technische universität
dortmund

Master's Thesis

One-Pass Strategies for Context-Free Games

Christian Cöster

August 2015

Examiners:
Prof. Dr. Thomas Schwentick
Prof. Dr. Christoph Buchheim

Technische Universität Dortmund
Fakultät für Mathematik & Fakultät für Informatik

Dedicated to my grandfather

who is the source of my mathematical interests

whom I never had the honor to meet or talk to in person

Acknowledgments

I would like to thank Thomas Schwentick for his support in and beyond the supervision of this thesis. Further, I would like to thank Martin Schuster for our discussions on the topic, and Christoph Buchheim for his interest in this thesis. I am grateful to Elias Kuthe for proof-reading important parts. This thesis could not exist without the unconditional support of my parents.

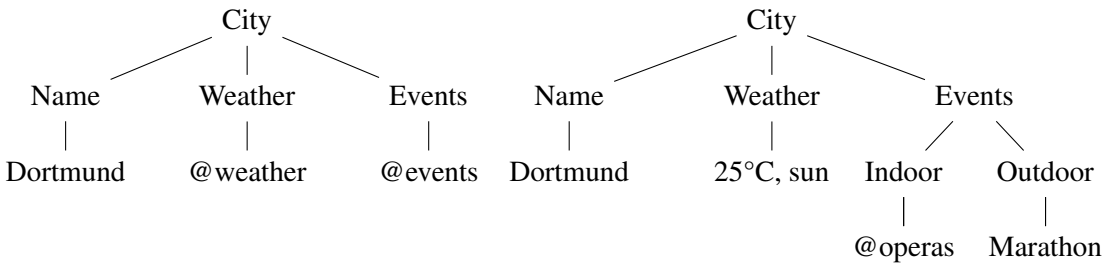
Contents

1. Introduction

The study of context-free games is motivated by issues that arose in the development of Active XML (AXML), an extension of the XML framework [12]. AXML documents are documents where some of the data is given explicitly while other parts are given by means of embedded calls to web services [10]. These embedded calls can be invoked to materialize more data. As an example (adapted from [10, 12]; see Figure 1.1), consider a document for the web page of a local newspaper. The document may contain some explicit data, such as the name of the city, whereas information about the weather and local events is given by means of calls to a weather forecast service and an events service (see Figure 1.1a). By invoking these calls, the data is materialized, i. e. replaced by concrete weather and events data (see Figure 1.1b). The data returned by the service call may contain further service calls.

When exchanging AXML documents between two applications, data can be materialized either by the sender before transferring the document or afterwards by the receiver. It is also possible that some of the data is materialized by the sender, some by the receiver and some is not materialized at all. In the example of Figure 1.1, data about the weather might be relevant only if there are outdoor events and otherwise it does not need to be materialized. The choice which data needs to be materialized by the sender and the receiver may be influenced by considerations about performance, capabilities, security and functionalities and can be specified, for instance, by a DTD [10]. An overview about AXML is given in [1].

An abstraction of this application are *(active) context-free games*, introduced by Muscholl, Schwentick and Segoufin [12]. A context-free game is determined by a set of production rules and a regular language (the *target language*). It is played on a string by two players, Romeo and Juliet, and consists of several rounds. In each round, first Juliet selects a position of the current string; then Romeo replaces the symbol at that position according to one of the production rules. Juliet wins if she can reach a situation where the current string belongs to the target language. In terms



(a) AXML document before invoking calls (b) AXML document after invoking calls

Figure 1.1.: An AXML document before and after the invocation of service calls

of the application described above, the string models the AXML document and Juliet's choice of a position in the string corresponds to the invocation of a service call. The replacement word chosen by Romeo corresponds to the return value of a service call. The target language represents the specification which the document has to satisfy before it is sent to the receiver. The question whether a document can be rewritten so that it satisfies the specification translates to the problem of deciding whether Juliet has a winning strategy for a given input string in a given context-free game.

In the general case, it is undecidable for a given game and string whether a winning strategy for Juliet exists [12]. However, the problem becomes decidable in several restricted cases or by restricting Juliet to a certain type of strategy [12]. A strategy for Juliet is called *left-to-right* if she never selects a position to the left of a previously selected position. In other words, Juliet traverses the string from left to right and has to decide for each symbol whether to play *Read* (go to the next symbol) or *Call* (select this symbol for Romeo to replace it). The decidability and complexity of deciding whether a winning left-to-right or general strategy exists is determined in [12] for both the general case and several restricted cases. Even for most of the decidable restricted cases studied in [12], the complexity is very high.

Abiteboul, Milo and Benjelloun [3] imposed further restrictions on strategies by introducing *one-pass strategies*. One-pass strategies are a special type of left-to-right strategies where the string arrives in a streaming fashion; Juliet makes her decisions without seeing the suffix starting at the immediate right of the position that she currently decides to read or call. The goal is to achieve a more efficient document parsing because the document is processed in a single pass (hence the name), without consideration of the symbols that have not been processed yet. Abiteboul et al. [3] also introduced *regular strategies*, a particularly simple type of one-pass strategies that achieve this goal with as little computational power as a finite state automaton (FSA).

Despite their advantages, one-pass strategies also entail several difficulties and challenges. For instance, consider the example in Figure 1.2. At the start, Juliet only sees that the input word begins with a. She must play Call on a to preserve her chance to reach a string that matches the target language. Romeo can choose either b or bab as replacement word, but Juliet does not know which replacement word he chooses. Initially, only the first symbol b of the replacement word is displayed to her. She must play Read on b because b is not a function symbol (there are no rules to replace b). So next, she sees that a is the symbol that follows b. But now – *even if she knew that the input word was aa* – she is on the horns of a dilemma: If Romeo chose b as the replacement word for a, then she better plays Read now because that will give her the win. On the other hand, if Romeo chose bab, then she better plays another Call in order to win. The incomplete information means that there is no winning one-pass strategy for the input word aa.

What makes the situation especially difficult in this example is that Romeo could choose between two replacement words where one is a prefix of the other. In such a case, Juliet does not know what the replacement word is even after seeing it completely. We call replacement languages *self-delimiting* if this problem does not exist, i. e. if they contain no two words where one is a prefix of the other. Note that this can easily be achieved in practice by suffixing strings with a special end-of-file symbol.

The study of different types of one-pass strategies is the subject of this thesis.

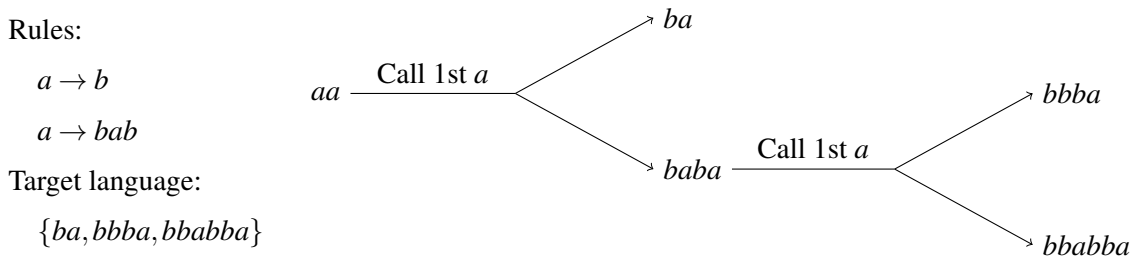


Figure 1.2.: A context-free game and three plays on the input word *aa* where Juliet wins

1.1. Scope

In total, we consider five types of one-pass strategies. Besides the general set of all one-pass strategies (type 1), we consider FSA-based strategies (type 2), strategies that do not make use of the history of Calls (type 3) and strategies that satisfy both of these restrictions at once (type 4; these are the regular strategies defined in [3]). An even more restricted class (type 5) are FSA-based strategies where the FSA is a simplification of the minimal deterministic finite automaton (DFA) for the target language. The FSA-based strategy types ensure efficient parsing, and especially type 5 guarantees that a specification of the strategy itself is not too complex. Ideally, one would wish to have a type 5 strategy that is as good as the best type 1 strategies.

Besides winning strategies for a single word, we are particularly interested in strategies that maximize the *set* of words on which they win. Roughly speaking, a strategy is *optimum* if it is at least as good as any other strategy and it is *optimal* if no other strategy is strictly better (but strategies may be incomparable). It was shown by Abiteboul et al. [3] that an optimum one-pass strategy does not always exist for a context-free game, and it was asked whether an optimal one-pass strategy always exists. It is also interesting to see whether the existence of a winning/optimum/optimal strategy of one type implies the existence of a winning/optimum/optimal strategy of a restricted type. Some questions of this kind were already considered in [3], but not fully resolved. Further interesting questions are concerned with the complexity of deciding whether a given one-pass strategy is winning (on a given word), optimum or optimal or whether a strategy exists that is winning, optimum or optimal. We study these and related questions, obtaining answers to the general case for some questions and to restricted cases for other questions.

1.2. Contributions

Towards the goal of having a type 5 strategy that is as good as the best one-pass strategy, we obtain the following generalization of a result from [3]: If the rules are non-recursive or if the target language is finite or if there are at most finitely many optimum one-pass strategies, then the existence of an optimum one-pass strategy implies the existence of an equivalent strategy of type 5. However, answering a question asked in [3], we show that even several weaker statements do not hold in general. As another positive result we show that the existence of an optimum one-pass

strategy implies the existence of an equivalent FSA-based strategy if replacement languages are self-delimiting.

We also consider these questions for optimal strategies, i. e. we ask whether the existence of an optimal strategy of one type implies the existence of an optimal strategy of a restricted type. It turns out that even if the conditions hold that guarantee a positive answer in the case of optimum strategies (non-recursiveness and a finite target language), the according implications and also several weaker implications are still not guaranteed to hold for optimal strategies.

Regarding the question whether an optimal strategy exists for each context-free game, we obtain a partial answer. We show that in games with non-recursive rules, games with self-delimiting replacement languages, and games with a finite target language, an optimal one-pass strategy always exists. More generally, we identify a condition that is sufficient for the existence of optimal one-pass strategies. It is conceivable that *all* games satisfy this condition. Proving that this is the case would be a means to show that an optimal one-pass strategy exists for each game, or disproving it might be a step towards proving that an optimal one-pass strategy does not exist for each game. For games with a finite target language, we show that even an optimal strategy that is FSA-based (type 2) always exists.

Finally, we study the complexity of problems relevant in the context of one-pass strategies. Given a game G, a word w and a strategy automaton $\mathcal{A}$, it is **PSPACE**-complete to decide whether the strategy specified by $\mathcal{A}$ wins on w in G. The problem becomes **P**-complete if the target language of G is given by a DFA. Deciding whether a winning type 5 strategy exists for a word w in a game G (with target language given by a DFA) is **NP**-complete. All of these hardness results hold even if the set of rules is restricted to be finite and non-recursive. For comparison, the counterpart of the latter problem for left-to-right strategies is **EXPTIME**-complete if the set of rules is finite and **PSPACE**-complete if, additionally, the rules are non-recursive (as shown in [12]); so the complexity is at least somewhat lower for type 5 strategies compared to left-to-right strategies (assuming that **NP** $\neq$ **EXPSPACE** and **NP** $\neq$ **PSPACE** respectively).

It was stated by Abiteboul et al. [3] that the winning language of a type 4 strategy is regular and a "universal FSA" recognizing the winning language can be constructed in polynomial time (if the target language is given by a DFA). We extend the construction from their proof sketch so that it is correct also for strategies that may lead to infinite play and prove the claim more generally for type 2 strategies. Based on this we show that, for a target language given by a DFA, it is **PSPACE**-complete to decide whether one given FSA-based strategy is at least as good as another given FSA-based strategy.

1.3. Related work

Further background about AXML is given in [1, 2, 10]. Context-free games were introduced in [11], which is the conference paper corresponding to [12]. The article studies the decidability and complexity of deciding whether a winning unrestricted or left-to-right strategy exists for a word in the general case and several restricted cases. One-pass strategies and regular strategies were introduced in [3]. The complexity of deciding, for a given context-free game, whether Juliet has a

winning left-to-right strategy for every word for which she has a winning unrestricted strategy is studied in [4]. An extended setting of context-free games with nested words (resembling the tree structure of (A)XML documents) is examined in [13].

1.4. Organisation

Most definitions are given in Chapter 2. We study sufficient conditions for the existence of optimal strategies in Chapter 3. In Chapter 4, we identify conditions under which the existence of an optimum, optimal or winning strategy of one type implies the existence of an optimum, optimal or winning strategy of a restricted types. In Chapter 5 we construct an automaton recognizing the language of words on which an FSA-based strategy wins, showing that this language is regular. The computational complexity of several problems is examined in Chapter 6. Chapter 7 contains an overview of results and open questions and a conclusion. Some variants of results of lesser importance and alternative proofs are moved to an appendix.

2. Preliminaries

We start with a few definitions that are not directly related to context-free games. For an alphabet Σ, we denote the set of strings over Σ by Σ^* and the set of non-empty strings over Σ by Σ^+. For $k \in \mathbb{N}_0$ we write Σ^k for the set of strings over Σ of length k and $\Sigma^{\leq k}$ for the set of strings over Σ of length at most k. We call a language $L \subset \Sigma^*$ *self-delimiting* if for each $u, v \in L$ where u is a prefix of v it holds that $u = v$. In the sections below, we formally define context-free games, strategies for Juliet and Romeo, and several related terms.

2.1. Context-free games

An *(active) context-free game*, or a *game* for short, is defined as a tuple $G = (\Sigma, R, T)$ consisting of

- a finite alphabet Σ,
- a set $R \subseteq \Sigma \times \Sigma^+$ of *rules* such that for each $a \in \Sigma$, the *replacement language* $\{v \in \Sigma^+ \mid (a,v) \in R\}$ of a is a regular language,
- a nondeterministic finite automaton (NFA) $T = (Q, \Sigma, \delta, q_0, F)$, where Q is the set of states, Σ the alphabet, $\delta\colon Q \times \Sigma \to \mathcal{P}(Q)$ the transition function, $q_0 \in Q$ the initial state and $F \subseteq Q$ the set of accepting states.

The language $L(T)$ recognized by T is the *target language*. By $\Sigma_f = \{a \in \Sigma \mid \exists v \in \Sigma^+\colon (a,v) \in R\}$ we denote the set of *function symbols*, i. e. the symbols occurring as the left hand side of a rule. These are the symbols which can be called by Juliet. We assume that Σ_f is non-empty to avoid trivial cases. The set R of rules is encoded in terms of regular expressions R_a for each $a \in \Sigma_f$, recognizing its replacement language $L(R_a) = \{v \in \Sigma^+ \mid (a,v) \in R\}$. Note that the set $L(R_a)$ of replacement strings for a does not contain the empty string ε. To define R_a we will often write $a \to r_i$ for $i = 1, \ldots, n$, where $n \in \mathbb{N}$ (mostly $n = 1$) and r_i are regular expressions; this notation means that $R_a = r_1 + \cdots + r_n$, where $+$ denotes alternation of regular expressions.

A game $G = (\Sigma, R, T)$ is called *recursive* if some symbol can be derived from itself by a sequence of rules, i. e. there exist $a_0, \ldots, a_n \in \Sigma_f$, $n \geq 1$, such that $a_0 = a_n$ and for each $k = 1, \ldots, n$ there exists a word in $L(R_{a_{k-1}})$ containing a_k. Otherwise, G is called *non-recursive*. Subsequently, we assume that a game $G = (\Sigma, R, T)$ is given.

We need more terms and notation to define how a game is played. Let $\widehat{\Sigma_f} = \{\widehat{a} \mid a \in \Sigma_f\}$ be a disjoint copy of the set of function symbols Σ_f, and let $\overline{\Sigma} = \Sigma \dot\cup \widehat{\Sigma_f}$. A *configuration* is a tuple $(\overline{u}, v) \in \overline{\Sigma}^* \times \Sigma^*$. We call $\overline{u}$ the *history string* and v the *remaining string*. Intuitively, if the ith symbol of the history string is $a \in \Sigma$ then this shall denote that Juliet's ith move was to read the symbol a,

and if it is $\widehat{a} \in \widehat{\Sigma_f}$ then this shall denote that Juliet's ith move was to call a. The remaining string is the string of symbols that have not been processed yet. To distinguish history strings from other strings we denote them by $\overline{u}, \overline{u_i}, \overline{u}', \overline{v}, \overline{v_i}$ etc. The *read prefix* of a configuration $(\overline{u}, v)$ is the string obtained from $\overline{u}$ by removing all symbols that are in $\widehat{\Sigma_f}$, denoted $\natural\overline{u}$. This is the string of symbols that have been read and cannot be changed any more. To allow us to omit parentheses, we define that $\natural$ has lower precedence than concatenation, i.e. $\natural\overline{uv} = \natural(\overline{uv}) = \natural\overline{u}\natural\overline{v}$. The set $\delta^*(q_0, \natural\overline{u})$ of states of T reached after reading the read prefix is called the *set of T-states* of the configuration $(\overline{u}, v)$. Here, $\delta^* \colon Q \times \Sigma^* \to \mathcal{P}(Q)$ denotes the extended transition function of T, given by

$$\delta^*(q, v) = \begin{cases} \{q\}, & \text{if } v = \varepsilon, \\ \bigcup\limits_{p \in \delta^*(q,u)} \delta(p, a), & \text{if } v = ua \text{ for } u \in \Sigma^* \text{ and } a \in \Sigma. \end{cases}$$

If T is deterministic, i.e. $\delta(q, a)$ is a singleton for each $q \in Q$ and $a \in \Sigma$, then we regard δ and δ^* as functions $Q \times \Sigma \to Q$ and $Q \times \Sigma^* \to Q$ respectively and call $\delta^*(q_0, \natural\overline{u})$ the *T-state* of the configuration $(\overline{u}, v)$. If $v = av'$ for $a \in \Sigma$ and $v' \in \Sigma^*$, then

- $\natural\overline{u}a$ is the *visible prefix*,
- a is the *current symbol*, and
- v' is the *invisible suffix*

of the configuration $(\overline{u}, av')$.

A *play* of a game in the one-pass setting[1] is a finite or infinite sequence $\Pi = (K_0, K_1, K_2, \dots)$ of configurations with the following properties:

(i) The *initial configuration* is of the form $K_0 = (\varepsilon, w)$, where $w \in \Sigma^*$ is called the *input word*.

(ii) If $K_n = (\overline{u}, av)$ with $a \in \Sigma$, then either $K_{n+1} = (\overline{u}a, v)$ or $K_{n+1} = (\overline{u}\widehat{a}, xv)$ with $x \in L(R_a)$.

(iii) If $K_n = (\overline{u}, \varepsilon)$, then K_n is the last configuration of the sequence. The history string $\overline{u}$ of K_n is the *final history string* of Π and the read prefix $\natural\overline{u}$ of K_n is the *final string* of Π.

Observe that for any configuration $K_n = (\overline{u}, v)$ it holds that $n = |\overline{u}|$, i.e. n is the length of the history string. In our imagination, the decision whether the configuration K_{n+1} in (ii) is $(\overline{u}a, v)$ or of the form $(\overline{u}\widehat{a}, xv)$ for some x is made by Juliet; in the latter case, the choice of $x \in L(R_a)$ is made by Romeo. If Juliet chooses $(\overline{u}a, v)$ as the next configuration after a configuration $(\overline{u}, av)$, then we say she plays *Read* or she *reads* a; otherwise we say she plays *Call* or she *calls* a.

We think of the configurations of a play to occur over time. For an input word of the form ww' with $w, w' \in \Sigma^*$, we refer to the time of the first configuration with remaining string w' as the time when w is *completely processed* (if such a configuration exists, which might not be the case if the play is infinite).

[1] The same definition could be used to define plays in the left-to-right setting. The only difference between the two settings is that in the one-pass setting, Juliet's decisions to call or to read in a configuration have to be defined independently of the invisible suffix.

A play is *winning* (from Juliet's point of view) if it is finite and its final string is in the target language $L(T)$, i.e. the set of T-states of the last configuration intersects the set F of accepting states. A play is *losing* if it is finite and its final string is not in $L(T)$, i.e. the set of T-states of the last configuration is disjoint from F. An infinite play is neither winning nor losing.

A way to think about plays that is more visual and sometimes more suitable than a list of configurations is via play trees. Play trees for context-free games are similar to derivation trees for context-free grammars. The *play tree* $\mathcal{T}(\Pi)$ of a play Π has a node for each configuration in Π except for the last configuration (if a last configuration exists, i.e. Π is finite), plus a root. The inner nodes (except of the root) are marked with configurations where Juliet plays Call while the leaves are marked with configurations where she plays Read. To allow for an easier terminology we often pretend that a node *is* a configuration rather than that it is marked by one.

The children of the root of $\mathcal{T}(\Pi)$ are defined as follows. Let $w = b_1 \dots b_\ell$ be the input word of Π, where $b_1, \dots, b_\ell \in \Sigma$. Let

$$i_j = \min\{i \in \mathbb{N}_0 \mid K_i = (\overline{v}, b_j \dots b_\ell) \text{ for some } \overline{v} \in \overline{\Sigma}^*\}$$

be the index of the configuration when $b_1 \dots b_{j-1}$ is completely processed for $j = 1, \dots, k$ where $k \in \{1, \dots, \ell\}$ is the maximal value of j for which the minimum exists. So $k = \ell$ if and only if Π is finite. The root of $\mathcal{T}(\Pi)$ has k children, which are marked – from left to right – with the configurations $K_{i_1}, \dots, K_{i_k}$. Informally, K_{i_j} is the configuration where the jth symbol of the input word becomes visible as the current symbol (however, Juliet might not know that this symbol comes from the input word and not from a replacement word of a Call move).

The children of non-root nodes are defined similarly. Let $K_n = (\overline{u}, av)$ be a configuration in Π for some $\overline{u} \in \overline{\Sigma}^*$, $a \in \Sigma$ and $v \in \Sigma^*$. If $K_{n+1} = (\overline{u}a, v)$, then K_n has no children. If $K_{n+1} = (\overline{u\widehat{a}}, xv)$ and $x = b_1 \dots b_\ell$ for some $b_1, \dots, b_\ell \in \Sigma$, $\ell \in \mathbb{N}$, let

$$i_j = \min\{i \in \mathbb{N} \mid K_i = (\overline{u\widehat{a}\overline{v}}, b_j \dots b_\ell v) \text{ for some } \overline{v} \in \overline{\Sigma}^*\}$$

for $j = 1, \dots, k$, where $k \in \{1, \dots, \ell\}$ is the maximal value of j for which the minimum exists. In this case, K_n has k children, marked with $K_{i_1}, \dots, K_{i_k}$ from left to right. Informally, K_{i_j} is the configuration where the jth symbol of the replacement string for a becomes visible.

There is indeed a node in $\mathcal{T}(\Pi)$ for each configuration K_n of Π (except for the last configuration, if Π is finite); this can be seen by induction on n: If the current symbol of K_n comes from the input word, then K_n is a root's child. If the current symbol of K_n comes from a replacement string, then K_n is a child of the configuration $K_{n'}$ where the Call was made. Since $n' < n$, the node with marking $K_{n'}$ exists by the induction hypothesis.

It is easy to see that a pre-order traversal of $\mathcal{T}(\Pi)$ yields the configurations in the order in which they occur in Π.

The *depth* of Π is the depth of $\mathcal{T}(\Pi)$ minus 1. This is the nesting depth of Call moves; subtracting 1 ensures that the depth of a play that consists of only Read moves is 0. By König's lemma, the depth of Π is ∞ if and only if Π is infinite. Note that the depth of a play in a non-recursive game is no more than $|\Sigma_f|$. Therefore, every play in a non-recursive game is finite.

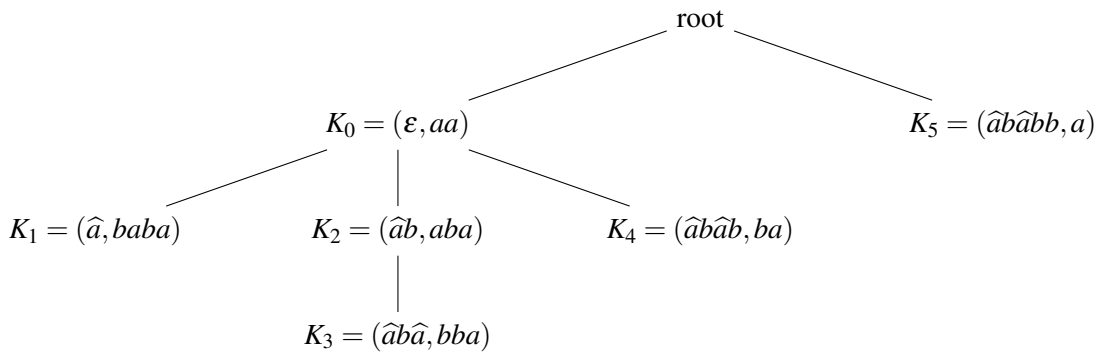


Figure 2.1.: A play tree

2.1 Example. Consider again the context-free game from Figure 1.2 in the introduction with rules given by $a \to b + bab$ and target language $L(T) = \{ba, bbba, bbabba\}$. A play tree of a play for this game is shown in Figure 2.1. The input word of this play is aa, which can be deduced either from the initial configuration $K_0 = (\varepsilon, aa)$ or because the current symbols of the root's childs are both a. Juliet plays Call in configuration K_0 and Romeo chooses bab as the replacement word. Accordingly, the children of K_0 have b, a and b as their current symbols. Juliet plays Call again in configuration K_2 and this time Romeo chooses b as replacement word. All other moves are Read. The last configuration of the play, $K_6 = (\widehat{a}b\widehat{a}bba, \varepsilon)$, is not part of the play tree. The final history string is $\widehat{a}b\widehat{a}bba$ and the final string is $bbba$. The play is winning because $bbba$ is in the target language. Its depth is 2.

Consider the configuration $K_4 = (\widehat{a}b\widehat{a}b, ba)$. Its history string $\widehat{a}b\widehat{a}b$ shows that Juliet's previous moves were to call a, read b, call a and read b. Therefore, the read prefix is $\natural\widehat{a}b\widehat{a}b = bb$ and the set of T-states of K_4 is the set of states of the target language automaton reachable by reading bb. The remaining string of K_4 is ba, the visible prefix is bbb and the invisible suffix is a. Juliet cannot see the invisible suffix, so from her point of view it would also be possible that $K_4 = (\widehat{a}b\widehat{a}b, b)$ – e. g. if the input word were aab and Romeo chose b as replacement word both times.

Observe that the history string and current symbol of a configuration K_n is uniquely determined by the history string of K_{n+1}. Thus, the history string of a configuration contains the information about the history strings and current symbols of *all* previous configurations. The restriction of incomplete information in the one-pass setting means that Juliet must make her decisions to call or to read based on only the history string and the current symbol of the current configuration. We will define this formally in the next section.

2.2. Strategies for Juliet

We are interested in a certain type of strategies for Juliet, namely one-pass strategies. One-pass strategies were introduced by Abiteboul et al. [3] as left-to-right strategies where Juliet can only see the current symbol at all times. The authors impose no restrictions on Juliet's memory. Thus, Juliet has full knowledge of the symbols that she saw in the past and her decisions on them, i. e. she

knows the history string. We will also consider a restricted type of one-pass strategies where Juliet only remembers the symbols on which she played Read, i. e. she only knows the visible prefix. In this section, we provide the formal definition of the most general type of one-pass strategies and related terms.

A *one-pass strategy* is a map $\sigma\colon \overline{\Sigma}^* \times \Sigma_f \to \{Call, Read\}$. We may sometimes simply say strategy when it is clear from the context what kind of strategy is meant (and in fact, all strategies for Juliet considered here are one-pass strategies).

