Finite Automata, Digraph Connectivity, and Regular Expression Size
(Extended Abstract)

Hermann Gruber[1] and Markus Holzer[2]

[1] Institut für Informatik, Ludwig-Maximilians-Universität München,
Oettingenstraße 67, D-80538 München, Germany
gruberh@tcs.ifi.lmu.de
[2] Institut für Informatik, Technische Universität München,
Boltzmannstraße 3, D-85748 Garching bei München, Germany
holzer@in.tum.de

Abstract. Recently lower bounds on the minimum required size for the conversion of deterministic finite automata into regular expressions and on the required size of regular expressions resulting from applying some basic language operations on them, were given by Gelade and Neven [8]. We strengthen and extend these results, obtaining lower bounds that are in part optimal, and, notably, the presented examples are over a binary alphabet, which is best possible. To this end, we develop a different, more versatile lower bound technique that is based on the star height of regular languages. It is known that for a restricted class of regular languages, the star height can be determined from the digraph underlying the transition structure of the minimal finite automaton accepting that language. In this way, star height is tied to cycle rank, a structural complexity measure for digraphs proposed by Eggan and Büchi, which measures the degree of connectivity of directed graphs.

1 Introduction

One of the most basic theorems in formal language theory is that every regular expression can be effectively converted into an equivalent finite automaton, and *vice versa* [16]. While algorithms accomplishing these tasks have been known for a long time, there has been a renewed interest in these classical problems during the last few years. For instance, new algorithms for converting regular expressions into finite automata outperforming classical algorithms have been found only recently, as well as a matching lower bound of $\Omega(n \cdot \log^2 n)$ on the number of transitions required by any equivalent nondeterministic finite automaton (NFA). The lower bound is, however, only attained for growing alphabet size, and a better algorithm is known for constant alphabet size, see [26] for the current state of the art.

In contrast, much less is known about the converse direction, namely of converting finite automata into regular expressions. Apart from the fundamental

L. Aceto et al. (Eds.): ICALP 2008, Part II, LNCS 5126, pp. 39–50, 2008.

Table 1. Comparing the lower bound results for conversion problems of deterministic finite automata (DFA) and regular expressions (RE), where $\cap$ denotes intersection, $\neg$ complementation, and ш the shuffle operation on formal languages. Entries with a bound in $\Theta(\cdot)$ indicate that the result is best possible, i.e., refers to a lower bound matching a known upper bound.

Conversion	Gelade and Neven [8]	this paper with $\lvert\Sigma\rvert = 2$
planar DFA to RE[1]	—	$2^{\Theta(\sqrt{n})}$ [Thm. 11]
DFA to RE	$2^{\Omega(\sqrt{n/\log n})}$ for $\lvert\Sigma\rvert = 4$	$2^{\Theta(n)}$ [Thm. 16]
RE $\cap$ RE to RE	$2^{\Omega(\sqrt{n})}$ for $\lvert\Sigma\rvert = O(n)$	$2^{\Omega(n)}$ [Cor. 8]
RE ш RE to RE	—	$2^{\Omega(n)}$ [Cor. 9]
$\neg$ RE to RE	$2^{2^{\Omega(n)}}$ for $\lvert\Sigma\rvert = 4$	$2^{2^{\Omega(\sqrt{n \log n})}}$ [Thm. 10]

nature of the problem, some applications lie in control flow normalization, including uses in software engineering such as automatic translation of legacy code [20]. All known algorithms covering the general case of infinite languages are based on the classical ones, which are compared in the survey [25]. The drawback is that all of these (structurally similar) algorithms return expressions of size $2^{O(n)}$ in the worst case, and Ehrenfeucht and Zeiger exhibit a family of languages over an alphabet of size n^2 for which this exponential blow-up is inevitable [6]. These examples naturally raise the question whether a size blow-up of $2^{\Omega(n)}$ can also occur for constant alphabet size, a question posed in [7]. One of the main results in this paper is a positive answer to this question, even in the case of a binary alphabet; note that the conversion problem becomes polynomial for unary languages [7]. Currently, there are not many lower bound techniques for regular expression size. A notable exception is the technique used in the above mentioned work [6], which however requires, in its original version, a largely growing alphabet. Recently, a variation of Ehrenfeucht and Zeiger's method was used in [8] to get similar but weaker lower bounds on the conversion problem for small alphabets. The above mentioned question, however, was left open. A technique based on communication complexity that applies only for finite languages, is proposed in [10]. They give an optimal bound of $n^{\Theta(\log n)}$ for the conversion problem in the case of finite languages.

