input alphabet. The fact that we can examine only words of the same length helps to halve the set of all analyzed word pairs. All the other studied cases may bring interesting insight into the problem but do not significantly decrease the amount of examined words.

3. Related work

In 1986 Goralčík and Koubek proved that for given words x, y with length $\leq n$ there exists an automaton discerning those words with $o(n)$ states. In 1989 the upper bound was improved by Robson with the result of $O(n^{2/5} \log^{3/5} n)$. In 2021 the bound was further improved by Zachary Chase to $O(n^{1/3} \log^7 n)$. The lower bound was proved to be $\Omega(\log n)$ by Demaine et all in 2011. In this chapter, we present the main ideas of those proofs, which may be useful in further research on the topic.

3.1 Upper bound

3.1.1 First result

The first result on the topic was obtained in 1986 at Charles University by Goralčík and Koubek [1986]. In that work, they solve some trivial cases such as the unary alphabet or words of different lengths, state that the result is independent of the alphabet size, and, most importantly, establish an upper bound for the worst-case discernibility function to be $o(n)$. The proof they provided is based on a careful enumeration of all possible cases of the words' compositions and differences. In this section, we sketch some ideas instead of presenting the full proof, as it is, by its nature, rather long and complicated. All details can be found in [Goralčík and Koubek, 1986].

The proof is done by contradiction. We assume that the worst-case discernibility function (see Definition 18) does not belong to $o(n)$, and there exists d such that $\limsup f(n)/n = d > 0$. We then consider different cases - subsets of all word pairs possessing some specific quality such as too big or too small number of zero blocks, number of zeros in the block containing the first difference and so on - and for each case C prove that $\limsup f_C(n)/n < d$. When studied cases cover all existing words, the proof is complete.

To deal with every case, we should reduce the problem to one which can be solved by an automaton with logarithmic number of states - in most cases, we want to get the words of different lengths or with different number of ones in them (in Section 2.3 we show how to deal with such pairs). The proof contains several automata constructions, which are then used to settle each case together with proof that with this particular conditions on the input pairs the constructed automaton will be small enough to achieve the desired bound of less than dn states. Main idea of those helping automata is to stop after encountering specific number of zeros / blocks of zeros in the word, and usually the counting occur by some modulo to decrease the number of states of a constructed automata. We present several examples of those constructions to give the reader an idea of the proof.

Definition 33. *We define a* discerning block *automaton K_{01}^m to be an automaton accepting the words with at least m blocks of zeros and ones by its only terminal state. We create K_{01}^m as an m-times concatenation $K_{01}^m = K_{01} \circ \cdots \circ K_{01}$ of an automaton K_{01} with 3 states accepting only the word $w = 01$.*

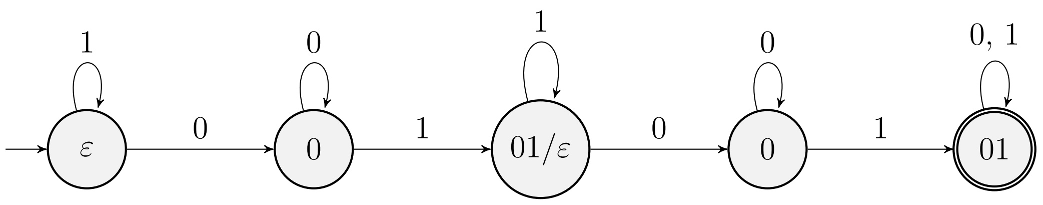


Figure 3.1: Example of a discerning block automaton K_{01}^2.

An example of K_{01}^2 is shown in Figure 3.1. It accepts exactly the words containing at least two disjoint occurrences of 01, i.e. those with at least two zero blocks followed by one blocks.

Definition 34. *We define a* zero modulo counter *automaton $K_{r:x}$ as the one accepting the words containing a block of zeros of lengths $l \equiv x \mod r$ by its only terminal state. We create it from an automaton accepting only the word 0^r by replacing its terminal state with a state $0^x \cdot 1$ and redefining $\delta(0^{r-1}, 0) = \varepsilon$ instead of 0^r and $\delta(0^x, 1) = 0^x \cdot 1$ instead of ε.*

Figure 3.2 shows an example of $K_{4:2}$ accepting exactly the words having a zero block of lengths $l \equiv 2 \mod 4$.

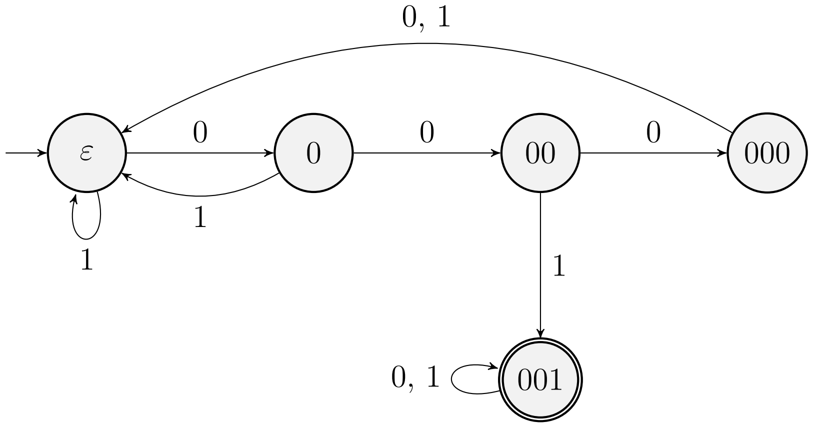


Figure 3.2: Example of a counter $K_{4:2}$

Both shown automata have one terminal state, which we may use to concatenate it with the discerning one. So, whenever we have the first difference between the words u and v in the m-th zero blocks, K_{01}^m will reach the terminal state at the different indices for those words. Thus, we may concatenate the *discerning block* automaton with the logarithmic one discerning words of different lengths (see Lemma 24) and be done. The same holds with the *zero modulo counter* automaton as long as we find two blocks of the size congruent by some modulo at the different positions in words – even if there are some preceding blocks of the same size, we can concatenate $K_{r:x}$ with itself several times to achieve the situation where both words reach the terminal state at the different indices. Having this, it is remaining to prove under which conditions those constructed automata would be small enough.

A slightly different principle is used for the next shown automaton.

Definition 35. *We define a* zero translator *automaton $T_{r:x,y}$ as a one with two terminal states t_0, t_1, where all the words having a block of zeros of size $l_0 \equiv x$*

mod r *will finish in* t_0 *and all the words having a block of zeros of size* $l_1 \equiv x \mod r$ *will finish in* t_1 *(depending on what block is coming first). To achieve this, we take an automaton accepting exactly one word* 0^r *and replace its terminal state* 0^r *with two new ones* $0^x \cdot 1$ *and* $0^y \cdot 1$*, redefining* $\delta(0^{r-1}, 0) = \varepsilon$*,* $\delta(0^x, 1) = 0^x \cdot 1$*,* $\delta(0^y, 1) = 0^y \cdot 1$.

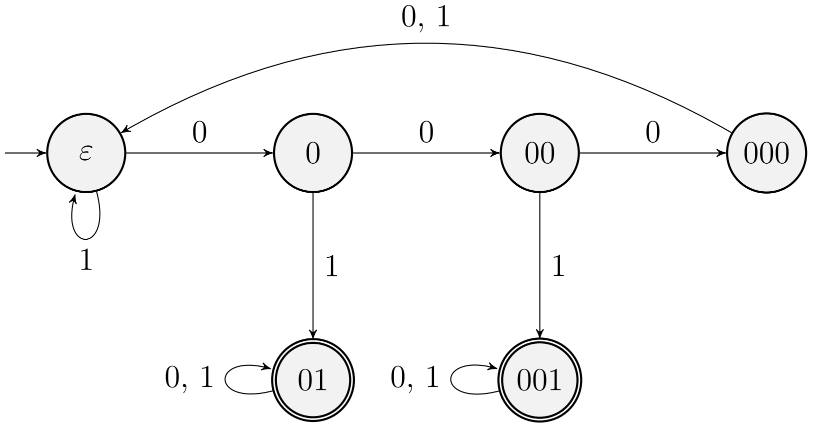


Figure 3.3: Example of a translator $T_{4:1,2}$

An example of $T_{4:1,2}$ is shown in Figure 3.3. We see that T ends in $t_0 = 01$ after reading zero block of length $l_0 \equiv 1 \mod 4$ and in $t_1 = 001$ after reading zero block of length $l_1 \equiv 2 \mod 4$. Let us create $T_{4:1,2}$-translation $T(u)$ of the word u by writing down 0 for each occurrence of the zero block of the first type and 1 for each occurrence of the zero block of the second type. Doing the same procedure for v, we obtain $T(v)$, which may have a different number of ones than $T(u)$ if the sizes of the blocks in u and v differ. If we have an automaton discerning $T(u)$ and $T(v)$ (e.g. by Lemma 29), we may replace each state in it with an instance of $T_{4:1,2}$ and obtain an automaton discerning u and v.

Those presented are examples of used automata, which, together with careful computations, help to achieve the desired bound of the number of states in each considered case. We refer the reader to [Goralčík and Koubek, 1986] for the full proof.

3.1.2 Second result

The second result for the upper bound was proposed by Robson [1989], not long after Goralčík's and Koubek's research. In his work, J.M. Robson proved the upper bound to be $O(n^{2/5} \log^{3/5} n)$. The main idea of the work combines the periodicity in words with congruence, when we focus on a subset of indices belonging to some remainder class of a small modulo. Those can be easily selected with a suitable automaton. The proof starts with a simpler bound $O(\sqrt{n \log n})$, which we will discuss here without going into detail. The improved bound is based on this one and consists of rather technical improvements and enumeration of cases, so we refer the reader to the original work Robson [1989] for the complete proof as well as proofs of some lemmas we present in this section.

Definition 36 (Word's period)**.** *A word* $w = w_{1\dots l}$ *has* a period p *if* $w_i = w_{i+p}$ *for any* $1 \leq i \leq l - p$.

Definition 37 (Periodic word)**.** *A word S is* periodic *if it has a period that is not greater than half of its length.* The period *of the word is the smallest period it has.*

Lemma 38. *If $w \cdot 0$ is periodic, then $w \cdot 1$ is not.*

Lemma 39. *If a string s has a period p and a word w contains two occurrences of s starting at indices i and j, then $|i - j| \geq p$.*

Lemma 40. *Given $w \in \{0,1\}^n$ and its non-periodic substring $s = w_{i \ldots i+\ell-1}$ of length ℓ, there exists prime $p \in O(\frac{n}{\ell} \log n)$ such that there is no other occurrence of s starting on indices congruent to i modulo p. I.e.,*

$$w_{k \ldots k+\ell-1} \neq s$$

for every $k \neq i$, $k \equiv i \mod p$.

Proof. First, let us observe that s is non-periodic, thus its period is greater than $\ell/2$, and by Lemma 39 there is at most $2n/\ell$ occurrences of s in w. Let us assume that those occurrences (besides one starting at index i) start at the indices $i_1, i_2, \ldots, i_m$. Consider $I = \prod_{j=1}^{m} |i_j - i| \leq n^{2n/\ell}$. By Lemma 22 there exists a prime $p \in O(\log I) = O(\frac{n}{\ell} \log n)$ such that $p \not| I$. Thus, $p \not| |i_j - i|$ for any $1 \leq j \leq m$ and $i_j \not\equiv i \mod p$. □

Definition 41. *An automaton M* finds *a word w if, after reading an input w, it enters the accepting state f for the first time.*

The construction of a discerning automaton in both simple and sophisticated proofs uses the following method. Given the words u and v as input, construct an automaton M which finds $u_{1 \ldots i}$ by the state f and does not accept $v_{1 \ldots i}$. If M never enters the state f while reading v, we may make a state f in the M terminal and be done - M will distinguish u and v. Otherwise, if M enters q for the first time after reading v_j, construct an automaton M' distinguishing between $u_{i+1 \ldots n}$ and $v_{j+1 \ldots n}$, which has logarithmic size, as the words $u_{i+1 \ldots n}$ and $v_{j+1 \ldots n}$ have different lengths. Composition $M \circ M'$ will distinguish u and v. As long as the size of M is $\Omega(\log n)$, the asymptotic size of $M \circ M'$ is fully determined by the size of M, so we can omit M'.

Now, in order to present the idea of Robson's work, we define an automaton $M(w,k,p)$ with $|w| + p$ states, which finds the first occurrence of the word w starting on position k modulo p. To do this, we combine a p-state counting automaton with the one distinguishing w from all other words. Figure 3.4 shows an example of such an automaton.

Definition 42. *We define an automaton $M(w,k,p) = (Q, \{0,1\}, \delta, a_0, \{w\})$, where $Q = \{a_0, \ldots a_{p-1}\} \cup P(w) \backslash \{\varepsilon\}$, $a_k = \varepsilon$, and δ is defined as follows $\forall s \in \{0,1\}$:*

$$\begin{aligned}
&\delta(a_i, s) = a_{i+1 \mod p}, && i \not\equiv k \mod p,\\
&\delta(w, s) = w, && s \in \{0,1\},\\
&\delta(w', s) = w' \cdot s, && \text{if } w' \cdot s \in P(w),\\
&\delta(w', s) = a_{k+|w'|+1 \mod p}, && w' \cdot s \notin P(w), |w'| < p\\
&\delta(w', s) = w'_{-r\ldots} \cdot s, && \text{if } w'_{-r\ldots} \cdot s \in P(w),\ \text{where } r = k + |w'| \mod p,\\
&\delta(w', s) = a_{k+|w'|+1 \mod p}, && \text{otherwise.}
\end{aligned}$$

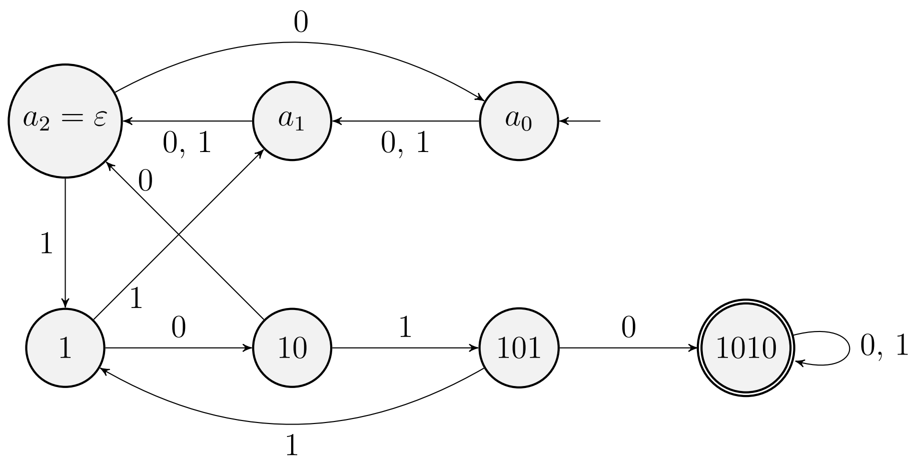


Figure 3.4: Example of an automaton $M(w, k, p)$ for $w = 1010, k = 2, p = 3$.

Now, we have everything we need to formulate the theorem about a simple upper bound.

Theorem 43. *Words u and v can be separated by an automaton with $O(\sqrt{n \log n})$ states.*

Proof. Let i be the first index where $u_i \neq v_i$. If $i < \sqrt{n \log n}$, then we may trivially construct small M which finds $u_{1\dots i}$ and does not accept $v_{1\dots i}$ (by Lemma 30).

Otherwise, consider two words $u' = u_{i-\sqrt{n \log n}\dots i}$ and $v' = v_{i-\sqrt{n \log n}\dots i}$. By Lemma 38, at least one of them is not periodic, w.l.o.g. we say that is it u'. Now, let p be a number from Lemma 40 for the word u'. We claim that $M(u', k, p)$, where $k = (i - \sqrt{n \log n}) \mod p$ finds $u_{1\dots i}$ and does not accept $v_{1\dots i}$. Size of $M(u', k, p)$ is $|u'| + p = \sqrt{n \log n} + 1 + p$. By Lemma 40 $p \in O(\frac{n}{l} \log n)$, where $l = \sqrt{n \log n}$, thus $\frac{n}{l} = \sqrt{\frac{n}{\log n}}$ and $p \in O(\sqrt{n \log n})$, which gives us the desired upper bound. □

To improve the upper bound, one should consider the fact that the part distinguishing w from all other words in $M(w, k, p)$ does a lot more than we require it to do. It accepts w and all the words having w as a prefix, separating all the other ones. However, as we have limited input, it is sufficient to distinguish u' from all other suffixes of u starting on the congruent positions. Using this, a more sophisticated automaton is created to achieve the desired upper bound $O(n^{2/5} \log^{3/5} n)$. The proof is based on determining the set of the prefixes of u and v starting on indices congruent by some modulo and constructing the automaton which will distinguish them. We refer the reader to [Robson, 1989] for the full proof.

3.1.3 Third result

The third result in the field was obtained by Chase [2021] after a significant amount of time. The contribution of his work consists in replacing the $O(n^{2/5})$ bound by $O(n^{1/3} \log^7 n)$ and connecting the studied problem to other areas such as trace reconstruction and "k-deck" problem. The main ideas of the proof include:

- restriction of the input to the subset of indices represented by a residue class of some small modulo,
- counting the occurrences of a given substring in both words,
- complex analysis methods used to prove the desired bounds.

We start with a few definitions:

Definition 44. *For a given word w and string s we denote by $pos_s(w)$ a set of starting indices of all occurrences of s in w. I.e.,*

$$pos_s(w) = \{1 \le i \le |w| \mid w_{i\dots i+|s|-1} = s\}.$$

Definition 45 (Residue class)**.** *For a given set A and numbers $m \in \mathbb{N}$, $0 \le i < m$ we define* residue class $A_{m,i}$ *of A as a subset of its elements belonging to a residue class i modulo m. I.e.,*

$$A_{m,i} = \{x \in A \mid x \equiv i \mod m\}.$$

Definition 46. *Set $A \subseteq \mathbb{N}$ is d-*separated *if for any $a \neq a' \in A$*

$$|a - a'| \ge d.$$

Now, we construct a small automaton, which checks if the number of occurrences of some substring at some residue class indices belongs to a given residue class, i.e., for given numbers i, p, a, q and a substring s it accepts the word w if and only if $|pos_s(w)_{p,i}| \equiv a \mod q$.

Lemma 47. *Given the numbers $p, q \in \mathbb{N}, 0 \le i < p$, $0 \le a < q$ and a string $s \in \{0,1\}^\ell$, $\ell \le p$, there exists an automaton with $2pq$ states accepting $w \in \{0,1\}^n$ if and only if $|pos_s(w)_{p,i}| \equiv a \mod q$.*

Proof. By construction. We create an automaton $M = (Q, \{0,1\}, \Sigma, q_0, F)$ with states represented as triples, $Q = \mathbb{Z}_p \times \{0,1\} \times \mathbb{Z}_q$, where the first element shows a next position to be read from the input modulo p, the second one is a flag whether we are in the middle of s occurrence right now, and the third one counts $|pos_s(w)_{p,i}|$ modulo q. More precisely, we define $q_0 = (1,0,0)$, $F = \mathbb{Z}_p \times \{0,1\} \times \{a\}$ and

$$\delta((j,f,c),a) = \begin{cases} (j+1,0,c), & \text{if } f=0,\ j \not\equiv i \mod p, \\ (j+1,1,c), & \text{if } f=0,\ j \equiv i \mod p, a = s_1, \\ (j+1,0,c), & \text{if } f=0,\ j \equiv i \mod p, a \neq s_1, \\ (j+1,1,c), & \text{if } f=1,\ j \not\equiv i+l-1 \mod p, a = s_{j-i+1}, \\ (j+1,0,c), & \text{if } f=1,\ j \not\equiv i+l-1 \mod p, a \neq s_{j-i+1}, \\ (j+1,0,c+1), & \text{if } f=1,\ j \equiv i+l-1 \mod p, a = s_l, \\ (j+1,0,c), & \text{if } f=1,\ j \equiv i+l-1 \mod p, a \neq s_l. \end{cases}$$

□

Having this automaton, we need to determine an appropriate substring w so that $pos_w(u) \neq pos_w(v)$ and there exists two small moduli p, q and a remainder $0 \leq i < p$ such that $|pos_w(u)_{p,i}| \not\equiv |pos_w(v)_{p,i}| \mod q$. We are always able to choose $q \in O(\log n)$ due to Lemma 21, but what about p? It depends on the choice of a separating substring w. A natural decision would be to take $pos_1(u)$ and $pos_1(v)$, which are clearly different. However, it turns out that one can not provide a sufficient upper bound for p. It can be proved that $p \in O(n^{1/2} \log^{1/2} n)$, with a total upper bound to the problem being $O(n^{1/2} \log^{3/2} n)$, which is too big (we refer to [Chase, 2021] for the proof).

To present a sufficient lower bound for p, one should choose a longer substring w. While longer possible w does not affect the size of the discerning automaton, it makes the sets $pos_w(x)$ and $pos_w(y)$ smaller, which may help with finding an appropriate p value. Ideally, we want to prove the following fact:

Conjecture 48. *For any sets $A \neq B \subseteq [n]$, $|A|, |B| \leq n^{2/3}$ there exists $p \in O(n^{1/3})$ and $0 \leq i < p$ such that $|A_{p,i}| \neq |B_{p,i}|$.*

This conjecture is hard to solve, but it becomes easier by introducing a condition that both A and B should be $n^{1/3}$-separated. The proof of the following lemmas can be found in the original work.

Lemma 49. *For any $n^{1/3}$-separated sets $A \neq B \subseteq [n]$, $|A|, |B| \leq n^{2/3}$, there exist $p \in \Theta(n^{1/3} \log^6 n)$ and $0 \leq i < p$ such that $|A_{p,i}| \neq |B_{p,i}|$.*

We also state that we can construct a "good" set by choosing an appropriate substring w.

Lemma 50. *For any word $u \in \{0,1\}^n$ and a substring w, $|w| = 2n^{1/3}$, such that w has no period of length $\leq n^{1/3}$, the set $pos_w(u)$ is $n^{1/3}$-separated.*

Now, we are ready to present the main steps of the proof.

Theorem 51. *Any $u \neq v \in \{0,1\}^n$ can be separated by an automaton with $O(n^{1/3} \log^7 n)$ states.*

Proof. Let i be the index of the first difference between u and v. If $i < 2n^{1/3}$, then the proof is complete (by Lemma 30). Otherwise, assign $w' = u_{i-2n^{1/3}+1\dots i-1}$ a substring of u of length $2n^{1/3} - 1$. Choose $w = w' \cdot 0$ or $w = w' \cdot 1$ so that $A = pos_w(u)$ and $B = pos_w(v)$ are both $n^{1/3}$-separated (see Lemmas 38 and 50). Clearly, $A \neq B$. By Lemma 49 there exist $p \in \Theta(n^{1/3} \log^6 n)$ and $0 \leq i < p$ so that $|A_{p,i}| \neq |B_{p,i}|$. As $|A|, |B| < n$, there exists $q \in O(\log n)$ such that $|A|_{p,i} \not\equiv |B|_{p,i} \mod q$ (by Lemma 21). Thus, we can construct a discerning automaton as described above with $2pq \in O(n^{1/3} \log^7 n)$ states. □

Remark. We should mention that in all the original statements, the numbers p, q are considered prime. Their primeness neither affects the existence and functionality of the separating automaton nor helps or harms the obtained asymptotic results. That is why we decided to omit this consideration, although it can be helpful in related areas.

3.2 Lower bound

This section presents the proof of the $\Omega(\log n)$ lower bound, which can be found in Demaine et al. [2011]. The proof is done by construction - we show a word pair that can not be distinguished by an automaton of smaller than logarithmic size. We use the following definition.

Definition 52. *A sequence $(p_i)_{i\geq 0}$ is* ultimately periodic *if there exist integers $r > 0$, $s \geq 0$ such that $p_i = p_{i+r}$ for any $i \geq s$. In this case, s is called* the preperiod *and r* the period.

Lemma 53. *For any DFA $M = (Q, \Sigma, \delta, q_0)$ of size m, any state $q \in Q$ and any symbol $a \in \Sigma$*

$$\hat{\delta}(q, a^{m-1}) = \hat{\delta}(q, a^{m-1+\mathrm{lcm}(1,2,\ldots,m)}).$$

Proof. Let us define a sequence (p_i) such that $p_i = \hat{\delta}(q, a^i)$. As an automaton has m states, it is clear that the first $m+1$ elements of (p_i) can not be unique, and some of them will appear there at least twice. After the first duplicate, the sequence will repeat, as the automaton M is deterministic, so there is exactly one state we can get to from another one by a given symbol a. Thus, it is clear that (p_i) is ultimately periodic with a period $r \leq m$ and preperiod $s \leq m-1$.