A *play of* σ *on* w is a play $\Pi = (K_0, K_1, K_2, \dots)$ with input word w satisfying that

- if $K_n = (\overline{u}, av)$ and $\sigma(\overline{u}, a) = Read$, then $K_{n+1} = (\overline{u}a, v)$,
- if $K_n = (\overline{u}, av)$ and $\sigma(\overline{u}, a) = Call$, then $K_{n+1} = (\overline{u}\widehat{a}, xv)$ for some $x \in L(R_a)$.

A strategy is *terminating* if each of its plays is finite. The *depth* of σ is the supremum of depths of plays of σ. Note that each strategy with finite depth is terminating. The converse, however, is not true and it is easy to construct counter-examples of a game and a strategy σ where each play of σ has finite depth but depths are arbitrarily large.

A strategy σ *wins* on a string $w \in \Sigma^*$ if every play of σ on w is winning. So a strategy wins if, independently of Romeo's replacement words, the input string will be rewritten to a string in the target language. By $L(\sigma)$ we denote the set of words on which σ wins. In contrast, σ *loses* on w if there exists a losing play of σ on w. Note that σ neither wins nor loses on w if there exists an infinite play of σ on w but no losing play of σ on w.

Two strategies σ and σ' are *equivalent* if they win on the same input words, i. e. $L(\sigma) = L(\sigma')$. Two strategies are *semi-equivalent* if there exists no string on which one strategy wins and the other strategy loses. Note that two terminating one-pass strategies (e. g. any two one-pass strategies for non-recursive games) are equivalent if and only if they are semi-equivalent.

A strategy σ is *optimum* if it is at least as good as any other strategy, i. e. $L(\sigma') \subseteq L(\sigma)$ for each one-pass strategy σ'. It was shown by Abiteboul et al. [3] that not every game has an optimum one-pass strategy. A strategy σ is *optimal* if no strictly better strategy exists, i. e. there does not exist σ' with $L(\sigma) \subsetneq L(\sigma')$. It is unclear whether each game has an optimal one-pass strategy, but we will identify sufficient conditions for their existence. Of course, for games where an optimum one-pass strategy exists, a strategy is optimal if and only if it is optimum. It is also clear from the definitions that a strategy is optimum (or optimal) if and only if it is equivalent to a strategy that is optimum (or optimal).

We cannot in general expect a one-pass strategy to be computable, since the set $\{(\overline{u}, a) \in \overline{\Sigma}^* \times \Sigma_f \mid \sigma(\overline{u}, a) = Call\}$ might be undecidable. If σ is computable, then it is desirable to have an efficient algorithm for it. This motivates the study of restricted classes of one-pass strategies. We will define classes of strategies where the decisions whether to call or to read is performed by a finite state automaton. Another interesting restriction, though it does not guarantee computability, is to require strategies to make their decisions to call or to read independently of the history of Calls.

2.3. Five types of one-pass strategies

In total we introduce five types of one-pass strategies. A study of their relationships constitutes a major part of this thesis. A strategy of *type 1* is exactly what we defined as a one-pass strategy above. This is the most general of the five types; the types 2, 3, 4 and 5 are restricted types. We will define the types 3, 4 and 5 first and type 2 last. The hierarchy of the five types is shown in Figure 2.2.

A strategy is of *type 3* if its decisions depend only on the visible prefix but not on the history of Calls. Formally, σ is a strategy of type 3 if $\sigma(\overline{u},a) = \sigma(\natural\overline{u},a)$ for each $\overline{u} \in \overline{\Sigma}^*$ and $a \in \Sigma_f$.

Strategies of type 4 and type 5 are strategies that can be specified by a DFA in the following sense: A *type 4 automaton* is a DFA $\mathcal{A} = (Q_\mathcal{A}, \Sigma, \delta_\mathcal{A}, q_0, F_\mathcal{A})$ where some transitions for function symbols may be missing. We formalize this by allowing the transition function $\delta_\mathcal{A}$ to be undefined on some $(q,a) \in Q \times \Sigma_f$, which we denote by $\delta_\mathcal{A}(q,a) = \bot$. The strategy of a type 4 automaton reads the input word like a standard DFA except that when reading a symbol $a \in \Sigma_f$ in a state that has no transition for a, then a is called and replaced by the word from $L(R_a)$ that Romeo selects. Formally,

$$\sigma_\mathcal{A}(\overline{u},a) = \begin{cases} \mathit{Read}, & \text{if } \delta^*_\mathcal{A}(q_0, \natural\overline{u}a) \neq \bot, \\ \mathit{Call}, & \text{if } \delta^*_\mathcal{A}(q_0, \natural\overline{u}a) = \bot, \end{cases}$$

where the extended partial transition function $\delta^*_\mathcal{A} \colon Q_\mathcal{A} \times \Sigma^* \to Q_\mathcal{A}$ is given by

$$\delta^*_\mathcal{A}(q,u) = \begin{cases} q, & \text{if } u = \varepsilon, \\ \delta_\mathcal{A}(\delta^*_\mathcal{A}(q,v),a), & \text{if } u = va \text{ and } \delta^*_\mathcal{A}(q,v) \neq \bot \text{ for } v \in \Sigma^* \text{ and } a \in \Sigma, \\ \bot, & \text{otherwise.} \end{cases} \tag{2.1}$$

A strategy σ is of *type 4* if $\sigma = \sigma_\mathcal{A}$ for a type 4 automaton $\mathcal{A}$. Typ 4 strategies have already been introduced by Abiteboul et al. [3] under the name *regular strategies*. It is immediate from the definition that each type 4 strategy is also a type 3 strategy.

Of course, to compute $\delta^*_\mathcal{A}(q_0, \natural\overline{u}a)$ when a decision to read or to call has to be made, it is not necessary to start the computation from the initial state q_0 each time. Rather, we define $q = \delta^*_\mathcal{A}(q_0, \natural\overline{u})$ as the *$\mathcal{A}$-state* of a configuration $(\overline{u}, v)$; if $v = av'$ for $a \in \Sigma$ and $v' \in \Sigma^*$ and the next configuration is $(\overline{u}a, v')$, then the new $\mathcal{A}$-state $\delta^*_\mathcal{A}(q_0, \natural\overline{u}a)$ can be computed as $\delta_\mathcal{A}(q,a)$; if the next configuration is of the form $(\overline{u}\widehat{a}, xv')$, then the new $\mathcal{A}$-state is still q. Therefore, type 4 strategies can be computed very efficiently.

A special case of type 4 strategies that was also already considered in [3] are type 5 strategies. A type 4 automaton is called a *type 5 automaton* if it equals the minimal DFA for the target language $L(T)$ except that some transitions may be missing. A strategy σ is of *type 5* if $\sigma = \sigma_\mathcal{A}$ for a type 5 automaton $\mathcal{A}$. By definition, a type 5 strategy is also a type 4 strategy.

Finally, let us define type 2 strategies. These strategies extend type 4 strategies in a natural way such that history can be used for the decisions to call or read. A *type 2 automaton* is a deterministic automaton $\mathcal{A} = (Q_\mathcal{A}, \overline{\Sigma}, \delta_\mathcal{A}, q_0, F_\mathcal{A})$ with a partial transition function such that for each $q \in Q_\mathcal{A}$ and $a \in \Sigma_f$, if $\delta_\mathcal{A}(q,a) = \bot$ then $\delta_\mathcal{A}(q,\widehat{a}) \neq \bot$. So a type 2 automaton is like a type 4 automaton with the difference that if a state does not have a transition for $a \in \Sigma_f$, then it has a transition for $\widehat{a}$. A

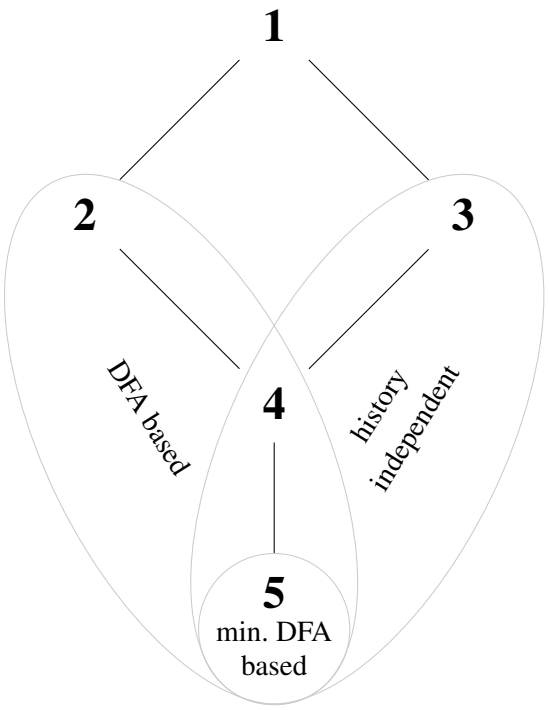


Figure 2.2.: Hasse diagram showing the hierarchy of the five types of one-pass strategies

type 2 automaton computes a strategy just like a type 4 strategies, but if a function symbol a is called in a state q then, instead of staying in state q, the transition for $\widehat{a}$ is followed. This can be formalized as

$$\sigma_{\mathcal{A}}(\overline{u},a) = \begin{cases} \mathit{Read}, & \text{if } \delta^*_{\mathcal{A}}(q_0,\overline{u}a) \neq \bot, \\ \mathit{Call}, & \text{if } \delta^*_{\mathcal{A}}(q_0,\overline{u}a) = \bot \end{cases}$$

for $\delta^*_{\mathcal{A}}$ defined as in (2.1) except that Σ is replaced by $\overline{\Sigma}$. Note that the only difference of the definition of $\sigma_{\mathcal{A}}$ compared to the previous definition is that we have written $\overline{u}$ instead of $\natural\overline{u}$. A strategy σ is of *type 2* if $\sigma = \sigma_{\mathcal{A}}$ for a type 2 automaton $\mathcal{A}$. The *$\mathcal{A}$-state* of a configuration $(\overline{u},v)$ occurring in a play of a type 2 strategy $\sigma_{\mathcal{A}}$ is defined as $\delta^*_{\mathcal{A}}(q_0,\overline{u})$. Similarly as above, the $\mathcal{A}$-state and the decisions of a type 2 strategy can be computed very efficiently.

Observe that every type 4 automaton $\mathcal{A} = (Q_{\mathcal{A}},\Sigma,\delta_{\mathcal{A}},q_0,F_{\mathcal{A}})$ can be transformed into a type 2 automaton for the same strategy by letting $\delta_{\mathcal{A}}(q,\widehat{a}) = q$ for each $q \in Q_{\mathcal{A}}$ and $\widehat{a} \in \widehat{\Sigma_f}$. Thus, every type 4 strategy is also of type 2.

A *strategy automaton* is a type 2 or type 4 automaton. We allow accepting states in strategy automata even though they have no meaning for the strategy that they specify. Accepting states are convenient, however, if they can be used to indicate the winner of a play. We say that a play Π of $\sigma_{\mathcal{A}}$ *terminates in $\mathcal{A}$-state q* if Π is finite and q is the $\mathcal{A}$-state of the last configuration of Π. Observe that a play of a type 5 strategy $\sigma_{\mathcal{A}}$ is winning if and only if it terminates in an accepting $\mathcal{A}$-state (because the $\mathcal{A}$-state where it terminates is the state where the minimal DFA for $L(T)$ terminates when reading the final string). For type 2 and 4 strategies it is also always possible to find a strategy

automaton $\mathcal{A}$ such that this property holds (which was already mentioned for the case of type 4 automata in [3]).

2.2 Proposition. *For each type 2 (or 4) automaton $\mathcal{A}$ for a game $G = (\Sigma, R, T)$ there exists a type 2 (or 4) automaton $\mathcal{B}$ such that $\sigma_{\mathcal{A}} = \sigma_{\mathcal{B}}$ and a play of $\sigma_{\mathcal{B}}$ terminates in an accepting $\mathcal{B}$-state if and only if it is winning. If T is a DFA, then $\mathcal{B}$ can be constructed in polynomial time.*

Proof. If $T = (Q, \Sigma, \delta, q_0, F)$ is a DFA and $\mathcal{A} = (Q_{\mathcal{A}}, \overline{\Sigma}, \delta_{\mathcal{A}}, q_0', F_{\mathcal{A}})$ is a type 2 automaton, then define $\mathcal{B} = (Q \times Q_{\mathcal{A}}, \overline{\Sigma}, \delta_{\mathcal{B}}, (q_0, q_0'), F \times Q_{\mathcal{A}})$ with

$$\delta_{\mathcal{B}}((q,q'),a) = \begin{cases} (\delta(q,a), \delta_{\mathcal{A}}(q',a)), & \text{if } \delta_{\mathcal{A}}(q',a) \neq \bot, \\ \bot, & \text{if } \delta_{\mathcal{A}}(q',a) = \bot, \end{cases}$$

$$\delta_{\mathcal{B}}((q,q'),\widehat{a}) = \begin{cases} (q, \delta_{\mathcal{A}}(q',\widehat{a})), & \text{if } \delta_{\mathcal{A}}(q',\widehat{a}) \neq \bot, \\ \bot, & \text{if } \delta_{\mathcal{A}}(q',\widehat{a}) = \bot. \end{cases}$$

It is straightforward to verify correctness and polynomial construction time. The construction for type 4 automata is similar. If T is an NFA, then it can be transformed into a DFA first. □

The usefulness of this proposition is that it allows to ignore the target language automaton if a strategy is specified by an automaton $\mathcal{B}$ like in the proposition. Next, we change the point of view and consider strategies for Romeo.

2.4. Strategies for Romeo

We can think of Romeo as an antagonistic adversary who, unlike Juliet, can see the invisible suffix and also knows what strategy Juliet uses to play by. Despite this, we can define even a Romeo strategy as a map that does not take the invisible suffix into account, as we will justify shortly. Namely, a *Romeo strategy* is a map $\tau\colon \overline{\Sigma}^* \times \Sigma_f \to \Sigma^+$ where $\tau(\overline{u}, a) \in L(R_a)$ for each $(\overline{u}, a) \in \overline{\Sigma}^* \times \Sigma_f$.

The *play of σ against τ on w*, denoted $\Pi(\sigma, \tau, w)$, is the (unique) play $(K_0, K_1, K_2, \dots)$ of σ on w satisfying that

- if $K_n = (\overline{u}, av)$ and $\sigma(\overline{u}, a) = \mathit{Call}$, then $K_{n+1} = (\overline{u}\widehat{a}, \tau(\overline{u}, a)v)$.

A *play of τ* is a play of some one-pass strategy σ against τ on some input word.

The definition of Romeo strategies, which does not take the invisible suffix into account, is justified by the following proposition.

2.3 Proposition. *Let σ be a one-pass strategy for a game $G = (\Sigma, R, T)$ and let $w \in \Sigma^*$. Then for every play $\Pi = (K_0, K_1, K_2, \dots)$ of σ on w there exists a Romeo strategy τ such that $\Pi = \Pi(\sigma, \tau, w)$.*

Proof. If $K_n = (\overline{u}, av)$ and $\sigma(\overline{u}, a) = \mathit{Call}$ then we define $\tau(\overline{u}, a)$ as the string $x \in L(R_a)$ with $K_{n+1} = (\overline{u}\widehat{a}, xv)$. A straight-forward induction on n shows that the history string of configuration K_n has length n. Therefore, so far we have defined $\tau(\overline{u}, a)$ at most once for each $\overline{u}$ and a, i. e. there are no contradictions in the definition. On other elements of the domain, τ can be defined arbitrarily. It is immediate that Π is the play of σ against τ on w. □

A way to think about why Romeo is omniscient even though his strategies neglect the information about the input word is that the dependency of his moves on the input word lies within his choice of τ itself, rather than within τ.

Given a Romeo strategy and a configuration occurring in a play of this strategy, all previous configurations are uniquely determined, as the next proposition shows.

2.4 Proposition. *Let $\Pi = (K_0, K_1, K_2, \dots)$ and $\Pi' = (K'_0, K'_1, K'_2, \dots)$ be plays of a Romeo strategy τ in a game $G = (\Sigma, R, T)$ such that $K_i = K'_j$ for some indices $i \leq j$. Then $i = j$ and $K_n = K'_n$ for all $n \leq i$.*

Proof. We show the proposition by induction on j. If $j = 0$ then the statement obviously holds. For $j > 0$, the common history string of K_i and K'_j is non-empty and we distinguish two cases depending whether the last symbol of the history string is in Σ or in $\widehat{\Sigma_f}$.

If $K_i = K'_j = (\overline{u}a, v)$ for some $\overline{u} \in \overline{\Sigma}^*$, $a \in \Sigma$ and $v \in \Sigma^*$, then $K_{i-1} = K'_{j-1} = (\overline{u}, av)$ and the statement follows from the induction hypothesis.

In the other case, $K_i = K'_j = (\overline{u\widehat{a}}, \tau(\overline{u}, a)v)$ for some $\overline{u} \in \overline{\Sigma}^*$, $a \in \Sigma_f$ and $v \in \Sigma^*$. Again, $K_{i-1} = K'_{j-1} = (\overline{u}, av)$ and the statement follows from the induction hypothesis. □

2.5. Operations on strategies

To have a precise terminology for strategies that are an adaptation or combination of other strategies, we define operations on strategies. Namely we introduce the notions of a substrategy of a (one-pass or Romeo) strategy and a concatenation strategy of two Romeo strategies.

For a one-pass strategy σ and a history string $\overline{v} \in \overline{\Sigma}^*$, the *substrategy of σ after $\overline{v}$* is the strategy $\sigma[\overline{v}]$ defined by $\sigma[\overline{v}](\overline{u}, a) = \sigma(\overline{vu}, a)$. Intuitively, $\sigma[\overline{v}]$ is the strategy that plays like σ continues to play if the previous part of the play has led to the history $\overline{v}$. Analogously, for a Romeo strategy τ and $\overline{v} \in \overline{\Sigma}^*$, the *substrategy of τ after $\overline{v}$* is the strategy $\tau[\overline{v}]$ given by $\tau[\overline{v}](\overline{u}, a) = \tau(\overline{vu}, a)$.

If τ_1 and τ_2 are two Romeo strategies and $w \in \Sigma^*$, then the *concatenation strategy* $\tau_1\langle w\rangle\tau_2$ is, informally, the following strategy: The strategy $\tau_1\langle w\rangle\tau_2$ starts to play like τ_1. If w is a prefix of the input word, then, once w is completely processed (if that happens), $\tau_1\langle w\rangle\tau_2$ continues to play like τ_2 would play if the history up to this point had not happened.

Towards a formal definition of $\tau_1\langle w\rangle\tau_2$, let $M \subseteq \overline{\Sigma}^*$ be the set of strings occurring as the final history string of some play $\Pi(\sigma, \tau_1, w)$, where σ is a one-pass strategy. Then $\tau_1\langle w\rangle\tau_2$ is defined via

$$\big(\tau_1\langle w\rangle\tau_2\big)(\overline{u}, a) = \begin{cases} \tau_2(\overline{u_2}, a), & \text{if } \overline{u} = \overline{u_1 u_2} \text{ for some } \overline{u_1} \in M \text{ and } \overline{u_2} \in \overline{\Sigma}^*, \\ \tau_1(\overline{u}, a), & \text{otherwise.} \end{cases}$$

The next proposition states that the formal definition of $\tau_1\langle w\rangle\tau_2$ captures exactly what the informal definition describes.

2.5 Proposition. *For Romeo strategies τ_1 and τ_2 in a game $G = (\Sigma, R, T)$ and $w \in \Sigma^*$, the concatenation strategy $\tau_1\langle w\rangle\tau_2$ is well-defined. Let σ be a one-pass strategy, $w'' \in \Sigma^*$ and*

$\Pi(\sigma, \tau_1\langle w\rangle\tau_2, w'') = (K_0, K_1, \dots)$. *If* $w'' = ww'$ *for some* $w' \in \Sigma^*$ *and* Π *contains a configuration* $(\overline{u_1}, w')$ *for some* $\overline{u_1} \in \overline{\Sigma}^*$*, with* K_n *being the first such configuration, then*

- $K_0, K_1, \dots, K_n$ *are the first* $n+1$ *configurations of* $\Pi(\sigma, \tau_1, w'')$*, and*
- *for* $\overline{v_i}$ *and* v_i *with* $K_{n+i} = (\overline{u_1 v_i}, v_i)$ *for* $i \geq 0$*, the sequence* $((\overline{v_0}, v_0), (\overline{v_1}, v_1), (\overline{v_2}, v_2), \dots)$ *is a play of* τ_2*.*

Otherwise, $\Pi(\sigma, \tau_1\langle w\rangle\tau_2, w'') = \Pi(\sigma, \tau_1, w'')$.

Proof. To prove well-definedness we need to show that for each $\overline{u}$ there is at most one possible choice for $\overline{u_1} \in M$ and $\overline{u_2} \in \overline{\Sigma}^*$ such that $\overline{u} = \overline{u_1 u_2}$. This is true because M is self-delimiting. To see this, note that M can be defined inductively by the rules

- $w \in M$,
- if $\overline{u}av \in M$ for $\overline{u} \in \overline{\Sigma}^*$, $a \in \Sigma_f$ and $v \in \Sigma^*$, then $\overline{u\widehat{a}}\tau_1(\overline{u}, a)v \in M$.

Suppose M is not self-delimiting, then there exist $\overline{v_1}, \overline{v_2} \in M$ where $\overline{v_1}$ is a proper prefix of $\overline{v_2}$. Choose $\overline{v_1}$ and $\overline{v_2}$ such that the number of occurrences of elements of $\widehat{\Sigma_f}$ in $\overline{v_2}$ is minimal. Clearly, w does not have a proper prefix that is also in M. Thus, we have that $\overline{v_2} = \overline{u\widehat{a}}\tau_1(\overline{u}, a)v$ for some $\overline{u} \in \overline{\Sigma}^*$, $a \in \Sigma_f$ and $v \in \Sigma^*$. By the assumption of minimality, $\overline{u}av$ cannot have $\overline{v_1}$ as a proper prefix. So $\overline{v_1}$ is a prefix of $\overline{u\widehat{a}}\tau_1(\overline{u}, a)v$ but not of $\overline{u}$. Hence, since $\overline{v_1} \in M$, we can write $\overline{v_1} = \overline{u\widehat{a}}\tau_1(\overline{u}, a)v'$ for some $v' \in \Sigma^*$ that is a proper prefix of v. But then $\overline{u}av'$ and $\overline{u}av$ would also be two strings in M where one is a proper prefix of the other, contradicting the minimality in the choice of $\overline{v_1}$ and $\overline{v_2}$.

We now prove the second part of the proposition. By definition of a play, the history string of each configuration extends the history string of the previous configuration by one symbol. Thus, once a configuration with a history string of the form $\overline{u} = \overline{u_1 u_2}$ with $\overline{u_1} \in M$ and $\overline{u_2} \in \overline{\Sigma}^*$ occurs in $\Pi(\sigma, \tau_1\langle w\rangle\tau_2, w'')$, all proceeding configurations also have a history string of this form. Moreover, if a configuration with a history string of this form occurs in $\Pi(\sigma, \tau_1\langle w\rangle\tau_2, w'')$, then the first such configuration is of the form $(\overline{u_1}, w')$ with $\overline{u_1} \in M$ and $w' \in \Sigma^*$. This tells us, together with the definition of $\tau_1\langle w\rangle\tau_2$, that $\tau_1\langle w\rangle\tau_2$ is a strategy that does the following: It starts to play like τ_1. Once a configuration $(\overline{u_1}, w')$ with $\overline{u_1} \in M$ and $w' \in \Sigma^*$ occurs, $\tau_1\langle w\rangle\tau_2$ changes to play like τ_2 would play if the history string at this time were empty instead of $\overline{u_1}$. This proof is finished if we can show the following:

(i) If $w'' = ww'$ and a configuration $(\overline{u_1}, w')$ with $\overline{u_1} \in \overline{\Sigma}^*$ occurs in $\Pi(\sigma, \tau_1\langle w\rangle\tau_2, w'')$, then $\overline{u_1} \in M$ for the first such configuration.

(ii) Conversely, a configuration $K_n = (\overline{u_1}, w')$ with $\overline{u_1} \in M$ and $w' \in \Sigma^*$ occurs in $\Pi(\sigma, \tau_1\langle w\rangle\tau_2, w'')$ only if $w'' = ww'$.

For part (i), consider the history string $\overline{u_1}$ of the first configuration $(\overline{u_1}, w')$ with $\overline{u_1} \in \overline{\Sigma}^*$ that occurs in $\Pi(\sigma, \tau_1\langle w\rangle\tau_2, w'')$. Clearly, $\overline{u_1}$ must be the final history string of the play of σ against $\tau_1\langle w\rangle\tau_2$ on w (instead of ww'). Thus, $\overline{u_1} \in M$ by definition of M.

For part (ii), let σ' be a one-pass strategy such that $\overline{u_1}$ is the final history string of $\Pi(\sigma', \tau_1, w)$ (recall that σ' exists by definition of M). Then the configuration $K_n = (\overline{u_1}, w')$ also occurs in $\Pi(\sigma', \tau_1, ww')$. By definition of $\tau_1\langle w\rangle\tau_2$, the configurations $K_0, \ldots, K_n$ are also the first $n+1$ configurations of $\Pi(\sigma, \tau_1, w'')$. Thus, by Proposition 2.4, $w'' = ww'$. □

Of course we can also build the concatenation of more than just two Romeo strategies. To simplify notation we postulate that the concatenation operation is right-associative, meaning that $\tau_1\langle w_1\rangle\tau_2\langle w_2\rangle\tau_3 = \tau_1\langle w_1\rangle\big(\tau_2\langle w_2\rangle\tau_3\big)$ for Romeo strategies τ_1, τ_2 and τ_3 and $w_1, w_2 \in \Sigma^*$. This entails that $\tau_1\langle w_1\rangle\tau_2\langle w_2\rangle\tau_3$ plays as one would expect; namely, on an input word $w_1w_2w_3$ for some $w_3 \in \Sigma^*$, it starts to play like τ_1 until w_1 is completely processed, then continues to play like τ_2 would play if the history at this point were empty, and finally, once w_1w_2 is completely processed, it plays like τ_3 would play if the history at this point were empty.

When we use substrategies and concatenation strategies in proofs later on, we will usually think of them in terms of their informal definitions. It is not difficult, though rather technical, to extend the proofs to work directly with the formal definitions. The formal definitions inhibit any ambiguity and justify that what we have defined informally exists as a Romeo strategy.

3. Existence of optimal strategies

This chapter is devoted to the question whether an optimal one-pass strategy of type 1 is guaranteed to exist for a context-free game. The answer to the analogous question for the case of optimum instead of optimal is negative, as Abiteboul et al. [3, Theorem 2.3(2)] have shown that there exist games for which no optimum one-pass strategy exists. This discriminates the one-pass setting from the more general left-to-right setting and the even more general setting of unrestricted strategies for context-free games [3, Theorem 2.3(1)]. Examples of games with no optimum one-pass strategy are also implicit in the proofs of Theorems 4.17 and 4.18 of this thesis. The reason why an optimum strategy does not always exist in the one-pass setting lies within the incomplete information that Juliet has to deal with. However, the question whether an optim*al* one-pass strategy exists for each game is still open.

In Section 3.1 we consider a setting where Juliet is slightly more powerful than in the one-pass setting and give an alternative proof to a theorem by Abiteboul et al. [3] that an optimal strategy always exists in this setting. In Section 3.2 we come back to the standard one-pass setting and identify a sufficient condition for the existence of optimal one-pass strategies. We call this condition the bounded depth property. Each non-recursive game and each game with a finite target language has the bounded depth property, so an optimal one-pass strategy exists for each of these games. In Section 3.3 we will show that also games with self-delimiting replacement languages have the bounded depth property, and therefore an optimal one-pass strategy.

