On Usefulness of Information: Framework and NFA Case

Branislav Rovan(✉) and Šimon Sádovský

Department of Computer Science, Comenius University,
Mlynská Dolina, 842 48 Bratislava, Slovakia
{rovan,sadovsky}@dcs.fmph.uniba.sk

Abstract. In this paper we present a brief overview of the results of our research aimed at identifying and formalising various aspects of the notion of information, especially its usefulness, i.e., its ability to simplify a solution of a problem. We formalize the problem via decompositions of automata and present new results in the nondeterministic finite automata setting. We characterise some subfamilies of regular languages that are nondeterministically decomposable/undecomposable and exhibit an infinite sequence of regular languages that are nondeterministically undecomposable but decomposable in the deterministic finite automata setting.

Keywords: Usefulness of information
Nondeterministic finite automata · Decomposition of finite automata
Supplementary information · Advice · Descriptional complexity
State complexity

1 Introduction

In this paper we present a brief overview of the research aimed at identifying and formalising various aspects of the notion of *information* initiated at the Department of Computer Science of the Comenius University about fifteen years ago and pursued by the first author and his master and PhD students. We also present some new recent results.

In the early days of Shannon's theory of information the main concern was in transferring information reliably and fast over possibly noisy channels. The *amount* of information was the important attribute of information considered. Over half a century later we can identify additional aspects, like usefulness, timeliness, etc. The first and so far the basic aspect of information we started to explore was the information usefulness. We shall now describe our approach to handling the problem.

The essence of our approach can be stated as follows: *Information is useful if it can help us to solve a problem easier*. We thus take a computational approach to studying information. There are several basic questions we need to clarify:

H.-J. Böckenhauer et al. (Eds.): Hromkovič Festschrift, LNCS 11011, pp. 85–99, 2018.
https://doi.org/10.1007/978-3-319-98355-4_6

(i) What is a problem?
(ii) What do we mean by solving the problem?
(iii) How do we measure the complexity of the solution?
(iv) How do we provide information (advice) that possibly simplifies the solution of the problem?

It is clear that taking some powerful advice (e.g., providing the solution to the problem) could render the solution to a problem trivial. In our approach we took the position that information should be 'reasonable', i.e., we should also be able to 'simply' check whether the advice information is correct. Thus the complexity of checking the information (advice) should not exceed that of the problem itself.

The idea of providing 'additional information' appeared in probability theory, automata theory (e.g., promise problems [3]) and more recently in on-line algorithms [2,8]. A brief overview of approaches and merging the views on additional information in formal languages, probability theory and Kolmogorov complexity can be found in [12] together with main aspects of our approach.

Following some notation and basic facts in Sect. 2 we present a brief overview of our framework and results on usefulness and other aspects of information in Sect. 3. In Sect. 4 we present some new results concerning usefulness of information in the nondeterministic finite automata setting and its comparison to the deterministic case.

2 Preliminaries

We shall develop our framework for dealing with the notion of usefulness of information using the formalism of formal languages and automata theory. The notions and basic results can be found, e.g., in [7].

We shall denote the length of a word by $|w|$ and a number of elements of a finite set S by $|S|$. The empty word is denoted by ε. To denote the prefix of a word u of length k we shall write $pref(u,k)$ and to denote the suffix of a word u of length k we shall write $suf(u,k)$. Notation $A \subseteq B$ means, that the set A is a (not necessarily proper) subset of the set B. A proper subset is denoted by $\subset$.

A deterministic finite state automaton (DFA) is a 5-tuple $A = (K, \Sigma, \delta, q_0, F)$ with a standard meaning of its components. We assume the transition function $\delta\colon K \times \Sigma \to K$ is total. In the nondeterministic finite state automata (NFA) case we do not allow ε transitions, i.e., $\delta\colon K \times \Sigma \to 2^K$. The language accepted by A, denoted by $L(A)$, is defined by $L(A) = \{w \in \Sigma^* \mid \exists q \in F\colon (q_0, w) \vdash^* (q, \varepsilon)\}$, where the relation 'step of computation' $\vdash$ on configurations in $K \times \Sigma^*$ is defined as usual.

We shall measure complexity of automata by the number of their states, i.e., we shall use the complexity measure $\#_S$ defined by $\#_S(A) = |K|$. We extend this complexity measure to regular languages in a natural way. The (deterministic) state complexity of L, denoted by $sc(L)$, is the number of states of the minimal DFA for L. The nondeterministic state complexity of L is defined by $nsc(L) =$

$\min\{\#_S(A) \mid A \text{ is } NFA \text{ and } L(A) = L\}$.[1] In this paper we shall often need to argue that we found a minimal NFA A for some regular language L. To obtain a lower bound for $nsc(L)$ we shall often employ the fooling set technique from [6].

Theorem 1 (Fooling Set Technique). *Let L be a regular language, $n \in \mathbb{N}$. Suppose there exists a set of pairs $P = \{(x_i, y_i) \mid 1 \leq i \leq n\}$ such that:*

(a) $x_i y_i \in L$ *for* $1 \leq i \leq n$
(b) $x_i y_j \notin L$ *for* $1 \leq i, j \leq n$ *and* $i \neq j$

Then any NFA accepting L has at least n states.

3 Overview of Past Results

As mentioned in the introduction there were many attempts to incorporate supplementary information in various settings. It is generally perceived (see, e.g., the discussion in [8]) that more effort should be spent on trying to better understand the very notion of information.