The statement $p_{m-1} = p_{m-1+\mathrm{lcm}(1,2,\ldots,m)}$ follows from the fact that $m-1 \geq s$ and $r \mid \mathrm{lcm}(1,2,\ldots,m)$, as $r \leq m$. Thus, by the definition of p_i, $\hat{\delta}(q, a^{m-1}) = \hat{\delta}(q, a^{m-1+\mathrm{lcm}(1,2,\ldots,m)})$. $\square$

Theorem 54. *There exists an infinite number of the words u, v of the same length n such that $D(u, v) \in \Omega(\log n)$.*

Proof. Consider two words u, v of the same length constructed as follows.

$$u = 0^{m-1}1^{m-1+\mathrm{lcm}(1,2,\ldots,m)}$$
$$v = 0^{m-1+\mathrm{lcm}(1,2,\ldots,m)}1^{m-1}$$

Consider any automaton $M = (Q, \{0,1\}, \delta, q_0)$ with m states. We show that $M(u) = M(v)$ by applying the Lemma 53 twice. First, we see that M will be in the same state after reading the sequence of zeros in u and v, i.e., $\hat{\delta}(q_0, 0^{m-1}) = \hat{\delta}(q_0, 0^{m-1+\mathrm{lcm}(1,2,\ldots,m)}) = q_1$. Then, we observe that the same holds for the sequences of ones in both words, i.e., $\hat{\delta}(q_1, 1^{m-1+\mathrm{lcm}(1,2,\ldots,m)}) = \hat{\delta}(q_1, 1^{m-1})$. Thus, $\hat{\delta}(q_0, u) = \hat{\delta}(q_0, v)$ and $M(u) = M(v)$.

From the prime number theorem it follows that $\mathrm{lcm}(1,2,\ldots,m) = e^{m(1+o(1))}$ [Hardy and Wright, 1979, Theorem 414]. Hence, for sufficiently large n, if $m \leq \log n$, then there are words of size n that are not distinguishable by any automaton of size m or smaller. $\square$

4. Discerning sets

In this chapter, we further develop the ideas of Wiedermann [2016]. In his technical report, Wiedermann suggests that $f(n) \in \Theta(\log n)$. However, the proposed proof of the upper bound contains a gap, and to the best of our knowledge, this gap was not resolved. We explain the gap later on.

In Section 4.1, we describe a reduction of the discernibility problem to a possibly simpler one, which solution provides a valid upper bound for the main problem. Next, we present the proof of the upper bound for this new problem by Wiedermann [2016] in Section 4.2 and explain the gap it contains. Then, in Section 4.3, we show how to deal with some easily solvable cases in a new problem formulation. We provide our empirical results on the topic in Section 4.4. Finally, in Section 4.5, we present our proof showing that Wiedermann's approach does not lead to improvement of the upper bound of the discerning problem. Nevertheless, we believe that obtained empirical results might be useful in other areas.

4.1 Problem statement

4.1.1 Formulation

Let us start with some necessary definitions and notions. First, we describe a *discerning automaton* $A(m, i)$ which accepts exactly the words with the odd number of 1's on positions congruent to i modulo m.

Definition 55 (Discerning automaton)**.** *We construct a* discerning automaton $A(m, i) = (Q, \Sigma, \delta, q_0, F)$ *as follows.*

- $Q = \{(q, p) \mid 0 \leq q < m, p \in \{0, 1\}\}$,
- $\Sigma = \{0, 1\}$,
- $q_0 = (0, 0)$,
- $F = \{(q, 1) \mid (q, 1) \in Q\}$.

While δ is given by the following definition.

$$\delta((q, p), s) = \begin{cases} ((q+1) \mod m,\ 1-p), & \textit{if } s = 1 \textit{ and } q = i, \\ ((q+1) \mod m,\ p), & \textit{otherwise.} \end{cases}$$

We can see that $A(m, i)$ has the size $2m \in O(m)$. Example of such an automaton with $m = 5$, $i = 3$ can be seen in Figure 4.1.

Definition 56 (Discerning set)**.** *For two words $u, v \in \{0, 1\}^n$ we define a* discerning set $\Delta(u, v) = \{i \mid 1 \leq i \leq n,\ u_i \neq v_i\}$.

We then show that the fact that $A(m, i)$ discerns u and v is equivalent to the fact that the residue class $\Delta(u, v)_{m,i}$ (as defined in Definition 45) has an odd number of elements. Thus, if we prove that for any u, v of length n, there exists

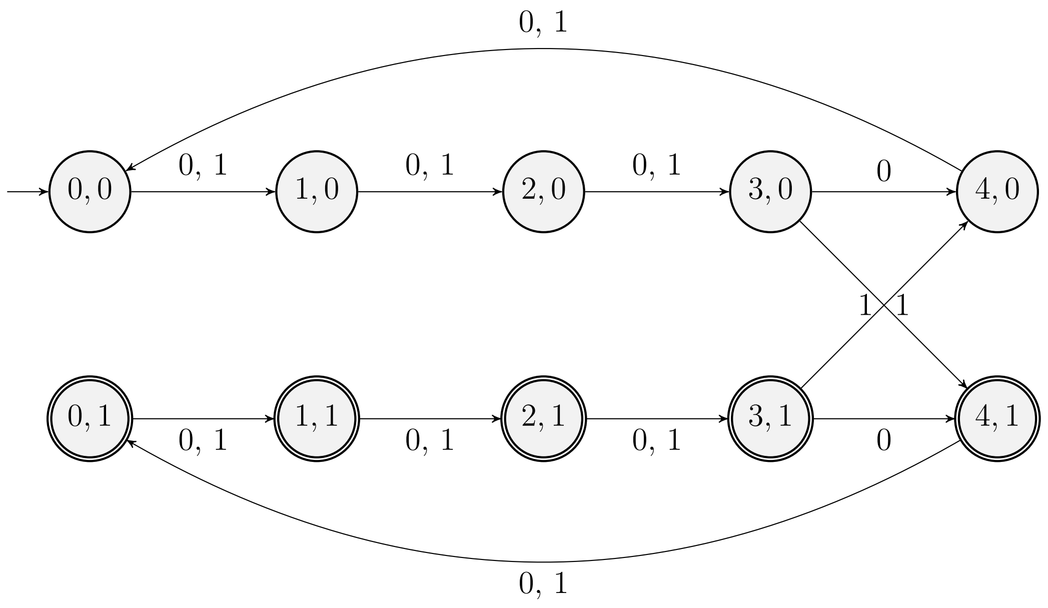


Figure 4.1: Discerning automaton $A(5, 3)$

sufficiently small m and i such that $\Delta(u, v)_{m,i}$ has odd cardinality, we will show the upper bound for the studied problem.

That is to say, we study the worst-case scenario - i.e., such $\Delta(u, v)$ (for given n) that we need m as large as possible. Moreover, actual words u, v do not matter for the discerning set, and it is clear that any subset of $[n]$ can be a discerning set for a pair of words of length n. Thus, studying sets $S \subseteq [n]$ without a connection to original words is sufficient. We examine a value M for each set S such that for all $i, m < M$ $|S_{m,i}|$ is even, but there exists i such that $|S_{M,i}|$ is odd. We are looking for a maximum such M over all possible sets $S \subseteq [n]$. Let us generalize this in the following definitions.

Definition 57. *For given $S \subseteq [n]$, we denote by $M(S)$ a number M such that*

$$\begin{aligned} \forall\, m < M, \quad \forall\, 0 \leq i < m \quad & |S_{m,i}| \textit{ is even}, \\ \exists\, 0 \leq i < M \quad & |S_{M,i}| \textit{ is odd.} \end{aligned}$$

Definition 58. *For $n \in \mathbb{N}$, we define $M(n)$ as a maximum $M(S)$ over all sets $S \subseteq [n]$.*

Definition 59 (Critical set)**.** *We call a "worst-case" set $S \subseteq [n]$ such that $M(S) = M(n)$ a* critical set*.*

Conjecture 60 (Discerning set problem)**.**

$$M(n) \in \Theta(\log n)$$

While the main goal should be to prove the preceding conjecture, any progress towards the upper bound will also be useful, as it is applicable to the main discerning problem. Any two words of the same length could be converted to a discerning set and distinguished using techniques described in this chapter.

4.1.2 Bounds

It is easy to see that $M(n)$ is at least logarithmic. For example, for given m we may take $S = \{1,\ 1 + \mathrm{lcm}(1, \dots, m)\}$ which will have $|S_{m',1}| = 2$ for each $2 \le m' \le m$ and $|S_{m',i}| = 0$ for each $i \ne 1$ and $2 \le m' \le m$. It is known that $\lim_{n\to\infty} \frac{\mathrm{lcm}(1,\dots,n)}{e^n} = 1$ [Hardy and Wright, 1979, Theorem 414], thus constructed S is of size around $2e^m$, which proves that there exist sets where smaller than logarithmic m is not sufficient. Actually, any stronger result would contradict already proven bounds for the main discerning problem.

Proving any higher lower bound will contradict Conjecture 60, but not the main Conjecture 19, as it is not clear if the chosen approach with converting the word pairs to the discerning sets is an optimal one.

4.2 Previous research

As we mentioned earlier, Wiedermann suggested in his technical report that $M(n) \in \Theta(\log n)$. Our aim is to prove or disprove this assumption by studying critical ("the worst") sets S. For completeness, we present the proof from Wiedermann's work here and point out the problematic part.

Conjecture 61 ([Wiedermann, 2016]). *Let $S \subseteq [n]$ be a set with $2 \le |S| \le n$, n is even. There exists $m \in O(\log n)$ and number $0 \le i < m$ such that the cardinality of $S_{m,i}$ is odd.*

Proof. By contradiction, assume that for all $m \in O(\log n)$ and $0 \le i < m$, $|S_{m,i}|$ is even. Consider two cases: $S = [n]$ and $S \subset [n]$.

In the former case, consider any $m \not| n$ (which exists by Lemma 21). The number of elements in all residue classes can not be the same and differ by at most one. Thus, at least one residue class will have odd cardinality.

In the latter case, consider the complement set $S' = [n] \setminus S$. $|S'|$ is even, so the assumption of the conjecture holds for it too, and according to the contradictory assumption, $|S'_{m,i}|$ is even for all m, i. For any i $S_{m,i} \cup S'_{m,i} = [n]_{m,i}$, and $S_{m,i} \cap S'_{m,i} = \emptyset$. Thus cardinality of $[n]_{m,i}$ is even for all m, i, which is in contradiction with the fact stated in the former case. □

Corollary ([Wiedermann, 2016]). Any words $u \ne v \in \{0,1\}^n$ can be discerned by an automaton with $O(\log n)$ states.

Proof. For even values of n by Conjecture 61, there exists logarithmic m and $0 \le i < m$ such that the cardinality of $S_{m,i}$ is odd. Then, a discerning automaton $A(m, i)$ of logarithmic size discerns the words.

Having the bound for $M(n)$ for all even n, it is simple to prove that we may transform the case for odd n to an even one without asymptotically increasing the upper bound. Given $u, v \in \{0,1\}^n$, where n is odd, consider the first symbols of u, v. If $u_1 \ne v_1$, we discern them with automata with $O(1)$ states (see Lemma 30). Otherwise, we create a trivial automaton skipping the first symbol and concatenate it to a discerning automaton $A(m, i)$ for $u_{2\dots}$ and $v_{2\dots}$. The resulting automaton has $(|A(m, i)| + 1) \in O(\log n)$ states. This consideration proves the upper bound for $M(n)$ for all n. □

The problem with the presented proof of Conjecture 61, however, is a classic logical mistake. We want to prove that for every $S \subseteq [n]$ $M(S)$ is logarithmic. Negation of this fact states that there exists $S \subseteq [n]$ having larger $M(S)$, not that all $S \subseteq [n]$ have larger $M(S)$. So, we may assume for contradiction that for some S, conjecture does not hold, but this does not imply that conjecture does not hold for every S, including S'. Thus $|S'_{m,i}|$ can be odd without violating the contradictory assumption, and this technique will not lead to a contradiction. We disprove this conjecture later on in Section 4.5

4.3 Trivial cases

In this section, we present some trivial limitations of the studied problem, which partly follow from the cases described in Section 2.3. As we can limit the main problem to some "interesting"/"worst-case" pairs of words, we may also present the following reductions of the studied sets.

Lemma 62. *W.L.O.G. we can consider only sets $S \subseteq [n]$ with even number of elements.*

Proof. Let us show that in all other cases, the main problem can be trivially solved with $f(n) \in O(\log n)$.

The fact that the discerning set S has an odd number of elements means that we are dealing with two words $u, v \in \{0,1\}^*$ with an odd number d of differences between them. Let u have k_u ones on the indices where it differs from v and ℓ_u ones on the remaining indices. Then the number of ones in v can be expressed as $\ell_u + (d - k_u)$, while u has $\ell_u + k_u$ ones in total. The difference in the number of ones between those two words is $|d - 2k_u|$, which is odd. Thus, by Lemma 28 we have $f(n) \in O(1)$. □

We may see that for the proof we used Lemma 28 about the different parity of the number of ones, not the stronger Lemma 29 about the different number of ones in general. Can this lemma introduce any more limitations to the studied problem? Surprisingly, the answer is no. Even if we limit ourselves to the words of the same compositions, we will still be dealing with all possible subsets of $[n]$ with even cardinality amongst their discerning sets. We prove it in the following lemma.

Lemma 63. *Collection of discerning sets of all pairs $u \neq v \in \{0,1\}^n$ with the same composition contains all non-empty sets of $[n]$ of even length.*

Proof. Consider $S \subseteq [n]$ of even length. Let us show that it can be constructed from two words $u \neq v$ of length n with the same number of ones in them. Let us divide S into two disjoint sets S_1, S_2 of the same length. Construct u from 0^n by placing ones at indices contained in S_1 and v similarly but placing ones on the indices from the S_2. It is clear, that the number of ones in both words is $|S_1| = |S|/2 = |S_2|$, and their discerning set $\Delta(u, v)$ is exactly S. Clearly, $u \neq v$ if $S \neq \emptyset$. □

We may furthermore employ lemmas from Subsection 2.3.6 about the number and positions of the differences in the words. As the main problem we are studying has a proven logarithmic lower bound, we may ignore the cases where the solution is clearly logarithmic. Those, in terms of discerning sets, we may omit the sets where the minimum element in $O(\log n)$ (see Lemma 30), sets where the maximum element is $O(\log n)$ close to n (see Lemma 31) and sets with $O(1)$ elements (see Lemma 32).

However, we did not find those limitations useful in our research. Sets with $O(1)$ size are easy to analyze, so their omission does not contribute to any significant reduction in computational time. In the next section, we also show that discerning sets can be shifted, preserving the cardinalities of their residue classes, so limitations about the smallest or the biggest element turn out to be not particularly useful. Also, the ratio of the sets to be omitted amongst all the sets is negligible.

4.4 Experiments

In this section, we describe different experiments which help us analyze the subject of discerning sets and present obtained results. All the experiments are implemented as Python scripts, and the source code is an inseparable part of this work. We describe the structure of the codebase in Appendix A and specify paths to corresponding scripts in the description of each experiment.

Our main goal is to take a look at critical sets. Obviously, it is not easy, as we need to go over all possible subsets of $[n]$ to find "the worst" ones. There is an exponentially large number of them, so we are able to do it only for small values of n. Hopefully, we will be able to deduct some pattern in their construction so that we may construct critical sets for larger values of n without the need to go through all possible sets. This would bring us closer to estimating some bounds for $M(n)$.

Experiment 64 (Critical sets)**.** *The goal of our first experiment is to compute $M(n)$ for various n and analyze what the critical sets look like. To achieve this, for given $n \in \mathbb{N}$, we go over all possible sets $S \subseteq [n]$ of even size and compute $M(S)$ for each of them. We then find a maximum $M(S)$ and store it together with all sets where $M(S)$ is the maximum. It is clear that the number of such sets grows exponentially, and we will be able to analyze only small n values.*

- Input*: n*
- Input range*: $6 - 34$*
- Output*: $M(n)$ values and the list of critical sets for given n*
- Path*: discerning_sets/generate/discerning_sets.py*

In Figure 4.2, we may see that obtained $M(n)$ values seem not to be logarithmic, but the very limited sample does not allow us to draw any firm conclusions about the asymptotic behavior of the function. Anyway, we may investigate the critical sets and determine some interesting facts about them. We refer to Table 4.1 for an idea of how the sets look like for several small values of n, while listing of all of them is rather long and is presented in Table B.1 in Appendix B. There,

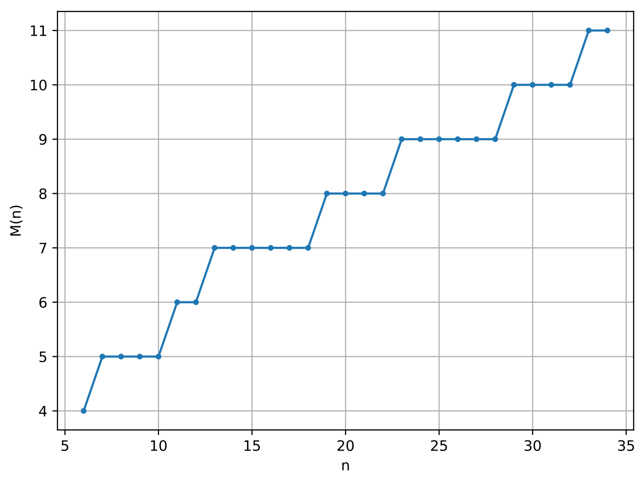


Figure 4.2: Results of Experiment 64: values of $M(n)$.

m	i	set	m	i	set
		$n=6$			$n=7$
4	0	{1, 2, 3, 4}	5	0	{1, 2, 3, 5, 6, 7}
4	0	{1, 3, 4, 6}			
4	1	{2, 3, 5, 6}			
		$n=8$			$n=9$
5	0	{1, 4, 5, 8}	5	1	{1, 3, 7, 9}
5	0	{1, 2, 3, 5, 6, 7}	5	0	{1, 4, 5, 8}
5	1	{2, 3, 4, 6, 7, 8}	5	0	{2, 5, 6, 9}
			5	0	{1, 2, 3, 5, 6, 7}
			5	0	{1, 2, 4, 6, 8, 9}
			5	0	{2, 3, 4, 6, 7, 8}
			5	0	{3, 4, 5, 7, 8, 9}

Table 4.1: Results of Experiment 64: critical sets for $n = 6, 7, 8, 9$.

we present all existing sets $S \subseteq [n]$ such that $M(S) = M(n)$ for n from 6 to 34. We introduce several lemmas and conjectures based on the results of the experiment. First, we note that $M(n)$ is piece-wise constant, and when it grows, it grows only by 1. We are interested in values of n where those "jumps" appear.

Lemma 65. *$M(n)$ is non-decreasing.*

Proof. Let $S \subseteq [n]$ be any critical set for a given n, i.e., $M(S) = M(n)$. Clearly, $S \subset [n+1]$, thus $M(n+1) \geq M(S) = M(n)$. □

Conjecture 66. *Values of $M(n)$ grow smoothly: for each n either $M(n) = M(n-1)$ or $M(n) = M(n-1) + 1$.*

Definition 67 (Cardinal n)**.** *We call $n \in \mathbb{N}$* cardinal *if $M(n) > M(n-1)$.*

The next observations we make are related to how the critical sets are constructed. We may see that for most of them, the sum of the first and the last element is equal to the sum of the second and the second-to-last ones, the third and the third-to-last ones, and so on till the middle of the critical set.

Definition 68 (Symmetric set)**.** *A critical set $S = \{a_1 < \cdots < a_k\}$ is* symmetric with a sum $\ell \in \mathbb{N}$ *if*

$$\forall\ i \leq \frac{k}{2} \quad a_i + a_{k+1-i} = \ell$$

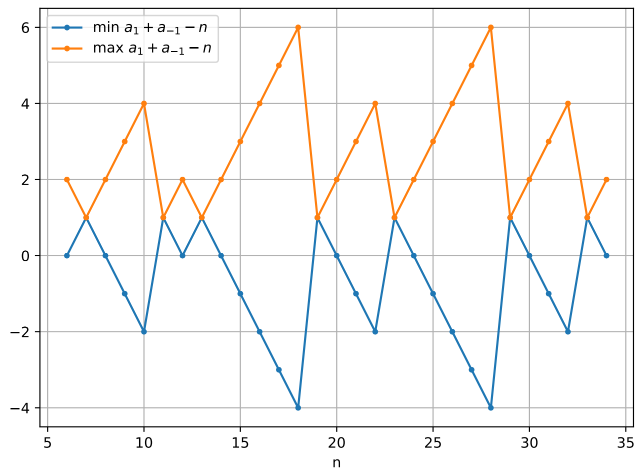


Figure 4.3: Results of Experiment 64: sums of the symmetric sets, shifted by n (e.g. critical sets for $n = 10$ have sums from $n - 2$ to $n + 4$).

In Figure 4.3, we see that most of the critical sets are symmetric with a sum close to n. The number of critical sets grows for every consecutive non-cardinal n, and so do the values of possible sums. For example, for $n = 8$ we have sets with sums $8, 9, 10$ (n to $n+2$), for $n = 9$ with sums from 8 to 12 ($n-1$ to $n+3$) and so on.

Definition 69. *We call the biggest interval of n's sharing the same $M(n)$ value* an interval*. For example, $13 - 18$ or $19 - 22$ are intervals. When we say i-th non-cardinal n, we mean that n is at the $(i+1)$-th position in its interval.*

We are now ready to present some conjectures about the sets' sums.

Conjecture 70. *Cardinal n's are always odd and have exactly one critical set S. It is symmetric with a sum $n + 1$.*

Conjecture 71. *Most of the critical sets for non-cardinal n are also symmetric, with a bigger interval of sums present. Symmetric sets for i-th non-cardinal n have sums from $n - i + 1$ to $n + i + 1$, with each value present at least once. In particular, a critical set with sum $n + 1$ is present for every n.*

Then we realize that some sets are similar to each other, and we do not need to consider them separately. For example, $S = \{a_1, \ldots, a_k\}$ and $S' = \{a_1 + 1, \ldots, a_k + 1\}$ behave equivalently in terms of the existence of an odd residue class, and $M(S) = M(S')$. Thus, if we are interested in some specific construction patterns of critical sets, we may filter out the sets which behave similarly to the ones we already have.

Definition 72 (Equal w.r.t. shift). *We say that sets* $S_1 = \{a_1, \ldots, a_k\}, S_2 = \{b_1, \ldots, b_k\}$ *are* equal w.r.t. shift *if*

$$\exists d \ : \forall\ i \in [k] \quad a_i - b_i = d.$$

It is easy to see that the relation "equal w.r.t. shift" is an equivalence.

Definition 73 (Canonical set). *We define* canonical sets *for given n as the representatives of equivalence classes of the relation "equal w.r.t. shift". From each class, we choose a representative to be a set where the smallest element is* 1*. It is clear that such an element always exists.*

It makes sense to stick only to canonical critical sets from now on. Also, every set which is critical for $n-1$ is also a critical one for n, as long as n is not cardinal. Thus, if we want to reduce the collection of considered critical sets even more, it makes sense to take a look only at *unique* sets.

Definition 74 (Unique set). *We call the critical set for given n* unique *if it is not a critical set for smaller n.*

Now, we are able to purify the collection of considered sets in order to analyze only sets possessing some unique characteristics.

Definition 75 (Proper set). *We call the critical set for given n* proper *if it is unique, canonical, and symmetric.*

Lemma 76. *Proper sets necessarily contain both* 1 *and n and thus have a sum* $n + 1$.

As follows from the results of Experiment 64, at least one proper set exists for every n we considered, and we conjecture that it holds even for larger n values. Thus we can decrease the number of sets considered in the experiment, which will decrease computational time, and we will be able to obtain more $M(n)$ values and critical sets for them thanks to it.

Conjecture 77. *For any n, there exists a proper set.*

Definition 78. *We denote by $M'(n)$ a restriction of $M(n)$ to proper sets, i.e.,*

$$M'(n) = \max(\{M(S) \mid S \subseteq [n], 1, n \in S, S \textit{ is symmetric}\}).$$

Clearly, $M'(n) \leq M(n)$, but based on the results of the first experiment, we assume that the values are actually equal, as a proper set exists amongst critical ones for any n.

Conjecture 79.

$$\forall n \in \mathbb{N} \quad M(n) = M'(n).$$

Before we move to the result of the next experiment, where we compute $M'(n)$ for higher n values, let us recall that by taking into account only proper sets, we cover all existing symmetric ones (those we do not analyze directly are those equal to considered w.r.t. to shift or those which are considered for smaller values of n). But what about the sets which are not symmetric? We empirically observe

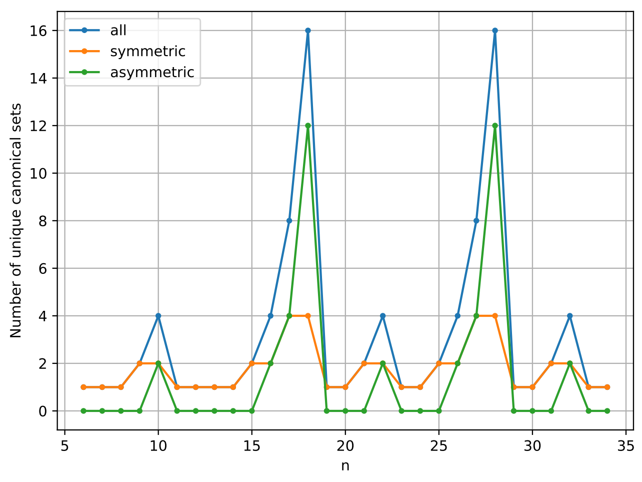


Figure 4.4: Results of Experiment 64: number of symmetric and asymmetric unique canonical sets.

symmetric	asymmetric	total
1	0	1
2	0 or 2	2 or 4
4	4 or 12	8 or 16

Table 4.2: Results of Experiment 64: correlation of the number of symmetric and asymmetric unique sets.

that the number of symmetric and asymmetric sets for each n correlate (see Table 4.2 and Figure 4.4). Interestingly, the total number of unique canonical sets is always a power of 2, and so is the number of proper sets. For now, we leave those observations as is and focus on proper sets, as we believe that this restriction is valid and results in a significant reduction of computational time for higher n's.