3.1. One-pass strategies with size

We now revisit a setting that was already considered and said to be relevant in practice by Abiteboul et al. [3]. They introduce *one-pass strategies with size*, which are like one-pass strategies with the difference that Juliet knows the length of the invisible suffix. This can be formalized by prefixing the input word and each replacement word by a number indicating its size, which can be read like a symbol of the alphabet. A one-pass strategy with size could then be defined as a map $(\overline{\Sigma} \cup \mathbb{N})^* \times \Sigma_f \to \{\mathit{Call}, \mathit{Read}\}$. By [3, Theorem 2.4], a computable optimal one-pass strategy with size exists for each game. The proof is only sketched in [3] and the authors say that it is quite involved. Neglecting the aspect of computability, we give a very simple non-constructive proof for the existence of an optimal one-pass strategy with size. Our proof does not even make use of the knowledge of the size of the replacement words but only of the size of the input string.

3.1 Proposition. *For each game $G = (\Sigma, R, T)$ there exists an optimal one-pass strategy with size.*

Proof. For $k \in \mathbb{N}$ let σ_k be a one-pass strategy with size that maximizes the number of input words of length k on which it wins. Since Σ^k is finite, such a strategy exists. Consider the strategy σ

defined by $\sigma(k\overline{u},a) = \sigma_k(k\overline{u},a)$ for $k \in \mathbb{N}$, $\overline{u} \in (\overline{\Sigma} \cup \mathbb{N})^*$ and $a \in \Sigma_f$ and defined arbitrarily if the history string does not begin with a number (which only happens in the initial configuration of a play, where Juliet is forced to play Read anyway because the initial current symbol – a number – is not in Σ_f). Clearly, σ is an optimal one-pass strategy with size. □

The same basic idea – maximizing the number of input words of a fixed size on which a strategy wins – can also be used in the normal one-pass setting (without size), at least if a game has the bounded depth property, which we will consider in the next section.

3.2. Optimal strategies and the bounded depth property

The bounded depth property, of which we will show that it is sufficient for the existence of an optimal one-pass strategy, is defined as follows.

3.2 Definition. A game G has the *bounded depth property* if there exists a sequence $(B_k)_{k\in\mathbb{N}_0} \subseteq \mathbb{N}$ such that for each one-pass strategy σ for G and each $k \in \mathbb{N}$ there exists a one-pass strategy σ_k such that each play of σ_k on a word $w \in L(\sigma) \cap \Sigma^{\leq k}$ is winning and has depth at most $B_{|w|}$.

Since each play in a non-recursive game has depth at most $|\Sigma_f|$, it is clear that non-recursive games have the bounded depth property – one can simply choose $B_k = |\Sigma_f|$ and $\sigma_k = \sigma$. We will also show for games with a finite target language, and for games with self-delimiting replacement languages (in Section 3.3), that they have the bounded depth property; even for these (B_k) can be chosen as a constant sequence. The bounded depth property seems like a considerably weaker condition, since it allows (B_k) to be an unbounded increasing sequence and it is only required that a strategy σ can be adjusted to a strategy that wins and satisfies the depth bound for the finitely many input words up to a given maximal length on which σ wins. It is unclear whether there even exists a game that does *not* satisfy the bounded depth property.

We now prove the fundamental theorem of this chapter.

3.3 Theorem. *For every game $G = (\Sigma, R, T)$ having the bounded depth property there exists an optimal one-pass strategy.*

Proof. Starting with the set of all one-pass strategies we will construct smaller and smaller sets of one-pass strategies by ensuring that the strategies win on an optimal subset of $\Sigma^{\leq k}$ for increasing k and fixing the first i moves for increasing i. Only a single strategy will remain in the intersection of all these sets, and we will show that this strategy is optimal.

Let $(B_k) \subseteq \mathbb{N}$ be as in the definition of the bounded depth property. For $k, i \in \mathbb{N}_0$ we define sets $S_{k,i}$ of one-pass strategies such that the following properties hold for all $k, i \in \mathbb{N}_0$:

(i) For all $\sigma \in S_{k,0}$ it holds that $S_{k,0}$ is exactly the set of one-pass strategies σ' with $L(\sigma) \cap \Sigma^{\leq k} = L(\sigma') \cap \Sigma^{\leq k}$ satisfying that each play of σ' on a word $w \in L(\sigma) \cap \Sigma^{\leq k+1}$ has depth at most $B_{|w|}$.

(ii) For $\sigma \in S_{k,i}$ there exists no one-pass strategy σ' with $L(\sigma) \cap \Sigma^{\leq k} \subsetneq L(\sigma') \cap \Sigma^{\leq k}$, i. e. the strategies in $S_{k,i}$ win on the same maximal subset of $\Sigma^{\leq k}$.

(iii) For all $\sigma, \sigma' \in S_{k,i}$ it holds that $\sigma\big|_{\overline{\Sigma}^{\leq i-1} \times \Sigma_f} = \sigma'\big|_{\overline{\Sigma}^{\leq i-1} \times \Sigma_f}$, i. e. all strategies in $S_{k,i}$ act the same during their first i moves.

(iv) $S_{k,i+1} \subseteq S_{k,i}$

(v) $S_{k+1,i} \subseteq S_{k,i}$

(vi) $S_{k,i}$ is non-empty.

Let $S_{0,0}$ be the set of all one-pass strategies for G. If $S_{k,0}$ is defined for $k \in \mathbb{N}_0$, pick some $\sigma \in S_{k,0}$ that maximizes $|L(\sigma) \cap \Sigma^{k+1}|$ and let

$$S_{k+1,0} = \Big\{ \sigma' \in S_{k,0} \;\Big|\; \text{each play of } \sigma' \text{ on a word } w \in L(\sigma) \cap \Sigma^{\leq k+1} \text{ is winning and has depth at most } B_{|w|} \Big\}.$$

This defines $S_{k,0}$ for all $k \in \mathbb{N}_0$. Note that $S_{k+1,0}$ is non-empty since G has the bounded depth property. By induction on k it is clear that the properties (i), (ii) and (v) hold for $k \in \mathbb{N}_0$ and $i = 0$. The properties (iii) and (iv) do not apply since we have defined $S_{k,i}$ only for $i = 0$ so far. The proof that $S_{k+1,0}$ is non-empty (property (vi)) is also by induction on k and uses that the one-pass strategy σ_{k+1} for which the bounded depth property says that it exists is automatically in $S_{k,0}$ due to the properties (i) and (ii).

For $i \in \mathbb{N}_0$ such that $S_{k,i}$ is defined for all $k \in \mathbb{N}_0$, we define $S_{k,i+1}$ as follows. We pick a map $\sigma_{i+1} \colon \overline{\Sigma}^{\leq i} \times \Sigma_f \to \{Call, Read\}$ such that for all $k \in \mathbb{N}_0$ there exists some $\sigma \in S_{k,i}$ with $\sigma\big|_{\overline{\Sigma}^{\leq i} \times \Sigma_f} = \sigma_{i+1}$. Note that such σ_{i+1} exists since $S_{k,i}$ is non-empty for each k by property (vi) and decreasing in k by property (v) and there are only finitely many maps $\overline{\Sigma}^{\leq i} \times \Sigma_f \to \{Call, Read\}$. For $k \in \mathbb{N}_0$ let

$$S_{k,i+1} = \Big\{ \sigma \in S_{k,i} \;\Big|\; \sigma\big|_{\overline{\Sigma}^{\leq i} \times \Sigma_f} = \sigma_{i+1} \Big\}.$$

It is straightforward to check by induction on i that the properties (i)–(vi) are satisfied for all $k, i \in \mathbb{N}_0$.

Consider the one-pass strategy σ that plays according to σ_i during its ith move, i. e. σ is given by $\sigma(\overline{u}, a) = \sigma_{|\overline{u}|+1}(\overline{u}, a)$. We claim that σ is optimal.

Suppose it is not, then there exists a one-pass strategy σ' with $L(\sigma) \subsetneq L(\sigma')$. Then there exists $k \in \mathbb{N}$ such that $L(\sigma) \cap \Sigma^{\leq k} \subsetneq L(\sigma') \cap \Sigma^{\leq k}$. Due to the properties (i) and (ii) there exists a word w of length at most k such that all the strategies in $S_{k,0}$ win on w but σ does not win on w.

Let $\Pi = (K_0, K_1, \dots)$ be a play of σ on w that is not winning. Observe that the depth of Π is at most $B_{|w|}$: Otherwise, Π would contain a Call of nesting depth $B_{|w|} + 1$ between two configurations K_i to K_{i+1} for some $i \in \mathbb{N}$. But σ plays according to σ_{i+1} until K_{i+1} is reached, and so do all strategies in $S_{|w|,i+1}$ by definition, which are contained in $S_{|w|,0}$ by property (iv) – so we would have a contradiction to property (i) if the depth of Π were greater than $B_{|w|}$. By König's lemma this means that Π must be finite. Let K_i be the last configuration of Π. By the same arguments as before, σ plays like any strategy in $S_{k,i}$ until K_i is reached, so Π is also a play of $\sigma'' \in S_{k,i}$ on w. But $\sigma'' \in S_{k,0}$ by property (iv) and we said earlier that the strategies in $S_{k,0}$ win on w – contradiction!□

It can be deduced from the proof of Theorem 3.3 that σ is contained in all of the sets $S_{k,i}$ and in fact it is the only strategy contained in the intersection of all sets $S_{k,i}$. Note that we needed the bounded depth property to ensure that certain plays of σ are finite. Justified by a later result (cf. Lemma 3.10), we could also have chosen $S_{0,0}$ as the set of *terminating* one-pass strategies. However, if we omitted the depth requirement in the definition of $S_{k+1,0}$ and defined σ analogously, then it would be conceivable that σ is not terminating (and not in $S_{0,0}$), even though all strategies in $S_{0,0}$ are terminating.

The next lemma is a result about games that have finitely many rules or a finite target language. In a following step we will deduce from this that every game with a finite target language has the bounded depth property. There are simpler ways to show that an optimal strategy exists for games with a finite target language than showing that they have the bounded depth property (it can be shown directly, similarly to Proposition 3.1), but we will need the next lemma also later in Chapter 4.

3.4 Lemma. *Let $G = (\Sigma, R, T)$ be a game for which R or $L(T)$ is finite. For each one-pass strategy σ for G and $w \in L(\sigma)$ there exists $B_{\sigma,w} \in \mathbb{N}$ such that every play of σ on w has at most $B_{\sigma,w}$ configurations.*

Proof. Consider the *game tree* of σ on w defined as follows: Like in a play tree, the nodes of the game tree are configurations. Unlike play trees, where paths from the root to a leaf correspond to a sequence of nested Calls within a single play, paths from the root to a leaf in a game tree correspond to entire plays of σ on w. The root of the game tree is (ε, w), i. e. the initial configuration of plays of σ on w. If $(\overline{u}, av)$ is a configuration occurring in the game tree and $\sigma(\overline{u}, a) = \mathit{Read}$, then the configuration $(\overline{u}a, v)$ is the only child of this configuration. If $\sigma(\overline{u}, a) = \mathit{Call}$, then the configuration $(\overline{u}, av)$ has a child $(\overline{u}\widehat{a}, xv)$ for each $x \in L(R_a)$. By definition, the plays of σ on w are exactly the sequences of configurations on a path from the root to a leaf.

We claim that every node of the game tree has finite degree. If R is finite, then this is immediate. Suppose this is not the case for a game where R is infinite but $L(T)$ is finite. Then there exists a play of σ on w with a configuration in which Juliet plays Call and Romeo has infinitely many options for the replacement word. If $L(T)$ is finite, then Romeo could choose a replacement word that is longer than any word in $L(T)$. But then the play would not be winning, since the current string (i. e. the concatenation of the read prefix and the remaining suffix; in the last configuration this is the final string) can never shrink, since replacement words are non-empty by definition of context-free games.

Therefore, every node has finite degree and, since each play of σ on w is finite (otherwise it would not hold that $w \in L(\sigma)$), the game tree has no infinite path. By König's lemma this implies that the game tree has only finitely many nodes. If d is the depth of the game tree, then $B_{\sigma,w} = d + 1$ is a bound on the number of configurations of plays of σ on w. □

We can now deduce that games with a finite target language satisfy a condition that is even stronger than the bounded depth property.

3.5 Lemma. *Let $G = (\Sigma, R, T)$ be a game for which $L(T)$ is finite. There exists $B \in \mathbb{N}$ such that for every one-pass strategy σ for G there exists a one-pass strategy σ' for G with $L(\sigma) \subseteq L(\sigma')$ and*

$\sigma'(\overline{u},a) = \mathit{Read}$ *for each* $(\overline{u},a) \in \overline{\Sigma}^* \times \Sigma_f$ *with* $|\overline{u}| > B$. *Hence, G has the bounded depth property with the constant sequence* $(B_k) = (B)$.

Proof. Let n be the maximal length of a word in $L(T)$. Since replacement words have length at least 1, a one-pass strategy cannot win on an input word of length greater than n. Let $\mathcal{W}$ be the set of sets $W \subseteq \Sigma^{\leq n}$ for which a one-pass strategy σ_W with $L(\sigma_W) = W$ exists. For $B_{\sigma,w}$ as in Lemma 3.4, we claim that the property stated in this lemma holds for B defined as

$$B = \max\{B_{\sigma_W,w} \mid W \in \mathcal{W}, w \in W\}.$$

Indeed, for a one-pass strategy σ we can choose σ' as the one-pass strategy defined via

$$\sigma'(\overline{u},a) = \begin{cases} \sigma_W(\overline{u},a), & \text{if } |\overline{u}| \leq B, \\ \mathit{Read}, & \text{otherwise,} \end{cases}$$

where $W = L(\sigma)$. □

We conclude this section with the following theorem, which now follows easily.

3.6 Theorem. *An optimal one-pass strategy exists for every non-recursive game and every game with a finite target language.*

Proof. Immediate from Theorem 3.3, Lemma 3.5 and the fact that non-recursive games have the bounded depth property with $B_k = |\Sigma_f|$ and $\sigma_k = \sigma$. □

In the next section, we will show that also games with self-delimiting replacement languages have the bounded depth property, and therefore always an optimal strategy.

3.3. Games with self-delimiting replacement languages

Recall that a language is self-delimiting if it contains no two words where one is a proper prefix of the other. By "games with self-delimiting replacement languages" we mean games $G = (\Sigma, R, T)$ where $L(R_a)$ is self-delimiting for each $a \in \Sigma_f$. This seems to be a realistic constraint in practice: An important special case are games where replacement strings are suffixed by a special end-of-file symbol. In this sense, *every* game $G = (\Sigma, R, T)$ can be transformed into a game $G' = (\Sigma', R', T')$ with self-delimiting replacement languages by letting $\Sigma' = \Sigma \cup \{\$\}$ for some new symbol $\$ \notin \Sigma$ that shall denote the end of replacement strings, and further letting $R'_a = R_a\$$ for each $a \in \Sigma_f$ to enforce that replacement words end with \$ and adding a loop transition for the symbol \$ to each state of T (accomplishing that the symbol \$ is "ignored" by the target language). Another special case of games with self-delimiting replacement languages, which is similar to the one-pass with size setting, are games where the alphabet Σ contains (besides other symbols) numbers $1,\ldots,N$ for some $N \in \mathbb{N}$ and all replacement strings are of the form nx where $x \in \Sigma^+$ and $n = |x|$.

We need some definitions and several lemmas before we obtain the result that games with self-delimiting replacement languages have the bounded depth property. These results and definitions will also be helpful later in Chapter 4. Some of them apply to context-free games in general and not only games with self-delimiting replacement languages.

3.7 Definition. For $w \in \Sigma^*$ let H_w be the set of final history strings of finite plays on w.

The next lemma says, intuitively, that in games with self-delimiting replacement languages, Juliet "can tell" when a prefix of the input word is completely processed. Indeed, in a play on some input word uv, the prefix u is completely processed when a configuration with a history string in H_u is reached. Due to the next lemma, this configuration is unique for each play.

3.8 Lemma. *Let $G = (\Sigma, R, T)$ be a game with self-delimiting replacement languages. For each $w \in \Sigma^*$, the set H_w is self-delimiting.*

Proof. We first consider the case that w is a single symbol. We show for each $a \in \Sigma$ and $\overline{w} \in H_a$ that there exists no $\overline{w}' \in H_a$ such that $\overline{w}$ is a proper prefix of $\overline{w}'$ or $\overline{w}'$ is a proper prefix of $\overline{w}$. The proof is by induction on $|\overline{w}|$.

The base case is that $|\overline{w}| = 1$. In this case, $\overline{w} = a$ and any symbol in H_a that does not equal a begins with $\widehat{a}$, so the statement is true.

If $|\overline{w}| > 1$ then $\overline{w} = \widehat{a}\overline{u}$ where $\overline{u} \in H_x$ for some $x \in L(R_a)$. Suppose for the sake of contradiction that there exists $\overline{w}' \in H_a$ such that $\overline{w}$ is a proper prefix of $\overline{w}'$ or vice versa. Then $\overline{w}' = \widehat{a}\overline{v}$ where $\overline{v} \in H_y$ for some $y \in L(R_a)$ and $\overline{u}$ is a proper prefix of $\overline{v}$ or vice versa. Let $x = a_1 \dots a_k$ and $y = b_1 \dots b_\ell$ with $a_i, b_j \in \Sigma$. Then $\overline{u} = \overline{u_1 u_2} \dots \overline{u_k}$ and $\overline{v} = \overline{v_1 v_2} \dots \overline{v_\ell}$ where $\overline{u_i} \in H_{a_i}$ for $i = 1, \dots, k$ and $\overline{v_j} \in H_{b_j}$ for $j = 1, \dots, \ell$. Since $\overline{u_1}$ and $\overline{v_1}$ are non-empty and one must be a prefix of the other (not necessarily a proper prefix), the first symbol of $\overline{u_1}$ must equal the first symbol of $\overline{v_1}$. But the first symbol of $\overline{u_1}$ is a_1 or $\widehat{a_1}$ and the first symbol of $\overline{v_1}$ is b_1 or $\widehat{b_1}$. Thus, $a_1 = b_1$. By the induction hypothesis, $\overline{u_1}$ cannot be a proper prefix of $\overline{v_1}$ or vice versa, therefore $\overline{u_1} = \overline{v_1}$. By repeating the argument, we see that $a_i = b_i$ and $\overline{u_i} = \overline{v_i}$ for each $i = 1, \dots, \min\{k, \ell\}$ and x is a proper prefix of y or vice versa. But this contradicts the premise that replacement languages are self-delimiting.

Now we consider the case that $w \in \Sigma^*$. Suppose there exist $\overline{w}, \overline{w}' \in H_w$ such that $\overline{w}$ is a proper prefix of $\overline{w}'$. Let $w = a_1 \dots a_n$ with $a_i \in \Sigma$. Then $\overline{w} = \overline{u_1 u_2} \dots \overline{u_n}$ and $\overline{w}' = \overline{u_1} \dots \overline{u_{k-1} u_k}' \dots \overline{u_n}'$ where $\overline{u_k}, \overline{u_k}' \in H_{a_k}$ and $\overline{u_k}$ is a proper prefix of $\overline{u_k'}$ or vice versa. But this is not possible since we have already shown that H_{a_k} is self-delimiting. □

The previous lemma is helpful especially because it allows us to define one-pass strategies that play in a certain way *as soon as* a particular prefix of the input word is completely processed. The next lemma states that Juliet can even construct the play tree as she plays, provided replacement languages are self-delimiting. Recall that this is not the case in general (cf. Example 2.1).

3.9 Lemma. *Let $\Pi = (K_0, K_1, \dots)$ be a play of a game $G = (\Sigma, R, T)$ with self-delimiting replacement languages. The structure of the subtree of $\mathcal{T}(\Pi)$ containing the configurations $K_0, \dots, K_n$ is uniquely determined by G and the history string of K_n.*

Proof. Recall that the history string of K_n contains all information about the history strings and current symbols of the configurations $K_0, \dots, K_{n-1}$.

Since the sets H_a are self-delimiting by Lemma 3.8, the subtree rooted at a configuration with history string $\overline{u_1}$ and current symbol a contains exactly the configurations that have a history string $\overline{u_1 u_2}$ where $\overline{u_2}$ is a proper prefix of a string in H_a. The node K_n has no children in the considered

subtree of $\mathcal{T}(\Pi)$. Thus, the history string of K_n (and G) determine the set of ancestors of each configuration $K_0, \ldots, K_n$. If two configurations K_i and K_j have the same set of ancestors, then K_i is placed to the left of K_j if and only if $i < j$. The lemma follows easily. □

The following lemma yields a convenient property of one-pass strategies that holds in general and not only for games with self-delimiting replacement languages.

3.10 Lemma. *For each one-pass strategy σ for a game G there exists a terminating strategy σ' with $L(\sigma) \subseteq L(\sigma')$.*

Proof. The strategy σ' plays like σ except that once it is apparent that Romeo can force infinite play against σ, all remaining moves of σ' are Read.

For a formal definition, consider some $\overline{u} \in \overline{\Sigma}^*$ and $a \in \Sigma_f$. If there exists a play Π of σ that contains a configuration $(\overline{u}, av)$ for some $v \in \Sigma^*$ such that no later configuration with remaining string v occurs in Π, then let $\sigma'(\overline{u}, a) = \mathit{Read}$ and $\sigma'(\overline{u}a\overline{v}, b) = \mathit{Read}$ for each $\overline{v} \in \overline{\Sigma}^*$ and $b \in \Sigma_f$. For all elements of the domain $\overline{\Sigma}^* \times \Sigma_f$ for which σ' is not already defined by this (for any $\overline{u} \in \overline{\Sigma}^*$ and $a \in \Sigma_f$), we define σ' like σ.

Let τ be some Romeo strategy and $w \in \Sigma^*$. We need to show that $\Pi(\sigma', \tau, w)$ is finite and, if $w \in L(\sigma)$, then $\Pi(\sigma', \tau, w)$ is winning.

If $\Pi(\sigma, \tau, w)$ is finite, then for each configuration $(\overline{u}, av)$ of $\Pi(\sigma, \tau, w)$ there exists a later configuration $(\overline{u}a\overline{v}, v)$ or $\overline{u}\widehat{a\overline{v}}, v)$. Thus, $\Pi(\sigma, \tau, w) = \Pi(\sigma', \tau, w)$. So $\Pi(\sigma', \tau, w)$ finite and if $w \in L(\sigma)$ then $\Pi(\sigma', \tau, w)$ is winning.

If $\Pi(\sigma, \tau, w)$ is infinite, then it contains a configuration $(\overline{u}, av)$ such that no later configuration $(\overline{u}a\overline{v}, v)$ or $(\overline{u}\widehat{a\overline{v}}, v)$ for any $\overline{v} \in \overline{\Sigma}^*$ occurs in $\Pi(\sigma, \tau, w)$. If $\Pi(\sigma', \tau, w)$ also contains this configuration then all subsequent moves of σ' are Read, so $\Pi(\sigma', \tau, w)$ is finite. If $\Pi(\sigma', \tau, w)$ does not contain this configuration, then σ' already started playing only Read moves in an earlier configuration, so $\Pi(\sigma', \tau, w)$ is again finite. Since $\Pi(\sigma, \tau, w)$ is infinite, $w \notin L(\sigma)$, so we are done. □

A concept that we will use in various proofs is the convergence of a sequence of one-pass strategies.

3.11 Definition. A sequence $(\sigma_k)_{k \in \mathbb{N}}$ of strategies *converges to* a strategy σ if for each $n \in \mathbb{N}$ there exists $k_0 \in \mathbb{N}$ such that for each $k \geq k_0$ and $(\overline{u}, a) \in \overline{\Sigma}^* \times \Sigma$ with $|\overline{u}| \leq n$ it holds that $\sigma(\overline{u}, a) = \sigma_k(\overline{u}, a)$.

This is the same as the convergence in the metric space of strategies where the metric d is given by $d(\sigma, \sigma) = 0$ and $d(\sigma, \sigma') = \frac{1}{n}$ for $\sigma \neq \sigma'$, where n is minimal such that there exists $(\overline{u}, a) \in \overline{\Sigma}^* \times \Sigma$ with $|\overline{u}| = n$ and $\sigma(\overline{u}, a) \neq \sigma'(\overline{u}, a)$. A property of converging sequences of one-pass strategies is given by the following Lemma.

3.12 Lemma. *Let $(\sigma_k)_{k \in \mathbb{N}}$ be a sequence of one-pass strategies that converges to some one-pass strategy σ. If σ loses on a word $w \in \Sigma^*$ then infinitely many strategies of the sequence also lose on w.*

Proof. Suppose σ loses on a word $w \in \Sigma^*$. Then there exists a losing (and thus finite) play $\Pi = (K_0, \ldots, K_n)$ of σ on w. Let $k_0 \in \mathbb{N}$ be such that for each $k \geq k_0$ and $(\overline{u}, a) \in \overline{\Sigma}^* \times \Sigma$ with $|\overline{u}| \leq n$ it holds that $\sigma(\overline{u}, a) = \sigma_k(\overline{u}, a)$. Since all configurations of Π have a history string of length at most n, it follows that Π is also a play of σ_k on w for each $k \geq k_0$. So σ_k also loses on w for each $k \geq k_0$. □

3.13 Definition. For $w \in \Sigma^*$ and a one-pass strategy σ, let $H_w^\sigma \subseteq H_w$ be the set of final history strings of finite plays of σ on w. If $T = (Q, \Sigma, \delta, q_0, F)$ is a DFA (!) for the target language, we define for $q \in Q$, $w \in \Sigma^*$ and a one-pass strategy σ the set $\mathit{states}(q, w, \sigma) = \{\delta^*(q, \natural\overline{w}) \mid \overline{w} \in H_w^\sigma\}$. This is the set of T-states that can be reached at the end of a play of σ on w if the initial state of T were q.

Since H_w^σ is a subset of H_w, the Lemmas 3.8 also holds for H_w^σ instead of H_w. Observe that if T is a DFA and σ is terminating, then $w \in L(\sigma)$ if and only if $\mathit{states}(q_0, w, \sigma) \subseteq F$.