In this section we illustrate the main ingredients of our framework for studying various aspects of information mentioned in the introduction. We also present a brief overview of the results obtained by our research group so far. The first and so far the basic aspect of information we started to explore was the information usefulness. There are many possible ways to specify the four basic ingredients of our approach: (i) what is a problem; (ii) what do we mean by solving the problem; (iii) how do we measure the complexity of the solution; and (iv) how do we provide information (advice) that possibly simplifies the solution of the problem. An important aspect of our framework is that the complexity of the advice should be smaller than that of the problem to be solved, i.e., the advice should be 'reasonable'.

We first addressed these questions in the deterministic finite automata setting [4]. Here the problem was given by some regular language L (more precisely, the problem is to answer a question, whether any given word w belongs to L) and solving the problem meant to find a DFA A such that $L = L(A)$. We decided to measure the complexity of the solution by taking the state complexity of A. Since we are interested in the simplest possible solutions it makes sense to take A to be the minimal automaton for L. Suppose we have some additional information about the input word (*advice*). It might be possible, that we could than find a simpler solution (a smaller automaton A_{new}) for deciding whether the input word belongs to L. We shall provide the advice information by another regular language L_{adv}, i.e., the advice is that the input word belongs to L_{adv} (given by the minimal DFA A_{adv}). Note that the minimal automaton A_{new} may by itself accept some superset of L but together with the advice it properly defines $L = L(A_{new}) \cap L_{adv}$.

[1] Note that our assumption to only consider NFA without ε moves does not influence the measure, since for each A there is an equivalent A' without ε moves having the same number of states. We made this assumption to simplify our presentation later.

To illustrate the above ideas let us take for L the language consisting of all words in a^* of length divisible by 6. Clearly the simplest solution to this problem is the automaton A having 6 states. Now suppose we have additional information, advice, that the input word is of even length (i.e., we take L_{adv} to be the language in a^* consisting of all words of even length). It thus suffices to check whether the length of the input word is divisible by 3 which can be done by A_{new} having just 3 states. Thus the information L_{adv} enabled us to find a simpler solution to the problem L.

Let us now return to the general considerations and illustrate what we mean by reasonable information (advice). Taking some powerful advice could render the solution to a problem trivial (e.g., taking $L_{adv} = L$, which would in the above example lead to A_{new} having just one state and recognising a^*). In this case we cannot 'simply' check, whether the advice information is correct. The complexity of this checking is the same as the complexity of the original problem. Thus we expect *both* A_{new} and A_{adv} to be simpler than A. In such case we consider the information useful.

The main question we considered in [4] was, whether there is a useful information, advice, that could simplify the solution to a given problem L. We reformulated this question in purely automata theoretic terms as follows: Does there exist a (nontrivial) parallel decomposition of a minimal DFA for L? It should be no surprise, that there are problems whose solution cannot be simplified by any useful information – advice. We also identify problems, for which a 'small' advice can substantially reduce the complexity of the solution. We also considered cases of a 'more precise' advice, e.g., instead of providing information whether the input word is accepted by A_{adv} we can provide information about the state A_{adv} reaches after reading the input word. This led to various types of decompositions of automata which we compared.

In order to further explore the notion of usefulness of information we decided to proceed by varying the four basic ingredients mentioned above. In [10] we take a problem L to be a deterministic context-free language and its solution the corresponding deterministic push-down automaton ($DPDA$) A. We measure the complexity of the solution via the number of states and auxiliary push-down symbols of A. The advice is given by a regular language L_{adv} (resp. its minimal DFA A_{adv}). Here we consider A_{adv} with any number of states to be a 'reasonable' advice since regular languages are 'simpler' than deterministic context-free languages formalising the problem here. We first show that no suitable complexity measure for DPDA combining the number of states and stack symbols exists. Next we prove tight bounds on the state complexity of an infinite sequence of deterministic context-free languages and show that no supplementary regular information can decrease the state complexity of the DPDAs recognizing them. We also exhibited an infinite sequence of deterministic context-free languages for which a suitable supplementary regular information enables a construction of a significantly simpler DPDA.

Our next exploration in this direction [13] was into the way the advice information is given. So far it was in a form to be directly used. This is not always

the case in real life situations. One may need to apply some transformation (e.g., into a foreign language or a different encoding) in order to use the advice. We proposed a framework for studying the possibility to transform the instance of a problem to match the format of the advisory information. We stay in the deterministic finite automata setting, i.e., the problem is a regular language L, the solution is a DFA A for L, and the complexity measure is the number of states of A. However, the advice information is given by a pair (L_{adv}, M) where M is a mapping and L_{adv} is an advisory language. The 'advice' is that the input word mapped by M belongs to L_{adv}. The mapping M is given by a finite state device (a-transducer or sequential transducer - see [5] for definitions). We made our reasonability requirement stricter, i.e., we consider the advice useful if all of the three finite state devices – A_{new}, A_{adv}, and M – have together fewer states than A. We have shown that nondeterminism of a-transducers can absorb too much of the (deterministic) complexity of the problem and concentrated on deterministic sequential transducers. We were again able to show the existence of an infinite sequence of regular languages for which no useful information simplifying their solution exists and another infinite sequence of regular languages, for which useful information simplifying their solution exists.