Independently of [8], we take a different direction, by relating the descriptional complexity of regular languages (alphabetic width) to their structural complexity (star height). The star height is a structural complexity measure of regular languages that has been intensively studied in the literature for more than 40 years, see [11,15] for a recent treatment. Determining the star height can be in some cases reduced to the easier task of determining the cycle rank of a certain digraph. The latter concept is related to the cycle rank of digraphs, a digraph connectivity measure defined by Eggan and Büchi [5] in the 1960s. Since measuring the connectivity of digraphs is a very active research area, see,

[1] The lower bound result on the conversion of planar deterministic finite automata to regular expressions holds for $\lvert\Sigma\rvert = 4$.

e.g., [1,2,14,22], and as we feel that cycle rank is a interesting concept in its own right, we summarize and further develop the theory of cycle rank. For a more thorough treatment, including all proofs and comparison to some other recently proposed measures we refer to [9]. These connections turn out to be fruitful, allowing not only for proving a tight lower bound on the problem of converting finite automata into regular expressions, but also for giving reasonably good lower bounds for the alphabetic width of some basic regular language operations, namely intersection, complement, and shuffle. In this way, we independently improve on and extend the recently obtained results in [8]—we summarize and compare the obtained results in Table 1.

2 Basic Definitions

We introduce some basic notions in formal language and automata theory—for a thorough treatment, the reader might want to consult a textbook such as [12]. In particular, let Σ be a finite alphabet and Σ^* the set of all words over the alphabet Σ, including the empty word ε. The length of a word w is denoted by $|w|$, where $|\varepsilon| = 0$. A *(formal) language* over the alphabet Σ is a subset of Σ^*.

The *regular expressions* over an alphabet Σ are defined recursively in the usual way:[2] $\emptyset$, ε, and every letter a with $a \in \Sigma$ is a regular expression; and when r_1 and r_2 are regular expressions, then $(r_1 + r_2)$, $(r_1 \cdot r_2)$, and $(r_1)^*$ are also regular expressions. The language defined by a regular expression r, denoted by $L(r)$, is defined as follows: $L(\emptyset) = \emptyset$, $L(\varepsilon) = \{\varepsilon\}$, $L(a) = \{a\}$, $L(r_1 + r_2) = L(r_1) \cup L(r_2)$, $L(r_1 \cdot r_2) = L(r_1) \cdot L(r_2)$, and $L(r_1^*) = L(r_1)^*$. The *size* or *alphabetic width* of a regular expression r over the alphabet Σ, denoted by $\mathrm{alph}(r)$, is defined as the total number of occurrences of letters of Σ in r. For a regular language L, we define its alphabetic width, $\mathrm{alph}(L)$, as the minimum alphabetic width among all regular expressions describing L.

It is well known that regular expressions and finite automata are equally powerful, i.e., for every regular expression one can construct an equivalent (deterministic) finite automaton and *vice versa*. Finite automata are defined as follows: A *nondeterministic finite automaton* (NFA) is a 5-tuple $A = (Q, \Sigma, \delta, q_0, F)$, where Q is a finite set of states, Σ is a finite set of input symbols, $\delta : Q \times \Sigma \to 2^Q$ is the transition function, $q_0 \in Q$ is the initial state, and $F \subseteq Q$ is the set of accepting states. The *language accepted* by the finite automaton A is defined as $L(A) = \{\, w \in \Sigma^* \mid \delta(q_0, w) \cap F \neq \emptyset \,\}$, where δ is naturally extended to a function $Q \times \Sigma^* \to 2^Q$. A nondeterministic finite automaton $A = (Q, \Sigma, \delta, Q_0, F)$ is *deterministic*, for short a DFA, if $|\delta(q, a)| \leq 1$, for every $q \in Q$ and $a \in \Sigma$. In this case we simply write $\delta(q, a) = p$ instead of $\delta(q, a) = \{p\}$. Two (deterministic or nondeterministic) finite automata are *equivalent* if they accept the same language.

[2] For convenience, parentheses in regular expressions are sometimes omitted and the concatenation is simply written as juxtaposition. The priority of operators is specified in the usual fashion: concatenation is performed before union, and star before both product and union.