Finally, we can show that games with self-delimiting replacement languages have the bounded depth property. In fact, they have a stronger property:

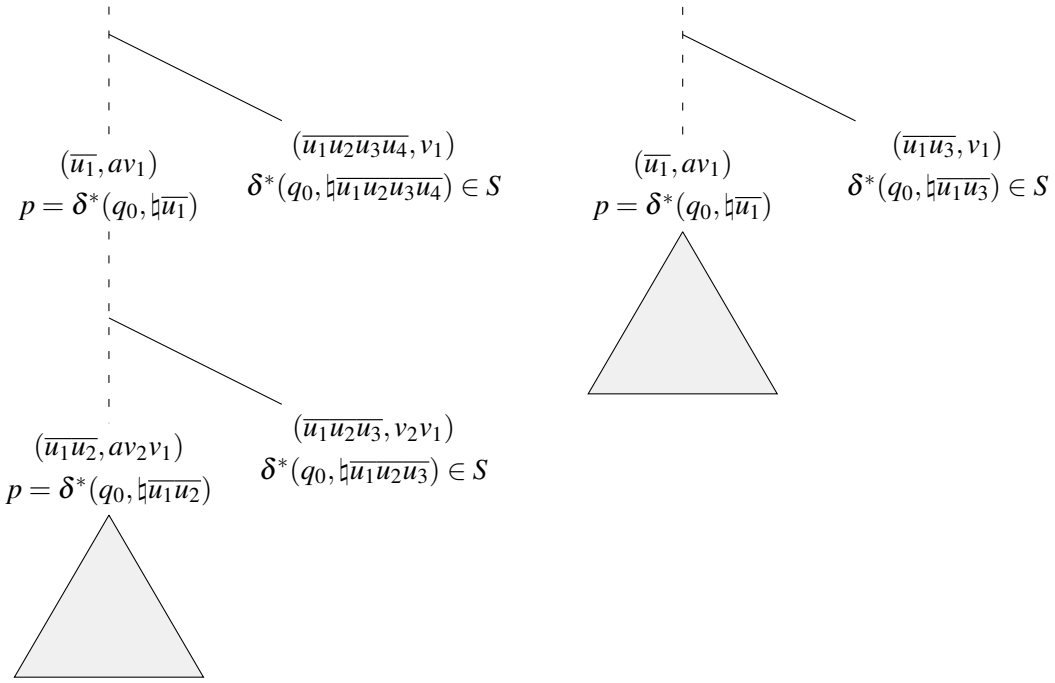
3.14 Lemma. *Let $G = (\Sigma, R, T)$ be a game with self-delimiting replacement languages and σ a one-pass strategy for G. Then there exists a strategy σ' with depth bounded by $|\Sigma_f| \cdot |Q| \cdot (2^{|Q|} - 1)$ and $L(\sigma) \subseteq L(\sigma')$, where $|Q|$ is the number of states of the minimal DFA for $L(T)$.*

Proof. By Lemma 3.10 we can assume that σ is terminating. Further, we assume without loss of generality that $T = (Q, \Sigma, \delta, q_0, F)$ is already the minimal DFA for $L(T)$.

We associate with a configuration K of a play Π with history string $\overline{u} \in \overline{\Sigma}^*$ and current symbol $a \in \Sigma$ the triple (p, a, S) consisting of

- the T-state $p = \delta^*(q_0, \natural\overline{u})$ of K,
- the current symbol a of K
- the set $S = \mathit{states}(p, a, \sigma[\overline{u}])$ of possible T-states of the next configuration of Π that is not part of the subtree of $\mathcal{T}(\Pi)$ rooted at K.

If a triple (p, a, S) with $a \in \Sigma \setminus \Sigma_f$ is associated to a configuration that occurs in a play of σ, then this configuration is a leaf of the play tree. Suppose the depth of σ is greater than $|\Sigma_f| \cdot |Q| \cdot (2^{|Q|} - 1)$. Since S is always non-empty, this means that there exists a play of σ such that the corresponding play tree has a path starting at the root that contains two configurations $(\overline{u_1}, av_1)$ and $(\overline{u_1 u_2}, av_2v_1)$ with the same associated triple (p, a, S), where $a \in \Sigma_f$. The idea of this proof is, roughly speaking, to change σ so as to replace the subtree rooted at the upper node by the subtree rooted at the lower node (see Figure 3.1).



(a) A possible play of σ (b) A possible play of σ'

Figure 3.1.: Schematic representation of possible plays of σ and σ' as constructed in the proof of Lemma 3.14

For each $q \in S$ we choose $\overline{a_q} \in H_a^{\sigma[\overline{u_1}]}$ such that $q = \delta^*(p, \natural\overline{a_q})$. We define a one-pass strategy σ' via

$$\sigma'(\overline{u}, b) = \begin{cases} \sigma(\overline{u_1u_2u'}, b), & \text{if } \overline{u} = \overline{u_1u'} \text{ such that either } (\overline{u'}, b) = (\varepsilon, a) \text{ or } \overline{u'} \in \overline{\Sigma}^+ \text{ is} \\ & \text{a proper prefix of some } \overline{u_3} \in H_a^{\sigma[\overline{u_1u_2}]}, \\ \sigma(\overline{u_1a_qu'}, b), & \text{if } \overline{u} = \overline{u_1u_3u'} \text{ and } q = \delta^*(p, \natural\overline{u_3}) \text{ for some } \overline{u_3} \in H_a^{\sigma[\overline{u_1u_2}]} \\ & \text{and } \overline{u'} \in \overline{\Sigma}^*, \\ \sigma(\overline{u}, b), & \text{otherwise.} \end{cases}$$

So σ' starts playing like σ, but once it reaches a configuration with history string $\overline{u_1}$ and current symbol a, it continues to play as if the history were $\overline{u_1u_2}$ instead of $\overline{u_1}$, simulating how σ would play in the subtree rooted at $(\overline{u_1u_2}, av_2v_1)$; once that subtree is left and the T-state is some $q \in S$ (i. e. when a history $\overline{u_1u_3}$ is reached with $\overline{u_3} \in H_a^{\sigma[\overline{u_1u_2}]}$ and $q = \delta^*(p, \natural\overline{u_3})$), σ' continues to play like σ when the subtree rooted at K is left and the T-state is q. Note that σ' is well-defined since $H_a^{\sigma[\overline{u_1u_2}]}$ is self-delimiting.

We show that σ' is terminating and that $L(\sigma) \subseteq L(\sigma')$. A play Π of σ' on some word $w \in \Sigma^*$ that is not also a play of σ on w must contain a configuration $(\overline{u_1}, av)$ for $v \in \Sigma^*$. Since σ is terminating, Π also contains a configuration $(\overline{u_1u_3}, v)$ where $\overline{u_3} \in H_a^{\sigma[\overline{u_1u_2}]}$. It follows easily from the definition of σ' and the fact that σ is terminating that Π is finite, hence σ' is terminating. Suppose that

$w \in L(\sigma)$ but Π is losing. Then, for $q = \delta^*(p, \natural\overline{u_3})$, it would hold that $states(p, av, \sigma[\overline{u_1}]) \subseteq F$ but $states(q, v, \sigma'[\overline{u_1 u_3}]) \nsubseteq F$. However, this is not possible since

$$\begin{aligned} states(q, v, \sigma'[\overline{u_1 u_3}]) &= states(q, v, \sigma[\overline{u_1 a_q}]) \\ &\subseteq \bigcup_{\overline{a} \in H_a^{\sigma[\overline{u_1}]}} states(\delta^*(p, \natural\overline{a}), v, \sigma[\overline{u_1 a}]) = states(p, av, \sigma[\overline{u_1}]). \end{aligned}$$

So we have shown that if a play of σ' on $w \in L(\sigma)$ is not a play of σ on w, then it is still finite and not losing, hence winning. Therefore, $L(\sigma) \subseteq L(\sigma')$.

In order to transform σ into a strategy with depth bounded by $|\Sigma_f| \cdot |Q| \cdot (2^{|Q|} - 1)$, we repeat this kind of adjustment several times. For the first adjustment, we pick a play Π of σ and configurations $K = (\overline{u_1}, av_1)$ and $K' = (\overline{u_1 u_2}, av_2 v_1)$ as above such that $|\overline{u_1}|$ is minimal and such that there exists no play of σ that contains the configuration K' and another configuration with associated triple (p, a, S) in the subtree rooted at K'. Such K' must exist: Suppose not, then Romeo could, after reaching K', select his replacement words so as to reach another configuration K'' with triple (p, a, S) in the subtree rooted at K' and, after reaching K'', choose the new replacement words to reach yet another configuration with triple (p, a, S) in the subtree rooted at K'' and so on. But if Romeo can repeat this procedure ad infinitum, then there would exist an infinite play of σ, but this contradicts that σ is terminating.

So the first adjustment of σ ensures that if a configuration K with current symbol a and history string $\overline{u_1}$ occurs in a play of σ', then no configuration with the same associated triple can occur in the subtree rooted at K. Moreover, any triple occurring in the subtree rooted at K could also previously occur in the subtree rooted at K (namely in the subtree rooted at K', which was part of the subtree rooted at K), so they cannot cause any new duplicate triple on a path, since the possible ascendants of K stay the same by Lemma 3.9. Also any other triple associated to a descendant of an ascendant of K (to the right of K after the subtree rooted at K is left) could previously occur in the same place of the play tree because the play of σ' after leaving the subtree rooted at K simulates a play of σ after leaving this subtree. Therefore, since $|\overline{u_1}|$ was chosen minimal, it is still not possible for configurations with a history string of length less than $|\overline{u_1}|$ to have a configuration with the same associated triple in the subtree below them.

After making finitely many such adjustments (at most once for each combination of a current symbol and a history string of length $|\overline{u_1}|$), it is not possible for any configuration with a history string of length at most $|\overline{u_1}|$ to have a configuration with the same associated triple in the subtree below it. Inductively we can guarantee for *any* $k \in \mathbb{N}_0$ that there exists a terminating one-pass strategy σ_k with $L(\sigma) \subseteq L(\sigma_k)$ such that no configuration with a history string of length k has a configuration with the same associated triple in the subtree below it. This induces a sequence (σ_k) of one-pass strategies that converges to some strategy σ' which satisfies this property for each $k \in \mathbb{N}_0$. Therefore, the depth of σ' is bounded as stated in the Lemma. If σ' loses on some word $w \in \Sigma^*$, then it follows from Lemma 3.12 that $w \notin L(\sigma_k)$ for infinitely many k and therefore $w \notin L(\sigma)$, since $L(\sigma) \subseteq L(\sigma_k)$. But σ' is terminating (due to its bounded depth), so it loses on any word on which it does not win. Hence, $L(\sigma) \subseteq L(\sigma')$. □

It is possible to achieve a slightly better bound on the depth than the one proved in the lemma. Similar to what will be done in Lemma 4.6, it would be possible to change σ to an at least as good strategy for which there never occur two triples (p,a,S) and (p,a,S') where S is a proper subset of S'. Since the improved bound this would yield is still exponential in $|Q|$, we will settle for the bound from Lemma 4.6 rather than dealing with more complicated terms.

With the results above, we can now state the following theorem.

3.15 Theorem. *An optimal one-pass strategy exists for every games with self-delimiting replacement languages.*

Proof. Games with self-delimiting replacement languages have the bounded depth property with $B_k = |\Sigma_f| \cdot |Q| \cdot (2^{|Q|} - 1)$ by Lemma 3.14. The result follows from Theorem 3.3. □

In the next chapter, we will see – among other things – whether we can deduce from the existence of an optimal type 1 strategy the existence of optimal strategies of restricted types.

4. Implications between strategy types

Recall the five types of one-pass strategies defined in Section 2.2. In search of good one-pass strategies, it would be helpful to know that one can restrict the search to one of the restricted types. For instance, if it were clear for a game that there must be an optimum strategy of type 5 or else there exists no optimum one-pass strategy at all, then it would be sufficient to compute all type 5 strategies, compare them and choose the best. Since there exist only finitely many type 5 strategies for each game, this procedure would terminate (we will study the complexity of comparing two automaton strategies in Chapter 6). Therefore, an interesting question is whether the existence of an optimum (or optimal) strategy of one type implies the existence of an optimum (or optimal) strategy of a more restricted type. Similarly one can ask whether, for a word w, the existence of a strategy of one type that wins on w implies the existence of a strategy of a restricted type that wins on w. We will study these questions in this chapter, obtaining some positive and some negative results, while some questions remain open.

Let us first introduce some terminology. Let $G = (\Sigma, R, T)$ be a game and $X, Y \in \{1,2,3,4,5\}$. We say

$$\textit{the implication } X \Longrightarrow Y \textit{ for optimum strategies holds for } G$$

if the existence of an optimum strategy of type X for G implies the existence of an optimum strategy of type Y for G. Worded differently, the implication $X \Longrightarrow Y$ for optimum strategies holds for G if the existence of an optimum strategy of type X for G implies the existence of a strategy of type Y for G that is equivalent to some optimum strategy (and therefore equivalent to all optimum strategies). We say

$$\textit{the implication } X \Longrightarrow Y \textit{ for optimum strategies semi-holds for } G$$

if the existence of an optimum strategy of type X for G implies the existence of a strategy of type Y for G that is semi-equivalent to some optimum strategy. We use the analogous definitions where "optimum" is replaced by "optimal". Note, however, that for an implication $X \Longrightarrow Y$ for optimal strategies to (semi-)holds, we do *not* require that *every* optimal type X strategy has a (semi-)equivalent type Y strategy, but only that at least one of them (if it exists) has a (semi-)equivalent type Y strategy. This seems like a slightly weaker statement, since it is conceivable that there are several inequivalent optimal type X strategies, but only some of them have a (semi-)equivalent strategy of type Y. Finally, we say

$$\textit{the implication } X \Longrightarrow Y \textit{ for winning strategies holds for } G$$

if, for each $w \in \Sigma^*$, the existence of a type X strategy that wins on w implies the existence of a type Y strategy that wins on w.

In the sections of this chapter, we study which of these implications hold or semi-hold for games with certain properties. Of course, an implication $X \Longrightarrow Y$ holds trivially if type X is a restriction of type Y.

Without loss of generality, we assume for each game $G = (\Sigma, R, T)$ considered in this chapter that $T = (Q, \Sigma, \delta, q_0, F)$ is the minimal DFA for the target language. Note that the existence of strategies of the different types is not affected by how the target language is represented. We allow T to be an NFA again in the next chapters when we deal with complexity questions, since the representation of the target language will make a difference there.

4.1. Implications for optimum strategies

To prepare for our first theorem in this section, we need to define the notion of indistinguishability of one-pass strategies.

4.1 Definition. Two one-pass strategies σ and σ' are *indistinguishable* if $\Pi(\sigma, \tau, w) = \Pi(\sigma', \tau, w)$ for each Romeo strategy τ and $w \in \Sigma^*$. Since indistinguishability is an equivalence relation, we can consider equivalence classes, which we call *indistinguishability classes*.

In other words, the decisions of indistinguishable strategies may only differ on pairs $(\overline{u}, a) \in \overline{\Sigma}^*$ that do not occur as the history string and current symbol of any configuration of a play. Of course, if two strategies are indistinguishable, then they are also equivalent (i. e. winning on the same input words). In particular, if a strategy is optimum (or optimal), then all strategies of its indistinguishability class are optimum (or optimal). The converse is not generally true, i. e. two strategies may be equivalent but not indistinguishable. For a trivial counter-example consider $L(T) = \emptyset$.

According to a theorem by Abiteboul et al. [3, Theorem 3.1], if a game has a unique optimum strategy, then this is a type 4 strategy, and if a game has an optimum strategy at all then it is semi-equivalent to a strategy of type 4; furthermore, provided a condition of "1-unambiguity" holds, the type 4 strategy is even of type 5. The definition of semi-equivalence in [3] differs slightly from ours, as they defined two strategies as semi-equivalent if they lose on the same strings. However, the following example shows that an optimum strategy does actually not always have a semi-equivalent (as per the definition in [3]) strategy of type 4 or even of type 3.

4.2 Example. Consider the game $G = (\Sigma, R, T)$ with $\Sigma = \{a, b\}$, R given by $a \to aa$ and the target language $L(T)$ consisting of all strings that start with aa and contain b. An optimum one-pass strategy σ for G is given by

$$\sigma(\overline{u}, a) = \begin{cases} \mathit{Call}, & \text{if } \overline{u} = \varepsilon \\ \mathit{Read}, & \text{if } \overline{u} \neq \varepsilon, \end{cases}$$

for $\overline{u} \in \{a, b, \widehat{a}\}^*$, i. e. σ plays Call once if the first symbol of the input word is a and otherwise Read. This strategy wins on all strings that start with a and contain b, and it is clear that this is optimum. A strategy σ' that is semi-equivalent to σ must satisfy $\sigma'(\varepsilon, a) = \mathit{Call}$ because otherwise σ' loses

on $ab \in L(\sigma)$. However, if σ' is of type 3 (and in particular if it is of type 4) then $\sigma'(\overline{u}, a) = \mathit{Call}$ for each $\overline{u} \in \{\widehat{a}\}^*$. Then, any play of σ' on a is infinite, so σ' does not lose on a. But σ loses on a, so σ and σ' are not semi-equivalent as per the definition in [3].

Barring our different definition of semi-equivalence, we can generalize the theorem by Abiteboul et al. [3, Theorem 3.1]. It is possible to drop the condition of 1-unambiguity, and rather than requiring a unique optimum strategy it suffices to require that there are only finitely many of them in order for one of them to be of type 5. Besides finitely many optimum strategies we also obtain two other sufficient conditions.

4.3 Theorem. *For each game $G = (\Sigma, R, T)$, all implications for optimum strategies semi-hold. Moreover, all implications for optimum strategies* hold *for G, if any of the following conditions is satisfied:*

- *G is non-recursive.*
- *$L(T)$ is finite.*
- *Only finitely many indistinguishability classes of optimum strategies for G exist.*

Proof. Since type 1 (all one-pass strategies) is the most general and type 5 the most restricted type, it suffices to show the statements for the implication $1 \Longrightarrow 5$. We assume that an optimum one-pass strategy exists because otherwise there is nothing to show.

Starting with some optimum strategy σ_1, we will construct a sequence (σ_k) of optimum strategies that is either finite and ends with a strategy that is indistinguishable from a type 5 strategy or is infinite and converges to such a strategy. The plan is to define σ_{k+1} by adjusting σ_k so as to postpone the earliest possible configuration that is a witness of why σ_k is *not* indistinguishable from a type 5 strategy.

The sequence (σ_k) starts with an arbitrary optimum one-pass strategy σ_1. Let σ_k be a strategy in the sequence. We distinguish two cases.

If – for all $q \in Q$ and $a \in \Sigma_f$ and all configurations $(\overline{u_1}, av_1)$ and $(\overline{u_2}, av_2)$ with $q = \delta^*(q_0, \natural\overline{u_1}) = \delta^*(q_0, \natural\overline{u_2})$ occurring in plays of σ_k – it holds that $\sigma_k(\overline{u_1}, a) = \sigma_k(\overline{u_2}, a)$, then σ_k is the last strategy of the sequence. In this case σ_k is indistinguishable from the strategy $\sigma_{\mathcal{A}}$ of the type 5 automaton $\mathcal{A} = (Q, \Sigma, \delta_{\mathcal{A}}, q_0, F)$, where

$$\delta_{\mathcal{A}}(q,a) = \begin{cases} \bot, & \text{if } a \in \Sigma_f \text{ and a configuration } (\overline{u}, av) \text{ with } q = \delta^*(q_0, \natural\overline{u}) \text{ occurs in} \\ & \text{some play of } \sigma_k \text{ such that } \sigma_k(\overline{u}, a) = \mathit{Call}, \\ \delta(q,a), & \text{otherwise} \end{cases}$$

for $q \in Q$ and $a \in \Sigma$. Indeed, due to the case we are considering, if $\delta_{\mathcal{A}}(q,a) = \bot$, then *all* configurations $(\overline{u}, av)$ with $q = \delta^*(q_0, \natural\overline{u})$ occurring in some play of σ_k satisfy that $\sigma_k(\overline{u}, a) = \mathit{Call}$. Therefore, if a configuration $(\overline{u}, av)$ occurs in a play of σ_k, then $\sigma_k(\overline{u}, a) = \sigma_{\mathcal{A}}(\overline{u}, a)$ by definition of type 4 and 5 strategies. Hence, σ_k and $\sigma_{\mathcal{A}}$ are indistinguishable.

In the other case, there exist $q \in Q$ and $a \in \Sigma_f$ and plays $\Pi^1 = (K_0^1, K_1^1, K_2^1, \dots)$ and $\Pi^2 = (K_0^2, K_1^2, K_2^2, \dots)$ of σ_k such that $K_{n_1}^1 = (\overline{u_1}, av_1)$, $K_{n_2}^2 = (\overline{u_2}, av_2)$, $q = \delta^*(q_0, \natural\overline{u_1}) = \delta^*(q_0, \natural\overline{u_2})$ and

$\sigma_k(\overline{u_1},a) \neq \sigma_k(\overline{u_2},a)$ for some $n_1,n_2 \in \mathbb{N}_0$, $\overline{u_1},\overline{u_2} \in \overline{\Sigma}^*$ and $v_1,v_2 \in \Sigma^*$. We choose these such that $n_1 \leq n_2$ and (n_2,n_1) is lexicographically minimal. Informally, this means that σ_k acts like a type 5 strategy during the first n_2 moves. We need the following claim.

4.4 Claim. *The play Π^1 starts with n_1 Read moves, i. e. $\overline{u_1} = \natural\overline{u_1} \in \Sigma^*$.*

Proof. First, we show that a play of σ_k on $\natural\overline{u_1}$ consists of only Read moves, i. e. $\sigma_k(u,b) = \mathit{Read}$ for each $u,v \in \Sigma^*$ and $b \in \Sigma_f$ with $\natural\overline{u_1} = ubv$. Suppose not and choose u, v and b that constitute a counter-example with u of minimal length. Let $\overline{u},\overline{v} \in \overline{\Sigma}^*$ be such that $\overline{u_1} = \overline{u}b\overline{v}$, $\natural\overline{u} = u$ and $\natural\overline{v} = v$. Then a play of σ_k on ub with its configuration (u,b) and the play Π^1 with the configuration $K^1_{|\overline{u}|}$ would contradict the minimality of n_2, since $|u| \leq |\overline{u}| < |\overline{u_1}| = n_1 \leq n_2$.

Now we show that $\overline{u_1} = \natural\overline{u_1}$. If this were not the case, then $|\natural\overline{u_1}| < |\overline{u_1}|$ and we would obtain a contradiction to the lexicographical minimality of (n_2,n_1) by considering a play of σ_k on $\natural\overline{u_1}a$ either instead of Π^1, if $\sigma_k(\natural\overline{u_1},a) = \sigma_k(\overline{u_1},a)$, or instead of Π^2, if $\sigma_k(\natural\overline{u_1},a) = \sigma_k(\overline{u_2},a)$. □

We define σ_{k+1} as the strategy that plays like σ_k except that if after n_2 moves it reaches a configuration with T-state q and current symbol a, then it continues to play like σ_k would play if the history string were $\overline{u_1}$. Formally,

$$\sigma_{k+1}(\overline{v},b) = \begin{cases} \sigma_k(\overline{u_1},a), & \text{if } |\overline{v}| = n_2,\ \delta^*(q_0,\natural\overline{v}) = q \text{ and } b = a, \\ \sigma_k(\overline{u_1v_2},b), & \text{if } \overline{v} = \overline{v_1v_2} \text{ for some } \overline{v_1},\overline{v_2} \in \overline{\Sigma}^* \text{ where } |\overline{v_1}| = n_2, \\ & \delta^*(q_0,\natural\overline{v_1}) = q \text{ and the first symbol of } \overline{v_2} \text{ is } a \text{ or } \widehat{a}, \\ \sigma_k(\overline{v},b), & \text{otherwise} \end{cases}$$

for $\overline{v} \in \overline{\Sigma}^*$ and $b \in \Sigma_f$.

We prove by induction that all strategies in the sequence (σ_k) are optimum. The base case is immediate from our choice of σ_1. The following induction step is illustrated in Figure 4.1. Suppose σ_k is optimum but σ_{k+1} is not. Then there exist $w \in L(\sigma_k)$ and a Romeo strategy τ such that $\Pi(\sigma_{k+1},\tau,w)$ is not winning. Thus, the plays of σ_{k+1} and σ_k against τ on w reach a configuration $(\overline{v},av)$ after n_2 moves with $\overline{v} \in \overline{\Sigma}^*$ and $v \in \Sigma^*$ such that $\delta^*(q_0,\natural\overline{v}) = q$ and σ_k does not win on $\overline{u_1}av$. Since σ_k is optimum, no strategy wins on $\overline{u_1}av$. However, consider the strategy $\sigma_k[\overline{u_1}/\overline{v}]$ that we define by

$$\sigma_k[\overline{u_1}/\overline{v}](\overline{u},b) = \begin{cases} \sigma_k(\overline{u},b), & \text{if } \overline{u_1} \text{ is not a prefix of } \overline{u}, \\ \sigma_k(\overline{vu_3},b), & \text{if } \overline{u} = \overline{u_1u_3} \text{ for some } \overline{u_3} \in \overline{\Sigma}^*. \end{cases}$$

This strategy does the following on the input word $\overline{u_1}av$:

- Initially, $\sigma_k[\overline{u_1}/\overline{v}]$ plays only Read moves until the prefix $\overline{u_1}$ is completely processed (just like σ_k).
- Upon reaching the configuration $(\overline{u_1},av)$, the strategy $\sigma_k[\overline{u_1}/\overline{v}]$ starts playing like σ_k would play after reaching the configuration $(\overline{v},av)$.

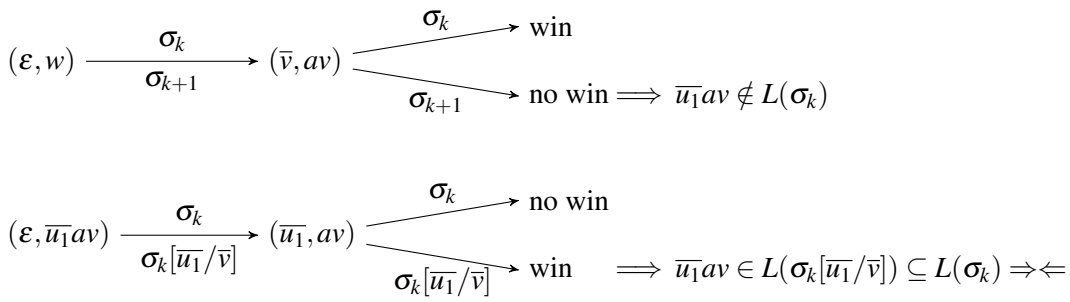


Figure 4.1.: Illustration of the proof by contradiction that σ_{k+1} is optimum in the proof of Theorem 4.3

Since there exist plays of σ_k on w that contain the configuration $(\overline{v}, av)$ (e. g. $\Pi(\sigma_k, \tau, w)$) and all these plays are winning independently of Romeo's subsequent moves, the strategy $\sigma_k[\overline{u_1}/\overline{v}]$ wins on $\overline{u_1}av$. Contradiction! So as claimed, (σ_k) is a sequence of optimum strategies.

Let $n_2(k)$, $a(k)$ and $q(k)$ be the values of n_2, a and q respectively used to define σ_{k+1}. The change from σ_k to σ_{k+1} does not affect the first $n_2(k)$ moves, i. e.

$$\sigma_{k+1}(\overline{v}, b) = \sigma_k(\overline{v}, b) \text{ for each } \overline{v} \in \overline{\Sigma}^* \text{ and } b \in \Sigma_f \text{ with } |\overline{v}| < n_2(k). \tag{4.1}$$

Hence, $n_2(k)$ is monotonously non-decreasing. Moreover, the definition of σ_{k+1} ensures that $(n_2(k), a(k), q(k)) \neq (n_2(\ell), a(\ell), q(\ell))$ for all $k < \ell$. Thus, $n_2(k)$ increases by at least 1 per $|\Sigma_f||Q|$ iterations.

If the sequence (σ_k) is finite then it ends with an optimum strategy that is indistinguishable from a type five strategy $\sigma_{\mathcal{A}}$. In this case, σ_1 is equivalent to $\sigma_{\mathcal{A}}$. Since the strategies of the sequence are pairwise not indistinguishable, the sequence must be finite if there are only finitely many indistinguishability classes. This proves the statement of the theorem for the case that the condition of the third bullet point is satisfied.