There is another aspect of information commonly used informally. We often say that information is 'actual' or 'received in the right time'. In [11] we introduced a framework enabling us to define and study *time criticality* of information (advice). It is based on the notion of usefulness discussed above. Once again we use the deterministic finite automata setting. We formalise time criticality using the following scenario. Given a problem L and its solution A and a particular instance w we obtain information about w at some point during the computation of A on w. This information (advice) A_{adv} might enable us to use a different (possibly simpler) automaton A_{new} to finish the computation. This automaton may be more or less simple depending on the time the advice is received. Note that we may have different A_{new} automaton depending on the time t the advice is received but we prefer not to overload the notation in this general description. We define time criticality (of L_{adv} given A and w) as the function of time, measuring the usefulness of this information via a possible decrease in the number of states needed to finish the computation. We define time criticality as the function giving the ratio of the number of states we can spare by obtaining the information at time t to the number of states required without the advice information. One might expect that the usefulness of advice decreases with time. However, we were able to show, that time criticality may have complex behaviour (in fact having any number of local maxima). We also considered a dynamic version of time criticality, where the advice concerned the remaining part of the input, obtaining similar results.

In [9] we departed slightly from our basic scenario. Inspired by the reoptimization approach (see, e.g., [1]) we embarked to develop a framework where the information, advice, received does not concern the input word itself but some 'close enough' or 'similar' word. To be able to process the additional information, or advice, about a similar word, the standard model of deterministic finite

automaton is augmented by an advice tape. There is a wide variety of possibilities how to specify similarity of words and especially, what is 'admissible' information. We explored some of them, many remain unexplored. We analysed the 'strength' of some of these scenarios by comparing and characterising the corresponding language families. However our main interest was not in the computational power of the new device. In line with our previous research we concentrated on regular languages and the possibility to reduce the complexity (measured again by the number of states) of their acceptance using the advice information on a similar word. We exhibited families of regular languages where the information provided by the given advice schema was useful and where no useful information could be found.

4 Usefulness and Nondeterministic Finite Automata

In this section we present new results of our exploration of the notion of usefulness of information in the nondeterministic finite state automata setting. Similar to the deterministic case studied in [4] the problem L and the advice L_{adv} are regular languages. However a solution to L is given by a nondeterministic finite state automaton A. Similarly A_{adv} for L_{adv} and a new solution utilising the advice A_{new} are nondeterministic finite state automata. We keep the number of states as a complexity measure used. We also keep our requirement of 'reasonability' of advice, i.e., for information to be useful we require both A_{adv} and A_{new} to be smaller than A.

We shall address two questions in this section. In Subsect. 4.1 we shall look for problems (regular languages) for which there exists useful information (advice) that simplifies their solution. On the other hand we shall also look for problems, where no useful information simplifying their solution exists. In Subsect. 4.2 we shall explore differences arising between the deterministic and nondeterministic finite state automata settings.

We shall again (just like in [4]) reformulate our problems in terms of parallel decompositions of NFAs. Given an NFA A we say that two NFAs A_1, A_2 such that $L(A) = L(A_1) \cap L(A_2)$ form a *parallel decomposition of the automaton* A. If it holds that $\#_S(A_1) < \#_S(A)$ and $\#_S(A_2) < \#_S(A)$, we call this decomposition *nontrivial*. If there exists nontrivial parallel decomposition of an NFA A we call A *decomposable*. We say that a regular language L is *nondeterministically decomposable* if a minimal NFA A for L is decomposable. It can be easily shown, that the decomposability of a language is well-defined and does not depend upon the choice of a minimal NFA for the language. In what follows we shall say just *decomposable* instead of nondeterministically decomposable when it is clear from the context.

Thus we can reformulate our question about existence of some useful advice for a given problem L to the question about decomposability of L. We shall use this formulation in the rest of this section.

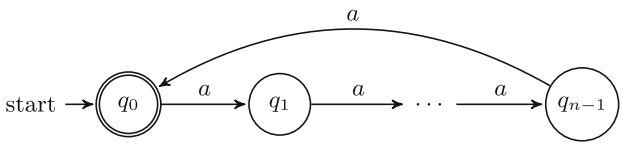

Fig. 1. The automaton A_n

4.1 Decomposability of Regular Languages

In this section we consider two infinite subfamilies of regular languages. We give a necessary and sufficient conditions for a language to be decomposable, i.e., we exhibit infinite number of problems – regular languages – for which a useful advice reducing the complexity of their solution exists and an infinite number of problems for which no such advice exists.

Note 1. There are some regular languages for which it is easy to see that they are undecomposable, e.g., a regular language L with $nsc(L) \leq 2$ is clearly undecomposable, since one-state NFAs can accept only one of the languages $\emptyset, \{\varepsilon\}$, or Σ^*.

First, let us consider a family of regular languages of the form $\{a^{kn} \mid k \in \mathbb{N}\}$ for a given $n \in \mathbb{N}$.

Theorem 2. *Let $n \in \mathbb{N}$, $L_n = \{a^{kn} \mid k \in \mathbb{N}\}$. The language L_n is decomposable if and only if n is not a power of a prime.*

Proof. First we shall show that $nsc(L_n) = n$, for all n. We construct an NFA A_n such that $L(A_n) = L_n$. We define A_n by the transition diagram in Fig. 1.

It is easy to see that $L(A_n) = L_n$. Thus $nsc(L_n) \leq \#_S(A_n) = n$. Using the fooling set technique of Theorem 1 and the set of pairs of words $M_n = \{(a^i, a^{n-i}) \mid 0 \leq i \leq n-1\}$ of size $|M_n| = n$ we can see that $nsc(L_n) \geq n$. Thus $nsc(L_n) = n$ and A_n is a minimal NFA for L_n.