Experiment 80 (Proper sets)**.** *This experiment is similar to Experiment 64, but we limit ourselves to proper sets, as we believe that at least one such set exists for every n (as stated in Conjecture 77). To do this, we go all over all sets $S' \subseteq [\lfloor n/2 \rfloor - 1]$ and for each of them construct*

$$S = \{1\} \cup \{a + 1 \mid a \in S'\} \cup \{n - a \mid a \in S'\} \cup \{n\}.$$

For each S, we compute $M(S)$, find a maximum $M(S)$ over all S, and store it together with all sets S where $M(S)$ is maximum. We pursue the same goals as in the previous experiment - find $M'(n)$ and take a look at the critical sets to find some clues about how they are constructed.

- Input*: n*
- Input range*: $6 - 62$*
- Output*: $M'(n)$, list of proper sets for given n*
- Path*: discerning_sets/generate/discerning_proper_sets.py*

This definition of an experiment allows us to save some computational time, as we do not need to go over all subsets of $[n]$. In Experiment 64, we analyzed all non-empty sets of $[n]$ of even size. There are $2^{n-1} - 1$ of them. In Experiment 80, we only need to construct half of the set (i.e., elements from $[\lfloor n/2 \rfloor]$), also knowing that 1 is always in the set. Thus, we examine only $2^{\lfloor n/2 \rfloor - 1}$ sets and can run the experiment for approximately twice bigger n.

This experiment allows us to obtain more values of the function $M'(n)$, which looks more promising for higher n, as the curve is starting to look flatter (see Figure 4.5). We are also able to examine more critical sets for some clues on their construction (see Table B.2).

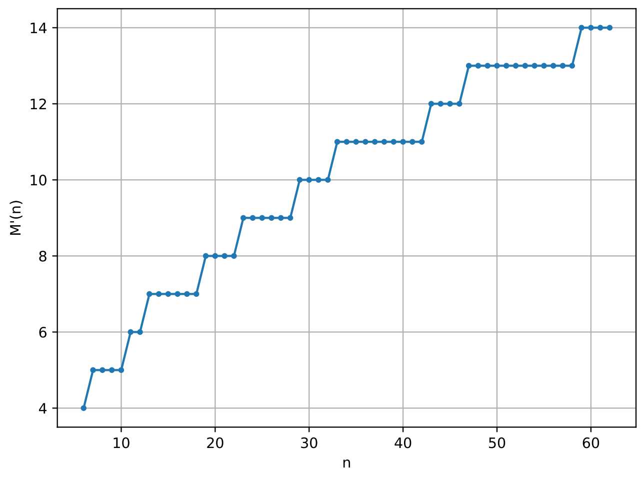


Figure 4.5: Results of Experiment 80: values of $M'(n)$.

We may also take a look at cardinal values of n and see if there is some pattern in whether the n is cardinal or not. In Table 4.3, we see that almost all (9 out of 10) cardinal n are prime, but not all prime numbers are present. This is an interesting result, which, however, can not be generalized as it seems that cardinal n's do not correspond to some obvious pattern.

$M'(n)$	**cardinal n**
5	7
6	11
7	13
8	19
9	23
10	29
11	33
12	43
13	47
14	59

Table 4.3: Results of Experiment 80: cardinal n's.

We again observe that the number of examined sets is always a power of 2, which we already saw in the result of the previous experiment. Moreover, if we restrict ourselves to an interval of n's with the same $M'(n)$, the number of proper sets is the same for consecutive odd and even n, while for the next odd n, it will be twice as big unless it is cardinal (see Figure 4.6). It is particularly well seen for bigger intervals of n's, we present one of them in Table 4.4. The number of the proper sets is 1 for a cardinal $n = 33$. Then, it doubles with every second value of n for n up to 42 until it becomes 1 again for a cardinal $n = 43$. The clear assumption would be that the proper sets for the odd non-cardinal n are somehow constructed from the sets for two previous n's (even and odd).

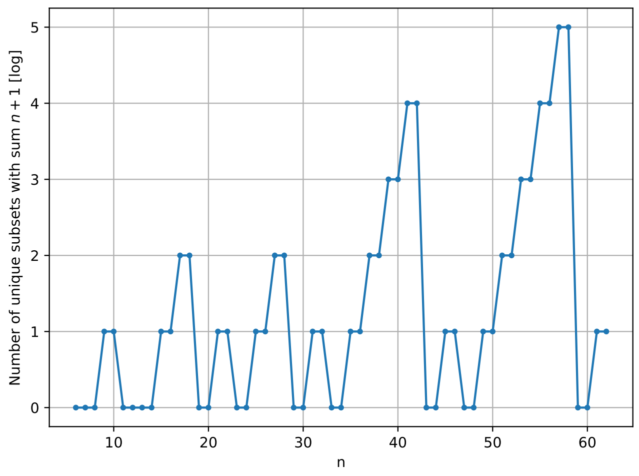


Figure 4.6: Results of Experiment 80: Number of proper sets (logarithmic).

n	number of proper sets
33	1
34	1
35	2
36	2
37	4
38	4
39	8
40	8
41	16
42	16
43	1

Table 4.4: Results of Experiment 80: number of proper sets for n from 33 to 43.

Now, it is time to take a closer look at how the proper sets are constructed so that we can prove or disprove our assumption. To do this, we construct a "critical matrix", where we visualize every set as a binary number depending on whether or not each element of $[n]$ is present in the set. Our main goal is to determine some patterns they possess.

Definition 81 (Critical binary representation)**.** *For every proper set $S \subseteq [n]$ with sum $n+1$ we denote its* binary representation *as a word $x \in \{0,1\}^{\lfloor n/2 \rfloor}$ defined* *as follows.*

$$x_i = \begin{cases} 1, & \textit{if } i \in S, \\ 0, & \textit{otherwise.} \end{cases}$$

This representation is a bijection because such set S is fully determined by the first half of its elements (others can be computed due to the set's symmetry), and the biggest element of the first half is $< (n+1)/2 \leq \lfloor n/2 \rfloor$.

Experiment 82 (Proper matrix)**.** *In this experiment, we convert every proper set $S \subseteq [n]$ to its binary representation $x \in \{0,1\}^{\lfloor n/2 \rfloor}$ as described above. We construct a binary matrix of sets for given n, where each proper set is represented as a column of the matrix. Columns are sorted lexicographically.*

- Input*: proper sets with sum $n+1$ (listed in Table B.2)*
- Input range*: $6 - 62$*
- Output*: visualization of sets as a matrix for each n*
- Path*: discerning_sets/plot/plot_proper_matrix.py*

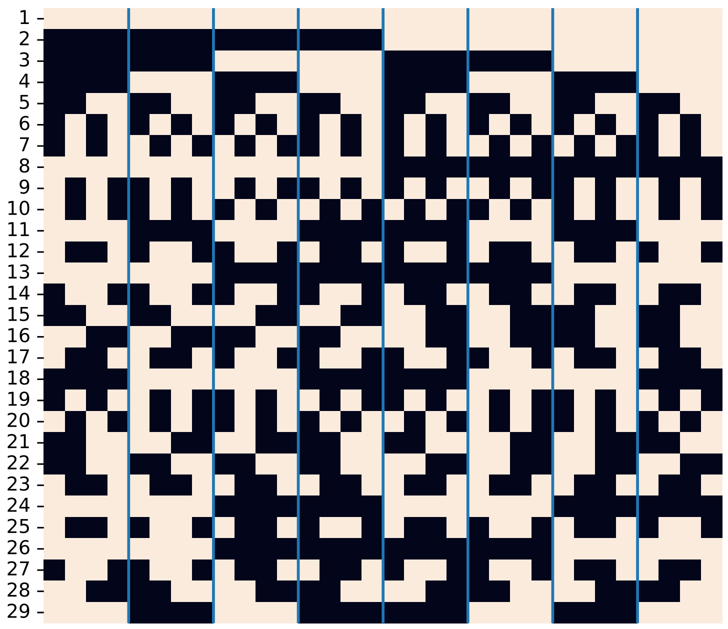


Figure 4.7: Results of Experiment 82: proper matrix for $n = 58$.

Figure 4.7 contains 32 proper sets for $n = 58$, divided into groups of four by visual guidelines. The picture does not look arbitrary at all, and we may note several interesting observations, which hold for other examined values of n as well (we do not present matrices for all examined values of n here, as it would take a lot of space, but all the observations could be verified in Table B.2). First, we may see that in those groups of four, which we separated, every number is presented 0, 2, or 4 times. Moreover, the presence of each number is balanced amongst all sets.

Conjecture 83. *Every element is contained in all, half, or none of the proper sets for given n.*

Moreover, we see that those appearances in the groups of four are codependent. Let us take 20, for example. It appears in the first and the third set of a group (i.e., we have 1010 in its row), and we observe that its appearance in the next group will always be the same (1010) or inverse (0101).

Conjecture 84. *In critical sets sorted lexicographically according to their binary representation and divided into groups of four, each element appears in an even number of the sets in each group (0 - none of them, 2 - exactly half, or 4 - all). Moreover, its appearance in the next groups coincides with the first group in a way that it is the same or completely inverse.*

Besides those facts, we may also take a closer look at the first 6 rows of the matrix. It turns out that they contain all binary representations of numbers from 100000 to 111111 in columns, each number exactly once.

Conjecture 85. *For any $i \geq 2$, proper sets for i-th non-cardinal n contain all possible combinations of numbers from 2 to $\lfloor i/2 \rfloor + 1$ (1 is always present in all sets).*

This allows us to state an even stronger conjecture. As none of the sets for one n share the same elements amongst the first ones, and the number of possible combinations of those elements perfectly correspond to the number of examined sets, we may assume that every proper set is completely determined by the appearance of the several smallest elements in it. We define it precisely in the following conjecture.

Conjecture 86. *For any i-th non-cardinal n, $i \geq 2$, every proper set has a unique combination of elements from 2 to $\lfloor i/2 \rfloor + 1$ and is completely determined by it.*

For now, we looked at the proper sets for each n separately. However, as consecutive non-cardinal n share $M(n)$ value, we assume that their proper sets should share some construction patterns and sets for higher n values could be constructed from the sets for smaller ones. Because of that, we concatenate the critical matrices for the two biggest intervals of non-cardinal n values we obtained, $33 - 42$ and $47 - 58$, in Figures 4.8 and 4.9.

The last conjecture we stated was related to the first elements of the sets. In Figure 4.8, we see that even the last elements of the set are somehow related. If we take a look at the last 5 elements of the sets for $n = 41$ and $n = 42$, we see that their appearances correspond perfectly, although they are not the same numbers (last 5 elements for $n = 41$ are number from 16 to 20, while for $n = 42$, those are 17 to 21). There is no such obvious correspondence between $n = 40$ and $n = 41$, but there is a similar one for $n = 39$ and $n = 40$, although it affects only 4 rows. It turns out that this holds for all the n values in this interval. More precisely, for i-th and $(i + 1)$-th non-cardinal n values, where i is even, last $\lfloor i/2 \rfloor + 1$ elements share the same pattern. Unfortunately, this statement is only true for this interval (with $M(n) = 11$), and the claim breaks for another examined interval $47 - 58$ ($M(n) = 13$). There are no obvious patterns in that interval.

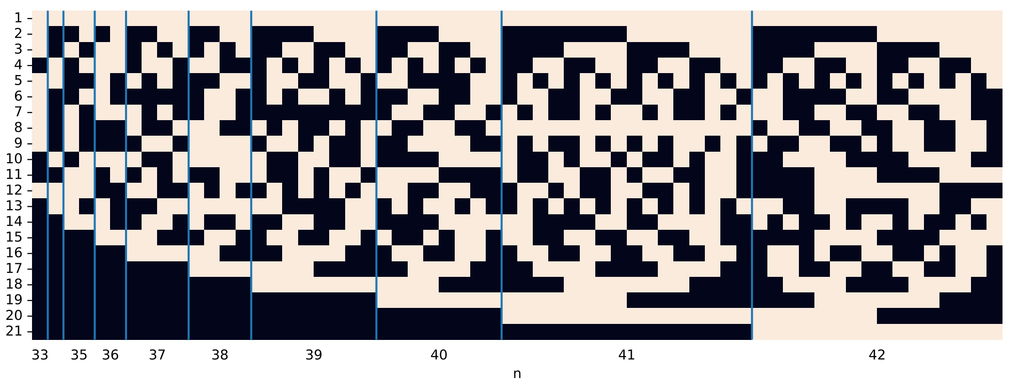


Figure 4.8: Results of Experiment 82: proper matrix for n from 33 to 42.

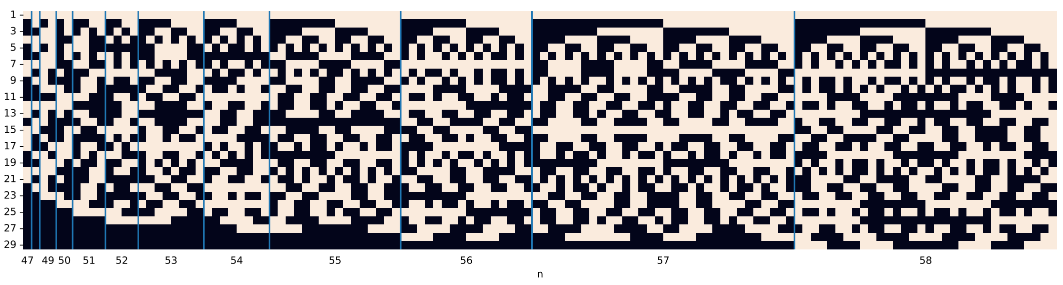


Figure 4.9: Results of Experiment 82: proper matrix for n from 47 to 58.

Unfortunately, we could not prove the conjectures stated in this section, but we believe this direction is promising. We list several essential ones which remain unproven.

Conjecture 87 (Discerning sets structure)**.** *We list the claims about the structure and construction of critical sets which we believe hold.*

- *Every n has a proper set S with $M(n) = M(S)$.*
- *Every cardinal n has exactly one critical set, which is also proper.*
- *There are $2^{\lfloor i/2 \rfloor}$ proper sets for i-th non-cardinal n amongst values of n sharing the same $M(n)$ value.*
- *Proper sets for i-th non-cardinal n are fully determined by the appearances of number from 2 to $\lfloor i/2 \rfloor + 1$ in them.*
- *Proper sets for i-th and $(i+1)$-th non-cardinal n's, where $i \geq 2$ is even, may be constructed as a combination of the proper sets for $n-1$ and $n-2$.*

We believe solving at least some of those conjectures will introduce valuable progress toward estimating $M(n)$ bounds. The ultimate goal, however, is to present a construction of one or more critical sets for a given n, which will allow to compute $M(n)$ without the need to go over different critical sets candidates.

4.5 Theory

In this section, we show that Conjecture 60 does not hold in general, and the approach described in this section does not help to achieve the desired upper bound for the discerning problem. To show this, we need to get back to the roots of the problem we are solving. The constructed discerning automaton accepts the words having an odd number of ones at indices belonging to some residue class. Thus, to discern two words, we need to find a residue class of indices where the numbers of ones in those words have different parity. It turns out that the number of those residue classes with logarithmic modulo is too small to distinguish all the words of length n sufficiently.

Lemma 88.
$$M(n) \in \Omega(\sqrt{n}).$$

Proof. Let us fix some word length n. For every pair of words u, v, we need to find $2 \leq m \leq M(n)$ and $0 \leq i \leq m$ such that $|u_{m,i}|_1 \not\equiv |v_{m,i}|_1 \mod 2$. Let us create a mapping $r : \{0,1\}^n \to \{0,1\}^k$ describing the words in terms of those remainders. I.e., for every modulo $2 \leq m \leq M(n)$ and every remainder $0 \leq i \leq m$, we write down 1 if the number of ones on corresponding indices is odd and 0 otherwise. How long would such a description be? Clearly, we write down $k = 2 + 3 + \cdots + M(n) = O(M(n)^2)$ numbers. Thus, the range of function r contains $2^k = 2^{O(M(n)^2)}$ values or even less, as probably not all combinations of parities are admissible. To be able to distinguish all pairs of input words with some modulo up to $M(n)$, we require r to be an injection, so the necessary condition is $2^n \leq 2^{O(M(n)^2)}$, which proves the claimed bound.

It is worth mentioning that we assume the equivalence between the discerning set problem, where $M(n)$ is defined as a maximum of $M(S)$ over all $S \subseteq [n]$, and discerning the words by a discerning automaton $A(m, i)$. Indeed, the existence of two words with the same parity of the number of ones in all residue classes implies that their discerning set has all residue classes of even cardinality. □

This consideration corresponds to one Chase [2021] presented in his work. He claimed that comparing the numbers of ones at some residue class indices is not sufficient by itself and leads to $O(\sqrt{n} \log^{3/2} n)$ upper bound. Clearly, the number of ones is a stronger criterion than its parity. Thus, obtaining any better result would actually contradict Chase's claim.

Although we proved that the approach described in this section will not help to improve the bounds for the discerning problem, this does not diminish the fact that the obtained results may be valuable for some other research involving all subsets of $[n]$ with small residue classes considered. We are convinced that the structure of the examined critical sets is interesting, and this topic deserves further research.

5. Random automata

In this chapter, we study an alternative approach to the discerning problem. Specifically, we assume that there is no need to construct a tailored automaton for two words given as input. Instead, we might take several randomly generated small automata and use them together to distinguish the input with high probability. Clearly, just some random automata may not be good enough, then one can consider a specific subclass of automata. Namely, we consider random permutation automata (as defined in Section 1.4) and compare their performance with the simple ones. We also introduce one slight improvement to the permutation automata construction. We analyze the behavior of randomly generated automata on randomly chosen subsets of words of a given length and claim that the random permutation automata perform well on average in contrast to the simple random ones.

We start with some definitions, theoretical considerations, and conjectures in Section 5.1 and support them with several conducted experiments and empirically obtained results in Section 5.2. We show that a specific subclass of random automata actually performs well on average for randomly generated pairs of words, even with a constant number of states. Thus it could be used as the first step of the discerning process, followed by a traditionally constructed automaton in case the random one will not be successful in distinguishing the words.

5.1 Theory

In this section, we discuss what we want to measure and how, what pitfalls this approach has, and how we can improve the performance of the automata. Clearly, when we analyze an automaton's performance, the most important part is the evaluation method. If we want to estimate an automaton's success rate for longer words empirically, we can not run the automaton over all of them, as it will demand enormous computational time with an exponentially growing number of words. Instead, we generate some number of word pairs randomly and test the automaton on them. However, as Demaine et al. [2011] mentioned, the discerning problem is small on average.

Lemma 89 (Demaine et al. [2011])**.** *The average size of an automaton discerning two words chosen uniformly at random from $\{0,1\}^n$ is constant.*

Proof. The probability that two randomly chosen words differ for the first time at i-th position is 2^{-i}. If the first difference between the words occurs at i-th position, the automaton with $i+2$ states is sufficient (see Lemma 30). Then

$$\mathbb{E}(\#\text{ states}) = \sum_{i=1}^{m} 2^{-i}(i+2) \leq 4.$$

□

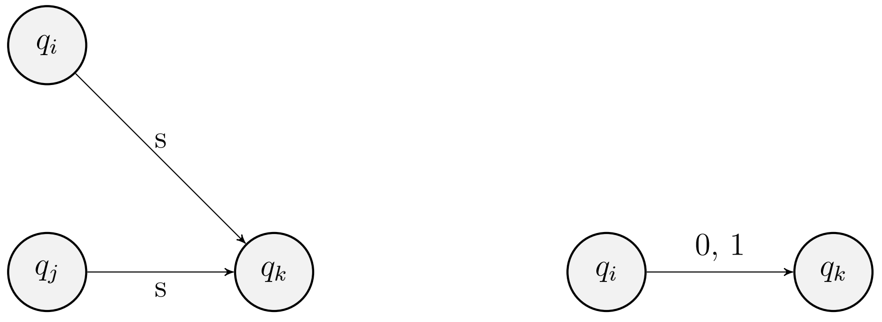
This consideration, however, uses a construction of an automaton specifically tailored for the input words. We, on the contrary, assume that even a not tailored automaton may be sufficient. Nevertheless, we can expect that some word pairs are more difficult to distinguish than others, and if there are not many of them, we may not encounter them during the evaluation of automata performance. Thus, it is not possible to guarantee absolute accuracy with this approach.

Let us return to the automata we will analyze: random ones (see Definition 14) and random permutation ones (see Definition 15). We call the latter just *permutation* ones from now on. It is clear that permutation automata feature a certain advantage compared to random ones. If we have the same substring in both input words, a permutation automaton can not finish in the same state after reading it unless it was already in the same state for both words at the beginning of the substring. Thus, once achieved, discernment is preserved until the next difference between the word. A random automaton can not guarantee this.

This consideration would be almost sufficient for the permutation automaton to always discern the words containing only one difference between them. However, it can happen that the transitions it makes reading the different symbols lead from the same state to the same state – then, the automaton will not be able to distinguish the words. Luckily, we can avoid this by introducing a class of improved permutation automata as follows.

Definition 90 (Shifted permutation automaton)**.** *We define a construction of* a shifted permutation automaton $M = (Q, \{0, 1\}, \delta, q_0)$*, where* $Q = \{q_0, \ldots, q_{k-1}\}$*, in a way that we choose a permutation* π *of the numbers from* 0 *to* $k-1$ *uniformly at random and assign* $\delta(q_i, 0) = q_{\pi(i)}$ *and* $\delta(q_i, 1) = q_{\pi(i+1 \mod k)}$ *for all* $0 \le i < k$*. It is clear that this construction yields a permutation automaton as introduced in Definition 13, and moreover* $\delta(q_i, 0) \neq \delta(q_i, 1)$ *for all the states* $q_i \in Q$*.*

There are clearly more ways to ensure that the transitions from any state lead to different states. For example, instead of $\delta(q_i, 1) = q_{\pi(i+1 \mod k)}$, one can assign $\delta(q_i, 1) = q_{\pi(i)+1 \mod k}$. Those automata types, however, behave similarly, so we present the analysis only for one of them. This gives us three classes of automata to analyze, which could be described by the parts they can not contain, which we show in Figure 5.1.



(a) for permutation automata (b) for shifted permutation automata

Figure 5.1: Forbidden configurations

- *Random automaton* may not preserve the discernment while reading the same substring (i.e., being in different states, the automaton can get into the same state even if it reads the same sequence of symbols in both runs).

- *Permutation automaton* preserves the discernment but does not guarantee its creation (i.e., it can not get to the same state from different ones reading the same input but can get to the same state from the same one after reading different inputs).

- *Shifted permutation automaton* preserves the discernment and guarantees its creation (i.e., being in the same state and reading different symbols, it would necessarily get to the different states).

None of the automata can guarantee to discern more than one difference in the input, as we can not say anything about the possibility of getting to the same state from the different ones by different symbols. Thus, even if we encountered a discernment, the automaton can get back to the same state after the next difference between the words. Now, let us see if we can assume something about the automata's behavior before we conduct the experiments.

5.1.1 Permutation automata

As we already mentioned, the transition function of a permutation automaton represents an injective mapping for each symbol. This gives us an opportunity to easily analyze what is happening when the automaton deals with the words having only one difference between them.

Lemma 91. *A randomly constructed permutation automaton with m states distinguishes the words of length n with one difference between them with probability* $\frac{m-1}{m}$.

Proof. Let us have an automaton $M = (Q, \{0,1\}, \delta, q_0)$. For any state $q \in Q$, by construction, it is clear that the probability that $\delta(q,0) \neq \delta(q,1)$ is $\frac{m-1}{m}$, as both permutations for the transition function are taken randomly. Let the words u and v have the one and only difference at index i. As we are dealing with the permutation automaton, we may be certain that if the automaton is in different states after reading the difference, it will remain so till the end and distinguish the words. If the automaton is in the same state after reading the difference, it will also remain so and will not distinguish the words. Thus, $P[\delta(q,u_i) \neq \delta(q,v_i)] = P[\hat{\delta}(q,u_{i\ldots}) \neq \hat{\delta}(q,v_{i\ldots})]$. So, after reading the beginning of the words $u_{\ldots i-1} = v_{\ldots i-1}$ the automaton will be in some state $q \in Q$, and, as we showed, the probability of distinguishing the words $P[\hat{\delta}(q_0,u) \neq \hat{\delta}(q_0,v)]$ will be the same as $P[\delta(q,u_i) \neq \delta(q,v_i)]$, which is $\frac{m-1}{m}$. □

Corollary. The probability of distinguishing the words with one difference by a permutation automaton does not depend on the length of the words.

Thus, if we want to achieve, for example, a 99% success ratio, an automaton with $m = 100$ states will be sufficient, regardless of the input length. The obtained result, however, is not strong or surprising, as we can construct even a 2-state deterministic automaton that is able to distinguish such words. Also, this consideration does not hold for simple random automata, as they can ”collapse” to the same state even while reading the same input - thus, the different states immediately after the difference do not guarantee the different states at the end.