If (σ_k) is infinite, consider the strategy σ given by $\sigma(\overline{u}, a) = \sigma_k(\overline{u}, a)$ where k is large enough such that $n_2(k) > |\overline{u}|$. By property (4.1), σ is well-defined and σ_k converges to σ. Similarly as earlier in this proof it can be seen that σ is indistinguishable from some type 5 strategy $\sigma_{\mathcal{A}}$. We claim that σ_1 is semi-equivalent to $\sigma_{\mathcal{A}}$: If $\sigma_{\mathcal{A}}$ loses on a word w, then so does σ. By Lemma 3.12 this implies that some σ_k does not win on w, so since σ_k is optimum also σ_1 does not win on w. Conversely, if σ_1 loses on a word, then $\sigma_{\mathcal{A}}$ cannot win on it either, since σ_1 is optimum. This proves the first statement of the theorem. If G is non-recursive, $\sigma_{\mathcal{A}}$ is even equivalent to σ_1 because two strategies in a non-recursive game are equivalent if and only if they are semi-equivalent – as already mentioned in Section 4.5. This proves the statement of the theorem for the case that the condition of the first bullet point is satisfied.

Lastly, we consider the case that $L(T)$ is finite, which surprisingly seems to be the most difficult to prove. Suppose $\sigma_{\mathcal{A}}$ is not optimum. The proof consists of two major steps. First, we show that there must exist $q \in Q$, $a_1, \ldots, a_n, a_{n+1} \in \Sigma_f$ with $a_{n+1} = a_1$ and $a_{i+1} \in L(R_{a_i})$, a word $w \in L(\sigma_1)$ and a Romeo strategy τ' such that $\Pi(\sigma_{\mathcal{A}}, \tau', w)$ contains a configuration with T-state q where all future configurations also have T-state q because all future moves by Juliet are to call some a_i and

Romeo keeps choosing a_{i+1} as the replacement word. The second major step is to construct an optimum strategy σ_1' such that, if we had chosen σ_1' instead of σ_1 as the first strategy of the sequence (σ_k), then the strategy that the sequence converges to might still not be optimum, but at least it will no longer be possible to choose the same q and $a_1,\dots,a_n$ with the properties above. After finitely many repetitions of this procedure, no possible combination of q and $a_1,\dots,a_n$ remains. Thus, the type 5 strategy that is semi-equivalent to the optimum strategies of the sequence must be optimum itself (i. e. equivalent to the strategies of the sequence). We will describe the steps in more detail below.

Since $L(T)$ is finite, T contains no directed circles except for circles in its error state, i. e. a non-accepting state q_e with $\delta(q_e,a)=q_e$ for all $a\in\Sigma$. We can assume that σ_1 plays Read whenever it is in a configuration with T-state q_e because there is no way to win once that state is reached. Thus, also $\sigma_{\mathcal{A}}$ plays Read whenever the T-state is q_e. If $\sigma_{\mathcal{A}}$ is not optimum then there exists $w\in L(\sigma_1)$ and a Romeo strategy τ such that $\Pi(\sigma_{\mathcal{A}},\tau,w)$ is infinite. Since the only circles in T are in q_e, this means that there exists some $q\in Q\setminus\{q_e\}$ such that $\Pi(\sigma_{\mathcal{A}},\tau,w)$ contains a configuration with T-state q where all subsequent configurations also have T-state q and all subsequent moves by $\sigma_{\mathcal{A}}$ are Call (because on a Read the T-state would change, due to the lack of circles). Let $(\overline{u},a_1v),(\overline{u}\widehat{a_1},a_2x_1v),(\overline{u}\widehat{a_1}\widehat{a_2},a_3x_2x_1v),\dots,(\overline{u}\widehat{a_1}\widehat{a_2}\dots\widehat{a_n},a_{n+1}x_nx_{n-1}\dots x_2x_1v)$ be consecutive configurations of $\Pi(\sigma_{\mathcal{A}},\tau,w)$ with T-state q such that $a_i,\dots,a_n$ are pairwise distinct and $a_1=a_{n+1}$. So $a_i\in\Sigma_f$, $x_i\in\Sigma^*$ and $a_{i+1}x_i\in L(R_{a_i})$ for $i=1,\dots,n$. We claim that $x_i=\varepsilon$ for all $i=1,\dots,n$.

Suppose not, then consider the Romeo strategy τ' given by

$$\tau'(\overline{v},a)=\begin{cases}a_{i+1}x_i, & \text{if } \overline{u} \text{ is a prefix of } \overline{v} \text{ and } a=a_i \text{ for some } i=1,\dots,n,\\ \tau(\overline{v},a), & \text{otherwise.}\end{cases}$$

Let $\ell\in\mathbb{N}$ be greater than the length of the longest word in $L(T)$. Note that the configuration $(\overline{u}(\widehat{a_1}\dots\widehat{a_n})^\ell,a_{n+1}(x_n\dots x_1)^\ell v)$ occurs in the play $\Pi(\sigma_{\mathcal{A}},\tau',w)$. For k large enough, this configuration also occurs in the play $\Pi(\sigma_k,\tau',w)$. But if $x_i\neq\varepsilon$ for some $i\in\{1,\dots,n\}$ then the remaining string of this configuration is longer than the longest word in $L(T)$. So also the final string of $\Pi(\sigma_k,\tau',w)$ is longer than the longest word in $L(T)$, thus $w\notin L(\sigma_k)$. But σ_k is optimum like σ_1 and we said that $w\in L(\sigma_1)$. Contradiction!

Note that $\delta^*(q_0,\natural\overline{u})=q$ since we have said that q is the T-state of the configuration $(\overline{u},a_1v)$. Also, on input words with prefix $\natural\overline{u}$, the strategy $\sigma_{\mathcal{A}}$ plays only Read moves until $\natural\overline{u}$ is completely processed because $\sigma_{\mathcal{A}}$ is of type 5 (and therefore of type 3) and we know that the configuration $(\overline{u},a_1v)$ occurs in a play of $\sigma_{\mathcal{A}}$ (namely on w). Since (σ_k) converges to σ, we can choose k large enough such that $\sigma(\overline{v},a)=\sigma_k(\overline{v},a)$ for all $(\overline{v},a)\in\overline{\Sigma}^{\leq|\overline{u}|}\times\Sigma_f$. As σ and $\sigma_{\mathcal{A}}$ are indistinguishable, the choice of k implies that the configuration $(\overline{u},a_1v)$ occurs in the play $\Pi(\sigma_k,\tau,w)$ and the configuration $(\natural\overline{u},a_1v)$ occurs in any play of σ_k on $\natural\overline{u}a_1v$. Since $w\in L(\sigma_k)$, we have that $\natural\overline{u}a_1v\in L(\widetilde{\sigma})$ for any one-pass strategy $\widetilde{\sigma}$ that initially plays Reads on $\natural\overline{u}a_1v$ until the configuration $(\natural\overline{u},a_1v)$

is reached and then continues to play like σ_k would play if the configuration were $(\overline{u}, a_1 v)$ instead, i.e. $\widetilde{\sigma}$ could be defined via

$$\widetilde{\sigma}(\overline{v}, a) = \begin{cases} \mathit{Read}, & \text{if } \natural\overline{u} \text{ is not a prefix of } \overline{v}, \\ \sigma_k(\overline{uv_2}, a), & \text{if } \overline{v} = (\natural\overline{u})\overline{v_2} \text{ for some } \overline{v_2} \in \overline{\Sigma}^*. \end{cases}$$

Since σ_k is optimum it follows that $\natural\overline{u}a_1 v \in L(\sigma_k)$. We (re)define a Romeo strategy τ' similar to above (when we showed that $x_i = \varepsilon$) except that $\overline{u}$ in the former definition is now replaced by $\natural\overline{u}$, i.e.

$$\tau'(\overline{v}, a) = \begin{cases} a_{i+1}, & \text{if } \natural\overline{u} \text{ is a prefix of } \overline{v} \text{ and } a = a_i \text{ for some } i = 1, \ldots, n, \\ \tau(\overline{v}, a), & \text{otherwise.} \end{cases}$$

Consider the (winning) play $\Pi(\sigma_k, \tau', \natural\overline{u}a_1 v)$. By choice of k it contains the configuration $(\natural\overline{u}, a_1 v)$. The strategy σ_k must play Read at some point after reaching this configuration because otherwise the play would be infinite instead of winning. By definition of τ', the next configuration where σ_k plays Read is of the form $((\natural\overline{u})(\widehat{a_1} \ldots \widehat{a_n})^j \widehat{a_1} \ldots \widehat{a_{i-1}}, a_i v)$ for some $j \in \mathbb{N}_0$, $i \in \{1, \ldots, n\}$. We claim that the strategy σ_1' given by

$$\sigma_1'(\overline{v}, a) = \begin{cases} \sigma_k((\natural\overline{u})(\widehat{a_1} \ldots \widehat{a_n})^j \widehat{a_1} \ldots \widehat{a_{i-1}}\overline{v_2}, a), & \text{if } \overline{v} = \overline{v_1 v_2} \text{ for some } \overline{v_1}, \overline{v_2} \in \overline{\Sigma}^* \text{ and} \\ & \delta^*(q_0, \natural\overline{v_1}) = q \text{ and either } (\overline{v_2}, a) = (\varepsilon, a_i) \\ & \text{or } \overline{v_2} \in \overline{\Sigma}^+ \text{ begins with } a_i, \\ \sigma_k(\overline{v}, a), & \text{otherwise.} \end{cases}$$

is optimum. This strategy plays like σ_k except that upon reaching a configuration with T-state q and current symbol a_i it will continue as if the history string were $(\natural\overline{u})(\widehat{a_1} \ldots \widehat{a_n})^j \widehat{a_1} \ldots \widehat{a_{i-1}}$. In particular, σ_1' *always* plays Read when the T-state is q and the current symbol is a_i (much like a type 5 strategy). Note that due to the lack of circles in T, this can happen at most once in a play, so σ_1' is well-defined. Suppose σ_1' were not optimum, then there would exist $w' \in L(\sigma_k) \setminus L(\sigma_1')$. A play of σ_1' on w' that is not winning must contain a configuration $(\overline{v_1}, a_i v')$ with T-state q, where $\overline{v_1} \in \overline{\Sigma}^*$ and $v' \in \Sigma^*$. Some play of σ_k on w' also contains this configuration because σ_1' plays like σ_k prior to reaching this configuration. Since the play of σ_1' is not winning after reaching the configuration $(\overline{v_1}, a_i v')$, it holds that $\natural\overline{u}a_1 v' \notin L(\sigma_k)$ by definition of σ_1'. On the other hand, since the play of σ_k on w' that contains the configuration $(\overline{v_1}, a_i v')$ is winning (since $w' \in L(\sigma_k)$), there exists a strategy that wins on $\natural\overline{u}a_i v'$, namely any strategy that starts by playing Reads until the history string is $\natural\overline{u}$ and then continues to play like σ_k would play if the history string were $\overline{v_1}$ instead of $\natural\overline{u}$. Hence, $\natural\overline{u}a_i v' \in L(\sigma_m)$ for all $m \in \mathbb{N}$ because the strategies σ_m are optimum. For m large enough, the play $\Pi(\sigma_m, \tau', \natural\overline{u}a_i v')$ contains the configuration $(\natural\overline{u}\widehat{a_i} \ldots \widehat{a_n}, a_1 v')$. Since this play is winning, it is easy to construct a strategy that wins on $\natural\overline{u}a_1 v'$ (with methods as above). But since σ_k is optimum this would mean that $\natural\overline{u}a_1 v' \in L(\sigma_k)$. Contradiction! So, as claimed, σ_1' is optimum.

Consider the sequence (σ_k') that we would have gotten instead of (σ_k) if we had started with the optimum strategy σ_1' instead of the optimum strategy σ_1. Since σ_1' always plays Read when the T-state is q and the current symbol is a_i, all other strategies of the sequence do the same. If the

sequence is finite then we are done, like above. Otherwise, also the strategy σ' that the sequence converges to plays Read whenever the T-state is q and the current symbol is a_i and the same holds for the type 5 strategy $\sigma_{\mathcal{A}'}$ that is indistinguishable from σ'. It could still be the case that $\sigma_{\mathcal{A}'}$ is not optimum, but if we repeat the procedure from above then it will not happen again that a_i occurs in the list of function symbols that are called infinitely often while the T-state is q. Therefore, after finitely many repetitions of the procedure, no combinations of a state $q \in Q$ and function symbols $a_1, \ldots, a_n \in \Sigma_f$ with the above properties is possible any more. We must have arrived at an optimum type 5 strategy! □

As we will see later, the implication $1 \Longrightarrow 5$ and even several weaker implications for optimum strategies do not hold in general. However, the implication $1 \Longrightarrow 2$ holds for games with self-delimiting replacement languages. This will be our next main result. We need several more lemmas before we can prove it.

The next lemma further improves our understanding of games with self-delimiting replacement languages. It augments the intuition given before Lemma 3.8: Juliet does not even need to know u in advance in order to tell when a prefix u of the input word is completely processed. As soon as that happens for some prefix u, she knows what u is.

4.5 Lemma. *Let $G = (\Sigma, R, T)$ be a game with self-delimiting replacement languages and $u, w \in \Sigma^*$. If a configuration $(\overline{u}, v) \in H_u \times \Sigma^*$ occurs in a play Π on w, then $w = uv$.*

Proof. Without loss of generality, all moves by Juliet after the configuration $(\overline{u}, v)$ is reached are Read. Hence, the final history string of Π is $\overline{u}v$. It is easy to see that there also exists a play Π' on uv with final history string $\overline{u}v$.

Consider the play trees $\mathcal{T}(\Pi)$ and $\mathcal{T}(\Pi')$. In $\mathcal{T}(\Pi)$, the current symbols of the root's children are the symbols of w. In $\mathcal{T}(\Pi')$, the current symbols of the root's children are the symbols of uv. By Lemma 3.9, both play trees have the same structure. Moreover, the current symbol of each configuration in the play tree is uniquely determined by the final history string, which is $\overline{u}v$ in both cases. Thus, $w = uv$. □

The following lemma allows us to assume that an optimum one-pass strategy σ in a game with self-delimiting replacement languages satisfies two specific properties. Property (i) of the following lemma says that the substrategy after completely processing some prefix u of the input word can be such that the set of possible T-states reached when the next longest prefix ua of the input word is completely processed depends only on

- the set S of states that *could* have been reached after completely processing u,
- the state $p \in S$ that was *really* reached and
- the next symbol a of the input word.

This is the crucial property that allows Juliet to forget the exact history a completely processed prefix and only remember S and p. Note that there are only finitely many possible combinations of $S \subseteq Q$ and $p \in S$, which is the key towards finitely many states in a strategy automaton. Property (ii)

of the next lemma intuitively makes sense as follows: After completely processing a prefix of the input word, it cannot be a bad idea to minimize the set of states reachable after completely processing the next longest prefix of the input word.

4.6 Lemma. *Let $G = (\Sigma, R, T)$ be a game with self-delimiting replacement languages for which an optimum one-pass strategy exists. Then there exists a terminating, optimum strategy σ as follows:*

Let $P = \{states(q_0, w, \sigma) \mid w \in \Sigma^*\} \subseteq \mathcal{P}(Q)$. For $S \in P$ there exist strings u_S with $S = states(\sigma, u_S)$ and for $p \in S$ history strings $\overline{u_{S,p}} \in H^\sigma_{u_S}$ with $\delta^*(q_0, \natural\overline{u_{S,p}}) = p$ such that the following properties are satisfied for each $S \in P$, $p \in S$ and $a \in \Sigma$:

(i) For each $u \in \Sigma^*$ and $\overline{u} \in H^\sigma_u$ with $S = states(q_0, u, \sigma)$ and $p = \delta^*(q_0, \natural\overline{u})$ it holds that $states(p, a, \sigma[\overline{u}]) = states(p, a, \sigma[\overline{u_{S,p}}])$.

(ii) There exists no terminating strategy σ' with $states(p, a, \sigma') \subsetneq states(p, a, \sigma[\overline{u_{S,p}}])$.

Proof. Let σ be an arbitrary terminating optimum one-pass strategy σ (it exists by Lemma 3.10). We will make several adjustments to σ until it satisfies the properties that we wish to prove. The adjustments consist of two major steps. The first step is given by Algorithm 4.1. This algorithm adjusts σ and constructs a set $P \subseteq \mathcal{P}(Q)$ and strings u_S for $S \in P$ with $S = states(\sigma, u_S)$ and $\overline{u_{S,p}} \in H^\sigma_{u_S}$ with $\delta^*(q_0, \natural\overline{u_{S,p}}) = p$ for $S \in P$ and $p \in S$ such that property (ii) of the lemma holds. In a second step, we will change σ further so that $P = \{states(q_0, w, \sigma) \mid w \in \Sigma^*\}$ and property (i) holds. During all the adjustments, σ remains an optimum terminating strategy. It should be noted that Algorithm 4.1 cannot be executed by a computer because there is no finite representation for one-pass strategies; the algorithm merely serves to define P, u_S, $\overline{u_{S,p}}$ and redefine σ.

Let us first explain the general structure of Algorithm 4.1. The algorithm initializes P as the empty set. An auxiliary set $W \subseteq \mathcal{P}(Q)$ is used in which subsets of Q are stored temporarily before they are added to P. Whenever a set S is added to W, the string u_S is defined directly afterwards (lines 2–3 and 18–19). The core of the algorithm is the while-loop, where in each iteration (lines 5–21)

- a set S is chosen (line 5),
- for each $p \in S$, the string $\overline{u_{S,p}}$ is defined (line 7) and some more preparations are made (8–12) in order to
- adjust σ (lines 13–14),
- possibly more sets will be added to W (lines 15–19) and
- eventually S is removed from W and added to P (lines 20–21).

We now examine the steps of Algorithm 4.1 in more detail. The first set that is added to W (and will be added to P later) is $\{q_0\}$, and $u_{\{q_0\}}$ is defined as the empty string (lines 2–3). It is clear that $\{q_0\} = states(q_0, \varepsilon, \sigma)$, so we have not done anything wrong yet. The algorithm then jumps into a while-loop over elements $S \in W$ and a nested for-loop over elements $p \in S$. Since it will always hold that $S = states(q_0, u_S, \sigma)$, it is possible to choose $\overline{u_{S,p}}$ such as it is done in

Algorithm 4.1

$P := \emptyset$
$W := \{\{q_0\}\}$
$u_{\{q_0\}} = \varepsilon$
while $W \neq \emptyset$ **do**
  Pick $S \in W$
  for all $p \in S$ **do**
    Pick $\overline{u_{S,p}} \in H_{u_S}^{\sigma}$ such that $\delta^*(q_0, \natural\overline{u_{S,p}}) = p$
    for all $a \in \Sigma$ **do**
      Let $\sigma_{p,a}$ be a terminating one-pass strategy that minimizes $states(p,a,\sigma_{p,a}) \subseteq states(p,a,\sigma[\overline{u_{S,p}}])$ under inclusion
      for all $q \in states(p,a,\sigma_{p,a})$ **do**
        Pick $\overline{a_q} \in H_a^{\sigma[\overline{u_{S,p}}]}$ such that $\delta^*(p, \natural\overline{a_q}) = q$
    Let σ_p be the one-pass strategy given by $\sigma_p(\overline{u}, b) = \sigma_{p,a}(\overline{u}, b)$ where $a \in \Sigma$ is such that $(\overline{u}, b) = (\varepsilon, a)$ or $\overline{u} \in \overline{\Sigma}^+$ begins with a or $\widehat{a}$
  Let σ' be the one-pass strategy defined by

$$\sigma'(\overline{u}, b) = \begin{cases} \sigma_p(\overline{u_2}, b), & \text{if } \overline{u} = \overline{u_1 u_2} \text{ and } p = \delta^*(q_0, \natural\overline{u_1}) \text{ where } \overline{u_1} \in H_{u_S}^{\sigma} \text{ and} \\ & \overline{u_2} \text{ is a proper prefix of a string in } \cup_{a \in \Sigma} H_a^{\sigma_p}, \\ \sigma(\overline{u_{S,p} a_q u_2}, b), & \text{if } \overline{u} = \overline{u_1 a u_2},\ p = \delta^*(q_0, \natural\overline{u_1}),\ q = \delta^*(p, \natural\overline{a}) \text{ and } a \in \Sigma \\ & \text{where } \overline{u_1} \in H_{u_S}^{\sigma},\ \overline{a} \in H_a^{\sigma_p} \text{ and } \overline{u_2} \in \overline{\Sigma}^*, \\ \sigma(\overline{u}, b), & \text{otherwise} \end{cases}$$

  $\sigma := \sigma'$
  for all $a \in \Sigma$ **do**
    $S' := states(q_0, u_S a, \sigma)$
    if $S' \notin W \dot{\cup} P$ **then**
      $W := W \cup \{S'\}$
      $u_{S'} := u_S a.$
  $W := W \setminus \{S\}$
  $P := P \cup \{S\}$

line 7. In line 9, a terminating one-pass strategy $\sigma_{p,a}$ that minimizes $states(p,a,\sigma_{p,a})$ as a subset of $states(p,a,\sigma[\overline{u_{S,p}}])$ is chosen. Note that if it is not possible to achieve a proper subset here, then $\sigma_{p,a} = \sigma[\overline{u_{S,p}}]$ is always a possibility. By definition of $states(p,a,\sigma[\overline{u_{S,p}}])$, it is also possible to choose $\overline{a_q}$ as in line 11. The strategy σ_p defined in line 12 is the strategy that plays like $\sigma_{p,a}$ for an input word that starts with a. In line 13, σ' is defined as the strategy that plays like σ unless the input word starts with $u_S a$ for some $a \in \Sigma$; in that case, after u_S is completely processed, σ' plays on the symbol a like σ_p (where p is the T-state reached when u_S was completely processed) and, once $u_S a$ is completely processed, σ' continues to play like σ would play if the history were $\overline{u_{S,p} a_q}$ after completely processing $u_S a$ (where q is the T-state reached when $u_S a$ was completely processed). Note that σ' is well-defined because

- the sets $H_{u_S}^{\sigma}$ and $H_a^{\sigma_p}$ are self-delimiting by Lemma 3.8,
- the first symbol of $\overline{u_2}$ (if it is non-empty) or $\overline{a}$ determines the only possibility of $a \in \Sigma$ such that $\overline{u_2}$ could be a proper prefix of a string in $H_a^{\sigma_p}$ or $\overline{a} \in H_a^{\sigma_p}$,
- the state $q = \delta^*(p, \natural\overline{a})$ in the second case is in $states(p,a,\sigma_{p,a})$ by definition of σ_p, so $\overline{a_q}$ was defined previously in line 11.

We will prove soon that σ' is also a terminating, optimum strategy, which justifies redefining σ as σ' in line 14. None of the changes to σ hereafter will affect its plays on input words $u_S a$ for $a \in \Sigma$. Therefore, if a set $S' = states(q_0, u_S a, \sigma)$ is not already added to P or W, then it needs to be added to W so that it will be added to P later. A string $u_{S'}$ with $S' = states(q_0, u_{S'}, \sigma)$ can be chosen as $u_{S'} = u_S a$. All this happens in lines 15–19. Eventually, S is removed from W and added to P (lines 20–21) and the while-loop continues until W is empty.

The algorithm terminates because each subset of Q is added to W at most once and a set is removed from W in each iteration of the while-loop. To show that the invariant holds that σ is terminating and optimum we need to show that this is the case for the strategy σ' defined in line 13.

Suppose σ is terminating and optimum before some execution of line 13, but the strategy σ' defined in line 13 is not. A play Π of σ' against some Romeo strategy τ on some $w \in \Sigma^*$ can differ from the corresponding play of σ only after a configuration $(\overline{u_1}, av)$ with $\overline{u_1} \in H_{u_S}^{\sigma}$, $a \in \Sigma$ and $v \in \Sigma^*$ occurs. By Lemma 4.5 this means that $w = u_S a v$. Since $\sigma_{p,a}$ is terminating (and hence also σ_p), Π will contain a configuration $(\overline{u_1 a}, v)$ with $\overline{a} \in H_a^{\sigma_p}$ corresponding to the moment when $u_S a$ is completely processed (where $p = \delta^*(q_0, \natural\overline{u_1})$). From this point onwards, σ' plays like σ if the configuration were $(\overline{u_{S,p} a_q}, v)$ instead of $(\overline{u_1 a}, v)$ (where $q = \delta^*(p, \natural\overline{a})$). Since such a configuration can indeed occur in a play of σ (namely this is possible – depending on Romeo's moves – if and only if the input word $u_S a v$) and σ is terminating, Π must be finite; hence σ' is terminating. If σ' is not optimum, then w and Π and can be chosen such that $w = u_S a v \in L(\sigma)$ and Π is losing. But Π is only losing if $u_S a v \notin L(\sigma)$. Contradiction!

Note that property (ii) of the lemma is satisfied, since the adjustment in lines 13–14 guarantee that $states(p,a,\sigma[\overline{u_{S,p}}]) = states(p,a,\sigma_{p,a})$ and $states(p,a,\sigma_{p,a})$ was chosen minimal under inclusion. Moreover, u_S and $\overline{u_{S,p}}$ are as specified by the lemma. However, it is not clear whether $P = \{states(q_0, w, \sigma) \mid w \in \Sigma^*\}$ and whether property (i) holds; so far, (i) necessarily holds only for

$u = u_S$. We will therefore make further adjustments to σ to ensure that (i) holds for all $u \in \Sigma^*$ and $P = \{states(q_0, w, \sigma) \mid w \in \Sigma^*\}$. We define a sequence $(\sigma_k)_{k \in \mathbb{N}_0}$ of one-pass strategies that converges to some strategy that has all the claimed properties.

Let σ_0 be the strategy σ at the end of Algorithm 4.1. We define σ_{k+1} via

$$\sigma_{k+1}(\overline{u}, a) = \begin{cases} \sigma_0(\overline{u_{S,p} u_2}, a), & \text{if } \overline{u} = \overline{u_1 u_2},\ p = \delta^*(q_0, \natural \overline{u_1}) \text{ and } S = states(q_0, u_1, \sigma_k) \\ & \text{where } u_1 \in \Sigma^{k+1},\ \overline{u_1} \in H^{\sigma_k}_{u_1} \text{ and } \overline{u_2} \in \overline{\Sigma}^* \\ \sigma_k(\overline{u}, a), & \text{otherwise.} \end{cases}$$

Informally, σ_{k+1} plays like σ_k except that after a prefix u_1 of length $k+1$ of the input word is completely processed (with some history string $\overline{u_1} \in H^{\sigma_k}_{u_1}$), σ_{k+1} continues to play like σ_0 if the history string were $\overline{u_{S,p}}$ instead of $\overline{u_1}$, where S is the set of states that could have been reached after completely processing u_1 and $p \in S$ is the state that was really reached.

Note that $\sigma_{k+1}(\overline{u}, a) \neq \sigma_k(\overline{u}, a)$ only if $|\overline{u}| > k$, since history strings in $H^{\sigma_k}_{u_1}$ have length at least $|u_1|$. Therefore, (σ_k) converges to the strategy, which we again call σ, given by $\sigma(\overline{u}, a) = \sigma_{|\overline{u}|}(\overline{u}, a)$.

The definition of σ_{k+1} ensures that σ_{k+1} satisfies property (i) for all $u = u_1$ of length at most $k+1$. Therefore, σ satisfies property (i) for *all* $u \in \Sigma^*$. Also property (ii) and that u_S and $\overline{u_{S,p}}$ are as specified in the lemma, which was already true for σ_0, is still satisfied also for σ because σ plays like σ_0 on input words of the form $u_S a$ (observe that each prefix of u_S is also of the form $u_{S'}$ for some $S' \in P$ by construction).

Since $states(q_0, u_S, \sigma_0) = S$ for all $S \in P$, we have that $P = \{states(q_0, u_S, \sigma_0) \mid S \in P\}$. In fact, we even have that $P = \{states(q_0, u_S a, \sigma_0) \mid S \in P, a \in \Sigma\}$ due to the for-loop in lines 15–19 of Algorithm 4.1 and since each set that is added to W is added to P eventually. Combined with property (i) and the fact that σ plays like σ_0 on input words of the form $u_S a$, we obtain that $P = \{states(q_0, w, \sigma) \mid w \in \Sigma^*\}$, as stated in the lemma.