(i) We shall now prove that if n is a power of a prime then the language L_n is undecomposable. Let $n = p^m$ for some prime p and $m \in \mathbb{N}$. From above we have $nsc(L_n) = p^m$.
Suppose that the language L_n is decomposable. Thus, there exists a nontrivial decomposition of the automaton A_n. Thus, there exist NFAs $A_1^{p^m}$ and $A_2^{p^m}$ such that it holds $\#_S(A_1^{p^m}) < p^m$, $\#_S(A_2^{p^m}) < p^m$, and $L(A_1^{p^m}) \cap L(A_2^{p^m}) = L_n$.
Consider the word a^{p^m} in $L(A_1^{p^m}) \cap L(A_2^{p^m})$. Let $(q_0, a^{p^m}) \vdash (q_1, a^{p^m-1}) \vdash \cdots \vdash (q_{p^m-1}, a) \vdash (q_{p^m}, \varepsilon)$, where q_0 is the initial state of $A_1^{p^m}$ and q_{p^m} is one of the accepting states of $A_1^{p^m}$, be an accepting computation of $A_1^{p^m}$ on a^{p^m}. Since $\#_S(A_1^{p^m}) < p^m$, there exist i and j in $\mathbb{N}$ such that $0 \leq i < j < p^m$ and $q_i = q_j$. It follows that we can pump part of the accepted word which is shorter than p^m, i.e., there exists $r_1 \in \mathbb{N}, 1 \leq r_1 < p^m$, such that for all k in

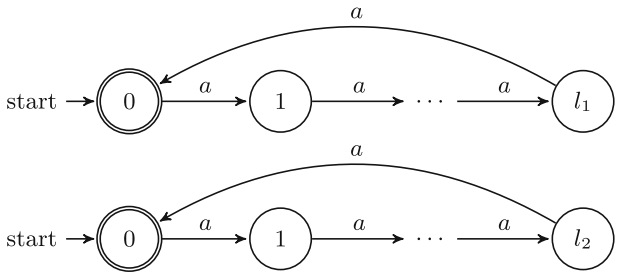

Fig. 2. Automata A_1^n and A_2^n forming a decomposition of A_n

$\mathbb{N}$ it holds $a^{p^m+kr_1}$ is in $L(A_1^{p^m})$. Similarly there exists $r_2 \in \mathbb{N}, 1 \leq r_2 < p^m$, such that for all k in $\mathbb{N}$ it holds $a^{p^m+kr_2}$ is in $L(A_2^{p^m})$.
The numbers r_1 and r_2 can be written as follows: $r_1 = p^{l_1}s_1$, where $0 \leq l_1 < m$ and s_1 is not divisible by p. Similarly $r_2 = p^{l_2}s_2$, where $0 \leq l_2 < m$ and s_2 is not divisible by p. Let $t = s_1 s_2 p^{max(l_1,l_2)}$. From above it follows that $a^{p^m+t} \in L(A_1^{p^m}) \cap L(A_2^{p^m})$. However, t is not divisible by p^m, thus, $a^{p^m+t} \notin L_n$ which is a contradiction to the assumption that the automata $A_1^{p^m}$ and $A_2^{p^m}$ form a nontrivial decomposition of the automaton A_n.

(ii) We shall now prove, that if n is not a power of prime, then the language L_n is decomposable. Let $p_1^{m_1}p_2^{m_2}\ldots p_r^{m_r}$ be the prime factorisation of n. Since n is not a power of prime $r \geq 2$ holds. Let us denote $l_1 = p_1^{m_1}$ and $l_2 = p_2^{m_2}\ldots p_r^{m_r}$. Let A_1^n and A_2^n be the NFA given by the transition diagrams in Fig. 2.
It is easy to see that these automata form a decomposition of A_n. [For, a word a^s belongs to $L(A_1^n) \cap L(A_2^n)$ iff s is divisible by both l_1 and l_2, i.e., iff s is divisible by $l_1 l_2$ which is iff s is divisible by $n = l_1 l_2$ and thus iff a^s belongs to $L_n = L(A_n)$.]
Since $\#_S(A_1^n) < \#_S(A_n)$ and $\#_S(A_2^n) < \#_S(A_n)$, this decomposition is nontrivial and the proof is complete. □

The second subfamily of regular languages considered is the family of singleton languages, i.e., languages consisting of one word only.

Theorem 3. *Let w be a word, let $L_w = \{w\}$. Then L_w is decomposable if and only if w contains at least two distinct symbols.*

Proof. First we shall show that $nsc(L_w) = |w| + 1$. Let $w = c_1 \ldots c_n$ where $c_1, \ldots, c_n$ are symbols. Let A_w be the automaton for L_w given by its transition diagram in Fig. 3. Thus $nsc(L_w) \leq \#_S(A_w) = |w|+1$. Using the fooling set technique of Theorem 1 and the set of pairs of words $F = \{(pref(w,i), suf(w,|w| - i)) \mid 0 \leq i \leq |w|\}$ of size $|w| + 1$ we can see that $nsc(L_n) \geq |w| + 1$. Thus $nsc(L_n) = |w| + 1$ and A_w is a minimal NFA for L_w.

(i) Suppose w does not contain two distinct symbols, i.e., $w = a^n$ for some $n \in \mathbb{N}$.

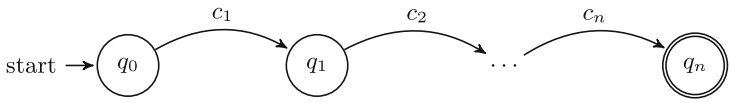

Fig. 3. The automaton A_w.