The computation of the success probability for a higher number of differences between the words is, unfortunately, much more complicated. We can not assume the probability of $\delta(q,0) \neq \delta(q,1)$ independently, as we need to take into account if we already were in a state q or not and the probability of finishing in the particular state after reading multiple symbols is hard to estimate without knowing the exact automaton's structure. Even if we want to analyze two consecutive differences one after another, we need to start with a result of the first transition, which could be two new states, the same new state, the same state as the automaton already was in, or one same and one different. In each of the cases, there are plenty of possibilities to analyze with regard to the situation after the second transition. However, for the case of 2 or more consecutive differences, we can provide a lower bound for the distinguishing probability by taking into account only the case where all the visited states are unique. Clearly, for a sufficiently large number of states, this case will be the most probable and thus gives us a reasonable estimate.

Lemma 92. *m-state permutation automaton distinguishes two words u and v with k **consecutive** differences, where $2k < m$, with probability P such that*

$$P \geq \frac{(m-1)\cdots(m-2k)}{m^2(m-1)^2\cdots(m-k+1)^2} \geq 1 - \frac{4k^2}{m}$$

Proof. Let us assume that the automaton is in the state q before the first differences. We consider a situation where after each read symbol in both words, the automaton gets to a new state. I.e., there exist $2k+1$ distinct states $Q_d = \{q, q_{u_1}, \dots, q_{u_k}, q_{v_1}, \dots, q_{v_k}\}$ such that the automaton goes through $q \to q_{u_1} \to \cdots \to q_{u_k}$ while reading the first word, and through $q \to q_{v_1} \to \cdots \to q_{v_k}$ while reading the second word. Let us call a probability of this happening $P_{distinct} = P[q \neq q_{u_i} \neq q_{v_j} \quad \forall i,j \in [k]]$. Clearly, as $q_{v_k} \neq q_{u_k}$, if the described situation happens, the automaton will distinguish the words. So, $P \geq P_{distinct}$. The first rough estimate we could obtain is to say that for any state in $Q_d \backslash \{q\}$ the possibility that some already existing state would be picked instead of it is $\frac{2k}{m}$. This gives us a second bound from the lemma.

$$P_{distinct} \geq 1 - 2k \cdot \frac{2k}{m}.$$

To obtain the stricter bound, we need to compute the probability using the fact that we are dealing with a permutation automaton. That is to say, no state can have more than one incoming arrow with the same symbol. Let us pick the states for Q_d one by one. We can do this under the assumption that we are not revisiting the states. Thus, we have no additional constraints for the values of the transition function for them. In the beginning, $Q_d = \{q\}$. When we pick q_{u_1}, we have m states to choose from, and $(m-1)$ of them are suitable. For q_{v_1} there are also m possibilities, but only $(m-2)$ suitable ones (as $q_{v_1} \neq q, q_{u_1}$). Now we have one arrow with each symbol in our automata (as $u_1 \neq v_1$). Thus, when we choose states q_{u_2} and q_{v_2}, there are only $(m-1)$ possibilities for each of them, from which $(m-3)$ and $(m-4)$ are suitable, respectively. As in each step we necessarily define one transition by zero and one transition by one, the number of possibilities at i-th step is always $(m-i+1)$, while the number of suitable possibilities is equal to the number of the states not visited yet, and will be $(m - 2i + 1)$ for q_{u_i} and $(m - 2i)$ for q_{v_i}. Continuing like this, we obtain that

$$P_{distinct} = \prod_{i=1}^{k} \frac{(m - 2i + 1)(m - 2i)}{(m - i + 1)^2},$$

which gives us a desired bound. □

This consideration gives us a lower bound on distinguishing the words with k consecutive differences by a permutation automaton, but the bound is getting smaller with the increasing number of differences - this is because the assumption that we will visit only previously unvisited states becomes less and less probable.

Nevertheless, while testing automata on randomly generated words, we can not guarantee that the differences in the words will be consecutive. Moreover, between all possible pairs with k differences, there is a negligible number of those ones with consecutive differences. Thus we are not able to make any general conclusions about the automata success rate based on this approach. We conclude this subsection with a simple conjecture.

Conjecture 93. *Permutation automata have a higher success rate than random ones.*

5.1.2 Shifted permutation automata

With a shifted permutation automaton, there are two statements that are certainly true.

Lemma 94. *A shifted permutation automaton always distinguishes words with one difference between them.*

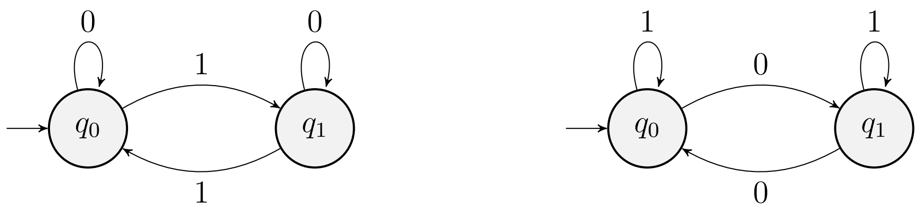
Proof. Let us consider a shifted permutation automaton $M = (Q, \{0, 1\}, \delta, q_0)$. By its construction, if M is in a state $q \in Q$ before reading the difference, it will get to the different states after it, as $\delta(q, 0) \neq \delta(q, 1)$ by definition, and remain there (by nature of permutation automata). □

Lemma 95. *A shifted permutation automaton with* 2 *states always distinguishes words with an odd number of differences between them.*

Proof. If we apply all the constraints in the definition of the shifted permutation automaton, we see that there are only two 2-state automata satisfying all of them. Figure 5.2 show both of them, and we may see that those are exactly the parity ones. I.e., they discern the words with different parity of the number of ones/zeros (See Lemma 28). Clearly, the words with an odd number of differences will have different parities of the numbers of zeros and ones. □

Other than that, the analysis is also complicated, so we only conjecture that the shifted permutation automaton will perform better than the simple permutation one.

Conjecture 96. *Shifted permutation automata have a higher success rate than permutation ones.*



(a) Automaton discerning parity of ones (b) Automaton discerning parity of zeros

Figure 5.2: Shifted permutation automata with 2 states

5.1.3 Random automata

With random automata, we can not predict anything - the main difference is that we can get to the same state from the different ones by the same symbol. So, we can say that the finishing state is also random and does not depend on the word we are reading. Then, if the probability of ending in every state is the same, the probability that automata distinguish two input words is $\frac{m-1}{m}$. However, it is obvious that the probability of finishing in each state is not the same - some of the states have more input arrows than others (otherwise, it would be a permutation automaton). This is the reason why we can not even compute the probability of the automaton distinguishing two words with one difference. Obviously, if the words differ in the last symbol, the probability will be $\frac{m-1}{m}$, but if the difference occurs somewhere in the middle, the result depends on the whole words' parts which come after the difference, and it is impossible to estimate finishing state(s) without knowing the automaton construction. Even the estimation of distinguishing probability for the words with consecutive differences, as we stated in Lemma 92 for a permutation automaton, is not possible because the fact that the automaton will be in different states after the last difference does not guarantee that it remains so. To the best of our knowledge, no analysis of such randomly generated automata was conducted previously.

The main problem of the random automaton is that it can converge to the same state from different ones while reading the same input. For example, if the words contain only one difference close to the beginning, it is natural to assume that the random automaton will distinguish them with a lower probability than if there is a difference close to the end of the words. As we generate the word pairs randomly, we do not take into account the positions of the differences in the word. However, with a higher number of differences, there probably will be a difference closer to the end of the words. Thus, we come up with a natural conjecture.

Conjecture 97. *The success rate of random automata increases with a growing number of differences between the input words.*

5.2 Experiments

In this section, we present empirical results on the success rate of automata from three different classes (random, permutation, and shifted permutation). We start with an input that differs only in one symbol in Experiment 98, where we see that both permutation classes drastically outperform the random one. We then proceed with Experiment 99, increasing the percentage of differences between the words, which helps random automata to distinguish the words better. After that, we recall what a simplified DFA is (see Definition 8) and realize that a fixed starting state may negatively affect the performance of the automata. In Experiment 100, we measure the success rate of the simplified versions of the automata and see that it improves the performance of both permutation types but not the random one. In Experiments 101,103 and 105, we are looking for the smallest number of states for a random shifted permutation automaton to be successful for all the words of a given length from a randomly generated sample. It turns out that it performs very well on average - even a constant size is sufficient with high probability. However, there exist words that are not distinguishable even with a linear size. Finally, we conduct the same experiments for the random automata to see that it does not perform well even on average - the successful size is twice the length of the word.

We start with analyzing the performance of three automata types on random words which differ only in one symbol.

Experiment 98 (One symbol difference)**.** *For given word length n, number of tried words w, number of tried automata t, and number of automata sizes to try s, we generate a specified number of automata sizes logarithmically on a scale from* 1 *to n. For each automata size, we generate w random words from $\{0,1\}^n$, and in each of them, randomly choose and swap one symbol to obtain a random pair. Then, we generate t automata of each type and compute the success rate - the number of tries where the automaton successfully distinguished the words (ended in different states) divided by the total number of tries. For a given automaton size, we compute the overall average success rate - i.e., the number of successful discernments divided by all $w \cdot t$ tries, and also the minimum and maximum success rate over all words (i.e., a ratio of successful tries for the best and worst distinguishable pair).*

- Input*: n (word length), w (number of words), t (number of tries), s (number of automata sizes).*
- Input values*: $n = 4000$, $w = 100$, $t = 1000$, $s = 100$.*
- Output*: average, minimum, and maximum success ratio for each automata type and size*
- Path*: random_automata/experiments/success_rate_one_symbol.py*

In Figure 5.3, we see that shifted permutation automata with more than one state always discern the words with one difference, as we stated in Lemma 94. Otherwise, in terms of performance, permutation automata with $\log n$ states distinguish the words in more than 85% of tries and more than 98% with $\sqrt{n}$ size. The obtained results also confirm the Lemma 91, as $m_1 = \lfloor \log n \rfloor = 8$ and $\frac{m_1-1}{m_1} = \frac{7}{8} = 0.875$, while $m_2 = \lfloor \sqrt{n} \rfloor = 63$ and $\frac{m_2-1}{m_2} = \frac{62}{63} \approx 0.984$. Random automata do not have those rates, even for bigger sizes. The average success ratio is around 65%, while the minimum and maximum ones are approximately 36% and 98%, respectively. We may assume that a hardly distinguishable pair of words with a low success ratio contained a difference toward the beginning of the

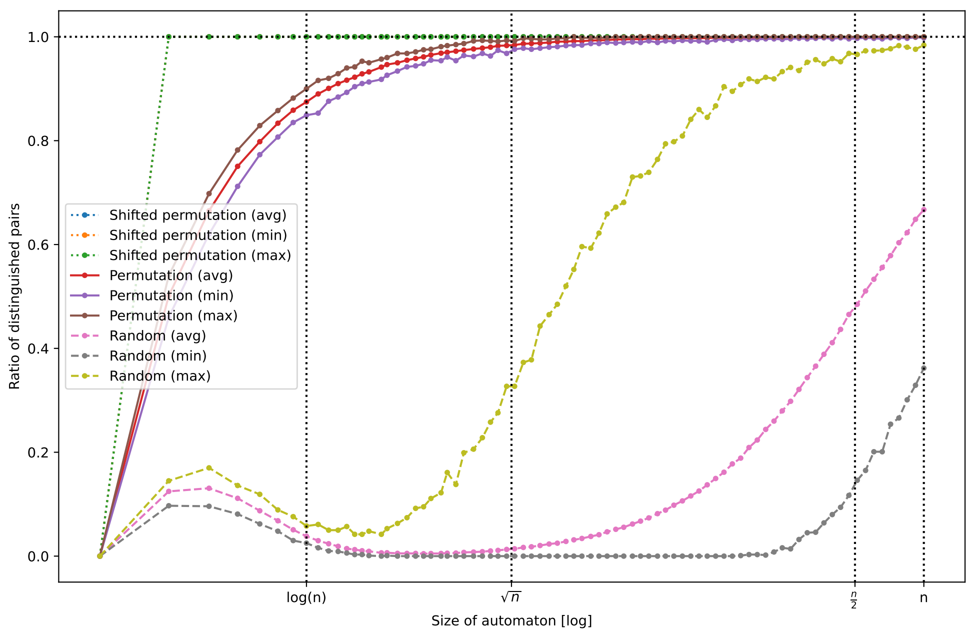


Figure 5.3: Result of Experiment 98: success ratio of automata of different types and sizes for $n = 4000$, input words differ in one symbol.

words, while the easiest one contained a difference near the end, which allowed the maximum success rates to at least tend to 100%.

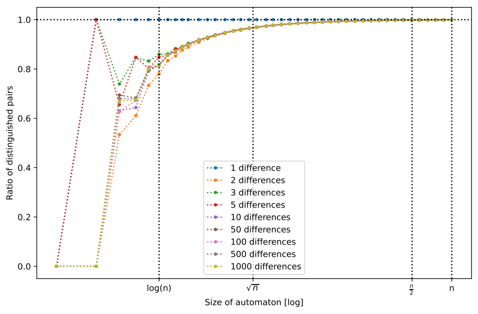
We proceed with a similar experiment, changing only the input words to contain more than 1 difference.

Experiment 99 (Multiple symbol difference)**.** *Given the number of differences to be done in the word, we choose a subset of positions of a given size uniformly at random and swap all the symbols on those positions. To reduce computational time, we process shorter words and try fewer automata sizes.*

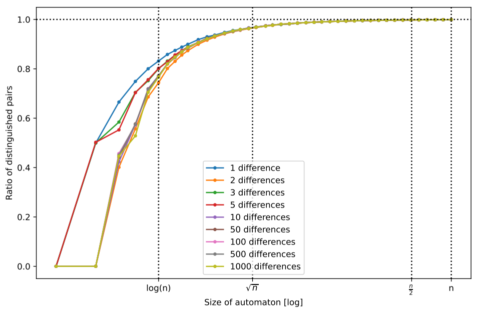
- Input*: n (word length), w (number of words), t (number of tries), s (number of automata sizes), D (list of numbers of differences in the words)*
- Input values*: $n = 1000, w = 100, t = 100, s = 50$ and numbers of differences $D = [1, 2, 3, 5, 10, 50, 100, 500, 1000]$*
- Output*: average success ratio for each number of differences, automata type, and size*
- Path*: random_automata/experiments/success_rate_more_symbols.py*

The results of the experiment are visualized in Figure 5.4, where we present the performance of individual automata types, and Figure 5.5, where we compare automata types with each other for several numbers of differences. Some interesting observations can be derived from the results.

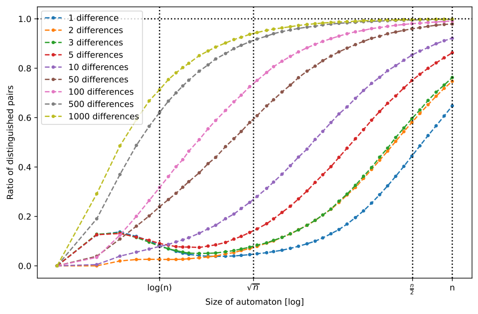
- We see that the performance of the random automata increases with the increasing number of differences in the words. This corresponds to our Conjecture 97.



(a) Shifted permutation automata

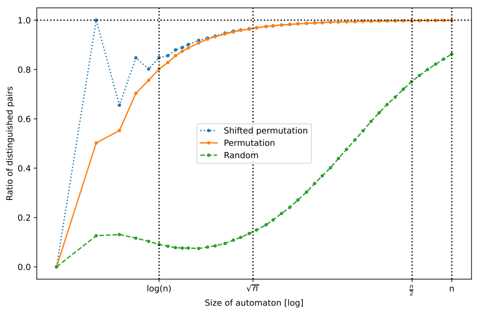


(b) Permutation automata

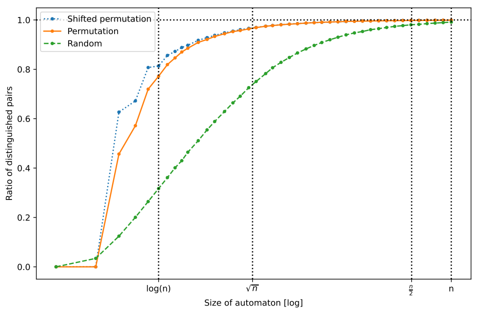


(c) Random automata

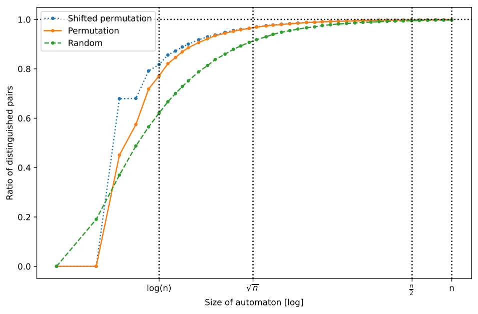
Figure 5.4: Result of Experiment 99: success ratio for different automata types and sizes, multiple differences between the words.



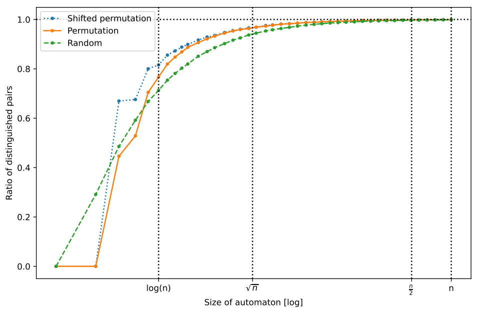
(a) 5 (0.5%) differences



(b) 100 (10%) differences



(c) 500 (50%) differences



(d) 1000 (100%) differences

Figure 5.5: Comparison of different automata types performance for 5, 100, 500, and 1000 differences

- When the words are completely different (i.e., they have $n = 1000$ differences in this case), random automata are almost as good as the permutation ones but still do not outperform them with increasing size.
- However, random automata present slightly better results amongst really small automata with 2 to 5 states. This can be explained by permutation constraints being too strict for the small automata (for example, as we already said, there exists only 2 shifted permutation automata with 2 states, while there are 16 automata with 2 states in total). In particular, no permutation automaton with 2 states can distinguish the words with an even number of differences (by construction), while random ones distinguish at least some of them.
- We see that shifted permutation automaton outperforms other types for any automata size bigger than 2, while the simple permutation one starts to continuously outperform the random one for any size bigger than 6. Both facts correspond to our Conjectures 93 and 96 with some tolerable deviation for smaller sizes, which we explained above.
- We see that both permutation automata types with a small number of states perform better for the odd number of differences. This, again, probably has something to do with their construction. Actually, the case with an odd number of differences is never our main concern, as it can be solved by an automaton with 2 states (see Lemma 28).
- Shifted permutation automata with two states always distinguish the words with the odd number of differences, which we stated in Lemma 95, and any shifted permutation automata always distinguish the words with one difference, which we already mentioned in the analysis of the previous experiment.
- We should also mention the improved performance of the shifted permutation automata with 4 states on the words with an odd number of differences. It turns out that the shifted permutation automata with 4 states share an interesting structure, but we do not describe it in detail as we were unable to describe the behavior of those automata analytically.

The next important observation is that our goal is not only to measure the success rate of different automata but to be able to find the best ones. We can improve our approach by choosing different initial states for each random automata. Up to now, we were starting in the state q_0 all the time, but the fact that the automaton is not successful starting from q_0 does not mean that it will not be successful starting from some other state. Thus we may improve the preceding experiment by taking this into account.

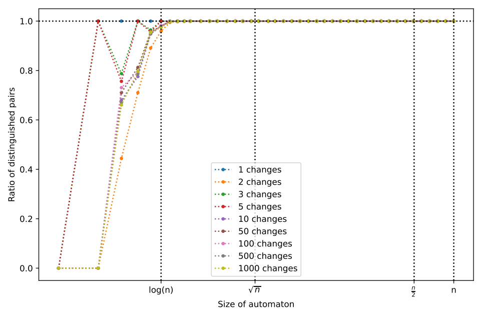
Experiment 100 (Choosing initial state)**.** *We proceed similarly as in Experiment 99, but for each generated automaton we iterate over all its states for each automaton and write down unsuccessful result only if the automaton is unsuccessful for all possible initial states.*

- Input*: n (word length), w (number of words), t (number of tries), s (number of automata sizes), D (list of numbers of differences in the words)*

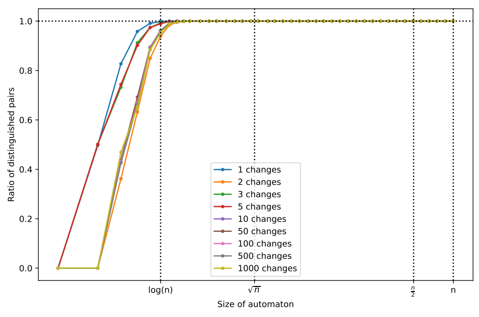
- Input values*: $n = 1000, w = 100, t = 100, s = 50$ and numbers of differences $D = [1, 2, 3, 5, 10, 50, 100, 500, 1000]$*

- Output*: average success ratio for each number of differences, automata type, and size*

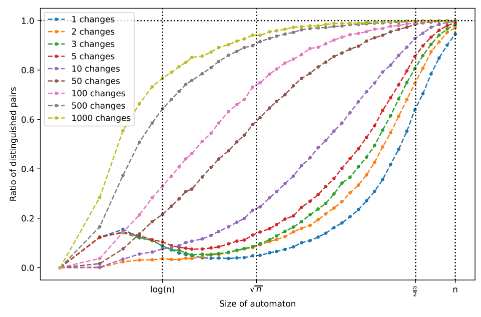
- Path*: random_automata/experiments/success_rate_choose_start.py*



(a) Shifted permutation automata



(b) Permutation automata



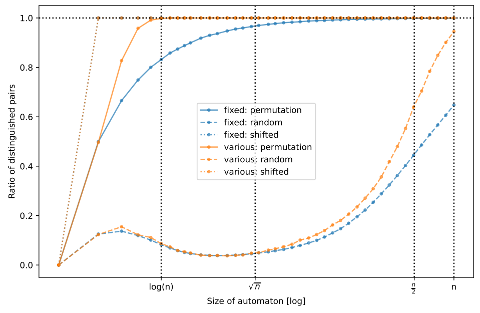
(c) Random automata

Figure 5.6: Result of Experiment 100: success ratio for different automata types and sizes, starting state can be chosen.

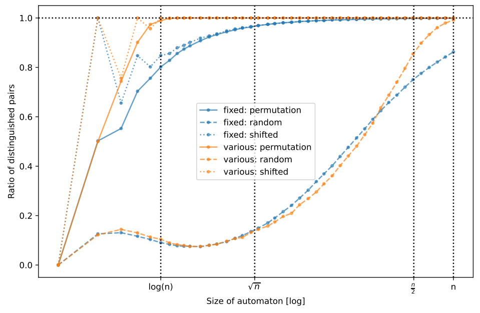
In Figure 5.6, we present the results of the experiment for each automaton type separately. We can see that choosing the initial state helps both permutation automata a lot: for an automaton of logarithmic size, we achieve a 95% success rate for permutation and 98% for shifted permutation automata. In comparison, without choosing the initial state, logarithmic permutation automata have $74 - 83\%$ success rate depending on the number of differences, while shifted permutation ones have $78 - 85\%$ (apart from the case with 1 difference, where they are 100% successful).

We also compare those results to results obtained with fixed initial state q_0 in Figure 5.7. We see that choosing the initial state does not help random automata that much, especially with the growing number of changes. We assume that it could be due to the existence of the state with a lot of input arrows, where the automaton could get with a high probability, even from different states. Then, choosing of initial state does not affect the behavior of automata that much (it will get to a "sink" state anyway).

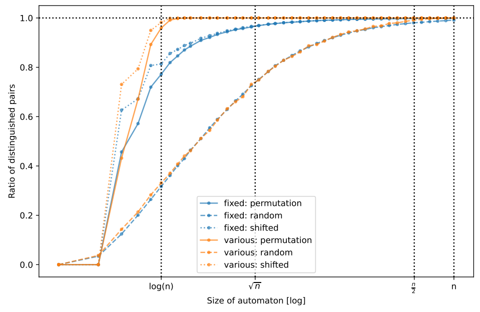
Now we move to the next question. It is nice to be able to compute that amongst 100 random words and 100 randomly generated automata, 98% tries



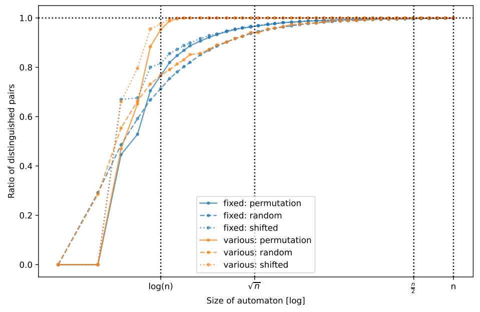
(a) 1 (0.1%) differences



(b) 5 (0.5%) differences



(c) 100 (10%) differences



(d) 1000 (100%) differences

Figure 5.7: Result of Experiment 100: comparison of automata performance for fixed and various initial states, for a number of differences $1, 5, 100$ and 1000.

were successful, but what if we want to achieve complete success? How many states are needed? We conduct the next experiment to find the minimum size of shifted permutation automaton, where it will be indistinguishable from an always successful one on all the words we try.