In the remainder of this section we fix some basic notions from graph theory. A *directed graph*, or *digraph*, $G = (V, E)$ consists of a finite set of vertices V with an associated set of edges $E \subseteq V \times V$. An edge whose start and end vertex are identical is called a *loop*. If G has no loops, then G is called *loop-free*. If the edge relation of G is symmetric, then G is an *undirected graph*, or simply *graph*. It is often convenient to view the set of edges of an undirected graph as a set of unordered pairs $\{u, v\}$, with u and v in V. Only if there is no risk of confusion, for an undirected graph G, we refer to the set $\{\, \{u, v\} \mid (u, v) \in E \,\}$ as the set of edges of G, and, abusing notation, denote it by E. A digraph $H = (U, F)$ is a *subdigraph*, or simply *subgraph*, of a digraph $G = (V, E)$, if $U \subseteq V$ and for each edge $(u, v) \in F$ with $u, v \in U$, the pair (u, v) is an edge in E. A subgraph H is called *induced*, if furthermore for each edge $(u, v) \in E$ with $u, v \in U$, the pair (u, v) is also an edge in F. In the latter case, H is referred to as the subgraph of G induced by U, and denoted by $G[U]$. When removing a set of vertices U, or a single vertex u, from G, it is often handy to write $G - U$ and $G - u$ to denote the induced subgraphs $G[V \setminus U]$ and $G[V \setminus \{u\}]$, respectively.

We recall the definitions of some other important concepts related to walks and reachability. A subgraph $H = (U, F)$ of G is *strongly connected*, if for every vertices u and v, both u is reachable from v and v is reachable from u. A strongly connected subgraph H is called *nontrivial* if H has at least one edge, otherwise it is called trivial. Note that every trivial strongly connected subgraph has at most one vertex, but if G is not loop-free, it also has nontrivial strongly connected subgraphs with only one vertex. A set of vertices $\emptyset \subset C \subseteq V$ is a strongly connected component if $G[C]$ is strongly connected, but for every proper superset $C' \supset C$, the induced subgraph $G[C']$ is not strongly connected.

3 Star Height of Regular Languages and Cycle Rank of Digraphs

3.1 Definitions and Early Results

For a regular expression r over Σ, the star height, denoted by $h(r)$, is a structural complexity measure inductively defined by: $h(\emptyset) = h(\varepsilon) = h(a) = 0$, $h(r_1 \cdot r_2) = h(r_1 + r_2) = \max(h(r_1), h(r_2))$, and $h(r_1^*) = 1 + h(r_1)$. The star height of a regular language L, denoted by $h(L)$, is then defined as the minimum star height among all regular expressions describing L. We will later establish a relation between star height and alphabetic width of regular languages. This relation will allow us to reduce the task of proving lower bounds on alphabetic width to the one of proving lower bounds on star height.

First, we call to attention a structural complexity measure for digraphs intimately related to the star height of regular languages, called the cycle rank, suggested by Eggan and Büchi in the course of investigating the star height of regular languages [5].

Definition 1. *The* cycle rank *of a directed graph $G = (V, E)$, denoted by $cr(G)$, is inductively defined as follows: (1) If G is acyclic, then $cr(G) = 0$. (2) If G is*

strongly connected, then $cr(G) = 1 + \min_{v \in V}\{cr(G - v)\}$*, where* $G - v$ *denotes the graph with the vertex set* $V \setminus \{v\}$ *and appropriately defined edge set. (3) If* G *is not strongly connected, then* $cr(G)$ *equals the maximum cycle rank among all strongly connected components of* G*.*

In the following, we will be sometimes concerned with the cycle rank of the digraph underlying the transition structure of finite automata, so for a given finite automaton A, let its cycle rank, denoted by $cr(A)$, be defined as the cycle rank of the underlying graph. The following relation between cycle rank of automata and star height of regular languages became known as Eggan's Theorem [5]:

Theorem 2 (Eggan's Theorem). *The star height of a regular language* L *equals the minimum cycle rank among all nondeterministic finite automata accepting* L*.*

The star height of a regular language appears to be a more difficult concept than alphabetic width, see, e.g., [11,15]. In light of this consideration, proving lower bounds on alphabetic width *via* lower bounds on star height appears to be trading a hard problem for an even harder one. But early research on the star height problem established a subclass of regular languages for which the star height is determined more easily, namely the family of bideterministic regular languages, which are defined as follows: A regular language L is *bideterministic* if there exists a deterministic finite automaton A with a single final state such that a deterministic finite automaton accepting the reversed language L^R is obtained from A by reverting the direction of each transition and exchanging the roles of the initial and final state. The star height of bideterministic languages was shown to be computable in [18], building on earlier work which was, however, published only later in [19]:

Theorem 3 (McNaughton's Theorem). *Let* L *be a bideterministic language, and let* A *be the minimal trim, i.e., without a dead state, deterministic finite automaton accepting* L*. Then* $h(L) = cr(A)$*.*

In fact, the minimality requirement in the above theorem is not needed, since every bideterministic finite automaton in which all states are useful is already a trim minimal deterministic finite automaton. Here, a state is useful if it is both reachable from the start state, and from which the final state is reachable from it.