As mentioned in Note 1, L_w is undecomposable for $n = 0$ or $n = 1$. Consider $n \geq 2$. Suppose the language L_w is decomposable, i.e., there exists a nontrivial decomposition of the automaton A_w given by automata A_1^w and A_2^w.

Let $(p_0, a^n) \vdash (p_1, a^{n-1}) \vdash \ldots \vdash (p_{n-1}, a) \vdash (p_n, \varepsilon)$, where p_0 is the initial state of A_1^w and p_n is one of the accepting states of A_1^w, be an accepting computation of A_1^w on w. Since $\#_S(A_1^w) < |w| + 1$, there exist i and j in $\mathbb{N}$ such that $0 \leq i < j \leq n$ and $q_i = q_j$. Thus there exists r_1 in $\mathbb{N}, 1 \leq r_1 \leq n$, such that for all k in $\mathbb{N}$ we have a^{n+kr_1} in $L(A_1^w)$. Similarly there exists r_2 in $\mathbb{N}, 1 \leq r_2 \leq n$, such that for all k in $\mathbb{N}$ we have a^{n+kr_2} in $L(A_2^w)$. We thus have $a^{n+r_1r_2} \in L(A_1^w) \cap L(A_2^w) = L_w$ which is a contradiction.

(ii) Let $w = c_1 \ldots c_n$ contain at least two distinct symbols which we shall denote a and b. Hence we can write $w = c_1 \ldots c_{i-1} a b c_{i+2} \ldots c_n$. Thus the automaton A_w defined above can be redrawn as shown in Fig. 4.

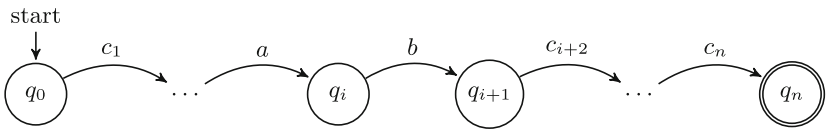

Fig. 4. The automaton A_w with two distinct symbols in w.

We shall construct a nontrivial decomposition of A_w. Automata A_w^a and A_w^b forming this decomposition are given by their transition diagrams in Fig. 5. Let $w_1 = c_1 \ldots c_{i-1}$ and $w_2 = c_{i+2} \ldots c_n$. Clearly $L(A_w^a) = \{w_1 a^k b w_2 \mid k \in \mathbb{N}\}$, $L(A_w^b) = \{w_1 a b^k w_2 \mid k \in \mathbb{N}\}$ and $L(A_w^a) \cap L(A_w^b) = \{w\}$. Since $\#_S(A_w^a) < \#_S(A_w)$ and $\#_S(A_w^b) < \#_S(A_w)$ the automata A_w^a and A_w^b form a nontrivial decomposition of A_w. □

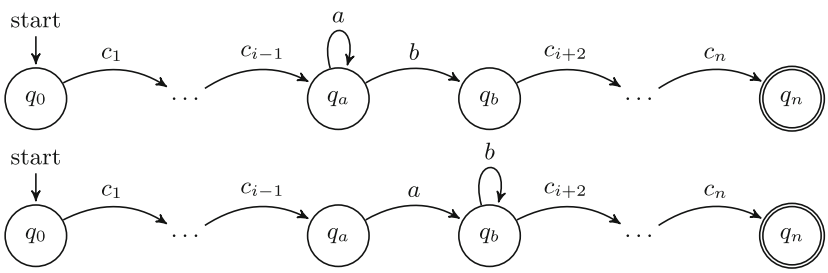

Fig. 5. A decomposition of A_w into automata A_w^a and A_w^b.

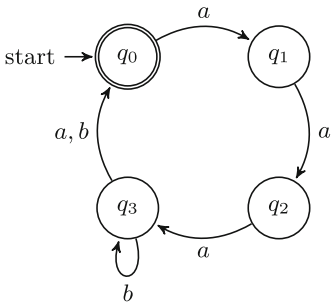

Fig. 6. The automaton A_4^N

4.2 Deterministic vs. Nondeterministic Decomposability

In this section we shall explore differences arising between the deterministic and nondeterministic finite state automata settings. We shall exhibit an infinite sequence of regular languages such that every language in that sequence is deterministically decomposable but nondeterministically undecomposable. Thus, for the problems in this sequence there is no useful information simplifying their solution in the nondeterministic setting but one can find useful advice to simplify the solution in the deterministic setting. Moreover, the advice in the deterministic setting helps substantially in the sense that for almost all decompositions in this sequence it holds that the size of both automata in the decomposition is about one half of the size of the original automaton.

Theorem 4. *There exists a sequence of regular languages* $(L_i)_{i=2}^{\infty}$ *such that the following holds:*

(a) The language L_i *is nondeterministically undecomposable but deterministically decomposable for every* $i \in \mathbb{N}, i \geq 2$.
(b) For each $i \in \mathbb{N}, i \geq 3$ *let* A_i *be the minimal DFA accepting* L_i. *Then there exists a decomposition of* A_i *into DFAs* A_1^i *and* A_2^i *such that* $\#_S(A_1^i) = \#_S(A_2^i) = \frac{\#_S(A_i)-1}{2} + 2$.

Proof. Let $(L'_i)_{i=2}^{\infty}$ be a sequence of languages defined by $L'_i = (\{a^{i-1}\}\{b\}^*\{a,b\})^*$ for every $i \geq 2$. We shall construct a sequence $(L_i)_{i=2}^{\infty}$ satisfying (a) and (b) of the theorem by selecting some (infinitely many) members of $(L'_i)_{i=2}^{\infty}$.