It only remains to show that σ is terminating and optimum. Observe that an input word of length k, the strategy σ plays exactly like σ_k. Therefore, it is enough to show that all the strategies σ_k are terminating and optimum.

That σ_k is terminating follows from the fact that σ_0 is terminating.

The proof that σ_k is optimum uses induction on k. We have shown before that σ_0 is optimum. Suppose σ_k is optimum but σ_{k+1} is not. Then there exists a string $u_1 v \in L(\sigma_k) \setminus L(\sigma_{k+1})$ with $u_1 \in \Sigma^{k+1}$ and $v \in \Sigma^*$. Since σ_{k+1} loses on $u_1 v$, there must be a play of σ_{k+1} on $u_1 v$ that reaches a configuration $(\overline{u_1}, v)$ with $\overline{u_1} \in H^{\sigma_k}_{u_1}$ where v is such that σ_k loses on $u_S v$, where $S = states(q_0, u_1, \sigma_k)$. Since σ_k is optimum, this means that no one-pass strategy wins on $u_S v$. But consider the strategy that plays like σ_k except that after completely processing u_S (with some history string $\overline{u} \in H^{\sigma_k}_{u_S}$), it continues as if the history string were $\overline{u_1}$ instead of $\overline{u}$, where $\overline{u_1} \in H^{\sigma_k}_{u_1}$ is such that $\delta^*(q_0, \natural \overline{u_1}) = \delta^*(q_0, \natural \overline{u})$. Such $\overline{u_1}$ must exist, since $states(q_0, u_S, \sigma_k) = S = states(q_0, u_1, \sigma_k)$. Since $u_1 v \in L(\sigma_k)$, this strategy wins on $u_S v$. Contradiction! □

We need two more lemmas before we can prove the theorem that the implication $1 \Longrightarrow 2$ for optimum strategies holds for games with self-delimiting replacement languages. The following is essentially a corollary of Lemma 3.14.

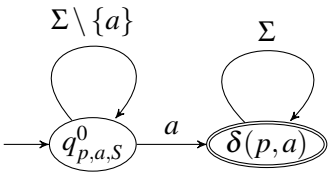


Figure 4.2.: Construction of $\mathcal{A}_{p,a,S}$ in the proof of Lemma 4.8 if $\sigma_{p,a,S}$ has depth 0

4.7 Lemma. *Let $G = (\Sigma, R, T)$ be a game with self-delimiting replacement languages. For each terminating one-pass strategy σ for G and each state $p \in Q$, symbol $a \in \Sigma$ there exists a strategy σ' with depth bounded by $|\Sigma_f| \cdot |Q| \cdot (2^{|Q|} - 1)$ and $states(p, a, \sigma') \subseteq states(p, a, \sigma)$.*

Proof. Consider the game obtained from G by changing the initial state of T to p and the set of accepting states to $states(p, a, \sigma)$. The strategy σ wins on a in this game, so by Lemma 3.14 there exists a strategy σ' that has depth bounded by $|\Sigma_f| \cdot |Q| \cdot (2^{|Q|} - 1)$ and also wins on a in this game. Therefore, $states(p, a, \sigma') \subseteq states(p, a, \sigma)$ and the lemma follows, since σ' is also a strategy for G. □

The core of the construction of a type 2 automaton of an optimum strategy lies within the proof of the following lemma. The strategy automata that we will construct make use of accepting states as particular distinguished states; however, these states bear no relation to the acceptance semantics of accepting states in "normal" FSAs that recognize a regular language.

4.8 Lemma. *Let $G = (\Sigma, R, T)$ be a game with self-delimiting replacement languages. Let Z be the set of triples $(p, a, S) \in Q \times \Sigma \times \mathcal{P}(Q)$ with the property that a strategy $\sigma_{p,a,S}$ with finite depth and $S = states(p, a, \sigma_{p,a,S})$ exists. For each $(p, a, S) \in Z$ there exists a type 2 automaton $\mathcal{A}_{p,a,S} = (Q_{p,a,S}, \overline{\Sigma}, \delta_{p,a,S}, q^0_{p,a,S}, F_{p,a,S})$ with the following properties:*

(i) The elements of S are a subset of the states of $\mathcal{A}_{p,a,S}$, namely $F_{p,a,S} = S$.

(ii) Every play Π of $\sigma_{\mathcal{A}_{p,a,S}}$ on a is finite. If $u \in \Sigma^*$ is its final string, then Π terminates in $\mathcal{A}_{p,a,S}$-state $\delta^*(p, u) \in S$ and no previous configuration of Π has an $\mathcal{A}_{p,a,S}$-state that is in S. In particular, $states(p, a, \sigma_{p,a,S}) \subseteq S$.

(iii) We call the states of $\mathcal{A}_{p,a,S}$ that are not in $\{q^0_{p,a,S}\} \cup S$ the inner states *of $A_{p,a,S}$. There are at most $\sum_{k=1}^{d} |\Sigma_f|^{k-1} |Q_f|^k |Q|^k$ inner states, where d is the depth of $\sigma_{p,a,S}$ and $|Q_f|$ is the maximal number of non-error states of a minimal DFA of a replacement language $L(R_b)$ for $b \in \Sigma_f$.*

Proof. We choose $\sigma_{p,a,S}$ as stated in the lemma such that its depth is minimal. We prove the lemma by induction on the depth of $\sigma_{p,a,S}$. Note that the depth of $\sigma_{p,a,S}$ is determined by the depth of its plays on a, since $\sigma_{p,a,S}$ could simply play Read moves if the history string is not a proper prefix of a history string in H_a or if the history string is empty but the current symbol is different from a.

If $\sigma_{p,a,S}$ has depth 0, then all moves of $\sigma_{p,a,S}$ are Read. We define $\mathcal{A}_{p,a,S}$ as shown in Figure 4.2. It is immediate that $\mathcal{A}_{p,a,S}$ satisfies the properties (i)–(iii).

If $\sigma_{p,a,S}$ has positive depth then $\mathcal{A}_{p,a,S}$ is composed of automata for strategies of less depths, as we will describe below. The construction of $\mathcal{A}_{p,a,S}$ is sketched exemplarily in Figure 4.3.

Since $L(R_a)$ is self-delimiting, its minimal DFA has a single accepting state f_a and a single error state e_a (i.e. a non-accepting state where all outgoing transitions are loops) and all outgoing transitions of f_a lead to e_a. Let $(Q_a, \Sigma, \delta_a, q_a^0, \{f_a\})$ be the automaton obtained from the minimal DFA for $L(R_a)$ by removing the error state e_a and its incident edges. Thus, f_a no longer has any outgoing edges. By definition, $|Q_f| = \max_{b \in \Sigma_f} |Q_b|$.

The states of $\mathcal{A}_{p,a,S}$ are initial state $q^0_{p,a,S}$, the pairs in $Q \times (Q_a \setminus \{f_a\})$, the elements of S, and possibly further states coming from automata built into it. For $s \in S$ we identify s with the pair (s, f_a), so we think of these pairs as states of $\mathcal{A}_{p,a,S}$ as well. The point of this is only to avoid case distinctions later because we can treat these states in the same way as pairs in $Q \times (Q_a \setminus \{f_a\})$. To satisfy property (i), we define $F_{p,a,S} = S \cong S \times \{f_a\}$.

The initial state has a transition for the symbol $\widehat{a}$ leading to the state (p, q_a^0). Further, the initial state has transitions to itself for every symbol in $\Sigma \setminus \{a\}$. For each $(p', q_a) \in Q \times (Q_a \setminus \{f_a\})$ and $b \in \Sigma$, the transitions determining the subsequent behavior of $\sigma_{\mathcal{A}_{p,a,S}}$ when the $\mathcal{A}_{p,a,S}$-state is (p', q_a) and the current symbol is b are defined as follows.

First check whether there exist $u, v \in \Sigma^*$ and $\overline{u} \in H_u^{\sigma_{p,a,S}[\widehat{a}]}$ such that $ubv \in L(R_a)$, $p' = \delta^*(p, \natural\overline{u})$ and $q_a = \delta_a^*(q_a^0, u)$. If not, then the state (p', q_a) has a transition for b to itself. This transition will never be used in a play of $\sigma_{\mathcal{A}_{p,a,S}}$ on a and only serves to satisfy the definition of type 2 automata.

If such u, v and $\overline{u}$ do exist, let $S' = \mathit{states}(p', b, \sigma_{p,a,S}[\widehat{a}\overline{u}])$. Observe that $(p', b, S') \in Z$ and the depth of $\sigma_{p',b,S'}$ is less than the depth of $\sigma_{p,a,S}$. We use a copy of the automaton $\mathcal{A}_{p',b,S'}$, which exists by the induction hypothesis; we remove all transitions leaving its initial state except for the transition for b or $\widehat{b}$, and remove all transitions leaving one of its accepting states; then we merge its initial state with (p', q_a) and merge each of its accepting states $s \in S'$ with $(s, \delta_a(q_a, b))$. Note that if $\delta_a(q_a, b) = f_a$, then $(s, \delta_a(q_a, b)) = s$ due to the aforementioned identification; indeed, $s \in S$ in this case by definition of S' and because $v = \varepsilon$ if $\delta_a(q_a, b) = f_a$, since $L(R_a)$ is self-delimiting.

To complete the construction and satisfy the definition of type 2 automata, we add for each $s \in S$ and $b \in \Sigma$ a transition from s reading b to itself.

It remains to show that $\mathcal{A}_{p,a,S}$ satisfies the properties (ii) and (iii).

We first show (iii). The inner states of $\mathcal{A}_{p,a,S}$ are the pairs in $Q \times (Q_a \setminus \{f_a\})$ plus the inner states of automata $\mathcal{A}_{p',b,S'}$ built into $\mathcal{A}_{p,a,S}$ (other states of nested automata have been merged with states of $\mathcal{A}_{p,a,S}$). If $b \notin \Sigma_f$ then $\sigma_{p',b,S'}$ has depth 0 and therefore $\mathcal{A}_{p',b,S'}$ has no inner states. If $b \in \Sigma_f$, then $\sigma_{p',b,S'}$ has at most $\sum_{k=1}^{d-1} |\Sigma_f|^{k-1} |Q_f|^k |Q|^k$ inner states by the induction hypothesis. Thus, $\mathcal{A}_{p,a,S}$ has at most

$$|Q|(|Q_a| - 1) + |Q|(|Q_a| - 1)|\Sigma_f| \sum_{k=1}^{d-1} |\Sigma_f|^{k-1} |Q_f|^k |Q|^k < \sum_{k=1}^{d} |\Sigma_f|^{k-1} |Q_f|^k |Q|^k$$

inner states, which proves (iii).

To prove (ii), consider a play of $\sigma_{\mathcal{A}_{p,a,S}}$ on a. The first two configurations of this play are (ε, a) and $(\widehat{a}, w)$ for some $w \in L(R_a)$, and the $\mathcal{A}_{p,a,S}$-state of $(\widehat{a}, w)$ is (p, q_a^0). For $(p', q_a) \in Q \times (Q_a \setminus \{f_a\})$ or $(p', q_a) \in S \times \{f_a\}$, let $\mathcal{B}_{p',q_a}$ be the automaton obtained from $\mathcal{A}_{p,a,S}$ by changing the initial

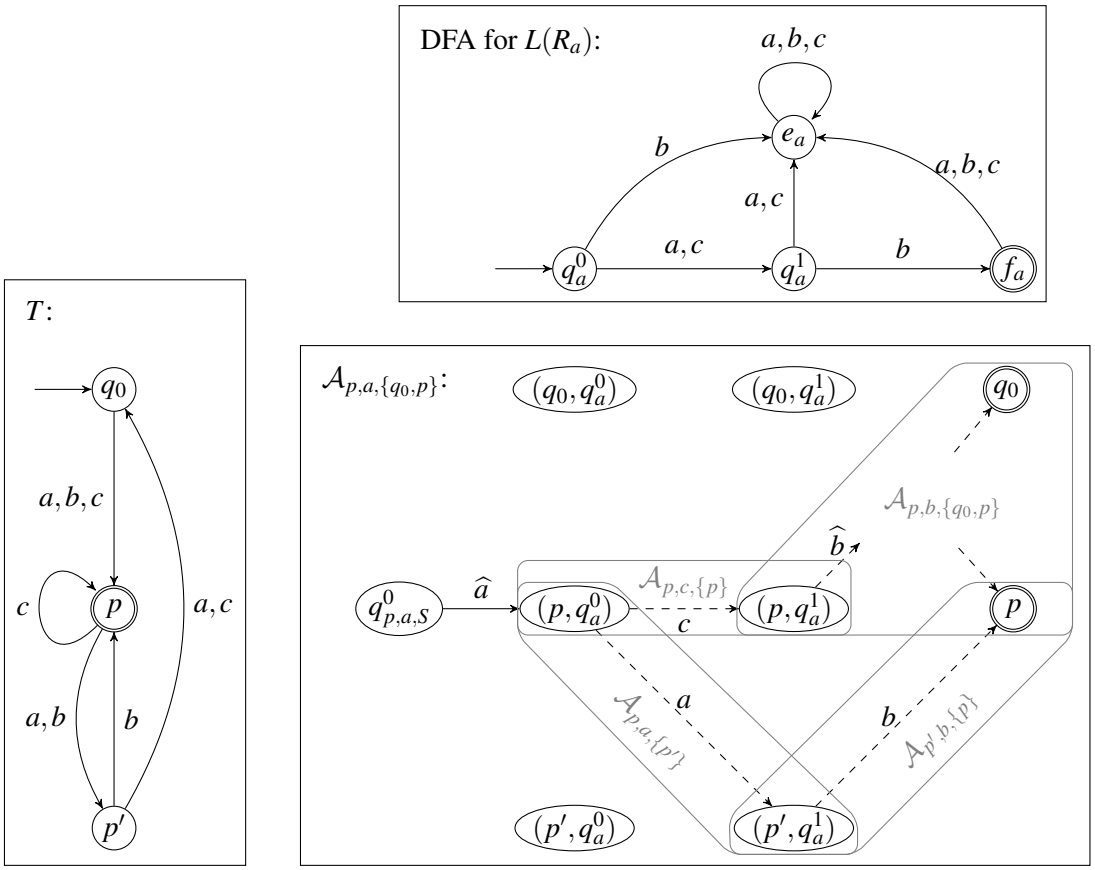


Figure 4.3.: Exemplary construction of $\mathcal{A}_{p,a,S}$ in the proof of Lemma 4.8 if $\sigma_{p,a,S}$ has positive depth, for given T and a given DFA for $L(R_a)$

state to (p', q_a). The property (ii) follows from the following claim for $p' = p$, $q_a = q_a^0$, $v = w$ and $u = \overline{u} = \varepsilon$.

4.9 Claim. *Let $(p', q_a) \in Q \times (Q_a \setminus \{f_a\})$ or $(p', q_a) \in S \times \{f_a\}$, and let $v \in \Sigma^*$. If there exist $u \in \Sigma^*$ and $\overline{u} \in H_u^{\sigma_{p,a,S}[\widehat{a}]}$ such that $uv \in L(R_a)$, $p' = \delta^*(p, \natural\overline{u})$ and $q_a = \delta_a^*(q_a^0, u)$, then every play Π of $\sigma_{\mathcal{B}_{p',q_a}}$ on v is finite. Moreover, if $\overline{v}$ is the final history string of Π, then Π terminates in $\mathcal{B}_{p',q_a}$-state $\delta^*(p', \natural\overline{v}) \in S$.*

Proof. We show the claim by induction on $|v|$. Since the previous induction proof has not been completed yet, we refer to the induction on the depth of $\sigma_{p,a,S}$ as the *outer induction* and the induction on $|v|$ as the *inner induction*.

If $|v| = 0$, then $u \in L(R_a)$, hence $q_a = f_a$. Furthermore, $\widehat{a}\overline{u} \in H_a^{\sigma_{p,a,S}}$ and therefore $p' = \delta^*(p, \natural\widehat{a}\overline{u}) \in S$. So by our identification, $(p', q_a) = p'$. The (unique) play of $\sigma_{\mathcal{B}_{p',q_a}}$ on $v = \varepsilon$ is finite with final history string $\overline{v} = \varepsilon$ and terminates in $\mathcal{B}_{p',q_a}$-state $\delta^*(p', \varepsilon) = p' \in S$.

For the induction step we have that $v = bv'$ for some $b \in \Sigma$ and $v' \in \Sigma^*$. Let Π be a play of $\sigma_{\mathcal{B}_{p',q_a}}$ on bv'. Since $L(R_a)$ is self-delimiting, it holds that $u \notin L(R_a)$, hence $q_a \neq f_a$. The premise of the claim is also true if u and $\overline{u}$ are the same as those chosen in the definition of the transitions determining the subsequent behavior of $\sigma_{\mathcal{A}_{p,a,S}}$ when the $\mathcal{A}_{p,a,S}$-state is (p', q_a) and the current symbol is b, so we can assume they are the ones chosen before.

Let $S' = \mathit{states}(p', b, \sigma_{p,a,S}[\widehat{a}\overline{u}])$, like in the definition of $\mathcal{A}_{p,a,S}$. By the induction hypothesis of the outer induction, the property (ii) holds for $\mathcal{A}_{p',b,S'}$. So every play of $\sigma_{\mathcal{A}_{p',b,S'}}$ on b is finite, and if $\overline{b}$ is its final history string, then it terminates in $\mathcal{A}_{p',b,S'}$-state $\delta^*(p', \natural\overline{b}) \in S'$. Thus, due to the way in which $\mathcal{A}_{p',b,S'}$ is built into $\mathcal{A}_{p,a,S}$, every play of $\sigma_{\mathcal{B}_{p',q_a}}$ on b is finite, and if $\overline{b}$ is its final history string, then it terminates in $\mathcal{B}_{p',q_a}$-state $(\delta^*(p', \natural\overline{b}), \delta_a(q_a, b)) \in S' \times Q_a$. Therefore, Π contains a configuration $(\overline{b}, v')$ with $\mathcal{B}_{p',q_a}$-state (s', q_a'), where $s' = \delta^*(p', \natural\overline{b}) \in S'$ and $q_a' = \delta_a(q_a, b)$.

Since $S' = \mathit{states}(p', b, \sigma_{p,a,S}[\widehat{a}\overline{u}])$, there also exists $\overline{b}' \in H_b^{\sigma_{p,a,S}[\widehat{a}\overline{u}]}$ with $\delta^*(p', \natural\overline{b}') = s'$, hence $\overline{u}\overline{b}' \in H_{ub}^{\sigma_{p,a,S}[\widehat{a}]}$. Moreover $s' = \delta^*(p, \natural\overline{u}\overline{b}')$ and $q_a' = \delta_a^*(q_a^0, ub)$. We can apply the induction hypothesis of the inner induction, which yields that every play of $\sigma_{\mathcal{B}_{s',q_a'}}$ on v' is finite and if $\overline{v}'$ is its final history string, then it terminates in $\mathcal{B}_{s',q_a'}$-state $\delta^*(s', \natural\overline{v}') \in S$. Hence, the play Π is finite, and if $\overline{b}\overline{v}'$ is its final history string, then it terminates in $\mathcal{B}_{p',q_a}$-state $\delta^*(p', \natural\overline{b}\overline{v}') \in S$. This completes the induction step of the inner induction and thereby proves the claim. □

As stated above, Claim 4.9 implies property (ii). This completes the proof of Lemma 4.8. □

Finally, we can prove the theorem announced earlier.

4.10 Theorem. *The implication $1 \Longrightarrow 2$ for optimum strategies holds for games with self-delimiting replacement languages. More precisely, if there exists an optimum one-pass strategy in a game $G = (\Sigma, R, T)$ with self-delimiting replacement languages, then there exists a terminating, optimum one-pass strategy specified by a type 2 automaton with less than $(|\Sigma_f| \cdot |Q_f| \cdot |Q|)^E$ states, where $E = |\Sigma_f| \cdot |Q| \cdot 2^{|Q|}$ and $|Q_f|$ is the maximal number of non-error states of a minimal DFA of a replacement language $L(R_a)$ for $a \in \Sigma_f$ and $|Q|$ is the number of states of the minimal DFA for $L(T)$.*

Proof. Let σ be an optimum one-pass strategy for G. We can assume that σ satisfies the properties of Lemma 4.6, and we choose P, u_S and $\overline{u_{S,p}}$ as in that lemma. We construct a type 2 automaton $\mathcal{A}$ that satisfies the stated size bound such that $\sigma_{\mathcal{A}}$ is terminating and equivalent to σ.

The basic idea of the proof is that when reaching a configuration with history string $\overline{u} \in H_u^\sigma$ for some $u \in \Sigma^*$, all the information contained in $\overline{u}$ can be forgotten except for the T-state $\delta^*(q_0, \natural\overline{u})$ and the set $\mathit{states}(q_0, u, \sigma)$ of states that would have been *possible* as T-states after completely processing u (depending on Romeo's moves). The automaton $\mathcal{A}$ is built by connecting several copies of the automata $\mathcal{A}_{p,a,S}$ from Lemma 4.8.

The type 2 automaton $\mathcal{A}$ has a state (S, p) for each $S \in P$ and $p \in S$ and possibly further states coming from automata that are built into it. The initial state of $\mathcal{A}$ is $(\{q_0\}, q_0)$. A state (S, p) shall be reached whenever a prefix u of the input string with $S = \mathit{states}(q_0, u, \sigma)$ is completely processed and the T-state of the current configuration is p. The construction of $\mathcal{A}$ is completed by doing the following for each $S \in P$, $p \in S$ and $a \in \Sigma$.

Let $S' = \mathit{states}(p, a, \sigma[\overline{u_{S,p}}])$. By Lemma 4.7 and property (ii) of Lemma 4.6, there exists a one-pass strategy σ' with depth bounded by $|\Sigma_f| \cdot |Q| \cdot (2^{|Q|} - 1)$ and $S' = \mathit{states}(p, a, \sigma')$. Therefore, the automaton $\mathcal{A}_{p,a,S'}$ from Lemma 4.8 exists. We take a copy of $\mathcal{A}_{p,a,S'}$, remove all transitions leaving its initial state except for the transition for a or $\widehat{a}$, and remove all transitions leaving one of its accepting states. Then, we merge its initial state with (S, p) and for each $s \in S'$ we merge its state s with $(\mathit{states}(q_0, u_S a, \sigma), s)$. This is indeed a state of $\mathcal{A}$, since $s \in S' \subseteq \mathit{states}(q_0, u_S a, \sigma)$. Note that $\mathcal{A}$ is a well-defined type 2 automaton.

The following claim will help us to show that $\sigma_{\mathcal{A}}$ is optimum.

4.11 Claim. *Let $w \in \Sigma^*$ and let Π be a play of $\sigma_{\mathcal{A}}$ on w. Then Π is finite (i. e. $\sigma_{\mathcal{A}}$ is terminating) and if $\overline{w} \in \overline{\Sigma}^*$ is the final history string of Π, then Π terminates in $\mathcal{A}$-state (S, p) where $S = \mathit{states}(q_0, w, \sigma_{\mathcal{A}}) = \mathit{states}(q_0, w, \sigma)$ and $p = \delta^*(q_0, \natural\overline{w})$.*

Proof. The proof is by induction on $|w|$. The base case $w = \varepsilon$ is immediate. For the induction step, let $w = ua$ with $u \in \Sigma^*$ and $a \in \Sigma$. By the induction hypothesis, Π contains a configuration $(\overline{u}, a)$ with $\mathcal{A}$-state (S', p'), where $S' = \mathit{states}(q_0, u, \sigma_{\mathcal{A}}) = \mathit{states}(q_0, u, \sigma)$ and $p' = \delta^*(q_0, \natural\overline{u})$. By construction of $\mathcal{A}$, a play of $\sigma_{\mathcal{A}}[\overline{u}]$ on a is the same as a play of $\sigma_{\mathcal{A}_{p',a,S''}}$ on a, where $S'' = \mathit{states}(p', a, \sigma[\overline{u_{S',p'}}])$. Therefore, Π is finite by property (ii) of Lemma 4.8. Let $\overline{a} \in H_a$ be such that $\overline{w} = \overline{ua}$ is the final history string of Π. By property (ii) of Lemma 4.8 and the way in which $\mathcal{A}_{p',a,S''}$ is built into $\mathcal{A}$, the play Π terminates in $\mathcal{A}$-state (S, p) where $S = \mathit{states}(q_0, u_{S'} a, \sigma)$ and $p = \delta^*(p', \natural\overline{a}) = \delta^*(q_0, \natural\overline{w})$. It remains to show that $S = \mathit{states}(q_0, ua, \sigma_{\mathcal{A}}) = \mathit{states}(q_0, ua, \sigma)$. Indeed,

$$\begin{aligned}
\mathit{states}(q_0, ua, \sigma_{\mathcal{A}}) &= \bigcup_{p' \in \mathit{states}(q_0, u, \sigma_{\mathcal{A}})} \mathit{states}(p', a, \sigma[\overline{u_{S',p'}}]) \\
&= \bigcup_{p' \in S'} \mathit{states}(p', a, \sigma[\overline{u_{S',p'}}]) \\
&= \mathit{states}(q_0, u_{S'} a, \sigma) \\
&= S
\end{aligned}$$

and the same equations hold when $\sigma_{\mathcal{A}}$ is replaced by σ (using property (i) of Lemma 4.6 for the first equation then). □

From the claim it follows that

$$L(\sigma_{\mathcal{A}}) = \{w \in \Sigma^* \mid states(q_0, w, \sigma_{\mathcal{A}}) \subseteq F\} = \{w \in \Sigma^* \mid states(q_0, w, \sigma) \subseteq F\} = L(\sigma),$$

i. e. $\sigma_{\mathcal{A}}$ is equivalent to σ and therefore also optimum.

It only remains to show that the number of states of $\mathcal{A}$ is bounded as stated in the theorem. Note that $|Q_f| \geq 2$ because the accepting state $f_a \in Q_a$ is different from the initial state $q_a^0 \in Q_a$ for each a, since $\varepsilon \notin L(R_a)$ by definition. We can also assume without loss of generality that $|Q| \geq 2$, since otherwise $L(T) = \Sigma^*$ or $L(T) = \emptyset$ and it would be trivial to find a type 2 automaton with just a single state that specifies a terminating, optimum strategy. There are less than $|Q| \cdot 2^{|Q|}$ states of the form (S, p) with $S \in P$ and $p \in S$. For each of these states, we have added inner states of at most $|\Sigma_f|$ automata of the form $\mathcal{A}_{p,a,S}$ that were built into it (again, if $a \notin \Sigma_f$ then $\mathcal{A}_{p,a,S}$ has no inner states). Since each $\sigma_{p,a,S}$ has depth at most $|\Sigma_f||Q|(2^{|Q|} - 1)$, the number of states of $\mathcal{A}$ adds up to less than

$$\begin{aligned}
&|Q| \cdot 2^{|Q|} \cdot \left(1 + |\Sigma_f| \sum_{k=1}^{|\Sigma_f||Q|(2^{|Q|}-1)} |\Sigma_f|^{k-1} |Q_f|^k |Q|^k\right) \\
=&|Q| \cdot 2^{|Q|} \cdot \sum_{k=0}^{|\Sigma_f||Q|(2^{|Q|}-1)} (|\Sigma_f||Q_f||Q|)^k \\
=&|Q| \cdot 2^{|Q|} \cdot \frac{(|\Sigma_f||Q_f||Q|)^{|\Sigma_f||Q|(2^{|Q|}-1)+1} - 1}{|\Sigma_f||Q_f||Q| - 1} \\
<&|Q| \cdot |Q_f|^{|Q|} \cdot \frac{(|\Sigma_f||Q_f||Q|)^{|\Sigma_f||Q|(2^{|Q|}-1)+1}}{|Q_f|} \\
\leq&(|\Sigma_f||Q_f||Q|)^{|\Sigma_f||Q|2^{|Q|}}.
\end{aligned}$$

We are done. □

The remaining results in this section will be negative. We show that the implications $1 \Longrightarrow 3$, $2 \Longrightarrow 4$ and $4 \Longrightarrow 5$ for optimum strategies do not in general hold for games that satisfy none of the three conditions of Theorem 4.3.