In order to relate star height to alphabetic width, and to find lower bound techniques for the cycle rank, we study the latter concept in more detail. First, we establish a basic fact about cycle rank, which is used throughout the following sections. The second part of the following statement is found in [19, Theorem 2.4.], and the other part is established by an easy induction:

Lemma 4. *Let* $G = (V, E)$ *be a digraph and let* $U \subseteq V$*. Then we have the inequalities* $cr(G) - |U| \leq cr(G - U) \leq cr(G)$*, where* $G - U$ *denotes the graph with vertex set* $V \setminus U$ *and appropriately defined edge set.* □

3.2 Cycle Rank *Via* Cops and Robbers

The characterization of cycle rank in terms of some "game against the graph" was already suggested in [19]. We give a modern formulation in terms of a cops and robber game. This characterization provides a useful tool in proving lower bounds on the cycle rank of specific families of digraphs. Moreover, many other digraph connectivity measures proposed recently admit a characterization in terms of some cops and robber game; this allows to compare the cycle rank with these other measures.

The *cops and strong visible robber game*, defined in [14], is given as follows: Let $G = (V, E)$ be a digraph. Initially, the cops occupy some set of $X \subseteq V$ vertices, with $|X| \leq k$, and the robber is placed on some vertex $v \in V \setminus X$. At any time, some of the cops can reside outside the graph, say, in a helicopter. In each round, the cop player chooses the next location $X' \subseteq V$ for the cops. The stationary cops in $X \cap X'$ remain in their positions, while the others go to the helicopter and fly to their new position. During this, the robber player, knowing the cops' next position X' from wire-tapping the police radio, can run at great speed to any new position v', provided there is both a (possibly empty) directed path from v to v', and a (possibly empty) directed path back from v' to v in $G-(X \cap X')$, i.e., he has to avoid to run into a stationary cop, and to run along a path in and to stay in the same strongly connected component of the remaining graph induced by the non-blocked vertices. Afterwards, the helicopter lands the cops at their new positions, and the next round starts, with X' and v' taking over the roles of X and v, respectively. The cop player wins the game if the robber cannot move any more, and the robber player wins if the robber can escape indefinitely.

The *immutable cops* variant of the above game restricts the movements of the cops in the following way: Once a cop has been placed on some vertex of the graph, he has to stay there forever. The *hot-plate* variant of the game restricts the movements of the robber in that he has to move along a nontrivial path in each move—even if the path consists only of a self-loop. These games are robust in the sense that small variations of rules, such as letting the robber player begin, or allowing only the placement of one cop at a time, do not alter the number of required cops. Also note that at most one additional cop is needed if we drop the hot-plate restriction. The following theorem gives a characterization of the cycle rank in terms of such a game. Due to space constraints, the proof is omitted.

Theorem 5. *Let G be a digraph and $k \geq 0$. Then k cops have a winning strategy for the immutable cops and hot-plate strong visible robber game if and only if the cycle rank of G is at most k, i.e., $cr(G) \leq k$.* □

4 Lower Bounds on Regular Expression Size

Now we have developed enough tools to derive lower bounds on alphabetic width in terms of star height.

Theorem 6. *Let $L \subseteq \Sigma^*$ be a regular language. Then* $\mathrm{alph}(L) \geq 2^{\frac{1}{3}(h(L)-1)} - 1$.

Proof. Let r be a regular expression over Σ of alphabetic width $n = \mathrm{alph}(L)$. Then the construction given in [13] shows how to transform this expression into an equivalent nondeterministic finite automaton A with ε-transitions having at most $n+1$ states. It is not hard to see that the digraph underlying the transition structure of the constructed automaton has undirected treewidth at most 2. With a graph separator technique, we show the following claim:

> Let G be a digraph with n vertices and undirected treewidth at most k. Then $cr(G) \leq 1 + (k+1) \cdot \log n$.

We argue as follows: First, we lift some notions and results concerning graph separators known for undirected graphs (see, e.g., [21]), to the case of digraphs: Let $G = (V, E)$ be a digraph and let $U \subseteq V$ be a set of vertices. A set of vertices S is a *weak separator* for U if every strongly connected component of $G[U \setminus S]$ contains at most $\frac{1}{2}|U|$ vertices. For real numbers $0 \leq k \leq |V|$, let $s(G, k)$ denote the maximum of the size of the smallest weak separator for U, where the maximum is taken over all subsets U of size at most k of V. The *weak separator number* of G, denoted by $s(G)$, is defined as $s(G, |V|)$.