First, we shall show that for (infinitely many) indices i such that i is a power of prime the language L'_i is nondeterministically undecomposable. Let us construct NFAs A_i^N such that $L(A_i^N) = L'_i$. Let $A_i^N = (K_i^N, \{a,b\}, \delta_i^N, q_0, \{q_0\})$ where $K_i^N = \{q_j \mid 0 \leq j < i\}$ and the transition function δ_i^N is defined as follows - $\delta_i^N(q_{i-1}, b) = \{q_0, q_{i-1}\}$, $\forall j \in \mathbb{N}, 0 \leq j \leq i-1$: $\delta_i^N(q_j, a) = \{q_{(j+1) \bmod i}\}$. We illustrate this construction by showing the automaton A_4^N by its transition diagram in Fig. 6.

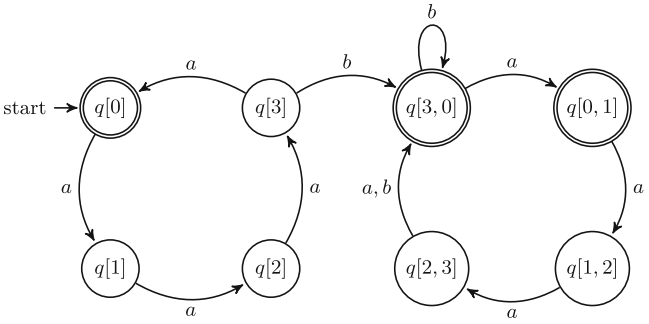

Fig. 7. The automaton A_4^D

Clearly $L(A_i^N) = L'_i$ for all i. Note that $\{a^{ki} \mid k \in \mathbb{N}\} \subset L'_i$. Using the fooling set technique of Theorem 1 with the set of pairs of words $M_i = \{(a^j, a^{i-j}) \mid 0 \leq j < i\}$ of size i we obtain $i \leq nsc(L'_i) \leq \#_S(A_i^N) = i$. Hence $nsc(L'_i) = i$ and the automaton A_i^N is a minimal NFA for the language L'_i.

We shall now show that for those i which are a power of prime the languages L'_i are nondeterministically undecomposable. Let us suppose that, to the contrary, there exists a nontrivial decomposition of A_i^N given by NFAs $A_1^{N,i}$ and $A_2^{N,i}$ with $\#_S(A_1^{N,i}) < i$, $\#_S(A_2^{N,i}) < i$, and $L(A_1^{N,i}) \cap L(A_2^{N,i}) = L'_i$ for each such i.

Since i is a power of a prime, $i = p^n$ for some prime p and n in $\mathbb{N}$. Since $a^{p^n} \in L(A_1^{N,i})$ and $a^{p^n} \in L(A_2^{N,i})$ there exist accepting computations of both $A_1^{N,i}$ and $A_2^{N,i}$ on the word a^{p^n}. Using an analogous argumentation as in part (i) of the proof of Theorem 2 one can find a word that belongs to $L(A_1^{N,i}) \cap L(A_2^{N,i})$ but does not belong to L'_i which is a contradiction.

We shall now show that L'_i is deterministically decomposable for arbitrary $i \geq 2$. Let the DFA A_i^D accepting L'_i be obtained from NFA A_i^N using the standard subset construction. Thus we define A_i^D as follows:
$A_i^D = (K_i^D \cup \{q_T\}, \{a, b\}, \delta_i^D, q[0], F_i^D)$, where $F_i^D = \{q[0], q[i-1, 0], q[0, 1]\}$, $K_i^D = \{q[j], q[j, (j+1) \bmod i] \mid 0 \leq j < i\}$ and the transition function δ_i^D is defined in an obvious way. For illustration we exhibit A_4^D by its transition diagram in Fig. 7. For simplicity we omit the trash state q_T in the diagram.

One can show minimality of A_i^D using standard techniques (e.g., Myhill-Nerode theorem).

We shall now construct a nontrivial decomposition of the DFA A_i^D. The main idea of the decomposition is, that the automata of the decomposition will not contain both cycles, which are present in A_i^D, but one of them will be 'collapsed' to one state. When considering words accepted by both automata, they will have each of their 'two parts' checked by one of the two automata thus guaranteeing they have the proper form for L'_i. The decomposition will work properly thanks to the fact that both cycles in A_i^D are separated by just one transition on b from the first cycle to the second one while we do not use transitions on b in the first