Experiment 101 (100% rate bound - shifted permutation automata). *In this experiment, we iterate over different word lengths n (50, 100, 150, ..., 4000) and percentages of differences d ($10\%, 20\%, 50\%, 70\%, 90\%, 100\%$) between the words. For each word length and differences' percentage, we start with an automaton of $\lfloor \log n \rfloor$ size. We generate w random word pairs with t randomly generated automata for each pair and evaluate a simplified version of the generated automaton in each of $w \cdot t$ tries. Then, we increase the automata size by 1 and repeat the process until we see that all automata in the previous 5 iterations were 100% successful - then we say that the first one of them was a minimal successful one. In that way, we ensure that the 100% success rate was not a coincidence or result of a specific automata structure (e.g. for a 2-state automaton with an odd differences' number).*

- Input*: N (list of word lengths), w (number of words), t (number of tries), D (list of percentages of differences in the words)*
- Input values*: $N = [50, 100, \dots, 4000], w = 100, t = 1000$, and percentages of differences $D = [10\%, 20\%, 50\%, 70\%, 90\%, 100\%]$*
- Output*: for each word length n we output the smallest automaton size m such that all $5 \cdot t$ generated automata of size m to $m+4$ were 100% successful on all w generated words.*

- Path*: random_automata/experiments/success_rate_linear.py*

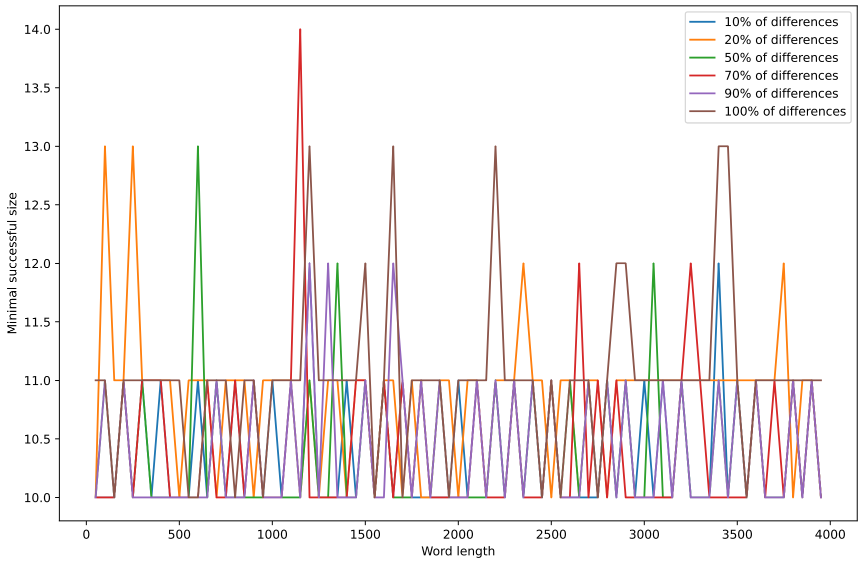


Figure 5.8: Result of Experiment 101: minimal successful size of shifted permutation automaton.

Figure 5.8 presents the result of the experiment. Interestingly, the 100% success rate border seems to be constant. Thus we may claim that, with high probability, an automaton of a constant size is enough.

Conjecture 102. *A randomly generated shifted permutation automaton of constant size will distinguish two distinct randomly generated words with high probability.*

This, however, as we discussed in the previous section, means only that permutation automata perform well on average but says nothing about the worst-case scenario. In our case, when we try only 100 word pairs from all possible $\binom{n}{d}2^n$, where n is from 50 up to 4000 and d is the number of differences, it makes sense that we will probably not encounter "bad" words. Clearly, we are not able to examine any significant portion of such long words. Thus, we modify our experiment to examine a bigger number of shorter words instead.

Experiment 103 (100% rate bound - shifted permutation automata, shorter words)**.** *We repeat Experiment 101 with a bigger number of shorter words to ensure that we test a significant percentage of all possible word pairs.*

- Input*: N (list of word lengths), w (number of words), t (number of tries), D (list of percentages of differences in the words)*
- Input values*: $N = [10, 15, \dots, 45]$, $w = 10\,000$, $t = 100$, and percentages of differences $D = [10\%, 20\%, 50\%, 70\%, 90\%, 100\%]$*
- Output*: for each word length n we output the smallest automaton size m such that all $5 \cdot t$ generated automata of size m to $m+4$ were 100% successful on all w generated words.*

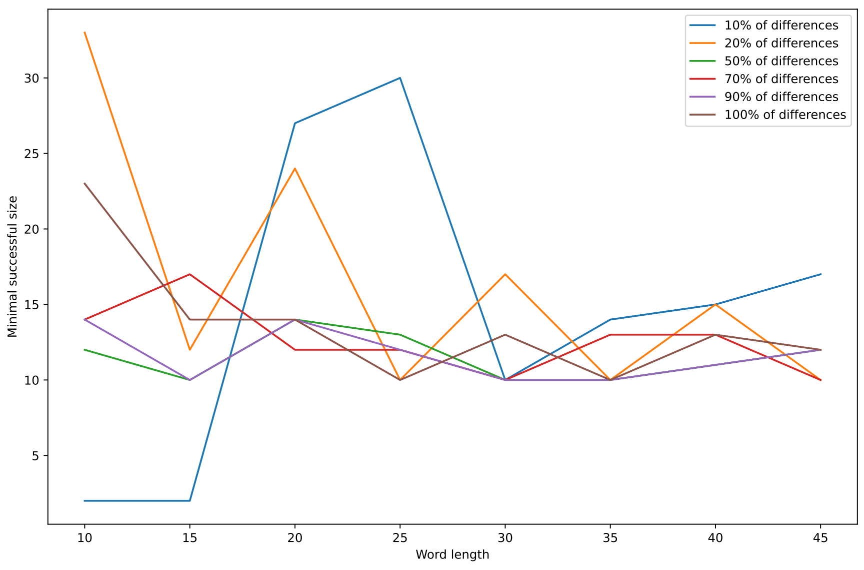


Figure 5.9: Result of Experiment 103: successful size of shifted permutation automaton, shorter words

- Path*: random_automata/experiments/success_rate_linear.py*

In Figure 5.9, we present the results for small n, where we try a significant percentage of all possible words. For example, for $n = 10$ and 20% of differences, we have $\binom{10}{2}2^{10} = 46080$ possible word pairs, and thus we examine $10^4/46080 \approx 22\%$ of words. We see that the needed automaton size is much higher - even bigger than the length of the words. Thus, we may state that even though shifted permutation automata perform well on average, they do not achieve such a good performance on all possible word pairs.

Conjecture 104. *There exist word pairs that a randomly generated shifted permutation automaton of sublinear size will not distinguish.*

Having all this, we try to formulate a more precise statement about the behavior of shifted permutation automata of constant size using statistical methods. We define a slightly different experiment to obtain the necessary values. We choose larger words to really show that the automaton size is negligible in comparison with the word length.

Experiment 105 (Constant size success rate - shifted permutation automata)**.** *For given automaton size m, word length n, and number of changes d, we conduct t Bernoulli trials. In each trial, we generate a random word pair and a shifted permutation automaton according to given parameters. We measure the success of the trial as "the simplified automaton distinguishes the pair".*

- Input*: n (word length), d (number of differences), m (automata size), t (number of trials)*
- Input values*: $n = 10\,000$, $d = 1000$, $m = 10$, $t = 1\,000\,000$*
- Output*: number of successful trials*

- Path*: random_automata/experiments/success_rate_linear_constant_size.py*

We obtain the result of **99.9993**% successful trials. Assuming that the probability of success in each trial is the same, we may calculate a binomial proportion confidence interval. As Brown et al. [2001] suggested, we use the Agresti–Coull interval, as we deal with a success rate extremely close to 1 and a large number of samples. Namely, we compute the following interval for a 99% confidence level.

$$CI_{AC} = \tilde{p} \pm \kappa\sqrt{\frac{\tilde{p}(1-\tilde{p})}{\tilde{n}}}, \text{ where } \tilde{n} = n + \kappa^2 \text{ and } \tilde{p} = \frac{n_s + \kappa^2/2}{\tilde{n}}.$$

Having $n = 1000000$, $n_s = 999993$, $\kappa = 2.576$, we estimate with 99% confidence that

$$p \in [99.9981\%,\ 99.9998\%].$$

This gives us the following claim.

Claim 106. *A randomly generated simplified shifted permutation automaton of size* 10 *will distinguish a randomly generated pair of words of length* 10 000 *with* 10% *differences between them with high probability. More precisely, we may estimate the probability to be* $\geq 99.9981\%$ *with* 99% *confidence.*

Having the randomly generated automata of small constant size m that will distinguish the words with high probability p allows us to decrease an average necessary automaton size not asymptotically but by a huge constant. Precisely, we are able to employ the following discerning method:

- Generate a random shifted permutation automaton of size m.
- If its simplified version discerns the input words, claim that the words are different.
- Otherwise, create an automaton of size $C \in O(\sqrt[3]{n}\log^7 n)$ using existing techniques and use it to distinguish the words.

The average number of needed states will be

$$\mathbb{E}(\#\text{ states}) = pm + (1-p)(m+C) = m + (1-p)C,$$

which allows us to decrease the average number of states for sufficiently big words by a big constant factor $1/(1-p)$.

This approach does not allow us to improve the asymptotic bounds for the main problem, but it is clearly useful in practice. Especially with no need to tailor the automaton to the specific words on input, which will significantly simplify its creation in case of dealing with extremely long words. This method should be, though, supported by theoretical proofs of the obtained probabilities, which are beyond the scope of this work.

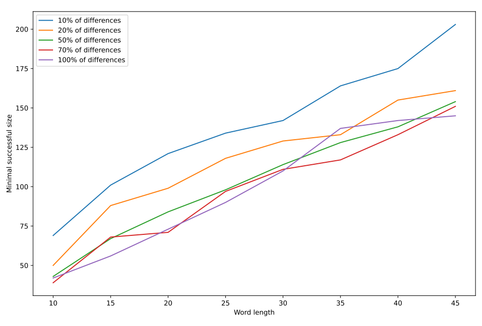
Finally, for completeness, we try the same approach for random automata, both for longer and shorter words.

Experiment 107 (100% rate bound - random automata)**.** *We adapt Experiment 101 for a random automaton, making some changes to improve running time: first, we increase automaton size m by 5 in each step instead of* 1*, as the resulting values tend to be much higher. We also require* 10 *subsequent successful sizes for a size to be marked as successful, as random automata tend to oscillate more.*

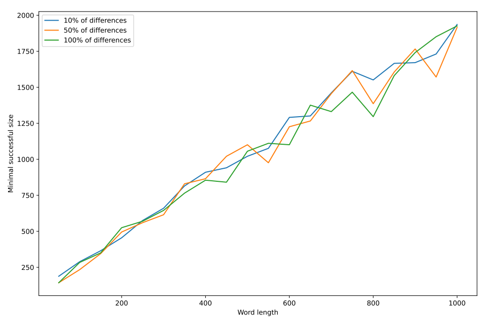
- Input*: N (list of word lengths), w (number of words), t (number of tries), D (list of percentages of differences in the words)*
- Input values*: $N = [50, 100, \ldots, 1000], w = 100, t = 1000,$ and percentages of differences $D = [10\%, 20\%, 50\%, 70\%, 90\%, 100\%]$*
- Output*: for each word length n we output the smallest automaton size m such that all $10 \cdot t$ generated automata of size $m, m + 5, \ldots, m + 45$ were* 100% *successful on all w generated words.*
- Path*: random_automata/experiments/success_rate_linear.py*

Experiment 108 (100% rate bound - random automata, shorter words)**.** *The setting of this experiment remains as in Experiment 107, but we again increase the size of an automaton by* 1 *in each step, as we expect smaller sizes for shorter words.*

- Input*: N (list of word lengths), w (number of words), t (number of tries), D (list of percentages of differences in the words)*
- Input values*: $N = [10, 15, \ldots, 45], w = 10\,000, t = 100,$ and percentages of differences $D = [10\%, 50\%, 100\%]$*
- Output*: for each word length n we output the smallest automaton size m such that all $10 \cdot t$ generated automata of size m to $m+9$ were* 100% *successful on all w generated words.*
- Path*: random_automata/experiments/success_rate_linear.py*



(a) Word length from 10 to 45



(b) Word length from 50 to 1000

Figure 5.10: Results of Experiments 107 and 108: minimal successful size of a random automaton.

Figure 5.10 shows the results for both experiments. The successful random automaton size is approximately twice the word length in both cases. This allows us to say that random automata do not perform well even on average, which makes the results we obtained for permutation ones even more valuable.

Conclusion

In this thesis, we examined the fundamental open problem in the informatics field, presented existing proven bounds, and dug deeper into several approaches not considered before. Although we were not able to push the bound for the main problem, we present several interesting conjectures that deserve further research.

In particular, the construction of discerning sets seems to be a promising topic, and we honestly think it is a puzzle that only misses some small piece to be solved completely. Even though we proved that this approach could not be used to prove the logarithmic upper bound and therefore close the whole problem, the presented results are too systematic to be a coincidence, and we hope they may be found useful in a different area.

We also presented some considerations and experiments about the usage of randomly generated automata in the problem. It turns out that while simple random automata do not perform well, there is an easily definable and generatable subclass of automata whose elements perform well on average and can distinguish randomly generated words with high probability. They do so even with a constant number of states, which is much better than a proven deterministic lower bound. Even though there exist words that such randomly generated automata can not distinguish, and this approach is not sufficiently generic, it can achieve excellent performance on average with the support of already-known deterministic methods.

Further research

This thesis contains several conjectures which we obtained empirically and not fully proved. We believe that it may be beneficial to explore them further.

In relation to discerning sets, we stated some conjectures about the structure of the critical sets at the end of Section 4.4. While the topic turned out to be not useful for the main discerning problem, we believe that there might be some other usage for the existence of a small modulo with an odd residue class in any $S \subseteq [n]$, and thus values of $M(n)$ are worth examining without the connection to discerning automata. As follows from the results of our experiments, critical sets possess some interesting structural patterns, which we believe might be further utilized for estimating the bounds for $M(n)$.

Regarding random automata, there are three main directions for further research. First, ”bad” word pairs are worth examining. Why are some word pairs harder or almost impossible to distinguish by randomly generated automata? How big a part of all pairs do they represent? Is it possible to distinguish them easily using other methods? Besides that, one can also do some state analysis and scrutinize the behavior of shifted permutation automata on different inputs to estimate the guaranteed probability of distinguishing specific word pairs. Finally, further theoretical research may be done regarding the performance of shifted permutation automata of constant size on a randomly chosen input. After some further analysis, we believe this approach can be utilized to solve the main discerning problem.

Bibliography

Eric Allender, Michal Koucky, Detlef Ronneburger, and Sambuddha Roy. Derandomization and distinguishing complexity. In *18th IEEE Annual Conference on Computational Complexity, 2003. Proceedings.*, pages 209–220. IEEE, 2003.

Daria Bilan. Discerning two words. `https://github.com/DariaLeb/discerning-two-words`, 2023.

Lawrence D Brown, T Tony Cai, and Anirban DasGupta. Interval estimation for a binomial proportion. *Statistical science*, 16(2):101–133, 2001.

Jan Černy. Poznamka k homogenym eksperimentom s konechnymi automatami. *Matematicko Fyzicalny Časopis*, 14:208–215, 1964.

Guillaume Chapuy and Guillem Perarnau. Short synchronizing words for random automata. In *Proceedings of the 2023 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 581–604. SIAM, 2023.

Zachary Chase. Separating words and trace reconstruction. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*. ACM, jun 2021. doi: 10.1145/3406325.3451118.

Erik D. Demaine, Sarah Eisenstat, Jeffrey Shallit, and David A. Wilson. Remarks on separating words. In *Descriptional Complexity of Formal Systems*, pages 147–157. Springer Berlin Heidelberg, 2011. doi: 10.1007/978-3-642-22600-7_12.

RA Gimadeev and Mikhail N Vyalyi. Identical relations in symmetric groups and separating words with reversible automata. In *CSR*, pages 144–155. Springer, 2010.

Pavel Goralčík and Václav Koubek. On discerning words by automata. In *International Colloquium on Automata, Languages, and Programming*, pages 116–122. Springer, 1986.

Thomas Eric Hall. Biprefix codes, inverse semigroups and syntactic monoids of injective automata. *Theoretical Computer Science*, 32(1-2):201–213, 1984.

Godfrey Harold Hardy and Edward Maitland Wright. *An introduction to the theory of numbers*. Oxford university press, 1979.

John E. Hopcroft, Rajeev Motwani, and Jeffrey D. Ullman. Introduction to automata theory, languages, and computation, 2nd edition. *ACM SIGACT News*, 32(1):45–52, mar 2001. doi: 10.1145/568438.568455.

J Howard Johnson. Rational equivalence relations. *Theoretical Computer Science*, 47:39–60, 1986.

Victor Namias. A simple derivation of stirling's asymptotic series. *The American Mathematical Monthly*, 93(1):25–29, jan 1986. doi: 10.1080/00029890.1986.11971738.

Cyril Nicaud. Fast synchronization of random automata. *arXiv preprint arXiv:1404.6962*, 2014.

Jean-Eric Pin. On reversible automata. In *Latin American Symposium on Theoretical Informatics*, pages 401–416. Springer, 1992.

Srinivasa Ramanujan. A proof of bertrand's postulate. *Journal of the Indian Mathematical Society*, 11(181-182):27, 1919.

J.M. Robson. Separating strings with small automata. *Information Processing Letters*, 30(4):209–214, feb 1989. doi: 10.1016/0020-0190(89)90215-9.

J Barkley Rosser and Lowell Schoenfeld. Approximate formulas for some functions of prime numbers. *Illinois Journal of Mathematics*, 6(1):64–94, 1962.

Jeffrey Shallit. *Asecond Course in Formal Languages And Automata Theory.* CAMBRIDGE UNIVERSITY, 2009.

Yaroslav Shitov. An improvement to a recent upper bound for synchronizing words of finite automata. *arXiv preprint arXiv:1901.06542*, 2019.

Satyaki Sikdar. Drawing finite state machines in latex using tikz a tutorial. 2017.

Jiří Wiedermann. Discerning two words by a minimum size automaton. 2016.

List of Figures

List of Tables

A. Source code

This appendix describes the source code used to obtain experimental results in Chapters 4 and 5. All code is written in Python 3.9 using NumPy, Matplotlib, and seaborn libraries. The *scripts* directory contains two subdirectories, each dedicated to one chapter of this thesis. As conducted experiments are costly in terms of computational power and time, we store the results in *data* subdirectories for reproducibility of the following analysis without the need to regenerate the input. In *discerning_sets* directory, one can find two scripts generating critical sets: *generate/discerning_sets.py* and *generate/discerning_proper_sets.py* corresponding to conducted Experiments 64 and 80. The *plot* directory contains various plotting functions used to visualize obtained results. In the following listing, we limit ourselves to the files used to produce results for a final version of this work, while there are more for an interested reader to examine. In *random_automata* directory one can find five scripts computing the success rate of different automata, each corresponding to an experiment in Chapter 5. Two *compare* scripts are used to create comparison plots in Figures 5.5 and 5.7. Described files are submitted together with this text but could also be found in [Bilan, 2023]. The source tree with a limited listing of files follows.

```
scripts
 ├─ discerning_sets
 │   ├─ data
 │   ├─ generate
 │   │   ├─ discerning_sets.py
 │   │   └─ discerning_proper_sets.py
 │   └─ plot
 │       ├─ plot_proper_matrix.py
 │       ├─ plot_proper_matrix_interval.py
 │       └─ plots.py
 └─ random_automata
     ├─ data
     ├─ experiments
     │   ├─ compare.py
     │   ├─ compare_initial.py
     │   ├─ success_rate_linear.py
     │   ├─ success_rate_linear_constant_size.py
     │   ├─ success_rate_choose_start.py
     │   ├─ success_rate_more_symbols.py
     │   └─ success_rate_one_symbol.py
     └─ automaton.py
```

B. Critical sets

In this appendix, we present all critical sets for values of n from 6 to 34 in Table B.1 and all proper sets for values of n from 6 to 62 in Table B.2.

Table B.1: Results of Experiment 64: critical sets for n from 6 to 34.

			unique &	
$\boldsymbol{m}$	$\boldsymbol{i}$	**sum**$-\boldsymbol{n}$	**canonical**	**set**
				$n = 6$
4	0	0		{1, 2, 4, 5}
4	0	1	✓	{1, 3, 4, 6}
4	1	2		{2, 3, 5, 6}
				$n = 7$
5	0	1	✓	{1, 2, 3, 5, 6, 7}
				$n = 8$
5	0	1	✓	{1, 4, 5, 8}
5	0	0		{1, 2, 3, 5, 6, 7}
5	1	2		{2, 3, 4, 6, 7, 8}
				$n = 9$
5	1	1	✓	{1, 3, 7, 9}
5	0	0		{1, 4, 5, 8}
5	0	2		{2, 5, 6, 9}
5	0	-1		{1, 2, 3, 5, 6, 7}
5	2	1	✓	{1, 2, 4, 6, 8, 9}
5	1	1		{2, 3, 4, 6, 7, 8}
5	0	3		{3, 4, 5, 7, 8, 9}
				$n = 10$
5	1	-	✓	{1, 2, 5, 10}
5	1	0		{1, 3, 7, 9}
5	0	-1		{1, 4, 5, 8}
5	0	-	✓	{1, 6, 9, 10}
5	0	2		{2, 4, 8, 10}
5	0	1		{2, 5, 6, 9}
5	0	3		{3, 6, 7, 10}
5	0	-2		{1, 2, 3, 5, 6, 7}
5	2	0		{1, 2, 4, 6, 8, 9}
5	1	0		{2, 3, 4, 6, 7, 8}
5	3	2		{2, 3, 5, 7, 9, 10}
5	0	2		{3, 4, 5, 7, 8, 9}
5	1	4		{4, 5, 6, 8, 9, 10}
5	0	1	✓	{1, 2, 3, 4, 7, 8, 9, 10}
5	2	1	✓	{1, 3, 4, 5, 6, 7, 8, 10}
				$n = 11$
6	1	1	✓	{1, 3, 4, 8, 9, 11}
				$n = 12$

Table B.1: (continued)

Table B.1: (continued)

m	i	sum$-n$	unique & canonical	set
6	1	0		{1, 3, 4, 8, 9, 11}
6	0	2		{2, 4, 5, 9, 10, 12}
6	0	1	✓	{1, 2, 3, 5, 8, 10, 11, 12}
				$n = 13$
7	2	1	✓	{1, 2, 6, 8, 12, 13}
				$n = 14$
7	2	0		{1, 2, 6, 8, 12, 13}
7	3	2		{2, 3, 7, 9, 13, 14}
7	2	1	✓	{1, 3, 6, 7, 8, 9, 12, 14}
				$n = 15$
7	2	-1		{1, 2, 6, 8, 12, 13}
7	3	1		{2, 3, 7, 9, 13, 14}
7	0	3		{3, 4, 8, 10, 14, 15}
7	2	0		{1, 3, 6, 7, 8, 9, 12, 14}
7	0	1	✓	{1, 4, 6, 7, 9, 10, 12, 15}
7	0	2		{2, 4, 7, 8, 9, 10, 13, 15}
7	0	1	✓	{1, 2, 3, 4, 6, 10, 12, 13, 14, 15}
				$n = 16$
7	2	-2		{1, 2, 6, 8, 12, 13}
7	3	0		{2, 3, 7, 9, 13, 14}
7	0	2		{3, 4, 8, 10, 14, 15}
7	1	4		{4, 5, 9, 11, 15, 16}
7	2	-1		{1, 3, 6, 7, 8, 9, 12, 14}
7	0	0		{1, 4, 6, 7, 9, 10, 12, 15}
7	0	1	✓	{1, 5, 6, 7, 10, 11, 12, 16}
7	0	1		{2, 4, 7, 8, 9, 10, 13, 15}
7	0	2		{2, 5, 7, 8, 10, 11, 13, 16}
7	0	3		{3, 5, 8, 9, 10, 11, 14, 16}
7	0	0		{1, 2, 3, 4, 6, 10, 12, 13, 14, 15}
7	1	2		{2, 3, 4, 5, 7, 11, 13, 14, 15, 16}
7	0	-	✓	{1, 2, 3, 5, 6, 9, 10, 11, 12, 13, 14, 16}
7	1	1	✓	{1, 2, 4, 5, 6, 8, 9, 11, 12, 13, 15, 16}
7	1	-	✓	{1, 3, 4, 5, 6, 7, 8, 11, 12, 14, 15, 16}
				$n = 17$
7	0	1	✓	{1, 7, 11, 17}
7	2	-3		{1, 2, 6, 8, 12, 13}
7	3	-1		{2, 3, 7, 9, 13, 14}
7	0	1		{3, 4, 8, 10, 14, 15}
7	1	3		{4, 5, 9, 11, 15, 16}
7	2	5		{5, 6, 10, 12, 16, 17}
7	0	-	✓	{1, 2, 3, 9, 11, 13, 14, 17}
7	5	1	✓	{1, 2, 5, 8, 10, 13, 16, 17}
7	2	-2		{1, 3, 6, 7, 8, 9, 12, 14}

Table B.1: (continued)