Next we prove the following relation: Let $G = (V, E)$ be a digraph with $n \geq 1$ vertices. Then

$$cr(G) \leq 1 + \sum_{0 \leq k \leq \log n - 1} s\left(G, \frac{n}{2^k}\right). \tag{1}$$

The proof proceeds by induction on n. In the case $n = 1$, we have $s(G) = 0$, and the sum in the statement of the lemma is empty, as desired. The induction step is as follows: By definition of weak separator number, G has a weak separator S of size at most $s(G, n)$. Let $C_1, C_2, \ldots, C_p$ be the strongly connected components of $G - S$. Each of these has cardinality at most $\frac{n}{2}$. With Lemma 4, we obtain

$$cr(G) \leq |S| + \max_{1 \leq i \leq p} cr(C_i) \leq s\left(G, \frac{n}{2^0}\right) + \max_{1 \leq i \leq p} cr(C_i).$$

Since for each $k \leq n$ and for each strongly connected component C_i obviously holds $s(C_i, k) \leq s(G, k)$, we have by induction hypothesis

$$\max_{1 \leq i \leq p} cr(C_i) \leq 1 + \sum_{0 \leq k \leq \log(n/2) - 1} s\left(G, \frac{n/2}{2^k}\right) = 1 + \sum_{1 \leq k \leq \log n - 1} s\left(G, \frac{n}{2^k}\right),$$

where the right hand side is obtained by simply shifting the summation index. By putting the two inequalities together, the proof of Inequality (1) is completed.

This establishes a relation between cycle rank and weak separator number, namely $cr(G) \leq 1 + s(G) \cdot \log n$, if G is a digraph with n vertices. Moreover, it is known from [24] that digraphs with undirected treewidth at most k have weak separator number at most $k + 1$, thus establishing our claim. Thus, we obtain $cr(A) \leq 1 + 3\log(n+1)$. Finally, the proof is completed by using Theorem 2. □

This bound is almost tight: Define the language L_n inductively by $L_0 = \varepsilon$ and $L_i = (a \cdot L_{i-1} \cdot b)^*$, for $i > 0$. Then $\mathrm{alph}(L_n)$ is clearly at most $2n$, but it is

known from [19] that $h(L_{2^k}) = k$, for each $k \geq 1$. In contrast, there cannot exist an upper bound on the alphabetic width in terms of star height, since all finite languages have star height 0, but there are only finitely many languages of bounded alphabetic width.

4.1 Lower Bounds on Alphabetic Width of Language Operations

As a first application of Theorem 6, we exhibit a family of languages over a binary alphabet that shows that several natural operations on regular languages such as complement, intersection and shuffle cannot be supported efficiently by regular expressions; most notably, complementation can require an almost doubly-exponential blow-up in regular expression size. These languages have an appealingly simple structure, and their star height was already studied, although not completely determined, in the very first paper on star height of regular languages [5].

Theorem 7. *For $m, n \in \mathbb{N}$, define $K_m = \{ w \in \{a,b\}^* \mid |w|_a \equiv 0 \mod m \}$ and $L_n = \{ w \in \{a,b\}^* \mid |w|_b \equiv 0 \mod n \}$. Then we have $h(K_m \cap L_n) = m$, if $m = n$, and $h(K_m \cap L_n) = \min(m,n) + 1$, otherwise.*

Proof. The stated upper bound on the star height is proved already in [5, Corollary 2, pp. 394f.], so it remains to show a matching lower bound. It is straightforward to construct deterministic finite automata with m (n, respectively) states describing the languages K_m and L_n, respectively. By applying the standard product construction on these automata, we obtain a deterministic finite automaton A accepting the language $K_m \cap L_n$. It is not hard to see that this automaton is a minimal trim deterministic finite automaton, and furthermore that it is bideterministic. Therefore Theorem 3 shows $h(K_m \cap L_n) = cr(A)$.