4.12 Theorem. *There exists a game G for which the implications $1 \Longrightarrow 3$ and $2 \Longrightarrow 4$ for optimum strategies do not hold.*

Proof. Consider the game $G = (\Sigma, R, T)$ with $\Sigma = \{a\}$, the only replacement rule being $a \to aa$ and $L(T) = \{a^k \mid k \geq 2\}$. We show that there exists an optimum type 2 strategy but no optimum type 3 strategy for G. Since every type 2 strategy is also of type 1 and every type 4 strategy is also of type 3, this implies the theorem.

An optimum one-pass strategy for G could play a single Call and followed by only Read moves. This guarantees that the strategy wins on every non-empty input word. A type 2 automaton specifying such a strategy is displayed in Figure 4.4

However, no optimum type 3 strategy for G exists. If σ is a type 3 strategy and $\sigma(\varepsilon, a) = \mathit{Call}$, then $\sigma(\hat{a}^k, a) = \mathit{Call}$ for each $k \in \mathbb{N}$ by definition of type 3 strategies. Each play of this strategy

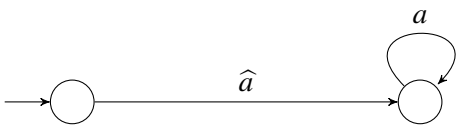
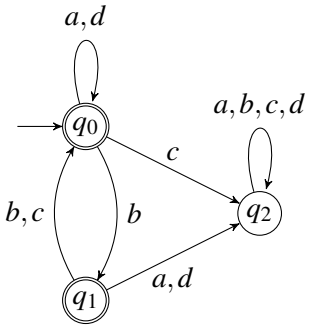
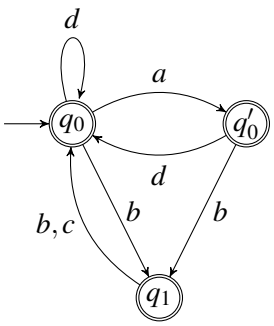


Figure 4.4.: Automaton of an optimum type 2 strategy for the game G in the proof of Theorem 4.12



(a) Target language automaton T



(b) Type 4 automaton $\mathcal{A}$

Figure 4.5.: Automata T and $\mathcal{A}$ used in the proof of Theorem 4.13

on a non-empty input word would be infinite, so the strategy would not be optimum. On the other hand, if $\sigma(\varepsilon, a) = \mathit{Read}$, then σ loses on a and is therefore also not optimum. □

The proof of Theorem 4.12 may seem to suggest that left-recursion is a necessary property of a counter-example, i. e. the possibility of deriving from a function symbol a word that begins with the same function symbol. While it is true that recursion is necessary (due to Theorem 4.3), neither left- nor right-recursion is necessary. An example (hence an alternative proof of Theorem 4.12) is given in Theorem A.1 in the appendix. We chose a left-recursive game in the proof above because it seems to be the simplest counter-example.

4.13 Theorem. *There exists a game G for which the implication $4 \Longrightarrow 5$ for optimum strategies does not hold.*

Proof. Consider the game $G = (\Sigma, R, T)$ with $\Sigma = \{a, b, c, d\}$, rules R given by

$$a \to b$$
$$c \to ac$$
$$d \to bad$$

and the target language automaton $T = (Q, \Sigma, \delta, q_0, F)$ depicted in Figure 4.5a. We claim that the strategy $\sigma_{\mathcal{A}}$ of the type 4 automaton $\mathcal{A}$ shown in Figure 4.5b is optimum.

Observe the correspondence of the states of $\mathcal{A}$ to states of T: For each $e \in \Sigma$ and $i, j = 1, 2$, if $\mathcal{A}$ has a transition from q_i or q'_i to q_j or q'_j reading e, then also T has a transition from q_i to q_j reading e. Moreover, the initial state of T and $\mathcal{A}$ is q_0 both times. Therefore, for any configuration in a play of $\sigma_{\mathcal{A}}$ it holds that its $\mathcal{A}$-state is q_0 or q'_0 if and only if its T-state is q_0, and its $\mathcal{A}$-state is q_1 if and only if its T-state is q_1. So a play of $\sigma_{\mathcal{A}}$ never leaves the accepting states q_0 and q_1 of T,

and as such it is winning if and only if it is finite. We show that *every* play of $\sigma_{\mathcal{A}}$ is finite. Thus, $L(\sigma_{\mathcal{A}}) = \Sigma^*$ and $\sigma_{\mathcal{A}}$ is optimum.

Let $\Pi = (K_0, K_1, \dots)$ be a play of $\sigma_{\mathcal{A}}$ and let $K_n = (\overline{u}, v)$ be a configuration in Π. We show by induction on the length of v that Π is finite.

If $v = \varepsilon$, then, by definition, K_n is the last configuration of Π, so Π is finite.

If $v = ev'$ for some $e \in \Sigma$ and $v' \in \Sigma^*$, then it is easy to check (separately for each combination of $e \in \{a,b,c,d\}$ and $\mathcal{A}$-states $q \in \{q_0, q_0', q_1\}$ of K_n) that Π contains a configuration of the form $K_m = (\overline{uv}, v')$, where $m \in \{n+1, \dots, n+6\}$ and $\overline{v} \in \overline{\Sigma}^*$. Hence, Π is finite by the induction hypothesis.

It remains to show that there exists no optimum type 5 strategy for G. Since the existence of the previous strategy implies that an optimum strategy for G must win on *all* words of Σ^*, we need to prove that for each type 5 automaton $\mathcal{B}$ it holds that $L(\sigma_{\mathcal{B}}) \subsetneq \Sigma^*$.

Let $\mathcal{B} = (Q, \Sigma, \delta_{\mathcal{B}}, q_0, F)$ be any type 5 automaton, i.e. an automaton obtained from T by removing transitions. We can assume that $\delta(q_0, c) = \delta(q_1, a) = \delta(q_1, d) = \perp$ since otherwise $\sigma_{\mathcal{B}}$ would lose on c, ba or bd.

If $\delta_{\mathcal{B}}(q_0, a) = q_0$, then the unique[1] play $(K_0, K_1, \dots)$ of $\sigma_{\mathcal{B}}$ on ac is infinite, since $K_{2k} = ((a\widehat{c})^k, ac)$ for each $k \in \mathbb{N}_0$ by induction. On the other hand, if $\delta_{\mathcal{B}}(q_0, a) = \perp$, then the play $(K_0, K_1, \dots)$ of $\sigma_{\mathcal{B}}$ on ad is infinite, since $K_{4k} = ((\widehat{ab}\widehat{db})^k, ad)$ for each $k \in \mathbb{N}_0$, again by induction. Thus, $ac \notin L(\sigma_{\mathcal{B}})$ or $ad \notin L(\sigma_{\mathcal{B}})$ and therefore $\sigma_{\mathcal{B}}$ is not optimum. □

Note that the counterexamples in the proofs of Theorems 4.12 and 4.13 are ones where Juliet basically plays alone because Romeo can never choose from a set of more than one replacement word. So Juliet has complete information except that she does not know the input word. But even if it were announced to her that the input word is a in the game of Theorem 4.12 or $acad$ in the game of Theorem 4.13 then there would not be a strategy of the respective restricted type that wins on these words (it is not difficult to see this for $acad$). An overview of the implications for optimum strategies is given in Figure 7.1 in the summary.

4.2. Implications for optimal strategies

Now, we will consider implications for optimal strategies. Recall that if an optimum strategy exists, then a strategy is optimal if and only if it is optimum. Therefore, any implication of which we can show that it holds for optimal strategies also holds for optimum strategies. Conversely, the implications $1 \Longrightarrow 3$, $2 \Longrightarrow 4$ and $4 \Longrightarrow 5$ that do not in general hold for optimum strategies also do not in general hold for optimal strategies. So the questions are: Under what (possibly stronger) conditions do the implications that hold for optimum strategies also hold for optimal strategies? Can we find conditions under which implications that hold for optimum strategies do not in general hold for optimal strategies?

[1] For the game G at hand, a play is uniquely determined by the one-pass strategy and the input word, since $|L(R_e)| = 1$ for each $e \in \Sigma_f$.

We begin with a positive result. It is a much weaker version of Theorem 4.3 for optimal strategies, and says that for each optimal strategy there also exists an automaton based optimal strategy if the target language is finite.

4.14 Theorem. *The implications* $1 \Longrightarrow 2$ *and* $3 \Longrightarrow 4$ *for optimal strategies hold for games with a finite target language.*

Proof. Let σ be an optimal one-pass strategy for a game $G = (\Sigma, R, T)$ with finite target language $L(T)$. We assume without loss of generality that $L(\sigma) \neq \emptyset$. For $w \in L(\sigma)$ let B_w be such that every play of σ on w has at most B_w configurations (see Lemma 3.4). Let B be the maximum of these numbers B_w, which are finitely many since $L(T)$ is finite. We construct a type 2 automaton or (if σ is of type 3) a type 4 automaton $\mathcal{A}$ such that the plays of σ on a word $w \in L(\sigma)$ are the same as the plays of $\sigma_{\mathcal{A}}$ on w.

Let $H \subseteq \overline{\Sigma}^{\leq B}$ be the set of history strings of configurations occurring in plays of σ on words from $L(\sigma)$. A type 2 automaton $\mathcal{A}$ that specifies a strategy $\sigma_{\mathcal{A}}$ that is equivalent to σ can be constructed with H as its set of states as follows. The initial state is ε. If $\overline{u}, \overline{ua} \in H$ with $\overline{a} \in \overline{\Sigma}$, then it has a transition from $\overline{u}$ reading $\overline{a}$ to $\overline{ua}$. Any remaining transitions required to satisfy the definition of type 2 automata can be chosen arbitrarily (they will not be used in a play of $\sigma_{\mathcal{A}}$ on a word from $L(\sigma)$). It is straightforward that for each $w \in L(\sigma)$ the plays of σ on word a w are the same as the plays of $\sigma_{\mathcal{A}}$ on w. Hence, $L(\sigma) \subseteq L(\sigma_{\mathcal{A}})$ and, since σ is optimal, $L(\sigma) = L(\sigma_{\mathcal{A}})$.

If σ is even a type 3 strategy, then we can construct a type 4 automaton of an equivalent strategy in much the same way except that state $\overline{u}$ and $\overline{v}$ with $\natural\overline{u} = \natural\overline{v}$ are merged to a single state and transitions for symbols in $\widehat{\Sigma_f}$ are omitted. That is, the set of states is $\{\natural\overline{u} \mid \overline{u} \in H\}$ and if $\overline{u}, \overline{ua} \in H$ then there is a transition from $\natural\overline{u}$ reading a to $\natural\overline{ua}$, and any missing transitions for symbols in $\Sigma \setminus \Sigma_f$ are chosen arbitrarily. Using the definition of type 3 strategies it is easy to check that this specifies an equivalent strategy. □

Since we know already that an optimal type 1 strategy exists for each game with a finite target language, it follows that there even exists an optimal type 2 strategy for these games.

4.15 Corollary. *For each game with a finite target language there exists an optimal type 2 strategy.*

Proof. Immediate from Theorem 3.6 and Theorem 4.14. □

Recall from Theorem 3.15 that an optimal one-pass strategy exists for each game with self-delimiting replacement languages. Therefore, if the analogon of Theorem 4.10 for optimal instead of optimum strategies is true, then every game with self-delimiting replacement languages has an optimal strategy of type 2. Indeed, it would be somewhat surprising if the implication $1 \Longrightarrow 2$ for optimal strategies does not hold for games with self-delimiting replacement languages – because not only the implication $1 \Longrightarrow 2$ for optimum strategies but, as we will see later, also the implication $1 \Longrightarrow 2$ for winning strategies holds for games with self-delimiting replacement languages. However, it is an open problem whether the same is true for optimal strategies. We briefly discuss difficulties in trying to answer this question.

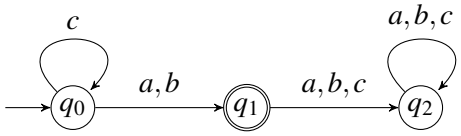


Figure 4.6.: Target language automaton T for Example 4.16

The proof in the case of optimum strategies was based on Lemma 4.6. In that lemma, we showed that every optimum strategy can be changed to an equivalent strategy σ where the moves after completely processing some prefix u of the input word (leading to a history string $\overline{u} \in H_u^\sigma$) depend only on $\delta^*(q_0, \natural\overline{u})$ and $states(q_0, u, \sigma)$. The according statement is not true for optimal strategies, as the following examples shows.

4.16 Example. Consider the game $G = (\Sigma, R, T)$ with $\Sigma = \{a,b,c\}$, rules $a \to c$ and $b \to c$ and the DFA T as per Figure 4.6.

The strategy σ with $\sigma(\varepsilon, a) = \sigma(\varepsilon, b) = Call$ and $\sigma(\overline{u}, a) = \sigma(\overline{u}, b) = Read$ for $\overline{u} \in \overline{\Sigma}^+$ is optimal with $L(\sigma) = \{dc^k e \mid d \in \{a,b,c\}, k \in \mathbb{N}_0, e \in \{a,b\}\}$. Indeed, a it is not hard to check that a strategy that wins on some word $w \notin L(\sigma)$ must lose on some other word that is in $L(\sigma)$. However, suppose there exists an equivalent strategy σ' such that for all $u_1, u_2 \in \Sigma^*$ and $\overline{u_i} \in H_{u_i}^{\sigma'}$ with $\delta^*(q_0, \natural\overline{u_1}) = \delta^*(q_0, \natural\overline{u_2})$ and $states(q_0, u_1, \sigma') = states(q_0, u_2, \sigma')$ it holds that $\sigma'[\overline{u_1}] = \sigma'[\overline{u_2}]$. Since σ' wins on ab, it must hold that $states(q_0, a, \sigma') = \{q_0\}$ and $\sigma'(\widehat{ac}, b) = Read$. With $u_1 = a$, $\overline{u_1} = \widehat{ac}$ and $u_2 = \overline{u_2} = \varepsilon$ it follows that $\sigma'(\varepsilon, b) = Read$. But then σ' is not equivalent to σ, since σ' loses on ba.

The example shows that *not every* optimal strategy in a game with self-delimiting replacement languages has an equivalent strategy that satisfies the properties of Lemma 4.6 (barring the requirement of being optimum). However, it could still be the case that *some* optimal strategy satisfies these properties. Further research is necessary to find out whether the implication $1 \implies 2$ for optimal strategies holds in games with self-delimiting replacement languages. The importance of this question is that a positive answer would imply that every game with self-delimiting replacement languages has an optimal strategy that is based on an automaton.

A similarly important question is whether the analogon of Theorem 4.3 is also true for optimal instead of optimum strategies. Due to Theorem 3.6, a positive answer would mean that every non-recursive game and every game with a finite target language has an optimal strategy that is even of type 5. We can answer this question, but unfortunately the answer is negative. We will show that the implications $1 \implies 3$, $2 \implies 4$ and $4 \implies 5$ for optimal strategies do not in general semi-hold even for non-recursive games with a finite target language and finitely many rules.

4.17 Theorem. *There exists a non-recursive game $G = (\Sigma, R, T)$ with R and $L(T)$ finite such that the implications $1 \implies 3$ and $2 \implies 4$ for optimal strategies do not semi-hold for G.*

Proof. We will construct a non-recursive game with finite set of rules and finite target language for which an optimal strategy of type 2 exists but no optimal strategy of type 3 exists. Since two one-pass strategies for a non-recursive game are equivalent if and only if they are semi-equivalent, this implies the theorem.

Consider the game $G = (\Sigma, R, T)$ over the alphabet $\Sigma = \{a, b, c, d, e\}$ with the set of rules R given by

$$a \to b + c$$
$$b \to cd$$
$$c \to e$$

and target language $L(T) = \{e, cd\}$.

The existence of an optimal type 2 strategies for G is guaranteed by Corollary 4.15. However, there exists no type 3 strategy for G that is optimal. Let σ be some type 3 strategy. We show first that σ does not win on a. Suppose σ does win on a. Then $\sigma(\varepsilon, a) = \sigma(\varepsilon, b) = \mathit{Call}$. If $\sigma(\varepsilon, c) = \mathit{Read}$, then the play

$$(\varepsilon, a), (\widehat{a}, c), (\widehat{ac}, \varepsilon)$$

of σ on a is losing – contradiction. If $\sigma(\varepsilon, c) = \mathit{Call}$, then the play

$$(\varepsilon, a), (\widehat{a}, b), (\widehat{ab}, cd), (\widehat{abc}, ed), (\widehat{abce}, d), (\widehat{abced}, \varepsilon)$$

of σ on a is losing – contradiction.

It is also easy to see that no strategy can win on a word that starts with a and has length at least 2. A one-pass strategy σ' that wins on all words from $L(\sigma)$ and also on a can be defined by

$$\sigma'(\overline{u}, f) = \begin{cases} \mathit{Call}, & \text{if } (\overline{u}, f) \in \{(\varepsilon, a), (\widehat{a}, b), (\widehat{a}, c)\}, \\ \mathit{Read}, & \text{if } (\overline{u}, f) = (\widehat{ab}, c), \\ \sigma(\overline{u}, f), & \text{otherwise} \end{cases}$$

for $\overline{u} \in \overline{\Sigma}^*$, $f \in \Sigma_f = \{a, b, c\}$. Since $L(\sigma) \subsetneq L(\sigma) \cup \{a\} = L(\sigma')$, it follows that σ is not optimal.□

As mentioned above, also the implication $4 \Longrightarrow 5$ for optimal strategies does not in general hold even for non-recursive games with a finite target language and finitely many rules.

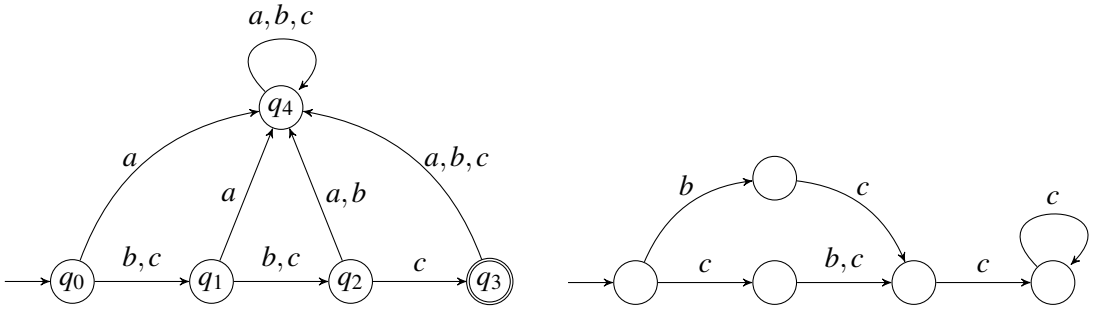
4.18 Theorem. *There exists a non-recursive game $G = (\Sigma, R, T)$ with R and $L(T)$ finite such that the implication $4 \Longrightarrow 5$ for optimal strategies does not semi-hold for G.*

Proof. Consider the game $G = (\Sigma, R, T)$ over the alphabet $\Sigma = \{a, b, c\}$ with rules given by

$$a \to bb + cbc$$
$$b \to cc$$

and target language automaton T as per Figure 4.7a. The finite target language is $L(T) = \{bbc, bcc, cbc, ccc\}$. Like in the proof of Theorem 4.17 it suffices to show that the implication $4 \Longrightarrow 5$ does not *hold* for G, since G is non-recursive.

We claim that the type 4 strategy $\sigma_{\mathcal{A}}$ specified by the type 4 automaton $\mathcal{A}$ shown in Figure A.2b is optimal. It can be checked easily that $L(\sigma_{\mathcal{A}}) = \{a, bb, bcc, cbc, ccc\}$. The only words that are not



(a) Target language automaton T (b) Type 4 automaton $\mathcal{A}$

Figure 4.7.: Automata T and $\mathcal{A}$ for the proof of Theorem 4.18

in $L(\sigma_{\mathcal{A}})$ even though a strategy exists that wins on them are *bc*, *bbc* and *cb*. However, a one-pass strategy that wins on *bc* cannot win on *bcc*; a strategy that wins on *bbc* cannot win on *bb*; a strategy that wins on *cb* cannot win on *cbc*. Therefore, every one-pass strategy that wins on a word on which $\sigma_{\mathcal{A}}$ does not win also loses on a word on which $\sigma_{\mathcal{A}}$ wins. Hence, there exists no strategy σ with $L(\sigma_{\mathcal{A}}) \subsetneq L(\sigma)$, which means that σ is optimal.

It remains to show that there exists no optimal type 5 strategy for G. Obviously, T is the minimal DFA for $L(T)$, so a type 5 automaton is obtained from T by removing transitions. Let $\mathcal{B}$ be a type 5 automaton specifying a type 5 strategy $\sigma_{\mathcal{B}}$. Observe that $\sigma_{\mathcal{B}}$ cannot win on a: In order to win on a, a strategy must play Call in the configurations (ε, a) and $(\widehat{a}b, b)$ and it must play Read in the configurations $(\widehat{a}, bb)$ and $(\widehat{a}c, bc)$. But a type 5 strategy must play the same in the configurations $(\widehat{a}b, b)$ and $(\widehat{a}c, bc)$, because both have T-state q_1 and current symbol b. However, we can find a one-pass strategy σ with $L(\sigma_{\mathcal{B}}) \subsetneq L(\sigma_{\mathcal{B}}) \dot{\cup} \{a\} = L(\sigma)$. Indeed, such a strategy can be defined by

$$\sigma(\overline{u}, d) = \begin{cases} \mathit{Call}, & \text{if } (\overline{u}, d) = (\varepsilon, a) \text{ or } (\overline{u}, d) = (\widehat{a}b, b), \\ \mathit{Read}, & \text{if } (\overline{u}, d) = (\widehat{a}, b) \text{ or } (\overline{u}, d) = (\widehat{a}c, b), \\ \sigma_{\mathcal{B}}(\overline{u}, d), & \text{otherwise} \end{cases}$$

for $\overline{u} \in \overline{\Sigma}^*$ and $d \in \Sigma_f = \{a, b\}$. Hence, $\sigma_{\mathcal{B}}$ is not optimal. □

For completeness, it shall be mentioned that there also exist non-recursive counter-examples for the implication $4 \Longrightarrow 5$ where Juliet plays "alone", i. e. $L(R_a)$ is a singleton for each $a \in \Sigma_f$. This is shown in Theorem A.2 in the appendix.

We have seen in this section that, compared to the case of optimum strategies, a lot stronger conditions are required to guarantee that implications for optimal strategies hold. Whereas all implications hold for optimum strategies in non-recursive games or if the target language is finite, we have shown that several implications for optimal strategies are not in general satisfied even if even if the game is non-recursive *and* has a finite target language. Open questions in this area are whether the implications $1 \Longrightarrow 2$ and $3 \Longrightarrow 4$ hold more generally than if the target language is finite. As it stands it is even conceivable that these implications hold for *all* context-free games, but a proof or a counter-example has yet to be found.

A summary of our results for optimal strategies can be found in Figure 7.2 in the last chapter. In the last section of this chapter we will take a look at implications for winning strategies.

4.3. Implications for winning strategies

Given our work on implications for optimal and optimum strategies, it is relatively easy to obtain results for implications for winning strategies. A convenient result is that in the case of finitely many rules or a finite target language, one can narrow the search to automaton strategies when trying to determine whether a winning strategy on a given word exists.

4.19 Theorem. *The implications* $1 \Longrightarrow 2$ *and* $3 \Longrightarrow 4$ *for winning strategies hold for games with finitely many rules and for games with a finite target language.*

Proof. The statement follows from Lemma 3.4 analogously to the of Theorem 4.14, except that only a single word needs to be considered instead of all words in $L(\sigma)$. Therefore, it is not required that $L(\sigma)$ is finite. That is why the theorem holds besides games with a finite target language also for games with finitely many rules. □

The next theorem is the counterpart of Theorem 4.10 for winning strategies.

4.20 Theorem. *The implication* $1 \Longrightarrow 2$ *for winning strategies holds for games with self-delimiting replacement languages. More precisely, if* $G = (\Sigma, R, T)$ *is a game with self-delimiting replacement languages and* $w \in L(\sigma)$ *for some one-pass strategy* σ*, then there exists a type 2 automaton* $\mathcal{A}$ *such that* $w \in L(\sigma_{\mathcal{A}})$ *and* $\mathcal{A}$ *has at most* $|w| \cdot \left(|\Sigma_f| \cdot |Q_f| \cdot |Q|\right)^E$ *states, where* $E = |\Sigma_f| \cdot |Q| \cdot 2^{|Q|}$ *and* $|Q_f|$ *is the maximal number of non-error states of a minimal DFA of a replacement language* $L(R_a)$ *for* $a \in \Sigma_f$ *and* $|Q|$ *is the number of states of the minimal DFA for* $L(T)$*.*

Proof. The proof is similar to that of the according implication for optimum strategies. Let $G = (\Sigma, R, T)$ be a game with self-delimiting replacement languages. Let σ be a one-pass strategy and let $w = a_1 \dots a_n \in L(\sigma)$ with $a_i \in \Sigma$. We construct a type 2 automaton $\mathcal{A}$ with $w \in L(\sigma_{\mathcal{A}})$.

By Lemma 3.14 we can assume that σ has depth at most $|\Sigma_f| \cdot |Q| \cdot (2^{|Q|} - 1)$. For $i = 0, \dots, n$ let $S_i = \mathit{states}(q_0, a_1 \dots a_i, \sigma)$. Note that $S_0 = \{q_0\}$ and $S_n \subseteq F$. The automaton $\mathcal{A}$ has a state (p, i) for each $i = 0, \dots, n$ and $p \in S_i$ and further states coming from automata nested into it.

A state (p, n) has a transition to itself for each $a \in \Sigma$. The outgoing transitions of a state (p, i) with $i < n$ are determined as follows. Let $\overline{u} \in H^{\sigma}_{a_1 \dots a_i}$ be such that $\delta^*(q_0, \natural\overline{u}) = p$. Let $S = \mathit{states}(p, a_{i+1}, \sigma[\overline{u}])$. Note that the automaton $\mathcal{A}_{p, a_{i+1}, S}$ from Lemma 4.8 exists. We take a copy of it and remove all transitions leaving one of its accepting states $s \in S$. Then, we merge its initial state with the state (p, i) of $\mathcal{A}$, and for each $s \in S$ we merge its state s with the state $(s, i+1)$ of $\mathcal{A}$. Indeed, $(s, i+1)$ is a state of $\mathcal{A}$ because $S = \mathit{states}(q, a_{i+1}, \sigma[\overline{u}]) \subseteq \mathit{states}(q_0, a_1 \dots a_{i+1}, \sigma)$.

Consider a play of $\sigma_{\mathcal{A}}$ on w. By induction on i one can see that every prefix $a_1 \dots a_i$ of w will be completely processed at some point, and if the T-state of the corresponding configuration is p, then $p \in S_i$ and the $\mathcal{A}$-state of the corresponding configuration is (p, i). For $i = n$ this means that each play of $\sigma_{\mathcal{A}}$ on w is finite and the T-state of the final configuration is in S_n. Since $S_n \subseteq F$, it follows that $w \in L(\sigma_{\mathcal{A}})$.