Table B.1: (continued)

m	***i***	**sum−*n***	**unique & canonical**	**set**
7	0	-	✓	{1, 4, 5, 7, 9, 15, 16, 17}
7	0	-1		{1, 4, 6, 7, 9, 10, 12, 15}
7	0	0		{1, 5, 6, 7, 10, 11, 12, 16}
7	0	0		{2, 4, 7, 8, 9, 10, 13, 15}
7	0	1		{2, 5, 7, 8, 10, 11, 13, 16}
7	0	2		{2, 6, 7, 8, 11, 12, 13, 17}
7	0	2		{3, 5, 8, 9, 10, 11, 14, 16}
7	0	3		{3, 6, 8, 9, 11, 12, 14, 17}
7	1	4		{4, 6, 9, 10, 11, 12, 15, 17}
7	0	1	✓	{1, 2, 3, 4, 5, 13, 14, 15, 16, 17}
7	0	-1		{1, 2, 3, 4, 6, 10, 12, 13, 14, 15}
7	1	-	✓	{1, 2, 4, 8, 9, 10, 11, 13, 15, 17}
7	1	1	✓	{1, 3, 4, 7, 8, 10, 11, 14, 15, 17}
7	3	-	✓	{1, 3, 5, 7, 8, 9, 10, 14, 16, 17}
7	1	1		{2, 3, 4, 5, 7, 11, 13, 14, 15, 16}
7	0	3		{3, 4, 5, 6, 8, 12, 14, 15, 16, 17}
7	0	-		{1, 2, 3, 5, 6, 9, 10, 11, 12, 13, 14, 16}
7	1	0		{1, 2, 4, 5, 6, 8, 9, 11, 12, 13, 15, 16}
7	1	-		{1, 3, 4, 5, 6, 7, 8, 11, 12, 14, 15, 16}
7	1	-		{2, 3, 4, 6, 7, 10, 11, 12, 13, 14, 15, 17}
7	2	2		{2, 3, 5, 6, 7, 9, 10, 12, 13, 14, 16, 17}
7	0	-		{2, 4, 5, 6, 7, 8, 9, 12, 13, 15, 16, 17}
				$n = 18$
7	1	1	✓	{1, 6, 13, 18}
7	0	0		{1, 7, 11, 17}
7	1	2		{2, 8, 12, 18}
7	2	-4		{1, 2, 6, 8, 12, 13}
7	3	-2		{2, 3, 7, 9, 13, 14}
7	0	0		{3, 4, 8, 10, 14, 15}
7	1	2		{4, 5, 9, 11, 15, 16}
7	2	4		{5, 6, 10, 12, 16, 17}
7	0	6		{6, 7, 11, 13, 17, 18}
7	1	-	✓	{1, 2, 3, 6, 7, 9, 14, 18}
7	0	-		{1, 2, 3, 9, 11, 13, 14, 17}
7	5	0		{1, 2, 5, 8, 10, 13, 16, 17}
7	0	1	✓	{1, 2, 7, 8, 11, 12, 17, 18}
7	2	-3		{1, 3, 6, 7, 8, 9, 12, 14}
7	0	-		{1, 4, 5, 7, 9, 15, 16, 17}
7	0	-2		{1, 4, 6, 7, 9, 10, 12, 15}
7	0	-1		{1, 5, 6, 7, 10, 11, 12, 16}
7	1	-	✓	{1, 5, 10, 12, 13, 16, 17, 18}
7	0	-		{2, 3, 4, 10, 12, 14, 15, 18}
7	0	2		{2, 3, 6, 9, 11, 14, 17, 18}
7	0	-1		{2, 4, 7, 8, 9, 10, 13, 15}

Table B.1: (continued)

Table B.1: (continued)

m	i	sum$-n$	unique & canonical	set
7	1	-		{2, 5, 6, 8, 10, 16, 17, 18}
7	0	0		{2, 5, 7, 8, 10, 11, 13, 16}
7	0	1		{2, 6, 7, 8, 11, 12, 13, 17}
7	0	1		{3, 5, 8, 9, 10, 11, 14, 16}
7	0	2		{3, 6, 8, 9, 11, 12, 14, 17}
7	1	3		{3, 7, 8, 9, 12, 13, 14, 18}
7	1	3		{4, 6, 9, 10, 11, 12, 15, 17}
7	0	4		{4, 7, 9, 10, 12, 13, 15, 18}
7	0	5		{5, 7, 10, 11, 12, 13, 16, 18}
7	0	0		{1, 2, 3, 4, 5, 13, 14, 15, 16, 17}
7	0	-2		{1, 2, 3, 4, 6, 10, 12, 13, 14, 15}
7	0	-	✓	{1, 2, 4, 6, 7, 8, 9, 10, 15, 18}
7	1	-		{1, 2, 4, 8, 9, 10, 11, 13, 15, 17}
7	0	-	✓	{1, 2, 5, 6, 7, 8, 10, 11, 16, 18}
7	0	-	✓	{1, 3, 4, 6, 8, 10, 13, 14, 15, 18}
7	1	0		{1, 3, 4, 7, 8, 10, 11, 14, 15, 17}
7	3	-		{1, 3, 5, 7, 8, 9, 10, 14, 16, 17}
7	0	-	✓	{1, 3, 8, 9, 11, 12, 13, 14, 17, 18}
7	4	-	✓	{1, 4, 5, 6, 9, 11, 13, 15, 16, 18}
7	2	-	✓	{1, 4, 9, 10, 11, 12, 13, 15, 17, 18}
7	0	2		{2, 3, 4, 5, 6, 14, 15, 16, 17, 18}
7	1	0		{2, 3, 4, 5, 7, 11, 13, 14, 15, 16}
7	0	-		{2, 3, 5, 9, 10, 11, 12, 14, 16, 18}
7	2	2		{2, 4, 5, 8, 9, 11, 12, 15, 16, 18}
7	4	-		{2, 4, 6, 8, 9, 10, 11, 15, 17, 18}
7	0	2		{3, 4, 5, 6, 8, 12, 14, 15, 16, 17}
7	0	4		{4, 5, 6, 7, 9, 13, 15, 16, 17, 18}
7	3	-	✓	{1, 2, 3, 4, 5, 6, 7, 11, 14, 15, 16, 18}
7	2	-	✓	{1, 2, 3, 4, 7, 10, 11, 12, 14, 15, 17, 18}
7	0	-		{1, 2, 3, 5, 6, 9, 10, 11, 12, 13, 14, 16}
7	1	1	✓	{1, 2, 3, 5, 7, 9, 10, 12, 14, 16, 17, 18}
7	1	-1		{1, 2, 4, 5, 6, 8, 9, 11, 12, 13, 15, 16}
7	0	-	✓	{1, 2, 4, 5, 7, 8, 9, 12, 15, 16, 17, 18}
7	1	-		{1, 3, 4, 5, 6, 7, 8, 11, 12, 14, 15, 16}
7	0	-	✓	{1, 3, 4, 5, 8, 12, 13, 14, 15, 16, 17, 18}
7	0	1	✓	{1, 3, 5, 6, 8, 9, 10, 11, 13, 14, 16, 18}
7	1	-		{2, 3, 4, 6, 7, 10, 11, 12, 13, 14, 15, 17}
7	2	1		{2, 3, 5, 6, 7, 9, 10, 12, 13, 14, 16, 17}
7	0	-		{2, 4, 5, 6, 7, 8, 9, 12, 13, 15, 16, 17}
7	2	-		{3, 4, 5, 7, 8, 11, 12, 13, 14, 15, 16, 18}
7	3	3		{3, 4, 6, 7, 8, 10, 11, 13, 14, 15, 17, 18}
7	1	-		{3, 5, 6, 7, 8, 9, 10, 13, 14, 16, 17, 18}
				$n = 19$

Table B.1: (continued)

Table B.1: (continued)

m	i	sum$-n$	unique & canonical	set
8	0	1	✓	{1, 6, 7, 8, 12, 13, 14, 19}
				$n = 20$
8	2	1	✓	{1, 2, 6, 9, 12, 15, 19, 20}
8	0	0		{1, 6, 7, 8, 12, 13, 14, 19}
8	0	2		{2, 7, 8, 9, 13, 14, 15, 20}
				$n = 21$
8	2	0		{1, 2, 6, 9, 12, 15, 19, 20}
8	0	-1		{1, 6, 7, 8, 12, 13, 14, 19}
8	0	2		{2, 3, 7, 10, 13, 16, 20, 21}
8	0	1		{2, 7, 8, 9, 13, 14, 15, 20}
8	1	3		{3, 8, 9, 10, 14, 15, 16, 21}
8	1	1	✓	{1, 2, 3, 6, 8, 10, 12, 14, 16, 19, 20, 21}
8	0	1	✓	{1, 3, 6, 7, 9, 10, 12, 13, 15, 16, 19, 21}
				$n = 22$
8	2	-1		{1, 2, 6, 9, 12, 15, 19, 20}
8	0	-2		{1, 6, 7, 8, 12, 13, 14, 19}
8	0	1		{2, 3, 7, 10, 13, 16, 20, 21}
8	0	0		{2, 7, 8, 9, 13, 14, 15, 20}
8	0	3		{3, 4, 8, 11, 14, 17, 21, 22}
8	1	2		{3, 8, 9, 10, 14, 15, 16, 21}
8	0	4		{4, 9, 10, 11, 15, 16, 17, 22}
8	1	0		{1, 2, 3, 6, 8, 10, 12, 14, 16, 19, 20, 21}
8	0	-	✓	{1, 2, 4, 6, 10, 11, 12, 16, 17, 19, 20, 22}
8	3	-	✓	{1, 3, 4, 6, 7, 11, 12, 13, 17, 19, 21, 22}
8	0	0		{1, 3, 6, 7, 9, 10, 12, 13, 15, 16, 19, 21}
8	2	2		{2, 3, 4, 7, 9, 11, 13, 15, 17, 20, 21, 22}
8	1	2		{2, 4, 7, 8, 10, 11, 13, 14, 16, 17, 20, 22}
8	0	1	✓	{1, 2, 3, 4, 6, 8, 9, 11, 12, 14, 15, 17, 19, 20, 21, 22}
8	1	1	✓	{1, 4, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 19, 22}
				$n = 23$
9	0	1	✓	{1, 5, 6, 7, 8, 10, 11, 13, 14, 16, 17, 18, 19, 23}
				$n = 24$
9	0	1	✓	{1, 2, 5, 9, 10, 12, 13, 15, 16, 20, 23, 24}
9	0	0		{1, 5, 6, 7, 8, 10, 11, 13, 14, 16, 17, 18, 19, 23}
9	1	2		{2, 6, 7, 8, 9, 11, 12, 14, 15, 17, 18, 19, 20, 24}
				$n = 25$
9	0	1	✓	{1, 2, 3, 5, 7, 8, 18, 19, 21, 23, 24, 25}
9	0	0		{1, 2, 5, 9, 10, 12, 13, 15, 16, 20, 23, 24}

Table B.1: (continued)

Table B.1: (continued)

m	i	sum$-n$	unique & canonical	set
9	1	2		{2, 3, 6, 10, 11, 13, 14, 16, 17, 21, 24, 25}
9	0	1	✓	{1, 3, 5, 6, 9, 11, 12, 14, 15, 17, 20, 21, 23, 25}
9	0	-1		{1, 5, 6, 7, 8, 10, 11, 13, 14, 16, 17, 18, 19, 23}
9	1	1		{2, 6, 7, 8, 9, 11, 12, 14, 15, 17, 18, 19, 20, 24}
9	2	3		{3, 7, 8, 9, 10, 12, 13, 15, 16, 18, 19, 20, 21, 25}
				$n = 26$
9	0	0		{1, 2, 3, 5, 7, 8, 18, 19, 21, 23, 24, 25}
9	0	-1		{1, 2, 5, 9, 10, 12, 13, 15, 16, 20, 23, 24}
9	1	1	✓	{1, 4, 5, 6, 7, 9, 18, 20, 21, 22, 23, 26}
9	0	2		{2, 3, 4, 6, 8, 9, 19, 20, 22, 24, 25, 26}
9	1	1		{2, 3, 6, 10, 11, 13, 14, 16, 17, 21, 24, 25}
9	0	3		{3, 4, 7, 11, 12, 14, 15, 17, 18, 22, 25, 26}
9	0	0		{1, 3, 5, 6, 9, 11, 12, 14, 15, 17, 20, 21, 23, 25}
9	0	-2		{1, 5, 6, 7, 8, 10, 11, 13, 14, 16, 17, 18, 19, 23}
9	0	2		{2, 4, 6, 7, 10, 12, 13, 15, 16, 18, 21, 22, 24, 26}
9	1	0		{2, 6, 7, 8, 9, 11, 12, 14, 15, 17, 18, 19, 20, 24}
9	2	2		{3, 7, 8, 9, 10, 12, 13, 15, 16, 18, 19, 20, 21, 25}
9	0	4		{4, 8, 9, 10, 11, 13, 14, 16, 17, 19, 20, 21, 22, 26}
9	5	-	✓	{1, 2, 4, 5, 8, 11, 12, 14, 15, 17, 19, 21, 22, 23, 24, 26}
9	1	-	✓	{1, 3, 4, 5, 6, 8, 10, 12, 13, 15, 16, 19, 22, 23, 25, 26}
9	2	1	✓	{1, 2, 3, 4, 5, 7, 9, 10, 11, 13, 14, 16, 17, 18, 20, 22, 23, 24, 25, 26}
				$n = 27$
9	3	1	✓	{1, 3, 6, 10, 18, 22, 25, 27}
9	0	-1		{1, 2, 3, 5, 7, 8, 18, 19, 21, 23, 24, 25}
9	0	1	✓	{1, 2, 4, 8, 9, 10, 18, 19, 20, 24, 26, 27}
9	0	-2		{1, 2, 5, 9, 10, 12, 13, 15, 16, 20, 23, 24}
9	1	-	✓	{1, 2, 11, 13, 14, 16, 17, 18, 21, 22, 24, 27}
9	1	0		{1, 4, 5, 6, 7, 9, 18, 20, 21, 22, 23, 26}
9	0	-	✓	{1, 4, 6, 7, 10, 11, 12, 14, 15, 17, 26, 27}
9	0	1		{2, 3, 4, 6, 8, 9, 19, 20, 22, 24, 25, 26}
9	1	0		{2, 3, 6, 10, 11, 13, 14, 16, 17, 21, 24, 25}

Table B.1: (continued)

Table B.1: (continued)

m	i	sum$-n$	unique & canonical	set
9	0	2		{2, 5, 6, 7, 8, 10, 19, 21, 22, 23, 24, 27}
9	1	3		{3, 4, 5, 7, 9, 10, 20, 21, 23, 25, 26, 27}
9	0	2		{3, 4, 7, 11, 12, 14, 15, 17, 18, 22, 25, 26}
9	1	4		{4, 5, 8, 12, 13, 15, 16, 18, 19, 23, 26, 27}
9	0	1	✓	{1, 2, 3, 4, 7, 12, 13, 15, 16, 21, 24, 25, 26, 27}
9	0	-1		{1, 3, 5, 6, 9, 11, 12, 14, 15, 17, 20, 21, 23, 25}
9	0	-3		{1, 5, 6, 7, 8, 10, 11, 13, 14, 16, 17, 18, 19, 23}
9	2	1	✓	{1, 6, 7, 8, 9, 12, 13, 15, 16, 19, 20, 21, 22, 27}
9	0	1		{2, 4, 6, 7, 10, 12, 13, 15, 16, 18, 21, 22, 24, 26}
9	1	-1		{2, 6, 7, 8, 9, 11, 12, 14, 15, 17, 18, 19, 20, 24}
9	0	3		{3, 5, 7, 8, 11, 13, 14, 16, 17, 19, 22, 23, 25, 27}
9	2	1		{3, 7, 8, 9, 10, 12, 13, 15, 16, 18, 19, 20, 21, 25}
9	0	3		{4, 8, 9, 10, 11, 13, 14, 16, 17, 19, 20, 21, 22, 26}
9	0	5		{5, 9, 10, 11, 12, 14, 15, 17, 18, 20, 21, 22, 23, 27}
9	5	-		{1, 2, 4, 5, 8, 11, 12, 14, 15, 17, 19, 21, 22, 23, 24, 26}
9	1	-		{1, 3, 4, 5, 6, 8, 10, 12, 13, 15, 16, 19, 22, 23, 25, 26}
9	0	-		{2, 3, 5, 6, 9, 12, 13, 15, 16, 18, 20, 22, 23, 24, 25, 27}
9	2	-		{2, 4, 5, 6, 7, 9, 11, 13, 14, 16, 17, 20, 23, 24, 26, 27}
9	1	-	✓	{1, 2, 3, 7, 8, 9, 10, 11, 12, 14, 15, 17, 19, 20, 22, 24, 25, 27}
9	0	-	✓	{1, 3, 4, 6, 8, 9, 11, 13, 14, 16, 17, 18, 19, 20, 21, 25, 26, 27}
9	2	0		{1, 2, 3, 4, 5, 7, 9, 10, 11, 13, 14, 16, 17, 18, 20, 22, 23, 24, 25, 26}
9	3	2		{2, 3, 4, 5, 6, 8, 10, 11, 12, 14, 15, 17, 18, 19, 21, 23, 24, 25, 26, 27}
				$n = 28$
9	3	0		{1, 3, 6, 10, 18, 22, 25, 27}
9	4	2		{2, 4, 7, 11, 19, 23, 26, 28}
9	0	-2		{1, 2, 3, 5, 7, 8, 18, 19, 21, 23, 24, 25}

Table B.1: (continued)

Table B.1: (continued)

m	i	sum$-n$	unique & canonical	set
9	0	0		{1, 2, 4, 8, 9, 10, 18, 19, 20, 24, 26, 27}
9	1	-	✓	{1, 2, 5, 6, 9, 11, 18, 19, 20, 21, 22, 28}
9	0	-3		{1, 2, 5, 9, 10, 12, 13, 15, 16, 20, 23, 24}
9	0	1	✓	{1, 2, 6, 10, 12, 14, 15, 17, 19, 23, 27, 28}
9	1	-		{1, 2, 11, 13, 14, 16, 17, 18, 21, 22, 24, 27}
9	0	1	✓	{1, 3, 4, 5, 8, 11, 18, 21, 24, 25, 26, 28}
9	1	-1		{1, 4, 5, 6, 7, 9, 18, 20, 21, 22, 23, 26}
9	0	-		{1, 4, 6, 7, 10, 11, 12, 14, 15, 17, 26, 27}
9	4	1	✓	{1, 5, 7, 8, 12, 14, 15, 17, 21, 22, 24, 28}
9	0	-	✓	{1, 7, 8, 9, 10, 11, 18, 20, 23, 24, 27, 28}
9	0	0		{2, 3, 4, 6, 8, 9, 19, 20, 22, 24, 25, 26}
9	1	2		{2, 3, 5, 9, 10, 11, 19, 20, 21, 25, 27, 28}
9	1	-1		{2, 3, 6, 10, 11, 13, 14, 16, 17, 21, 24, 25}
9	0	-		{2, 3, 12, 14, 15, 17, 18, 19, 22, 23, 25, 28}
9	0	1		{2, 5, 6, 7, 8, 10, 19, 21, 22, 23, 24, 27}
9	1	-		{2, 5, 7, 8, 11, 12, 13, 15, 16, 18, 27, 28}
9	1	2		{3, 4, 5, 7, 9, 10, 20, 21, 23, 25, 26, 27}
9	0	1		{3, 4, 7, 11, 12, 14, 15, 17, 18, 22, 25, 26}
9	0	3		{3, 6, 7, 8, 9, 11, 20, 22, 23, 24, 25, 28}
9	0	4		{4, 5, 6, 8, 10, 11, 21, 22, 24, 26, 27, 28}
9	1	3		{4, 5, 8, 12, 13, 15, 16, 18, 19, 23, 26, 27}
9	2	5		{5, 6, 9, 13, 14, 16, 17, 19, 20, 24, 27, 28}
9	0	0		{1, 2, 3, 4, 7, 12, 13, 15, 16, 21, 24, 25, 26, 27}
9	0	-	✓	{1, 2, 4, 5, 6, 8, 10, 13, 14, 16, 17, 18, 26, 28}
9	0	-2		{1, 3, 5, 6, 9, 11, 12, 14, 15, 17, 20, 21, 23, 25}
9	0	-	✓	{1, 3, 11, 12, 13, 15, 16, 19, 21, 23, 24, 25, 27, 28}
9	0	-4		{1, 5, 6, 7, 8, 10, 11, 13, 14, 16, 17, 18, 19, 23}
9	2	0		{1, 6, 7, 8, 9, 12, 13, 15, 16, 19, 20, 21, 22, 27}
9	0	2		{2, 3, 4, 5, 8, 13, 14, 16, 17, 22, 25, 26, 27, 28}
9	0	0		{2, 4, 6, 7, 10, 12, 13, 15, 16, 18, 21, 22, 24, 26}
9	1	-2		{2, 6, 7, 8, 9, 11, 12, 14, 15, 17, 18, 19, 20, 24}
9	0	2		{2, 7, 8, 9, 10, 13, 14, 16, 17, 20, 21, 22, 23, 28}
9	0	2		{3, 5, 7, 8, 11, 13, 14, 16, 17, 19, 22, 23, 25, 27}

Table B.1: (continued)

Table B.1: (continued)

m	i	sum$-n$	unique & canonical	set
9	2	0		{3, 7, 8, 9, 10, 12, 13, 15, 16, 18, 19, 20, 21, 25}
9	1	4		{4, 6, 8, 9, 12, 14, 15, 17, 18, 20, 23, 24, 26, 28}
9	0	2		{4, 8, 9, 10, 11, 13, 14, 16, 17, 19, 20, 21, 22, 26}
9	0	4		{5, 9, 10, 11, 12, 14, 15, 17, 18, 20, 21, 22, 23, 27}
9	0	6		{6, 10, 11, 12, 13, 15, 16, 18, 19, 21, 22, 23, 24, 28}
9	3	1	✓	{1, 2, 3, 4, 6, 7, 10, 11, 18, 19, 22, 23, 25, 26, 27, 28}
9	1	-	✓	{1, 2, 3, 5, 6, 7, 8, 10, 11, 12, 13, 15, 16, 22, 25, 28}
9	5	-		{1, 2, 4, 5, 8, 11, 12, 14, 15, 17, 19, 21, 22, 23, 24, 26}
9	1	-		{1, 3, 4, 5, 6, 8, 10, 12, 13, 15, 16, 19, 22, 23, 25, 26}
9	2	-	✓	{1, 3, 5, 9, 10, 13, 14, 16, 17, 18, 19, 20, 22, 24, 25, 28}
9	0	-	✓	{1, 4, 5, 7, 9, 10, 11, 12, 13, 15, 16, 19, 20, 24, 26, 28}
9	1	-	✓	{1, 4, 7, 13, 14, 16, 17, 18, 19, 21, 22, 23, 24, 26, 27, 28}
9	0	-		{2, 3, 5, 6, 9, 12, 13, 15, 16, 18, 20, 22, 23, 24, 25, 27}
9	2	-		{2, 4, 5, 6, 7, 9, 11, 13, 14, 16, 17, 20, 23, 24, 26, 27}
9	1	-		{3, 4, 6, 7, 10, 13, 14, 16, 17, 19, 21, 23, 24, 25, 26, 28}
9	3	-		{3, 5, 6, 7, 8, 10, 12, 14, 15, 17, 18, 21, 24, 25, 27, 28}
9	0	-	✓	{1, 2, 3, 4, 5, 6, 7, 9, 12, 14, 15, 17, 19, 20, 21, 25, 26, 28}
9	0	-	✓	{1, 2, 3, 6, 7, 8, 9, 13, 14, 16, 17, 18, 20, 21, 23, 25, 27, 28}
9	1	-		{1, 2, 3, 7, 8, 9, 10, 11, 12, 14, 15, 17, 19, 20, 22, 24, 25, 27}
9	2	-	✓	{1, 2, 4, 6, 8, 9, 11, 12, 13, 15, 16, 20, 21, 22, 23, 26, 27, 28}
9	0	-		{1, 3, 4, 6, 8, 9, 11, 13, 14, 16, 17, 18, 19, 20, 21, 25, 26, 27}
9	1	-	✓	{1, 3, 4, 8, 9, 10, 12, 14, 15, 17, 20, 22, 23, 24, 25, 26, 27, 28}

Table B.1: (continued)

Table B.1: (continued)

m	i	sum$-n$	unique & canonical	set
9	2	-		{2, 3, 4, 8, 9, 10, 11, 12, 13, 15, 16, 18, 20, 21, 23, 25, 26, 28}
9	0	-		{2, 4, 5, 7, 9, 10, 12, 14, 15, 17, 18, 19, 20, 21, 22, 26, 27, 28}
9	2	-1		{1, 2, 3, 4, 5, 7, 9, 10, 11, 13, 14, 16, 17, 18, 20, 22, 23, 24, 25, 26}
9	3	1		{2, 3, 4, 5, 6, 8, 10, 11, 12, 14, 15, 17, 18, 19, 21, 23, 24, 25, 26, 27}
9	0	3		{3, 4, 5, 6, 7, 9, 11, 12, 13, 15, 16, 18, 19, 20, 22, 24, 25, 26, 27, 28}
				$n = 29$
10	1	1	✓	{1, 4, 5, 6, 9, 11, 12, 13, 14, 16, 17, 18, 19, 21, 24, 25, 26, 29}
				$n = 30$
10	0	1	✓	{1, 2, 4, 7, 9, 10, 11, 15, 16, 20, 21, 22, 24, 27, 29, 30}
10	1	0		{1, 4, 5, 6, 9, 11, 12, 13, 14, 16, 17, 18, 19, 21, 24, 25, 26, 29}
10	0	2		{2, 5, 6, 7, 10, 12, 13, 14, 15, 17, 18, 19, 20, 22, 25, 26, 27, 30}
				$n = 31$
10	0	0		{1, 2, 4, 7, 9, 10, 11, 15, 16, 20, 21, 22, 24, 27, 29, 30}
10	1	2		{2, 3, 5, 8, 10, 11, 12, 16, 17, 21, 22, 23, 25, 28, 30, 31}
10	0	1	✓	{1, 3, 4, 5, 7, 8, 9, 12, 15, 17, 20, 23, 24, 25, 27, 28, 29, 31}
10	1	-1		{1, 4, 5, 6, 9, 11, 12, 13, 14, 16, 17, 18, 19, 21, 24, 25, 26, 29}
10	0	1		{2, 5, 6, 7, 10, 12, 13, 14, 15, 17, 18, 19, 20, 22, 25, 26, 27, 30}
10	0	3		{3, 6, 7, 8, 11, 13, 14, 15, 16, 18, 19, 20, 21, 23, 26, 27, 28, 31}
10	3	1	✓	{1, 2, 3, 4, 6, 8, 9, 10, 13, 14, 18, 19, 22, 23, 24, 26, 28, 29, 30, 31}
				$n = 32$
10	2	-	✓	{1, 2, 8, 10, 11, 12, 14, 17, 19, 28, 30, 32}
10	1	-	✓	{1, 3, 5, 14, 16, 19, 21, 22, 23, 25, 31, 32}
10	0	-1		{1, 2, 4, 7, 9, 10, 11, 15, 16, 20, 21, 22, 24, 27, 29, 30}
10	0	1	✓	{1, 5, 6, 7, 8, 11, 13, 15, 18, 20, 22, 25, 26, 27, 28, 32}
10	1	1		{2, 3, 5, 8, 10, 11, 12, 16, 17, 21, 22, 23, 25, 28, 30, 31}