The digraph underlying automaton A is the directed discrete $(m \times n)$-torus arising from the Cartesian graph product of two directed cycles, whose entanglement was determined by similar means in [2]. We give a lower bound on the cycle rank of this digraph using the game characterization given in Theorem 5. By symmetry, assume the torus has m rows and n columns, with $m \leq n$. At any stage of the game, we call a row (column, respectively) *free*, if each of the vertices in the row (column, respectively) is neither yet occupied, nor announced to be occupied in the current move of the cops. In the kth move of the cops, there are at least $m - k$ free rows and $n - k$ free columns. As long as $k < m$, the robbers' strategy is to reside on the subgraph induced by the rows and columns that are currently free. For $k < m$, each free row or column is strongly connected itself, and each pair of free columns is strongly connected to each other *via* the (nonempty) set of free rows. The strategy always yields a valid game position, and this already shows the desired lower bound in the case $m = n$. In the case $m > n$, as soon as the last free row is threatening to be occupied, the robber can still flee to one of the remaining free columns. Thus an additional cop is needed, since each free column itself forms a nontrivial strongly connected subgraph, even though the columns are no longer strongly connected to each other. □

Together with Theorem 6, we immediately obtain some results about the alphabetic width of operations on regular languages. The classical way to extend the syntax of regular expressions is to allow intersection, thus obtaining the semi-extended regular expressions, or to allow also complement, resulting in extended regular expressions. It is known that semi-extended regular expressions can be exponentially more succinct even than nondeterministic finite automata, and hence than ordinary regular expressions. The former fact no longer holds if the number of occurrences of the intersection operator is bounded. But for regular expressions, already a single intersection operation can infer a huge blow-up in the needed description size:

Corollary 8. *For every $m \geq n$, there exist languages K_m and L_n over a binary alphabet with* $\mathrm{alph}(K_m) \leq m$ *and* $\mathrm{alph}(L_n) \leq n$, *such that* $\mathrm{alph}(K_m \cap L_n) = 2^{\Omega(n)}$. □

This improves a lower bound independently obtained in [8]. Another language operation is the shuffle of two languages, which naturally arises in modeling the interleaving of the action traces of two processes. The shuffle of two languages L_1 and L_2 over alphabet Σ is $\{ w \in \Sigma^* \mid w \in x \sqcup\!\sqcup y \text{ for some } x \in L_1 \text{ and } y \in L_2 \}$, where the shuffle of two words x and y is defined as the set of all words of the form $x_1 y_1 x_2 y_2 \ldots x_n y_n$, where $x = x_1 \ldots x_n$, $y = y_1 \ldots y_n$ with $x_i, y_i \in \Sigma^*$, for $1 \leq i \leq n$ and $n \geq 1$, and is denoted by $x \sqcup\!\sqcup y$. While the shuffle operation preserves regularity, it is known that regular expressions extended with the shuffle operator can be exponentially more succinct than regular expressions—in fact, the same holds for nondeterministic finite automata [17]. As with intersection, a similar blow-up can be caused already by a single application of the shuffle operator (which cannot be deduced from an argument solely based on automaton size). Namely, the language from Theorem 7 can be written as $(a^m)^* \sqcup\!\sqcup (b^n)^*$.

Corollary 9. *For every $m \geq n$, there exist languages L_m and L_n over a binary alphabet with* $\mathrm{alph}(K_m) \leq m$ *and* $\mathrm{alph}(L_n) \leq n$, *such that* $\mathrm{alph}(K_m \sqcup\!\sqcup L_n) = 2^{\Omega(n)}$. □

For numbers n that have many distinct prime factors, the language $\{a, b\}^* \setminus (K_n \cap L_n)$, where K_n and L_n are defined as in Theorem 7, can be expressed very succinctly by a regular expression using a kind of Chinese Remainder Representation. In this way, we obtain for the complementation operation a lower bound that is roughly doubly exponential, even for binary alphabets, thus complementing a result given in [8] for 4-symbol alphabets—the proof is omitted due to lack of space:

Theorem 10. *There exists an infinite family of languages L_n over a binary alphabet Σ with* $\mathrm{alph}(L_n) \leq n$, *such that* $\mathrm{alph}(\Sigma^* \setminus L_n) = 2^{2^{\Omega(\sqrt{n \log n})}}$. □

4.2 A Lower Bound for Converting DFAs into Regular Expressions

From the results in the previous chapter, it can be deduced that there are very simple examples of languages over a binary alphabet for which a blow-up in

size of $2^{\Omega(\sqrt{n})}$ is inevitable when converting from an n-state deterministic finite automaton to an equivalent regular expression. Next, we can show that this bound can even be reached for *planar* deterministic finite automata, first studied in [4], thus complementing a corresponding algorithmic result from [7] with an optimal lower bound—again, the proof has to be omitted, but we note that the transition structure of the witness DFA are undirected grid graphs.

Theorem 11. *For alphabet size $|\Sigma| \geq 4$, there is an infinite family of languages L_n over alphabet Σ acceptable by n-state planar deterministic finite automata, such that* $\mathrm{alph}(L_n) \geq 2^{\Omega(\sqrt{n})}$. □

The obvious question is now if a lower bound of $2^{\Omega(n)}$ can be reached over a constant alphabet, when starting with non-planar deterministic finite automata. The rest of this section is devoted to a proof of this fact.