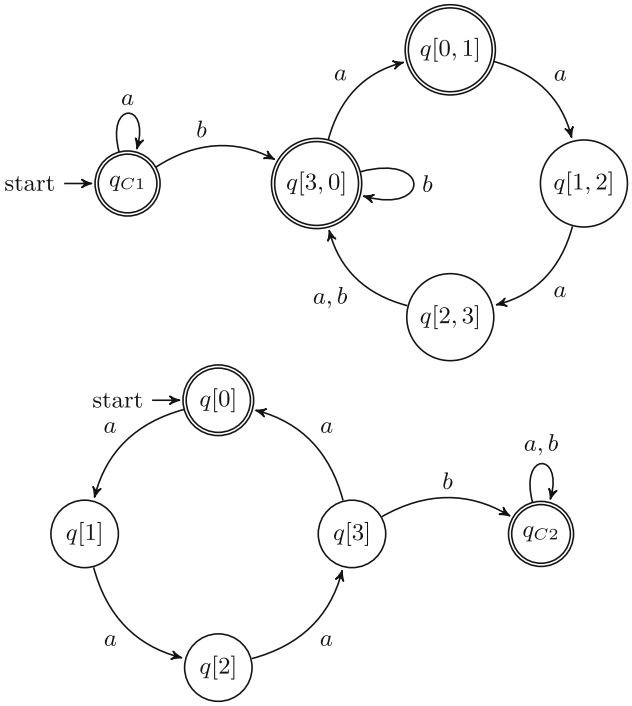

Fig. 8. A decomposition of A_4^D to $A_1^{D,4}$ and $A_2^{D,4}$

cycle. We denote the automata in the decomposition by $A_1^{D,i}$ and $A_2^{D,i}$. We shall illustrate the construction by presenting the decomposition of the automaton A_4^D to automata $A_1^{D,4}$ and $A_2^{D,4}$ by transition diagrams in Fig. 8. For simplicity we omit trash state q_T in the diagrams.

Formally, let us define the automata $A_1^{D,i}$ and $A_2^{D,i}$ as follows:

1. $A_1^{D,i} = (K_1^{D,i} \cup \{q_{C1}, q_T\}, \{a, b\}, \delta_1^{D,i}, q_{C1}, F_1^{D,i})$, where $K_1^{D,i} = \{q[j, (j + 1) \bmod i] \mid 0 \leq j < i\}$, $F_1^{D,i} = \{q_{C1}, q[i-1, 0], q[0, 1]\}$ where the transition function $\delta_1^{D,i}$ is defined as follows: $\delta_1^{D,i}(q_{C1}, a) = q_{C1}$, $\delta_1^{D,i}(q_{C1}, b) = q[i - 1, 0]$, $\delta_1^{D,i}(q[i-1, 0], b) = q[i-1, 0]$, $\delta_1^{D,i}(q[i-2, i-1], b) = q[i-1, 0]$, $\forall q[j, k] \in K_1^{D,i}$: $\delta_1^{D,i}(q[j, k], a) = q[(j + 1) \bmod i, (k + 1) \bmod i]$. All transitions not explicitly mentioned are defined to lead to the trash state q_T.
2. $A_2^{D,i} = (K_2^{D,i} \cup \{q_{C2}, q_T\}, \{a, b\}, \delta_2^{D,i}, q[0], F_2^{D,i})$, where $K_2^{D,i} = \{q[j] \mid 0 \leq j < i\}$, $F_2^{D,i} = \{q[0], q_{C2}\}$ where the transition function $\delta_2^{D,i}$ is defined as follows: $\delta_2^{D,i}(q[i-1], b) = q_{C2}$, $\delta_2^{D,i}(q_{C2}, a) = q_{C2}$, $\delta_2^{D,i}(q_{C2}, b) = q_{C2}, \forall q[j] \in K_2^{D,i}$: $\delta_2^{D,i}(q[j], a) = q[(j+1) \bmod i]$. All transitions not explicitly mentioned are defined to lead to the trash state q_T.

We show that $L(A_i^D) = L(A_1^{D,i}) \cap L(A_2^{D,i})$. Since the inclusion $L(A_i^D) \subseteq L(A_1^{D,i}) \cap L(A_2^{D,i})$ follows directly from the construction we shall concentrate on the reverse inclusion.

Let w be in $L(A_1^{D,i}) \cap L(A_2^{D,i})$. There has to be an accepting computation on w in both automata. Thus there exist states $q_{F1} \in F_1^{D,i}, q_{F2} \in F_2^{D,i}$ so that $(q_{C1}, w) \vdash^*_{A_1^{D,i}} (q_{F1}, \varepsilon), (q[0], w) \vdash^*_{A_2^{D,i}} (q_{F2}, \varepsilon)$. We shall consider two cases based on the possibilities for q_{F1}.

(i) $q_{F1} = q_{C1}$.
It follows from the construction of $A_1^{D,i}$ that the computation $(q_{C1}, w) \vdash^*_{A_1^{D,i}} (q_{F1}, \varepsilon)$ uses only state q_{C1}. Therefore there exists $n \in \mathbb{N}$ such that $w = a^n$. It follows from the construction of $A_2^{D,i}$ that the accepting computation $(q[0], w) \vdash^*_{A_2^{D,i}} (q_{F2}, \varepsilon)$ of $A_2^{D,i}$ on w uses only the states in $\{q[j] \mid 0 \leq j < i\}$. Thus $q_{F2} = q[0]$ and the computation $(q[0], w) \vdash^*_{A_i^D} (q[0], \varepsilon)$ is an accepting computation of A_i^D on the word w. Informally, the automaton A_i^D uses only its first cycle, which is completely contained also in $A_2^{D,i}$.