Table B.1: (continued)

Table B.1: (continued)

m	i	sum$-n$	unique & canonical	set
10	2	3		{3, 4, 6, 9, 11, 12, 13, 17, 18, 22, 23, 24, 26, 29, 31, 32}
10	0	0		{1, 3, 4, 5, 7, 8, 9, 12, 15, 17, 20, 23, 24, 25, 27, 28, 29, 31}
10	1	-2		{1, 4, 5, 6, 9, 11, 12, 13, 14, 16, 17, 18, 19, 21, 24, 25, 26, 29}
10	1	2		{2, 4, 5, 6, 8, 9, 10, 13, 16, 18, 21, 24, 25, 26, 28, 29, 30, 32}
10	0	0		{2, 5, 6, 7, 10, 12, 13, 14, 15, 17, 18, 19, 20, 22, 25, 26, 27, 30}
10	0	2		{3, 6, 7, 8, 11, 13, 14, 15, 16, 18, 19, 20, 21, 23, 26, 27, 28, 31}
10	0	4		{4, 7, 8, 9, 12, 14, 15, 16, 17, 19, 20, 21, 22, 24, 27, 28, 29, 32}
10	3	0		{1, 2, 3, 4, 6, 8, 9, 10, 13, 14, 18, 19, 22, 23, 24, 26, 28, 29, 30, 31}
10	0	1	✓	{1, 2, 3, 6, 7, 10, 12, 13, 15, 16, 17, 18, 20, 21, 23, 26, 27, 30, 31, 32}
10	0	2		{2, 3, 4, 5, 7, 9, 10, 11, 14, 15, 19, 20, 23, 24, 25, 27, 29, 30, 31, 32}
		$n = 33$		
11	2	1	✓	{1, 2, 3, 5, 6, 7, 8, 9, 12, 22, 25, 26, 27, 28, 29, 31, 32, 33}
		$n = 34$		
11	0	1	✓	{1, 4, 5, 10, 12, 13, 22, 23, 25, 30, 31, 34}
11	2	0		{1, 2, 3, 5, 6, 7, 8, 9, 12, 22, 25, 26, 27, 28, 29, 31, 32, 33}
11	0	2		{2, 3, 4, 6, 7, 8, 9, 10, 13, 23, 26, 27, 28, 29, 30, 32, 33, 34}

Table B.1: (concluded)

Table B.2: Results of Experiment 80: proper sets for n from 6 to 62

m	i	set
		$n = 6$
4	0	{1, 3, 4, 6}
		$n = 7$
5	0	{1, 2, 3, 5, 6, 7}
		$n = 8$
5	0	{1, 4, 5, 8}

Table B.2: (continued)

Table B.2: (continued)

m	i	set
		$n = 9$
5	1	{1, 3, 7, 9}
5	2	{1, 2, 4, 6, 8, 9}
		$n = 10$
5	0	{1, 2, 3, 4, 7, 8, 9, 10}
5	2	{1, 3, 4, 5, 6, 7, 8, 10}
		$n = 11$
6	1	{1, 3, 4, 8, 9, 11}
		$n = 12$
6	0	{1, 2, 3, 5, 8, 10, 11, 12}
		$n = 13$
7	2	{1, 2, 6, 8, 12, 13}
		$n = 14$
7	2	{1, 3, 6, 7, 8, 9, 12, 14}
		$n = 15$
7	0	{1, 4, 6, 7, 9, 10, 12, 15}
7	0	{1, 2, 3, 4, 6, 10, 12, 13, 14, 15}
		$n = 16$
7	0	{1, 5, 6, 7, 10, 11, 12, 16}
7	1	{1, 2, 4, 5, 6, 8, 9, 11, 12, 13, 15, 16}
		$n = 17$
7	0	{1, 7, 11, 17}
7	5	{1, 2, 5, 8, 10, 13, 16, 17}
7	0	{1, 2, 3, 4, 5, 13, 14, 15, 16, 17}
7	1	{1, 3, 4, 7, 8, 10, 11, 14, 15, 17}
		$n = 18$
7	1	{1, 6, 13, 18}
7	0	{1, 2, 7, 8, 11, 12, 17, 18}
7	1	{1, 2, 3, 5, 7, 9, 10, 12, 14, 16, 17, 18}
7	0	{1, 3, 5, 6, 8, 9, 10, 11, 13, 14, 16, 18}
		$n = 19$
8	0	{1, 6, 7, 8, 12, 13, 14, 19}
		$n = 20$
8	2	{1, 2, 6, 9, 12, 15, 19, 20}
		$n = 21$
8	1	{1, 2, 3, 6, 8, 10, 12, 14, 16, 19, 20, 21}
8	0	{1, 3, 6, 7, 9, 10, 12, 13, 15, 16, 19, 21}
		$n = 22$
8	0	{1, 2, 3, 4, 6, 8, 9, 11, 12, 14, 15, 17, 19, 20, 21, 22}
8	1	{1, 4, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 19, 22}
		$n = 23$
9	0	{1, 5, 6, 7, 8, 10, 11, 13, 14, 16, 17, 18, 19, 23}
		$n = 24$

Table B.2: (continued)

Table B.2: (continued)

m	i	**set**
9	0	{1, 2, 5, 9, 10, 12, 13, 15, 16, 20, 23, 24}
		$n = 25$
9	0	{1, 2, 3, 5, 7, 8, 18, 19, 21, 23, 24, 25}
9	0	{1, 3, 5, 6, 9, 11, 12, 14, 15, 17, 20, 21, 23, 25}
		$n = 26$
9	1	{1, 4, 5, 6, 7, 9, 18, 20, 21, 22, 23, 26}
9	2	{1, 2, 3, 4, 5, 7, 9, 10, 11, 13, 14, 16, 17, 18, 20, 22, 23, 24, 25, 26}
		$n = 27$
9	3	{1, 3, 6, 10, 18, 22, 25, 27}
9	0	{1, 2, 4, 8, 9, 10, 18, 19, 20, 24, 26, 27}
9	0	{1, 2, 3, 4, 7, 12, 13, 15, 16, 21, 24, 25, 26, 27}
9	2	{1, 6, 7, 8, 9, 12, 13, 15, 16, 19, 20, 21, 22, 27}
		$n = 28$
9	0	{1, 2, 6, 10, 12, 14, 15, 17, 19, 23, 27, 28}
9	0	{1, 3, 4, 5, 8, 11, 18, 21, 24, 25, 26, 28}
9	4	{1, 5, 7, 8, 12, 14, 15, 17, 21, 22, 24, 28}
9	3	{1, 2, 3, 4, 6, 7, 10, 11, 18, 19, 22, 23, 25, 26, 27, 28}
		$n = 29$
10	1	{1, 4, 5, 6, 9, 11, 12, 13, 14, 16, 17, 18, 19, 21, 24, 25, 26, 29}
		$n = 30$
10	0	{1, 2, 4, 7, 9, 10, 11, 15, 16, 20, 21, 22, 24, 27, 29, 30}
		$n = 31$
10	0	{1, 3, 4, 5, 7, 8, 9, 12, 15, 17, 20, 23, 24, 25, 27, 28, 29, 31}
10	3	{1, 2, 3, 4, 6, 8, 9, 10, 13, 14, 18, 19, 22, 23, 24, 26, 28, 29, 30, 31}
		$n = 32$
10	0	{1, 5, 6, 7, 8, 11, 13, 15, 18, 20, 22, 25, 26, 27, 28, 32}
10	0	{1, 2, 3, 6, 7, 10, 12, 13, 15, 16, 17, 18, 20, 21, 23, 26, 27, 30, 31, 32}
		$n = 33$
11	2	{1, 2, 3, 5, 6, 7, 8, 9, 12, 22, 25, 26, 27, 28, 29, 31, 32, 33}
		$n = 34$
11	0	{1, 4, 5, 10, 12, 13, 22, 23, 25, 30, 31, 34}
		$n = 35$
11	1	{1, 2, 4, 6, 10, 11, 12, 14, 22, 24, 25, 26, 30, 32, 34, 35}
11	0	{1, 3, 7, 8, 9, 11, 12, 13, 14, 22, 23, 24, 25, 27, 28, 29, 33, 35}
		$n = 36$
11	0	{1, 2, 3, 4, 7, 10, 11, 15, 22, 26, 27, 30, 33, 34, 35, 36}
11	1	{1, 3, 4, 5, 6, 7, 10, 13, 14, 15, 22, 23, 24, 27, 30, 31, 32, 33, 34, 36}
		$n = 37$
11	1	{1, 2, 3, 8, 10, 11, 13, 16, 22, 25, 27, 28, 30, 35, 36, 37}
11	1	{1, 5, 7, 8, 10, 12, 15, 16, 22, 23, 26, 28, 30, 31, 33, 37}
11	2	{1, 2, 4, 5, 7, 9, 13, 14, 16, 22, 24, 25, 29, 31, 33, 34, 36, 37}
11	7	{1, 3, 4, 9, 11, 12, 14, 15, 16, 22, 23, 24, 26, 27, 29, 34, 35, 37}

Table B.2: (continued)

Table B.2: (continued)

m	***i***	**set**
		$n = 38$
11	7	{1, 2, 3, 5, 9, 10, 11, 13, 14, 17, 22, 25, 26, 28, 29, 30, 34, 36, 37, 38}
11	0	{1, 4, 8, 9, 10, 12, 13, 14, 16, 17, 22, 23, 25, 26, 27, 29, 30, 31, 35, 38}
11	0	{1, 2, 5, 6, 7, 9, 10, 11, 12, 13, 15, 17, 22, 24, 26, 27, 28, 29, 30, 32, 33, 34, 37, 38}
11	2	{1, 3, 4, 6, 7, 8, 9, 10, 13, 15, 16, 17, 22, 23, 24, 26, 29, 30, 31, 32, 33, 35, 36, 38}
		$n = 39$
11	0	{1, 2, 6, 9, 10, 16, 18, 22, 24, 30, 31, 34, 38, 39}
11	2	{1, 8, 10, 11, 13, 17, 18, 22, 23, 27, 29, 30, 32, 39}
11	2	{1, 2, 3, 5, 6, 11, 13, 14, 15, 18, 22, 25, 26, 27, 29, 34, 35, 37, 38, 39}
11	2	{1, 2, 4, 5, 8, 11, 12, 15, 16, 18, 22, 24, 25, 28, 29, 32, 35, 36, 38, 39}
11	0	{1, 3, 5, 8, 9, 14, 15, 16, 17, 18, 22, 23, 24, 25, 26, 31, 32, 35, 37, 39}
11	0	{1, 4, 5, 6, 9, 12, 13, 15, 17, 18, 22, 23, 25, 27, 28, 31, 34, 35, 36, 39}
11	0	{1, 2, 3, 4, 8, 9, 10, 12, 13, 14, 18, 22, 26, 27, 28, 30, 31, 32, 36, 37, 38, 39}
11	2	{1, 3, 4, 6, 10, 11, 12, 14, 16, 17, 18, 22, 23, 24, 26, 28, 29, 30, 34, 36, 37, 39}
		$n = 40$
11	0	{1, 5, 8, 11, 12, 15, 18, 19, 22, 23, 26, 29, 30, 33, 36, 40}
11	1	{1, 2, 3, 4, 5, 6, 8, 10, 14, 19, 22, 27, 31, 33, 35, 36, 37, 38, 39, 40}
11	2	{1, 2, 8, 9, 10, 12, 13, 14, 17, 19, 22, 24, 27, 28, 29, 31, 32, 33, 39, 40}
11	1	{1, 3, 6, 7, 9, 11, 16, 17, 18, 19, 22, 23, 24, 25, 30, 32, 34, 35, 38, 40}
11	2	{1, 4, 5, 7, 11, 12, 13, 16, 18, 19, 22, 23, 25, 28, 29, 30, 34, 36, 37, 40}
11	0	{1, 2, 3, 5, 6, 7, 10, 13, 14, 15, 16, 19, 22, 25, 26, 27, 28, 31, 34, 35, 36, 38, 39, 40}
11	0	{1, 2, 4, 7, 9, 10, 12, 14, 15, 16, 17, 19, 22, 24, 25, 26, 27, 29, 31, 32, 34, 37, 39, 40}
11	0	{1, 3, 4, 6, 8, 9, 11, 13, 15, 17, 18, 19, 22, 23, 24, 26, 28, 30, 32, 33, 35, 37, 38, 40}
		$n = 41$
11	0	{1, 2, 3, 4, 5, 7, 8, 13, 20, 22, 29, 34, 35, 37, 38, 39, 40, 41}
11	2	{1, 2, 3, 8, 9, 14, 15, 17, 20, 22, 25, 27, 28, 33, 34, 39, 40, 41}
11	0	{1, 2, 7, 8, 10, 12, 15, 18, 20, 22, 24, 27, 30, 32, 34, 35, 40, 41}

Table B.2: (continued)

Table B.2: (continued)

m	i	**set**
11	2	{1, 3, 8, 11, 12, 15, 17, 18, 19, 20, 22, 23, 24, 25, 27, 30, 31, 34, 39, 41}
11	1	{1, 4, 5, 8, 10, 11, 13, 17, 19, 20, 22, 23, 25, 29, 31, 32, 34, 37, 38, 41}
11	0	{1, 7, 8, 9, 10, 11, 14, 15, 19, 20, 22, 23, 27, 28, 31, 32, 33, 34, 35, 41}
11	0	{1, 3, 4, 6, 8, 10, 14, 16, 18, 19, 20, 22, 23, 24, 26, 28, 32, 34, 36, 38, 39, 41}
11	2	{1, 4, 6, 7, 8, 9, 12, 16, 17, 19, 20, 22, 23, 25, 26, 30, 33, 34, 35, 36, 38, 41}
11	0	{1, 5, 6, 8, 12, 13, 14, 15, 16, 19, 20, 22, 23, 26, 27, 28, 29, 30, 34, 36, 37, 41}
11	0	{1, 2, 3, 4, 6, 8, 9, 10, 11, 12, 16, 20, 22, 26, 30, 31, 32, 33, 34, 36, 38, 39, 40, 41}
11	2	{1, 2, 4, 6, 7, 8, 11, 14, 16, 17, 18, 20, 22, 24, 25, 26, 28, 31, 34, 35, 36, 38, 40, 41}
11	0	{1, 2, 5, 6, 8, 9, 11, 13, 15, 16, 18, 20, 22, 24, 26, 27, 29, 31, 33, 34, 36, 37, 40, 41}
11	1	{1, 2, 4, 5, 8, 9, 10, 12, 13, 14, 17, 18, 20, 22, 24, 25, 28, 29, 30, 32, 33, 34, 37, 38, 40, 41}
11	0	{1, 3, 4, 5, 7, 8, 9, 11, 12, 13, 14, 18, 19, 20, 22, 23, 24, 28, 29, 30, 31, 33, 34, 35, 37, 38, 39, 41}
11	1	{1, 3, 5, 6, 7, 8, 9, 10, 13, 15, 16, 17, 18, 19, 20, 22, 23, 24, 25, 26, 27, 29, 32, 33, 34, 35, 36, 37, 39, 41}
11	1	{1, 2, 3, 5, 6, 7, 8, 10, 11, 12, 13, 14, 15, 16, 17, 20, 22, 25, 26, 27, 28, 29, 30, 31, 32, 34, 35, 36, 37, 39, 40, 41}
		$n = 42$
11	0	{1, 2, 7, 12, 14, 16, 19, 21, 22, 24, 27, 29, 31, 36, 41, 42}
11	1	{1, 4, 5, 9, 10, 18, 20, 21, 22, 23, 25, 33, 34, 38, 39, 42}
11	0	{1, 6, 7, 9, 13, 14, 20, 21, 22, 23, 29, 30, 34, 36, 37, 42}
11	1	{1, 2, 3, 4, 5, 11, 13, 14, 15, 21, 22, 28, 29, 30, 32, 38, 39, 40, 41, 42}
11	0	{1, 2, 3, 6, 7, 10, 11, 15, 18, 21, 22, 25, 28, 32, 33, 36, 37, 40, 41, 42}
11	0	{1, 2, 5, 7, 8, 9, 12, 17, 19, 21, 22, 24, 26, 31, 34, 35, 36, 38, 41, 42}
11	2	{1, 4, 8, 10, 14, 16, 17, 18, 20, 21, 22, 23, 25, 26, 27, 29, 33, 35, 39, 42}
11	0	{1, 5, 6, 7, 8, 13, 16, 17, 20, 21, 22, 23, 26, 27, 30, 35, 36, 37, 38, 42}
11	2	{1, 2, 3, 4, 8, 9, 11, 13, 15, 16, 17, 21, 22, 26, 27, 28, 30, 32, 34, 35, 39, 40, 41, 42}
11	2	{1, 2, 4, 5, 6, 10, 12, 13, 16, 18, 19, 21, 22, 24, 25, 27, 30, 31, 33, 37, 38, 39, 41, 42}
11	1	{1, 3, 4, 6, 8, 11, 12, 15, 17, 19, 20, 21, 22, 23, 24, 26, 28, 31, 32, 35, 37, 39, 40, 42}

Table B.2: (continued)

Table B.2: (continued)

m	i	set
11	1	{1, 2, 4, 6, 8, 9, 10, 12, 13, 14, 17, 18, 19, 21, 22, 24, 25, 26, 29, 30, 31, 33, 34, 35, 37, 39, 41, 42}
11	2	{1, 3, 4, 5, 6, 9, 11, 12, 14, 15, 16, 19, 20, 21, 22, 23, 24, 27, 28, 29, 31, 32, 34, 37, 38, 39, 40, 42}
11	0	{1, 3, 7, 9, 10, 11, 12, 13, 15, 16, 18, 19, 20, 21, 22, 23, 24, 25, 27, 28, 30, 31, 32, 33, 34, 36, 40, 42}
11	0	{1, 2, 3, 5, 6, 7, 8, 9, 10, 11, 14, 15, 16, 17, 18, 21, 22, 25, 26, 27, 28, 29, 32, 33, 34, 35, 36, 37, 38, 40, 41, 42}
11	0	{1, 3, 5, 7, 8, 10, 11, 12, 13, 14, 15, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 28, 29, 30, 31, 32, 33, 35, 36, 38, 40, 42}
		$n = 43$
12	3	{1, 3, 4, 6, 8, 13, 16, 18, 20, 21, 23, 24, 26, 28, 31, 36, 38, 40, 41, 43}
		$n = 44$
12	0	{1, 2, 3, 5, 6, 7, 8, 9, 13, 14, 16, 17, 18, 19, 20, 22, 23, 25, 26, 27, 28, 29, 31, 32, 36, 37, 38, 39, 40, 42, 43, 44}
		$n = 45$
12	1	{1, 4, 5, 10, 13, 15, 16, 21, 22, 24, 25, 30, 31, 33, 36, 41, 42, 45}
12	0	{1, 2, 7, 9, 10, 13, 14, 15, 16, 17, 19, 27, 29, 30, 31, 32, 33, 36, 37, 39, 44, 45}
		$n = 46$
12	0	{1, 3, 7, 8, 9, 11, 13, 18, 19, 20, 27, 28, 29, 34, 36, 38, 39, 40, 44, 46}
12	1	{1, 2, 4, 6, 10, 11, 13, 14, 15, 17, 21, 23, 24, 26, 30, 32, 33, 34, 36, 37, 41, 43, 45, 46}
		$n = 47$
13	0	{1, 4, 7, 8, 12, 13, 15, 16, 17, 20, 21, 27, 28, 31, 32, 33, 35, 36, 40, 41, 44, 47}
		$n = 48$
13	4	{1, 2, 4, 5, 7, 9, 12, 14, 15, 18, 20, 22, 27, 29, 31, 34, 35, 37, 40, 42, 44, 45, 47, 48}
		$n = 49$
13	1	{1, 2, 3, 4, 5, 6, 7, 10, 12, 17, 19, 20, 23, 27, 30, 31, 33, 38, 40, 43, 44, 45, 46, 47, 48, 49}
13	0	{1, 3, 4, 6, 7, 8, 9, 10, 12, 13, 14, 16, 18, 19, 20, 21, 22, 23, 27, 28, 29, 30, 31, 32, 34, 36, 37, 38, 40, 41, 42, 43, 44, 46, 47, 49}
		$n = 50$
13	0	{1, 2, 3, 5, 6, 11, 12, 15, 16, 17, 18, 24, 27, 33, 34, 35, 36, 39, 40, 45, 46, 48, 49, 50}
13	1	{1, 8, 10, 11, 12, 13, 17, 18, 19, 21, 23, 24, 27, 28, 30, 32, 33, 34, 38, 39, 40, 41, 43, 50}
		$n = 51$
13	2	{1, 4, 5, 7, 11, 13, 15, 19, 24, 25, 27, 28, 33, 37, 39, 41, 45, 47, 48, 51}

Table B.2: (continued)

Table B.2: (continued)

m	i	set
13	3	{1, 2, 3, 6, 8, 15, 18, 19, 20, 21, 25, 27, 31, 32, 33, 34, 37, 44, 46, 49, 50, 51}
13	1	{1, 2, 8, 9, 10, 14, 17, 20, 21, 22, 23, 25, 27, 29, 30, 31, 32, 35, 38, 42, 43, 44, 50, 51}
13	1	{1, 3, 4, 5, 6, 7, 9, 10, 11, 13, 14, 17, 18, 22, 23, 24, 25, 27, 28, 29, 30, 34, 35, 38, 39, 41, 42, 43, 45, 46, 47, 48, 49, 51}
		$n = 52$
13	0	{1, 2, 3, 8, 9, 12, 13, 15, 17, 19, 22, 26, 27, 31, 34, 36, 38, 40, 41, 44, 45, 50, 51, 52}
13	0	{1, 3, 8, 11, 14, 15, 17, 18, 20, 24, 25, 26, 27, 28, 29, 33, 35, 36, 38, 39, 42, 45, 50, 52}
13	3	{1, 4, 6, 7, 8, 9, 15, 16, 18, 22, 25, 26, 27, 28, 31, 35, 37, 38, 44, 45, 46, 47, 49, 52}
13	2	{1, 2, 4, 6, 7, 8, 11, 12, 13, 14, 15, 16, 19, 20, 24, 26, 27, 29, 33, 34, 37, 38, 39, 40, 41, 42, 45, 46, 47, 49, 51, 52}
		$n = 53$
13	0	{1, 7, 8, 11, 12, 14, 17, 22, 24, 26, 28, 30, 32, 37, 40, 42, 43, 46, 47, 53}
13	0	{1, 2, 5, 6, 7, 11, 16, 17, 20, 22, 24, 25, 29, 30, 32, 34, 37, 38, 43, 47, 48, 49, 52, 53}
13	3	{1, 2, 4, 5, 6, 10, 15, 17, 18, 19, 22, 23, 25, 29, 31, 32, 35, 36, 37, 39, 44, 48, 49, 50, 52, 53}
13	0	{1, 3, 4, 5, 6, 9, 11, 17, 19, 21, 24, 25, 26, 28, 29, 30, 33, 35, 37, 43, 45, 48, 49, 50, 51, 53}
13	4	{1, 2, 3, 7, 8, 9, 10, 12, 14, 15, 17, 18, 21, 23, 31, 33, 36, 37, 39, 40, 42, 44, 45, 46, 47, 51, 52, 53}
13	0	{1, 2, 3, 4, 8, 9, 11, 12, 14, 16, 17, 19, 20, 21, 24, 30, 33, 34, 35, 37, 38, 40, 42, 43, 45, 46, 50, 51, 52, 53}
13	5	{1, 4, 8, 10, 12, 14, 15, 16, 17, 18, 19, 20, 22, 23, 26, 28, 31, 32, 34, 35, 36, 37, 38, 39, 40, 42, 44, 46, 50, 53}
13	3	{1, 3, 5, 6, 7, 9, 10, 15, 16, 17, 18, 20, 21, 23, 25, 26, 28, 29, 31, 33, 34, 36, 37, 38, 39, 44, 45, 47, 48, 49, 51, 53}
		$n = 54$
13	0	{1, 3, 4, 7, 10, 11, 15, 16, 17, 20, 22, 24, 25, 26, 29, 30, 31, 33, 35, 38, 39, 40, 44, 45, 48, 51, 52, 54}
13	0	{1, 3, 5, 8, 11, 12, 16, 18, 20, 21, 22, 23, 24, 26, 29, 31, 32, 33, 34, 35, 37, 39, 43, 44, 47, 50, 52, 54}
13	4	{1, 4, 7, 11, 12, 13, 14, 16, 17, 19, 21, 22, 23, 24, 31, 32, 33, 34, 36, 38, 39, 41, 42, 43, 44, 48, 51, 54}
13	1	{1, 5, 8, 10, 11, 13, 14, 15, 16, 18, 19, 22, 24, 25, 30, 31, 33, 36, 37, 39, 40, 41, 42, 44, 45, 47, 50, 54}
13	0	{1, 2, 3, 4, 5, 8, 9, 11, 15, 19, 20, 22, 23, 24, 25, 27, 28, 30, 31, 32, 33, 35, 36, 40, 44, 46, 47, 50, 51, 52, 53, 54}