By Theorem 5, the cycle rank of an undirected graph G, i.e., a symmetric digraph, can be described in terms of the immutable cops and strong visible robber game. Note that in this case every connected component of size at least two is also a nontrivial strongly connected component. The *greedy* strategy for the robber player is to choose in each step the largest connected component he can reach in the remaining graph. We will identify a class of graphs in which the greedy strategy is particularly successful, namely expander graphs.

Definition 12. *Let $G = (V, E)$ be an undirected graph. For a subset $U \subset V$, the boundary of U, denoted by δU, is defined as $\delta U = \{\, v \in V \setminus U \mid \{u, v\} \in E \text{ for some } u \in U \,\}$. An (undirected) d-regular graph $G = (V, E)$ with n vertices is called a (n, d, c)-expander, for $c > 0$, if each subset $U \subset V$ of vertices satisfies $|\delta U| > c \cdot |U|$, if $|U| < n/2$ and $|\delta U| \geq c \cdot (n - |U|)$, if $|U| \geq n/2$.*

A now standard probabilistic argument, originally from [23], shows that expander graphs are the rule rather than the exception among d-regular graphs, for all $d \geq 3$.

Theorem 13 (Pinsker). *There exists a fixed $c > 0$ such that for any $d \geq 3$ and even integer n, there is an (n, d, c)-expander, which is furthermore d-edge-colorable.*[3]

The proof of the following theorem is similar to that of [3, Theorem 4], where it was shown that each directed expander graph contains a long directed path.

Theorem 14. *Let G be a (n, d, c)-expander with $n \geq 3$. Then the cycle rank of G is at least $\frac{c}{d+1}(n-1)$, i.e., $cr(G) \geq \frac{c}{d+1}(n-1)$.* □

The next lemma shows that such a graph, equipped with an edge coloring, can be easily converted into a bideterministic finite automaton that accepts a language of large star height and uses only the edge colors as input alphabet.

[3] That is, one can assign to its edges d colors such that no pair of incident edges receives the same color.

Lemma 15. *For every d-edge colorable, connected undirected graph G with n vertices of cycle rank k, there exists an n-state deterministic finite automaton A over a d-symbol alphabet such that the star height of $L(A)$ is k.*

Proof. Let $G = (V, E)$ be such a graph, with $V = \{1, 2, \ldots, n\}$. and maximum degree d, equipped with an edge coloring $c : E \to \{0, 1, \ldots, d\}$ such that no pair of incident edges receives the same color. Given this colored graph, we construct a deterministic finite automaton over the alphabet $\Sigma = \{a_1, a_2, \ldots, a_d\}$ with state set V, start and single final state $v_0 \in V$ (arbitrary), and whose transition relation is defined as follows: $\delta(p, a_i) = q$ if the colored graph G has an i-colored edge $\{p, q\}$. It is not hard to see that this automaton is a trim bideterministic automaton, and therefore minimal. Furthermore, its underlying digraph is symmetric, and its undirected version is isomorphic to G. By Theorem 3, the star height of $L(A)$ equals k. □

For the main theorem of this section we need the existence of a suitable homomorphism that preserves star height. The existence of reasonably economic binary encodings with this property have been already conjectured in [5], and their existence was proved constructively in [19]: Let $\Sigma = \{a_1, a_2, \ldots, a_d\}$ be a finite alphabet, $d \geq 1$, and let $\varphi : \Sigma^* \to \{a, b\}^*$ be the homomorphism defined by $\varphi(a_i) = a^i b^{d-i+1}$, for $i = 1, 2, \ldots, d$. Then for every regular language $L \subseteq \Sigma^*$ the star height of L equals the star height of $\varphi(L)$. Then Lemma 15 and Theorems 6, 13, and 14 can be combined with the above presented star height preserving homomorphism to give the following theorem.

Theorem 16. *For alphabet size $|\Sigma| \geq 2$, there is an infinite family of languages L_n over alphabet Σ acceptable by deterministic finite automata with at most n states, such that* $\mathrm{alph}(L_n) = 2^{\Omega(n)}$. □

This gives an affirmative answer to "Open Problem 3" in [7], which asked whether such a family of languages exists, over some constant alphabet.