(ii) $q_{F1} \in \{q[i-1, 0], q[0, 1]\}$.
It follows from the construction of $A_1^{D,i}$ that there exist some $n \in \mathbb{N}$ and $u \in \{a, b\}^*$ so that $w = a^n bu$. The computation of $A_1^{D,i}$ on the word w then looks as follows: $(q_{C1}, a^n bu) \vdash^*_{A_1^{D,i}} (q_{C1}, bu) \vdash_{A_1^{D,i}} (q[i-1, 0], u) \vdash^*_{A_1^{D,i}} (q_{F1,\varepsilon})$. Since w contains the symbol b it follows from the construction of $A_2^{D,i}$ that $q_{F2} = q_{C2}$ and the computation of the automaton $A_2^{D,i}$ on the word w looks as follows: $(q[0], a^n bu) \vdash^*_{A_2^{D,i}} (q[i-1], bu) \vdash_{A_2^{D,i}} (q_{C2}, u) \vdash^*_{A_2^{D,i}} (q_{C2}, \varepsilon)$. It follows from the construction of $A_1^{D,i}$ that the computation $(q[i-1, 0], u) \vdash^*_{A_1^{D,i}} (q_{F1}, \varepsilon)$ in $A_1^{D,i}$ is also a computation $(q[i-1, 0], u) \vdash^*_{A_i^D} (q_{F1}, \varepsilon)$ in A_i^D. It follows from the construction of $A_2^{D,i}$ that the computation $(q[0], a^n bu) \vdash^*_{A_2^{D,i}} (q[i-1], bu)$ in $A_2^{D,i}$ is also a computation $(q[0], a^n bu) \vdash^*_{A_i^D} (q[i-1], bu)$ in A_i^D. Moreover, it holds $\delta_i^D(q[i-1], b) = q[i-1, 0]$. Therefore we have $(q[0], a^n bu) \vdash^*_{A_i^D} (q[i-1], bu) \vdash_{A_i^D} (q[i-1, 0], u) \vdash^*_{A_i^D} (q_{F1,\varepsilon})$. Since $q_{F1} \in F_i^D$, this computation is an accepting computation in the automaton A_i^D on the word w. Informally, both automata in the decomposition check part of the input in one of the cycles of the original automaton and just wait in the other. So we have the input checked by both cycles in the intersection.

Thus in all possible cases for q_{F1} we have w in $L(A_i^D)$ and the proof is complete.

Clearly $\#_S(A_1^{D,i}) < \#_S(A_i^D), \#_S(A_2^{D,i}) < \#_S(A_i^D)$, so that automata $A_1^{D,i}$ and $A_2^{D,i}$ form a nontrivial decomposition of the automaton A_i^D. Therefore L'_i is deterministically decomposable for arbitrary $i \geq 2$.

Let us summarize what we have proven. For the sequence of languages $(L'_i)_{i=2}^{\infty}$ it holds that it contains infinitely many languages, which are nondeterministically undecomposable but deterministically decomposable. These languages are

those L'_i for which i is a power a prime. Thus, we obtain $(L_i)_{i=2}^{\infty}$ as a subsequence of $(L'_i)_{i=2}^{\infty}$ by selecting those L'_i for which i is a power of a prime. Clearly $(L_i)_{i=2}^{\infty}$ satisfies (a). Moreover, from the construction of automata $A_1^{D,i}$ and $A_2^{D,i}$ in the deterministic decomposition of A_i^D it follows that (b) of the statement of the theorem holds as well. □

5 Conclusion

We have shown that a formal framework can be defined to formalise and study some informally used attributes of information. Our framework opens many possible avenues to explore. Varying possible instances of (i) to (iv) in our setting should provide more insight and may bring us closer to understanding the notion of information. Formalizing additional aspects of information may lead to extensions of our framework.

References

1. Böckenhauer, H.-J., Hromkovič, J., Mömke, T., Widmayer, P.: On the hardness of reoptimization. In: Geffert, V., Karhumäki, J., Bertoni, A., Preneel, B., Návrat, P., Bieliková, M. (eds.) SOFSEM 2008. LNCS, vol. 4910, pp. 50–65. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-77566-9_5
2. Dobrev, S., Královič, R., Pardubská, D.: How much information about the future is needed? In: Geffert, V., Karhumäki, J., Bertoni, A., Preneel, B., Návrat, P., Bieliková, M. (eds.) SOFSEM 2008. LNCS, vol. 4910, pp. 247–258. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-77566-9_21
3. Even, S., Selman, A.L., Yacobi, Y.: The complexity of promise problems with applications to public-key cryptography. Inf. Control **61**(2), 159–173 (1984)
4. Gaži, P., Rovan, B.: Assisted problem solving and decompositions of finite automata. In: Geffert, V., Karhumäki, J., Bertoni, A., Preneel, B., Návrat, P., Bieliková, M. (eds.) SOFSEM 2008. LNCS, vol. 4910, pp. 292–303. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-77566-9_25
5. Ginsburg, S.: Algebraic and Automata-Theoretic Properties of Formal Languages. Elsevier Science Inc., New York (1975)
6. Glaister, I., Shallit, J.: A lower bound technique for the size of nondeterministic finite automata. Inf. Process. Lett. **59**(2), 75–77 (1996)
7. Hopcroft, J.E., Motwani, R., Ullman, J.D.: Introduction to Automata Theory, Languages, and Computation, 3rd edn. Addison-Wesley Longman Publishing Co., Inc., Boston (2006)
8. Hromkovič, J., Královič, R., Královič, R.: Information complexity of online problems. In: Hliněný, P., Kučera, A. (eds.) MFCS 2010. LNCS, vol. 6281, pp. 24–36. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-15155-2_3
9. Kováč, I.: Supplementary information and complexity of finite automata. Ph.D. thesis, Comenius University (2015)
10. Labath, P., Rovan, B.: Simplifying DPDA using supplementary information. In: Dediu, A.-H., Inenaga, S., Martín-Vide, C. (eds.) LATA 2011. LNCS, vol. 6638, pp. 342–353. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-21254-3_27

11. Rovan, B., Zeman, M.: Modeling time criticality of information. Inf. Process. Lett. **114**(3), 147–151 (2014)
12. Steskal, L.: On usefulness of information: a computational approach. Ph.D. thesis, Comenius University (2010)
13. Vida, B.: Using transformation in solving problems with supplementary information. Master thesis, Comenius University (2015)