Table B.2: (continued)

Table B.2: (continued)

m	***i***	**set**
13	0	{1, 2, 3, 7, 9, 10, 11, 12, 17, 18, 19, 20, 21, 22, 24, 27, 28, 31, 33, 34, 35, 36, 37, 38, 43, 44, 45, 46, 48, 52, 53, 54}
13	5	{1, 2, 4, 5, 8, 9, 10, 11, 12, 13, 14, 21, 22, 24, 26, 27, 28, 29, 31, 33, 34, 41, 42, 43, 44, 45, 46, 47, 50, 51, 53, 54}
13	1	{1, 2, 7, 9, 11, 13, 14, 15, 17, 18, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 37, 38, 40, 41, 42, 44, 46, 48, 53, 54}
		$n = 55$
13	1	{1, 2, 4, 7, 12, 15, 17, 18, 24, 32, 38, 39, 41, 44, 49, 52, 54, 55}
13	5	{1, 3, 4, 6, 8, 15, 21, 23, 24, 25, 26, 30, 31, 32, 33, 35, 41, 48, 50, 52, 53, 55}
13	0	{1, 4, 7, 8, 9, 13, 17, 23, 24, 25, 27, 29, 31, 32, 33, 39, 43, 47, 48, 49, 52, 55}
13	0	{1, 3, 5, 7, 9, 10, 13, 20, 21, 22, 24, 25, 31, 32, 34, 35, 36, 43, 46, 47, 49, 51, 53, 55}
13	8	{1, 5, 6, 10, 15, 17, 20, 22, 24, 25, 26, 27, 29, 30, 31, 32, 34, 36, 39, 41, 46, 50, 51, 55}
13	0	{1, 2, 3, 4, 6, 9, 12, 13, 18, 21, 24, 26, 27, 29, 30, 32, 35, 38, 43, 44, 47, 50, 52, 53, 54, 55}
13	1	{1, 3, 7, 8, 9, 10, 11, 12, 13, 16, 17, 19, 22, 34, 37, 39, 40, 43, 44, 45, 46, 47, 48, 49, 53, 55}
13	0	{1, 6, 8, 10, 11, 12, 15, 16, 19, 21, 22, 26, 27, 29, 30, 34, 35, 37, 40, 41, 44, 45, 46, 48, 50, 55}
13	0	{1, 2, 4, 5, 7, 8, 11, 15, 16, 18, 19, 20, 21, 25, 31, 35, 36, 37, 38, 40, 41, 45, 48, 49, 51, 52, 54, 55}
13	0	{1, 3, 4, 5, 6, 11, 12, 15, 16, 17, 19, 20, 23, 26, 30, 33, 36, 37, 39, 40, 41, 44, 45, 50, 51, 52, 53, 55}
13	1	{1, 4, 5, 7, 9, 11, 12, 13, 16, 19, 20, 21, 23, 27, 29, 33, 35, 36, 37, 40, 43, 44, 45, 47, 49, 51, 52, 55}
13	0	{1, 2, 3, 7, 10, 11, 15, 16, 17, 18, 19, 22, 23, 25, 27, 29, 31, 33, 34, 37, 38, 39, 40, 41, 45, 46, 49, 53, 54, 55}
13	6	{1, 2, 6, 9, 10, 11, 13, 16, 18, 19, 21, 22, 23, 25, 26, 30, 31, 33, 34, 35, 37, 38, 40, 43, 45, 46, 47, 50, 54, 55}
13	1	{1, 2, 3, 5, 7, 8, 10, 12, 15, 18, 20, 21, 22, 23, 24, 27, 29, 32, 33, 34, 35, 36, 38, 41, 44, 46, 48, 49, 51, 53, 54, 55}
13	0	{1, 2, 5, 6, 8, 9, 10, 12, 13, 17, 18, 20, 22, 23, 24, 26, 30, 32, 33, 34, 36, 38, 39, 43, 44, 46, 47, 48, 50, 51, 54, 55}
13	5	{1, 2, 3, 4, 5, 6, 8, 9, 11, 13, 16, 17, 18, 19, 20, 25, 26, 27, 29, 30, 31, 36, 37, 38, 39, 40, 43, 45, 47, 48, 50, 51, 52, 53, 54, 55}
		$n = 56$
13	0	{1, 2, 3, 4, 7, 14, 16, 18, 19, 20, 22, 23, 34, 35, 37, 38, 39, 41, 43, 50, 53, 54, 55, 56}
13	0	{1, 7, 8, 10, 11, 12, 13, 14, 16, 21, 25, 28, 29, 32, 36, 41, 43, 44, 45, 46, 47, 49, 50, 56}
13	0	{1, 2, 3, 5, 6, 7, 8, 9, 15, 16, 21, 22, 23, 27, 30, 34, 35, 36, 41, 42, 48, 49, 50, 51, 52, 54, 55, 56}

Table B.2: (continued)

Table B.2: (continued)

m	i	set
13	0	{1, 2, 3, 7, 11, 13, 15, 18, 19, 21, 23, 24, 26, 27, 30, 31, 33, 34, 36, 38, 39, 42, 44, 46, 50, 54, 55, 56}
13	1	{1, 2, 4, 5, 7, 10, 13, 14, 17, 18, 23, 26, 27, 28, 29, 30, 31, 34, 39, 40, 43, 44, 47, 50, 52, 53, 55, 56}
13	1	{1, 3, 4, 5, 7, 8, 12, 13, 15, 16, 17, 19, 24, 25, 32, 33, 38, 40, 41, 42, 44, 45, 49, 50, 52, 53, 54, 56}
13	1	{1, 3, 4, 6, 7, 9, 11, 12, 15, 17, 18, 22, 25, 26, 31, 32, 35, 39, 40, 42, 45, 46, 48, 50, 51, 53, 54, 56}
13	0	{1, 4, 7, 8, 10, 12, 15, 20, 22, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 35, 37, 42, 45, 47, 49, 50, 53, 56}
13	0	{1, 2, 3, 4, 5, 6, 7, 8, 9, 11, 13, 14, 20, 23, 24, 26, 31, 33, 34, 37, 43, 44, 46, 48, 49, 50, 51, 52, 53, 54, 55, 56}
13	8	{1, 2, 5, 7, 10, 11, 15, 16, 17, 18, 20, 21, 22, 23, 24, 28, 29, 33, 34, 35, 36, 37, 39, 40, 41, 42, 46, 47, 50, 52, 55, 56}
13	2	{1, 2, 6, 7, 8, 9, 10, 13, 15, 17, 19, 20, 21, 23, 26, 28, 29, 31, 34, 36, 37, 38, 40, 42, 44, 47, 48, 49, 50, 51, 55, 56}
13	2	{1, 3, 5, 7, 8, 11, 12, 14, 17, 19, 20, 21, 22, 25, 26, 27, 30, 31, 32, 35, 36, 37, 38, 40, 43, 45, 46, 49, 50, 52, 54, 56}
13	6	{1, 3, 6, 7, 9, 12, 13, 14, 16, 17, 18, 20, 21, 24, 25, 27, 30, 32, 33, 36, 37, 39, 40, 41, 43, 44, 45, 48, 50, 51, 54, 56}
13	0	{1, 5, 6, 7, 9, 10, 12, 14, 18, 19, 21, 22, 24, 25, 26, 28, 29, 31, 32, 33, 35, 36, 38, 39, 43, 45, 47, 48, 50, 51, 52, 56}
13	1	{1, 2, 4, 6, 7, 8, 9, 10, 11, 14, 16, 17, 19, 22, 23, 24, 27, 28, 29, 30, 33, 34, 35, 38, 40, 41, 43, 46, 47, 48, 49, 50, 51, 53, 55, 56}
13	0	{1, 4, 5, 6, 7, 9, 10, 11, 12, 13, 15, 16, 18, 19, 20, 25, 27, 28, 29, 30, 32, 37, 38, 39, 41, 42, 44, 45, 46, 47, 48, 50, 51, 52, 53, 56}
		$n = 57$
13	1	{1, 4, 5, 10, 15, 17, 21, 24, 27, 28, 30, 31, 34, 37, 41, 43, 48, 53, 54, 57}
13	0	{1, 6, 9, 10, 11, 15, 17, 18, 22, 23, 27, 31, 35, 36, 40, 41, 43, 47, 48, 49, 52, 57}
13	1	{1, 2, 4, 6, 8, 12, 13, 14, 15, 20, 22, 27, 31, 36, 38, 43, 44, 45, 46, 50, 52, 54, 56, 57}
13	2	{1, 2, 6, 7, 10, 12, 13, 15, 16, 18, 20, 28, 30, 38, 40, 42, 43, 45, 46, 48, 51, 52, 56, 57}
13	7	{1, 5, 7, 8, 14, 15, 16, 17, 18, 21, 22, 24, 34, 36, 37, 40, 41, 42, 43, 44, 50, 51, 53, 57}
13	2	{1, 10, 11, 12, 13, 15, 20, 21, 23, 25, 26, 27, 31, 32, 33, 35, 37, 38, 43, 45, 46, 47, 48, 57}
13	2	{1, 2, 3, 7, 13, 14, 15, 17, 19, 21, 22, 24, 25, 33, 34, 36, 37, 39, 41, 43, 44, 45, 51, 55, 56, 57}
13	0	{1, 2, 4, 8, 9, 14, 15, 17, 18, 21, 25, 26, 27, 31, 32, 33, 37, 40, 41, 43, 44, 49, 50, 54, 56, 57}
13	0	{1, 2, 7, 9, 10, 15, 16, 17, 21, 22, 25, 26, 28, 30, 32, 33, 36, 37, 41, 42, 43, 48, 49, 51, 56, 57}

Table B.2: (continued)

Table B.2: (continued)

m	***i***	**set**
13	8	{1, 3, 5, 6, 7, 8, 10, 12, 15, 19, 20, 25, 28, 30, 33, 38, 39, 43, 46, 48, 50, 51, 52, 53, 55, 57}
13	0	{1, 4, 6, 7, 8, 9, 11, 14, 15, 16, 17, 23, 28, 30, 35, 41, 42, 43, 44, 47, 49, 50, 51, 52, 54, 57}
13	0	{1, 2, 3, 6, 7, 9, 12, 14, 15, 18, 19, 20, 24, 26, 32, 34, 38, 39, 40, 43, 44, 46, 49, 51, 52, 55, 56, 57}
13	0	{1, 3, 4, 5, 9, 13, 14, 15, 16, 17, 19, 21, 26, 27, 31, 32, 37, 39, 41, 42, 43, 44, 45, 49, 53, 54, 55, 57}
13	1	{1, 3, 4, 5, 6, 12, 14, 15, 16, 18, 19, 20, 22, 25, 27, 31, 33, 36, 38, 39, 40, 42, 43, 44, 46, 52, 53, 54, 55, 57}
13	1	{1, 3, 4, 6, 7, 8, 10, 11, 13, 15, 17, 19, 23, 24, 26, 32, 34, 35, 39, 41, 43, 45, 47, 48, 50, 51, 52, 54, 55, 57}
13	0	{1, 5, 6, 7, 8, 9, 12, 13, 14, 15, 16, 20, 24, 25, 26, 32, 33, 34, 38, 42, 43, 44, 45, 46, 49, 50, 51, 52, 53, 57}
13	1	{1, 2, 4, 5, 6, 7, 10, 11, 15, 16, 17, 18, 23, 24, 25, 26, 32, 33, 34, 35, 40, 41, 42, 43, 47, 48, 51, 52, 53, 54, 56, 57}
13	7	{1, 2, 5, 6, 8, 11, 14, 15, 17, 22, 23, 24, 25, 26, 27, 28, 30, 31, 32, 33, 34, 35, 36, 41, 43, 44, 47, 50, 52, 53, 56, 57}
13	0	{1, 3, 5, 7, 8, 9, 10, 13, 15, 17, 18, 19, 21, 22, 26, 28, 30, 32, 36, 37, 39, 40, 41, 43, 45, 48, 49, 50, 51, 53, 55, 57}
13	0	{1, 3, 9, 11, 12, 14, 15, 16, 19, 20, 21, 23, 24, 25, 27, 28, 30, 31, 33, 34, 35, 37, 38, 39, 42, 43, 44, 46, 47, 49, 55, 57}
13	1	{1, 2, 3, 4, 5, 7, 11, 12, 14, 15, 19, 20, 21, 22, 23, 26, 28, 30, 32, 35, 36, 37, 38, 39, 43, 44, 46, 47, 51, 53, 54, 55, 56, 57}
13	1	{1, 2, 3, 4, 8, 10, 13, 15, 16, 17, 18, 19, 21, 24, 25, 27, 28, 30, 31, 33, 34, 37, 39, 40, 41, 42, 43, 45, 48, 50, 54, 55, 56, 57}
13	8	{1, 2, 3, 5, 8, 10, 11, 12, 15, 16, 18, 19, 20, 21, 23, 26, 27, 31, 32, 35, 37, 38, 39, 40, 42, 43, 46, 47, 48, 50, 53, 55, 56, 57}
13	0	{1, 2, 4, 5, 7, 9, 10, 11, 12, 13, 15, 16, 20, 21, 22, 23, 24, 34, 35, 36, 37, 38, 42, 43, 45, 46, 47, 48, 49, 51, 53, 54, 56, 57}
13	0	{1, 2, 5, 8, 9, 11, 12, 13, 14, 15, 18, 20, 21, 23, 24, 27, 28, 30, 31, 34, 35, 37, 38, 40, 43, 44, 45, 46, 47, 49, 50, 53, 56, 57}
13	2	{1, 3, 6, 11, 13, 14, 15, 16, 17, 18, 19, 22, 23, 24, 26, 27, 28, 30, 31, 32, 34, 35, 36, 39, 40, 41, 42, 43, 44, 45, 47, 52, 55, 57}
13	0	{1, 4, 5, 6, 9, 10, 12, 13, 15, 18, 20, 22, 24, 25, 26, 27, 28, 30, 31, 32, 33, 34, 36, 38, 40, 43, 45, 46, 48, 49, 52, 53, 54, 57}
13	0	{1, 2, 3, 4, 5, 6, 7, 9, 11, 13, 14, 15, 17, 18, 19, 23, 25, 28, 30, 33, 35, 39, 40, 41, 43, 44, 45, 47, 49, 51, 52, 53, 54, 55, 56, 57}
13	0	{1, 2, 3, 4, 6, 8, 9, 10, 12, 15, 16, 19, 20, 22, 24, 26, 27, 28, 30, 31, 32, 34, 36, 38, 39, 42, 43, 46, 48, 49, 50, 52, 54, 55, 56, 57}
13	0	{1, 2, 3, 5, 6, 8, 9, 10, 11, 13, 15, 16, 17, 19, 22, 23, 25, 27, 31, 33, 35, 36, 39, 41, 42, 43, 45, 47, 48, 49, 50, 52, 53, 55, 56, 57}
13	0	{1, 3, 4, 7, 8, 9, 10, 11, 12, 15, 18, 19, 20, 21, 22, 23, 24, 25, 33, 34, 35, 36, 37, 38, 39, 40, 43, 46, 47, 48, 49, 50, 51, 54, 55, 57}

Table B.2: (continued)

Table B.2: (continued)

m	i	set
13	1	{1, 4, 7, 8, 11, 12, 13, 14, 15, 16, 18, 20, 21, 22, 23, 25, 26, 28, 30, 32, 33, 35, 36, 37, 38, 40, 42, 43, 44, 45, 46, 47, 50, 51, 54, 57}
		$n = 58$
13	0	{1, 2, 3, 5, 7, 13, 15, 16, 18, 26, 33, 41, 43, 44, 46, 52, 54, 56, 57, 58}
13	0	{1, 2, 5, 10, 12, 17, 20, 21, 24, 27, 32, 35, 38, 39, 42, 47, 49, 54, 57, 58}
13	1	{1, 3, 4, 8, 10, 12, 15, 19, 23, 27, 32, 36, 40, 44, 47, 49, 51, 55, 56, 58}
13	0	{1, 2, 5, 6, 7, 9, 14, 19, 21, 23, 24, 25, 34, 35, 36, 38, 40, 45, 50, 52, 53, 54, 57, 58}
13	0	{1, 2, 10, 14, 15, 16, 20, 22, 23, 24, 25, 28, 31, 34, 35, 36, 37, 39, 43, 44, 45, 49, 57, 58}
13	1	{1, 3, 4, 6, 7, 8, 9, 14, 15, 17, 20, 25, 34, 39, 42, 44, 45, 50, 51, 52, 53, 55, 56, 58}
13	2	{1, 2, 3, 4, 6, 7, 11, 12, 13, 20, 22, 25, 26, 29, 30, 33, 34, 37, 39, 46, 47, 48, 52, 53, 55, 56, 57, 58}
13	0	{1, 2, 3, 6, 9, 10, 13, 18, 19, 20, 21, 22, 26, 28, 31, 33, 37, 38, 39, 40, 41, 46, 49, 50, 53, 56, 57, 58}
13	1	{1, 2, 4, 5, 6, 10, 11, 12, 14, 18, 23, 24, 28, 29, 30, 31, 35, 36, 41, 45, 47, 48, 49, 53, 54, 55, 57, 58}
13	0	{1, 2, 6, 7, 9, 12, 15, 16, 17, 19, 22, 24, 27, 28, 31, 32, 35, 37, 40, 42, 43, 44, 47, 50, 52, 53, 57, 58}
13	1	{1, 3, 4, 5, 8, 10, 14, 16, 17, 19, 21, 22, 25, 28, 31, 34, 37, 38, 40, 42, 43, 45, 49, 51, 54, 55, 56, 58}
13	0	{1, 3, 5, 7, 8, 9, 11, 12, 14, 16, 17, 18, 22, 29, 30, 37, 41, 42, 43, 45, 47, 48, 50, 51, 52, 54, 56, 58}
13	0	{1, 3, 7, 8, 9, 11, 15, 18, 21, 23, 25, 27, 28, 29, 30, 31, 32, 34, 36, 38, 41, 44, 48, 50, 51, 52, 56, 58}
13	2	{1, 4, 7, 8, 13, 16, 17, 18, 19, 20, 21, 23, 24, 26, 33, 35, 36, 38, 39, 40, 41, 42, 43, 46, 51, 52, 55, 58}
13	0	{1, 6, 7, 8, 11, 13, 14, 16, 19, 24, 26, 27, 28, 29, 30, 31, 32, 33, 35, 40, 43, 45, 46, 48, 51, 52, 53, 58}
13	2	{1, 2, 3, 4, 9, 10, 11, 13, 14, 17, 19, 22, 23, 26, 27, 29, 30, 32, 33, 36, 37, 40, 42, 45, 46, 48, 49, 50, 55, 56, 57, 58}
13	0	{1, 2, 3, 7, 12, 13, 14, 17, 18, 21, 22, 23, 25, 26, 27, 28, 31, 32, 33, 34, 36, 37, 38, 41, 42, 45, 46, 47, 52, 56, 57, 58}
13	1	{1, 2, 4, 5, 7, 9, 11, 17, 18, 19, 20, 24, 25, 27, 28, 29, 30, 31, 32, 34, 35, 39, 40, 41, 42, 48, 50, 52, 54, 55, 57, 58}
13	1	{1, 2, 4, 6, 10, 11, 15, 16, 17, 18, 21, 22, 24, 25, 27, 29, 30, 32, 34, 35, 37, 38, 41, 42, 43, 44, 48, 49, 53, 55, 57, 58}
13	1	{1, 3, 4, 5, 6, 7, 8, 9, 12, 16, 20, 21, 22, 23, 27, 28, 31, 32, 36, 37, 38, 39, 43, 47, 50, 51, 52, 53, 54, 55, 56, 58}
13	0	{1, 3, 5, 6, 8, 10, 11, 16, 18, 19, 20, 22, 23, 25, 27, 29, 30, 32, 34, 36, 37, 39, 40, 41, 43, 48, 49, 51, 53, 54, 56, 58}

Table B.2: (continued)

Table B.2: (continued)

m	i	set
13	0	{1, 3, 6, 8, 10, 11, 12, 14, 15, 17, 18, 19, 20, 21, 28, 29, 30, 31, 38, 39, 40, 41, 42, 44, 45, 47, 48, 49, 51, 53, 56, 58}
13	8	{1, 4, 5, 6, 8, 9, 10, 13, 15, 17, 18, 22, 23, 24, 26, 28, 31, 33, 35, 36, 37, 41, 42, 44, 46, 49, 50, 51, 53, 54, 55, 58}
13	2	{1, 4, 6, 8, 9, 10, 12, 13, 14, 16, 18, 21, 24, 25, 26, 27, 32, 33, 34, 35, 38, 41, 43, 45, 46, 47, 49, 50, 51, 53, 55, 58}
13	0	{1, 5, 8, 9, 10, 11, 13, 14, 15, 20, 21, 22, 24, 26, 27, 29, 30, 32, 33, 35, 37, 38, 39, 44, 45, 46, 48, 49, 50, 51, 54, 58}
13	0	{1, 8, 9, 10, 11, 12, 13, 16, 17, 20, 23, 24, 25, 26, 28, 29, 30, 31, 33, 34, 35, 36, 39, 42, 43, 46, 47, 48, 49, 50, 51, 58}
13	8	{1, 2, 3, 4, 5, 9, 10, 11, 12, 13, 15, 16, 19, 21, 25, 26, 28, 29, 30, 31, 33, 34, 38, 40, 43, 44, 46, 47, 48, 49, 50, 54, 55, 56, 57, 58}
13	1	{1, 2, 4, 7, 9, 11, 12, 14, 15, 16, 18, 19, 20, 21, 22, 23, 24, 29, 30, 35, 36, 37, 38, 39, 40, 41, 43, 44, 45, 47, 48, 50, 52, 55, 57, 58}
13	3	{1, 4, 5, 7, 8, 12, 13, 14, 15, 18, 19, 20, 22, 24, 25, 26, 27, 28, 31, 32, 33, 34, 35, 37, 39, 40, 41, 44, 45, 46, 47, 51, 52, 54, 55, 58}
13	0	{1, 5, 6, 7, 8, 11, 12, 13, 15, 17, 19, 21, 22, 23, 24, 25, 26, 29, 30, 33, 34, 35, 36, 37, 38, 40, 42, 44, 46, 47, 48, 51, 52, 53, 54, 58}
13	3	{1, 2, 3, 4, 5, 6, 7, 11, 13, 14, 15, 16, 17, 20, 21, 23, 26, 27, 28, 29, 30, 31, 32, 33, 36, 38, 39, 42, 43, 44, 45, 46, 48, 52, 53, 54, 55, 56, 57, 58}
13	0	{1, 2, 3, 5, 6, 9, 10, 12, 13, 14, 15, 16, 17, 18, 19, 20, 23, 25, 26, 27, 32, 33, 34, 36, 39, 40, 41, 42, 43, 44, 45, 46, 47, 49, 50, 53, 54, 56, 57, 58}
		$n = 59$
14	5	{1, 2, 3, 7, 12, 14, 16, 17, 18, 19, 20, 21, 22, 23, 24, 26, 29, 31, 34, 36, 37, 38, 39, 40, 41, 42, 43, 44, 46, 48, 53, 57, 58, 59}
		$n = 60$
14	0	{1, 4, 7, 8, 12, 13, 14, 15, 16, 25, 26, 27, 29, 30, 31, 32, 34, 35, 36, 45, 46, 47, 48, 49, 53, 54, 57, 60}
		$n = 61$
14	0	{1, 2, 4, 5, 7, 9, 12, 17, 25, 28, 29, 33, 34, 37, 45, 50, 53, 55, 57, 58, 60, 61}
14	1	{1, 3, 5, 7, 8, 9, 12, 13, 15, 18, 19, 20, 21, 22, 23, 24, 27, 28, 29, 30, 32, 33, 34, 35, 38, 39, 40, 41, 42, 43, 44, 47, 49, 50, 53, 54, 55, 57, 59, 61}
		$n = 62$
14	1	{1, 2, 3, 4, 5, 6, 7, 10, 12, 14, 15, 16, 18, 25, 27, 31, 32, 36, 38, 45, 47, 48, 49, 51, 53, 56, 57, 58, 59, 60, 61, 62}
14	2	{1, 3, 4, 6, 7, 8, 9, 10, 12, 13, 17, 18, 25, 26, 28, 30, 33, 35, 37, 38, 45, 46, 50, 51, 53, 54, 55, 56, 57, 59, 60, 62}

Table B.2: (concluded)