References

1. Berwanger, D., Dawar, A., Hunter, P., Kreutzer, S.: Dag-width and parity games. In: Durand, B., Thomas, W. (eds.) STACS 2006. LNCS, vol. 3884, pp. 524–536. Springer, Heidelberg (2006)
2. Berwanger, D., Grädel, E.: Entanglement—A measure for the complexity of directed graphs with applications to logic and games. In: Baader, F., Voronkov, A. (eds.) LPAR 2004. LNCS (LNAI), vol. 3452, pp. 209–223. Springer, Heidelberg (2005)
3. Björklund, A., Husfeldt, T., Khanna, S.: Approximating longest directed paths and cycles. In: Díaz, J., Karhumäki, J., Lepistö, A., Sannella, D. (eds.) ICALP 2004. LNCS, vol. 3142, pp. 222–233. Springer, Heidelberg (2004)
4. Book, R.V., Chandra, A.K.: Inherently nonplanar automata. Acta Informatica 6, 89–94 (1976)
5. Eggan, L.C.: Transition graphs and the star height of regular events. Michigan Mathematical Journal 10, 385–397 (1963)

6. Ehrenfeucht, A., Zeiger, H.P.: Complexity measures for regular expressions. Journal of Computer and System Sciences 12(2), 134–146 (1976)
7. Ellul, K., Krawetz, B., Shallit, J., Wang, M.: Regular expressions: New results and open problems. Journal of Automata, Languages and Combinatorics 10(4), 407–437 (2005)
8. Gelade, W., Neven, F.: Succinctness of the complement and intersection of regular expressions. In: Albers, S., Weil, P. (eds.) Symposium on Theoretical Aspects of Computer Science. Dagstuhl Seminar Proceedings, vol. 08001, pp. 325–336. IBFI (2008)
9. Gruber, H., Holzer, M.: Finite automata, digraph connectivity and regular expression size. Technical report, Technische Universität München (December 2007)
10. Gruber, H., Johannsen, J.: Optimal lower bounds on regular expression size using communication complexity. In: Amadio, R. (ed.) Foundations of Software Science and Computation Structures. LNCS, vol. 4962, pp. 273–286. Springer, Heidelberg (2008)
11. Hashiguchi, K.: Algorithms for determining relative star height and star height. Information and Computation 78(2), 124–169 (1988)
12. Hopcroft, J.E., Ullman, J.D.: Introduction to Automata Theory, Languages and Computation. Addison-Wesley, Reading (1979)
13. Ilie, L., Yu, S.: Follow automata. Information and Computation 186(1), 140–162 (2003)
14. Johnson, T., Robertson, N., Seymour, P.D., Thomas, R.: Directed tree-width. Journal of Combinatorial Theory, Series B 82(1), 138–154 (2001)
15. Kirsten, D.: Distance desert automata and the star height problem. RAIRO – Theoretical Informatics and Applications 39(3), 455–509 (2005)
16. Kleene, S.C.: Representation of events in nerve nets and finite automata. In: Shannon, C.E., McCarthy, J. (eds.) Automata Studies. Annals of Mathematics Studies, pp. 3–42. Princeton University Press, Princeton (1956)
17. Mayer, A.J., Stockmeyer, L.J.: Word problems – This time with interleaving. Information and Computation 115(2), 293–311 (1994)
18. McNaughton, R.: The loop complexity of pure-group events. Information and Control 11(1/2), 167–176 (1967)
19. McNaughton, R.: The loop complexity of regular events. Information Sciences 1, 305–328 (1969)
20. Morris, P.H., Gray, R.A., Filman, R.E.: Goto removal based on regular expressions. Journal of Software Maintenance 9(1), 47–66 (1997)
21. Nešetřil, J., de Mendez, P.O.: Tree-depth, subgraph coloring and homomorphism bounds. European Journal of Combinatorics 27(6), 1022–1041 (2006)
22. Obdržálek, J.: Dag-width: Connectivity measure for directed graphs. In: ACM-SIAM Symposium on Discrete Algorithms, pp. 814–821. ACM Press, New York (2006)
23. Pinsker, M.S.: On the complexity of a concentrator. In: Annual Teletraffic Conference, pp. 318/1–318/4 (1973)
24. Robertson, N., Seymour, P.D.: Graph minors. II. Algorithmic aspects of tree-width. Journal of Algorithms 7(3), 309–322 (1986)
25. Sakarovitch, J.: The language, the expression, and the (small) automaton. In: Farré, J., Litovsky, I., Schmitz, S. (eds.) CIAA 2005. LNCS, vol. 3845, pp. 15–30. Springer, Heidelberg (2006)
26. Schnitger, G.: Regular expressions and NFAs without ε-transitions. In: Durand, B., Thomas, W. (eds.) STACS 2006. LNCS, vol. 3884, pp. 432–443. Springer, Heidelberg (2006)