Finally, let us estimate the number of states of $\mathcal{A}$. We can assume that $F \subsetneq Q$ because otherwise the theorem is trivial. Since $|S_0| = 1$ and $|S_n| \leq |F| \leq |Q| - 1$, there are at most

$$1 + (n-1) \cdot |Q| + |Q| - 1 = n \cdot |Q|$$

states of the form (p, i). For each of them, we added at most

$$\sum_{k=1}^{|\Sigma_f||Q|(2^{|Q|}-1)} |\Sigma_f|^{k-1} |Q_f|^k |Q|^k$$

more states, coming from inner states of the automata built into $\mathcal{A}$. Using that $|Q_f| \geq 2$ (since replacement languages do not contain ε), this makes in total at most

$$\begin{aligned}
& n \cdot |Q| \cdot \left(1 + \sum_{k=1}^{|\Sigma_f||Q|(2^{|Q|}-1)} |\Sigma_f|^{k-1} |Q_f|^k |Q|^k\right) \\
\leq & n \cdot |Q| \cdot \left(1 + |Q_f||Q| \frac{\left(|\Sigma_f||Q_f||Q|\right)^{|\Sigma_f||Q|(2^{|Q|}-1)} - 1}{|\Sigma_f||Q_f||Q| - 1}\right) \\
\leq & n \cdot |Q| \cdot \left(1 + |Q_f| \left(\left(|\Sigma_f||Q_f||Q|\right)^{|\Sigma_f||Q|(2^{|Q|}-1)} - 1\right)\right) \\
\leq & n \cdot \left(|\Sigma_f||Q_f||Q|\right)^{|\Sigma_f||Q|2^{|Q|}}
\end{aligned}$$

states in $\mathcal{A}$. □

The remaining results in this section are negative. They are analogous to the negative results that we showed for optimal strategies.

4.21 Theorem. *There exists a non-recursive game $G = (\Sigma, R, T)$ with R and $L(T)$ finite such that the implications $1 \Longrightarrow 3$ and $2 \Longrightarrow 4$ for winning strategies do not hold for G.*

Proof. An example is the game from the proof of Theorem 4.17 with input word a. □

4.22 Theorem. *There exists a non-recursive game $G = (\Sigma, R, T)$ with R and $L(T)$ finite such that the implication $4 \Longrightarrow 5$ for winning strategies does not hold for G.*

Proof. An example is the game from the proof of Theorem 4.18 with input word a. □

An overview of the results of this section is given in Figure 7.3.

This concludes our chapter about implications for one-pass strategies. Besides several positive results for restricted cases, we showed that the implications $1 \Longrightarrow 3$, $2 \Longrightarrow 4$ and $4 \Longrightarrow 5$ do not hold in general (neither for winning strategies, nor for optimum or optimal strategies). The general case of the implications $1 \Longrightarrow 2$ and $3 \Longrightarrow 4$ is open.

In the next chapter, we will describe the language of words on which an automaton strategy wins. The theorem proved in that chapter will also be helpful later to obtain upper bounds on the complexity of decision problems.

5. Winning sets of automaton strategies

We show that the language $L(\sigma_{\mathcal{A}})$ of words on which the strategy of a automaton $\mathcal{A}$ wins is regular. We will do so by constructing a *universal finite automaton (UFA)* that recognizes $L(\sigma_{\mathcal{A}})$. A UFA is the same as an NFA except that it accepts a word w if w is accepted in *all* possible runs (instead of just *some* possible run). It is possible to construct a DFA that recognizes the same language as a given UFA by using a powerset construction (analogously to how an equivalent DFA can be found for a given NFA). Thus, the language recognized by a UFA is regular. If the target language automaton T is deterministic, then the UFA can be constructed in time polynomial in the size of G and $\mathcal{A}$. This result was stated already by Abiteboul et al. [3, Proposition 3.1] for a type 4 automaton $\mathcal{A}$. However, the construction from their proof sketch does not take into account the possibility that the play of $\sigma_{\mathcal{A}}$ may not terminate on some input words. We give a construction that also treats this case and generalize the result to type 2 automata.

Intuitively, the universal quantification in the semantics of the UFA corresponds to the fact that $\sigma_{\mathcal{A}}$ must win against *all* Romeo strategies in order to win on an input word.

5.1 Theorem. *For a game $G = (\Sigma, R, T)$ and a type 2 or 4 automaton $\mathcal{A}$, the set $L(\sigma_{\mathcal{A}})$ is regular. If T is a DFA, then a UFA $\mathcal{U}$ recognizing $L(\sigma_{\mathcal{A}})$ can be constructed time polynomial in the size of G and $\mathcal{A}$.*

Proof. It is enough to prove the theorem for type 2 automata since each type 4 automaton can be transformed into a type 2 automaton. By Proposition 2.2 we can assume that $\mathcal{A}$ does the test for the target language. This is the only step where we require T to be a DFA to ensure polynomial running time. We do not need to consider T any more from now on. Let $\mathcal{A} = (Q, \overline{\Sigma}, \delta, q_0, F)$.

The UFA recognizing $L(\sigma_{\mathcal{A}})$ is $\mathcal{U} = (Q \dot\cup \{q_\infty\}, \Sigma, \delta_{\mathcal{U}}, q_0, F\}$ for a transition relation $\delta_{\mathcal{U}}$ that we will define below. So $\mathcal{U}$ uses the same states as $\mathcal{A}$ plus an additional state q_∞, and $\mathcal{U}$ has the same initial state and the same accepting states as $\mathcal{A}$. The state q_∞ will be needed to treat the case that $\sigma_{\mathcal{A}}$ might play infinitely long.

For a state q of $\mathcal{A}$, let $\mathcal{A}_q$ denote the strategy automaton obtained from $\mathcal{A}$ by changing its initial state to q.

Let E be the set of all subexpressions of the regular expressions R_a for $a \in \Sigma_f$. We will define a relation $\mathit{Move} \subseteq Q \times (E \cup \Sigma) \times Q$, where a triple $(q, a, q') \in \mathit{Move}$ with $a \in \Sigma$ shall denote that a play of $\sigma_{\mathcal{A}_q}$ on a may terminate in $\mathcal{A}_q$-state q (depending on Romeo's strategy). These triples will constitute the core of the transition relation of $\mathcal{U}$. Note that the union $E \cup \Sigma$ is not disjoint since replacement words are non-empty by definition, so some alphabet symbols must occur as atomic subexpressions.

For $(q, r, q') \in Q \times (E \cup \Sigma) \times Q$, we define inductively that $(q, r, q') \in \mathit{Move}$ if any of the following conditions holds for some $a \in \Sigma$, $r_1, r_2 \in E$ and $q'' \in Q$:

(i) $r = a$ and $q' = \delta(q,a)$

(ii) $r = a$, $\delta(q,a) = \bot$ and $(\delta(q,\widehat{a}),R_a,q') \in \mathit{Move}$

(iii) $r = \varepsilon$ and $q = q'$

(iv) $r = r_1 r_2$ and $(q,r_1,q''),(q'',r_2,q') \in \mathit{Move}$

(v) $r = r_1 + r_2$ and $(q,r_1,q') \in \mathit{Move}$

(vi) $r = r_1 + r_2$ and $(q,r_2,q') \in \mathit{Move}$

(vii) $r = r_1^*$ and $q' = q$

(viii) $r = r_1^*$ and $(q,r_1,q''),(q'',r_1^*,q') \in \mathit{Move}$

The semantics of the *Move* relation is described by the following claim.

5.2 Claim. *For $q,q' \in Q$ and $r \in E \cup \Sigma$, the following are equivalent:*

- $(q,r,q') \in \mathit{Move}$
- *There exist $w \in L(r)$ and a Romeo strategy τ such that $\Pi(\sigma_{\mathcal{A}_q},\tau,w)$ terminates in $\mathcal{A}_q$-state q'.*

The proof of this claim is postponed until the end of this proof.

If it were guaranteed that every play of $\sigma_{\mathcal{A}}$ is finite, then we could finish the construction of $\mathcal{U}$ by letting $\delta_{\mathcal{U}} = \{(q,a,q') \in \mathit{Move} \mid a \in \Sigma\}$. However, this is problematic if a word w exists such that every finite play of $\sigma_{\mathcal{A}}$ on w is winning, but there also exist infinite plays of $\sigma_{\mathcal{A}}$ on w. In this case, $\sigma_{\mathcal{A}}$ does not win on w, but w would be universally accepted by $\mathcal{U}$ if we defined $\delta_{\mathcal{U}}$ as suggested. Hence, we need to pay special attention to the case that Romeo can force infinite play against $\sigma_{\mathcal{A}}$. We will do so by adding further transitions to $\mathcal{U}$ that ensure that if an infinite play of $\sigma_{\mathcal{A}}$ on w exists, then there exists a run of $\mathcal{U}$ on w that terminates in the non-accepting state q_∞.

The idea to detect the possibility of an infinite play is the following: For a play to be infinite there must be a function symbol $a \in \Sigma_f$ and a state $q \in Q$ such that a is called in q and in the "sub-play" on the replacement word of a there occurs another configuration with $\mathcal{A}$-state q and current symbol a. To detect the possibility of such a play, we introduce a relation $\mathit{Next} \subseteq Q \times (E \cup \Sigma) \times Q \times \Sigma$. A tuple (q,r,q',a) shall be in *Next* if and only if there exists a play of $\sigma_{\mathcal{A}_q}$ on some word from $L(r)$ that contains a configuration with $\mathcal{A}_q$-state q' and current symbol a. For $(q,r,q',a) \in Q \times (E \cup \Sigma) \times Q \times \Sigma$, we define inductively that $(q,r,q',a) \in \mathit{Next}$ if any of the following conditions holds for some $b \in \Sigma$, $r_1,r_2 \in E$ and $q'' \in Q$:

(i) $r = a$ and $q = q'$

(ii) $r = b$, $\delta(q,b) = \bot$ and $(\delta(q,\widehat{b}),R_b,q',a) \in \mathit{Next}$

(iii) $r = r_1 r_2$ and $(q,r_1,q',b) \in \mathit{Next}$

(iv) $r = r_1 r_2$, $(q, r_1, q'') \in \mathit{Move}$ and $(q'', r_2, q', a) \in \mathit{Next}$

(v) $r = r_1 + r_2$ and $(q, r_1, q', a) \in \mathit{Next}$

(vi) $r = r_1 + r_2$ and $(q, r_2, q', a) \in \mathit{Next}$

(vii) $r = r_1^*$, $(q, r_1^*, q'') \in \mathit{Move}$ and $(q'', r_1, q', a) \in \mathit{Next}$

5.3 Claim. *For $q, q' \in Q$, $r \in E \cup \Sigma$ and $a \in \Sigma$, the following are equivalent:*

- $(q, r, q', a) \in \mathit{Next}$
- *There exist $w \in L(r)$ and a Romeo strategy τ such that $\Pi(\sigma_{\mathcal{A}_q}, \tau, w)$ contains a configuration with $\mathcal{A}_q$-state q' and current symbol a.*

Also the proof of this claim will be provided later.

As a last ingredient before we can define $\delta_{\mathcal{U}}$, we define the relation $\mathit{Inf} \in Q \times (\Sigma \cup E)$ that shall contain a pair (q, r) if an infinite play of $\sigma_{\mathcal{A}_q}$ on some word from $L(r)$ exists. For $q \in Q$ and $r \in E \cup \Sigma$, we define inductively that $(q, r) \in \mathit{Inf}$ if any of the following conditions holds for some $a \in \Sigma$, $r_1, r_2 \in E$ and $q \in Q$:

(i) $r = a$, $\delta(q, a) = \bot$ and $(\delta(q, \widehat{a}), R_a, q, a) \in \mathit{Next}$

(ii) $r = a$, $\delta(q, a) = \bot$ and $(\delta(q, \widehat{a}), R_a) \in \mathit{Inf}$

(iii) $r = r_1 r_2$ and $(q, r_1) \in \mathit{Inf}$

(iv) $r = r_1 r_2$, $(q, r_1, q') \in \mathit{Move}$ and $(q', r_2) \in \mathit{Inf}$

(v) $r = r_1 + r_2$ and $(q, r_1) \in \mathit{Inf}$

(vi) $r = r_1 + r_2$ and $(q, r_2) \in \mathit{Inf}$

(vii) $r = r_1^*$, $(q, r_1^*, q') \in \mathit{Move}$ and $(q', r_1) \in \mathit{Inf}$

5.4 Claim. *For $q \in Q$ and $r \in E \cup \Sigma$, the following are equivalent:*

- $(q, r) \in \mathit{Inf}$
- *There exist $w \in L(r)$ and a Romeo strategy τ such that $\Pi(\sigma_{\mathcal{A}_q}, \tau, w)$ is infinite.*

Again, the proof of the claim is postponed until later.

The transition relation of $\mathcal{U}$ is defined as

$$\begin{aligned}\delta_{\mathcal{U}} = &\{(q, a, q') \mid (q, a, q') \in \mathit{Move}, a \in \Sigma\} \cup \\ &\{(q, a, q_\infty) \mid (q, a) \in \mathit{Inf}, a \in \Sigma\} \cup \\ &\{(q_\infty, a, q_\infty) \mid a \in \Sigma\}.\end{aligned}$$

Observe that each state of $\mathcal{U}$ has at least one transition for every symbol.

We argue briefly that $\mathcal{U}$ can be constructed in polynomial time. The bottleneck is the computation of the relations *Move*, *Next* and *Inf*. The inductive definition of these relations can be implemented naïvely with several nested loops. The outmost loop terminates when no new tuple was added during its last iteration. Since Q, Σ and E are all of polynomial size, the entire computation can be execute in polynomial time.

Apart from Claims 5.2, 5.3 and 5.4, it only remains to show that $L(\sigma_{\mathcal{A}}) = L(\mathcal{U})$. We prove both inclusions separately.

"$\subseteq$": Suppose $w \notin L(\mathcal{U})$ for some $w = a_1 \dots a_n$ with $a_i \in \Sigma$. Then there exists a sequence $q_1, \dots, q_n \in Q \cup \{q_\infty\}$ such that $(q_{i-1}, a_i, q_i) \in \delta_{\mathcal{U}}$ for each $i = 1, \dots, n$ and $q_n \notin F$. We first consider the case that $q_i \neq q_\infty$ for each $i = 1, \dots, n$. Then $(q_{i-1}, a_i, q_i) \in \mathit{Move}$ for each $i = 1, \dots, n$. By Claim 5.2 this implies that there exist Romeo strategies $\tau_1, \dots, \tau_n$ such that $\Pi(\sigma_{\mathcal{A}_{q_{i-1}}}, \tau_i, a_i)$ terminates in $\mathcal{A}_{q_{i-1}}$-state q_i for each $i = 1, \dots, n$. Then the play of $\sigma_{\mathcal{A}}$ against the concatenation strategy $\tau_1\langle a_1\rangle\tau_2\langle a_2\rangle \dots \langle a_{n-1}\rangle\tau_n$ on w terminates in $\mathcal{A}$-state $q_n \notin F$. Therefore, $w \notin L(\sigma_{\mathcal{A}})$.

Now we consider the case that $q_i = q_\infty$ for some $i \in \{1, \dots, n\}$. Let i be minimal with this property, then $(q_{j-1}, a_j, q_j) \in \mathit{Move}$ for each $j = 1, \dots, i-1$ and $(q_{i-1}, a_i) \in \mathit{Inf}$. By Claims 5.2 and 5.4 there exist Romeo strategies $\tau_1, \dots, \tau_i$ such that $\Pi(\sigma_{\mathcal{A}_{q_{j-1}}}, \tau_j, a_j)$ terminates in $\mathcal{A}_{q_{i-1}}$-state q_j for each $j = 1, \dots, i-1$ and $\Pi(\sigma_{\mathcal{A}_{q_{i-1}}}, \tau_i, a_i)$ is infinite. Thus, the play of $\sigma_{\mathcal{A}}$ against $\tau_1\langle a_1\rangle\tau_2\langle a_2\rangle \dots \langle a_{i-1}\rangle\tau_i$ on w is infinite (because it is already infinite on the prefix $a_1 \dots a_i$ of w). Hence, $w \notin L(\sigma_{\mathcal{A}})$.

"$\supseteq$": Suppose $w \notin L(\sigma_{\mathcal{A}})$ for some $w = a_1 \dots a_n$ with $a_i \in \Sigma$. Then there exists a Romeo strategy τ such that $\Pi(\sigma_{\mathcal{A}}, \tau, w)$ is either losing or infinite. If it is losing, let q_i be the $\mathcal{A}$-state of the configuration $(\overline{u_i}, a_{i+1} \dots a_n)$ at the time when $a_1 \dots a_i$ is completely processed, for $i = 1, \dots, n$, and let $\overline{u_0} = \varepsilon$ (the history string of the initial configuration). Then $\Pi(\sigma_{\mathcal{A}_{q_i}}, \tau[\overline{u_i}], a_{i+1})$ terminates in $\mathcal{A}_{q_i}$-state q_{i+1} for each $i = 0, \dots, n-1$ (recall that $\tau[\overline{u_i}]$ is the substrategy of τ after $\overline{u_i}$). Thus, by Claim 5.2, $(q_i, a_{i+1}, q_{i+1}) \in \mathit{Move}$ for each $i = 0, \dots, n-1$, so there exists a run of $\mathcal{U}$ on w that terminates in q_n. Since $\Pi(\sigma_{\mathcal{A}}, \tau, w)$ is losing, $q_n \notin F$, so $w \notin L(\mathcal{U})$.

For the case that $\Pi(\sigma_{\mathcal{A}}, \tau, w)$ is infinite, let $a_1 \dots a_j$ be the longest prefix of w such that the play $\Pi(\sigma_{\mathcal{A}}, \tau, a_1 \dots a_j)$ is finite. Define $q_1, \dots, q_j$ like in the case where $\Pi(\sigma_{\mathcal{A}}, \tau, w)$ is losing. Like above, $(q_i, a_{i+1}, q_{i+1}) \in \mathit{Move}$ for each $i = 0, \dots, j-1$. Let $\overline{u}$ be the history string of the configuration of the time when $a_1 \dots a_j$ is completely processed. Then $\Pi(\sigma_{\mathcal{A}_{q_j}}, \tau[\overline{u}], a_{j+1})$ is infinite, hence $(q_j, a_{j+1}) \in \mathit{Inf}$ by Claim 5.4. Thus, by definition of $\delta_{\mathcal{U}}$, there exists a run of $\mathcal{U}$ on w that terminates in the non-accepting state q_∞. Therefore, $w \notin L(\mathcal{U})$.

We are still in debt to prove the Claims 5.2, 5.3 and 5.4. Each of these proofs is done by separately showing the two implications of the claimed equivalence, where the implication from the first to the second statement is shown by structural induction and the converse implication is shown by induction on a less obvious variable.

Proof of Claim 5.2. We show both implications separately by induction.

"$\Longrightarrow$": The proof of the implication from the first to the second statement is by structural induction on elements of *Move*.

If $r = a \in \Sigma$ and $q' = \delta(q,a)$, then the second statement holds for $w = a$ and an arbitrary Romeo strategy τ, since $\sigma_{\mathcal{A}_q}$ plays *Read* on a.

If $r = a \in \Sigma$, $\delta(q,a) = \bot$ and $(\delta(q,\widehat{a}), R_a, q') \in \mathit{Move}$, then by the induction hypothesis there exist $w' \in L(R_a)$ and a Romeo strategy τ' such that $\Pi(\sigma_{\mathcal{A}_{\delta(q,\widehat{a})}}, \tau', w')$ terminates in $\mathcal{A}_{\delta(q,\widehat{a})}$-state q'. The second statement holds for $w = a$ and any Romeo strategy τ with $\tau(\varepsilon, a) = w'$ and $\tau(\widehat{a}\overline{u}, b) = \tau'(\overline{u}, b)$ for $\overline{u} \in \overline{\Sigma}^*$ and $b \in \Sigma_f$. That is, if the input word is a and Juliet's first move is *Call*, then τ replaces a by w' and continues to play like τ' plays on the input word w'.

If $r = \varepsilon$ and $q = q'$, then the second statement holds for $w = \varepsilon$ and any τ.

If $r = r_1 r_2$ and $(q, r_1, q''), (q'', r_2, q') \in \mathit{Move}$, then by the induction hypothesis there exist words $w_1 \in L(r_1)$, $w_2 \in L(r_2)$ and Romeo strategies τ_1 and τ_2 such that $\Pi(\sigma_{\mathcal{A}_q}, \tau_1, w_1)$ terminates in $\mathcal{A}_q$-state q'' and $\Pi(\sigma_{\mathcal{A}_{q''}}, \tau_2, w_2)$ terminates in $\mathcal{A}_{q''}$-state q'. The second statement holds for $w = w_1 w_2$ and the concatenation strategy $\tau = \tau_1 \langle w_1 \rangle \tau_2$.

The remaining cases are similar.

"$\Longleftarrow$": Let $w \in L(r)$ and a Romeo strategy τ be such that $\Pi(\sigma_{\mathcal{A}_q}, \tau, w)$ terminates in $\mathcal{A}_q$-state q'. Let $x \in \Sigma^*$ be the final string of this play. For $a \in \Sigma_f$, let n_a be the number of Calls of a occurring during this play. We prove the implication by induction on $|x| + |r| + \sum_{a \in \Sigma_f} n_a |R_a|$. We distinguish several cases and deduce the first statement in each case either directly or by applying the induction hypothesis.

- If $r = \varepsilon$, then $w = \varepsilon$ and therefore $q' = q$. Hence, $(q, r, q') \in \mathit{Move}$ follows directly from the definition, since condition (iii) is satisfied.
- If $r = a \in \Sigma$ and $\delta(q,a) \neq \bot$, then $w = a$ and $q' = \delta(q,a)$. Again $(q, r, q') \in \mathit{Move}$ follows directly, since condition (i) of the definition is satisfied.
- If $r = a \in \Sigma$ and $\delta(q,a) = \bot$, then $w = a$ and $\Pi(\sigma_{\mathcal{A}_{\delta(q,\widehat{a})}}, \tau[\widehat{a}], \tau(\varepsilon, a))$ terminates in $\mathcal{A}_{\delta(q,\widehat{a})}$-state q'. We can apply the induction hypothesis and get $(\delta(q,\widehat{a}), R_a, q') \in \mathit{Move}$. Thus, $(q, r, q') \in \mathit{Move}$ holds by condition (ii) of the definition.
- If $r = r_1^*$ then we can write $w = w_1 w_2 \ldots w_n$ with $n \in \mathbb{N}_0$ and $w_j \in L(r_1) \setminus \{\varepsilon\}$. If $n = 0$, then $w = \varepsilon$ and $q = q'$, so $(q, r, q') \in \mathit{Move}$ by condition (vii). If $n \geq 1$, let $\overline{u}$ be the final history string of $\Pi(\sigma_{\mathcal{A}_q}, \tau, w_1)$ and let $q'' = \delta(q, \overline{u})$ be the $\mathcal{A}_q$-state in which this play terminates. Since $|r_1^*| = |r_1| + 1 > |r_1|$, we can apply the induction hypothesis, which yields $(q, r_1, q'') \in \mathit{Move}$. Observe that $\Pi(\sigma_{\mathcal{A}_{q''}}, \tau[\overline{u}], w_2 w_3 \ldots w_n)$ terminates in $\mathcal{A}_{q''}$-state q', and its final string is the string x' with $\natural \overline{u} x' = x$. From $|\natural \overline{u}| \geq |w_1| > 0$ it follows that $|x| > |x'|$, so another application of the induction hypothesis yields $(q'', r_1^*, q') \in \mathit{Move}$. We get $(q, r, q') \in \mathit{Move}$ from condition (viii) of the definition.
- The cases $r = r_1 r_2$ and $r = r_1 + r_2$ are similar to and slightly easier than the previous case. □

Proof of Claim 5.3.

"$\Longrightarrow$": The proof is by structural induction on elements of *Next* and uses the same techniques that were already in the proof of the implication "$\Longrightarrow$" of Claim 5.2.

"$\Longleftarrow$": Let $w \in L(r)$ and τ be a Romeo strategy such that $\Pi(\sigma_{\mathcal{A}_q}, \tau, w)$ contains a configuration with $\mathcal{A}_q$-state q' and current symbol a and w is of minimal length with these properties. Let $\overline{u}$ be the history string of the first such configuration. For $b \in \Sigma_f$ let n_b be the number of occurrences of $\widehat{b}$ in $\overline{u}$, i. e. the number of Calls of b before this configuration is reached. The proof is by induction on $|r| + \sum_{b \in \Sigma_f} n_b |R_b|$. Note that $w \neq \varepsilon$, because the only configuration in a play on ε is $(\varepsilon, \varepsilon)$, which does not have current symbol a. So in particular $r \neq \varepsilon$. There are several cases.

- If $r = a$ and $q = q'$, then $(q, r, q', a) \in \mathit{Next}$ by condition (i) of the definition.
- If $r = b \in \Sigma$, but $b \neq a$ or $q \neq q'$, then $w = b$ and the initial configuration (ε, b) does not have both $\mathcal{A}_q$-state q' and current symbol a. It is necessary that $\delta(q, b) = \bot$ because otherwise also the second and last configuration (b, ε) does not have current symbol a. Thus, $\overline{u} = \widehat{b}\overline{v}$ for some $\overline{v} \in \overline{\Sigma}^*$ and $\Pi(\sigma_{\mathcal{A}_{\delta(q,\widehat{b})}}, \tau[\widehat{b}], \tau(\varepsilon, b))$ must have a configuration with $\mathcal{A}_{\delta(q,\widehat{b})}$-state q' and current symbol a. By the induction hypothesis, $(\delta(q, \widehat{b}), R_b, q', a) \in \mathit{Next}$. Hence, $(q, r, q', a) \in \mathit{Next}$ by condition (ii) of the definition.
- If $r = r_1^*$, then, since $w \neq \varepsilon$ by the argument above, $w = w_1 \dots w_n$ for some $n \geq 1$ and $w_i \in L(r_1) \setminus \{\varepsilon\}$. By minimality of the length of w, the play $\Pi(\sigma_{\mathcal{A}_q}, \tau, w_1 \dots w_{n-1})$ must be finite. Let q'' be the $\mathcal{A}_q$-state in which it terminates and $\overline{v}$ the final history string. By Claim 5.2, $(q, r_1^*, q'') \in \mathit{Move}$. Moreover, $\Pi(\sigma_{\mathcal{A}_{q''}}, \tau[\overline{v}], w_n)$ contains a configuration with $\mathcal{A}_{q''}$-state q' and current symbol a. By the induction hypothesis, $(q'', r_1, q', a) \in \mathit{Next}$. It follows from condition (vii) of the definition that $(q, r, q', a) \in \mathit{Next}$.
- The cases $r = r_1 r_2$ and $r = r_1 + r_2$ are similar. □

Proof of Claim 5.4.

"$\Longrightarrow$": The prove is by structural induction on elements of *Inf*.

We first consider the case that $r = a \in \Sigma$, $\delta(q, a) = \bot$ and $(\delta(q, \widehat{a}), R_a, q, a) \in \mathit{Next}$. The idea of a strategy for Romeo that achieves infinite play against $\sigma_{\mathcal{A}_q}$ on a is to reach a configuration with $\mathcal{A}_q$-state q and current symbol a (like the initial configuration), and then to repeat the moves from the beginning so that such a configuration occurs over and over again.

Let $w \in L(R_a)$ and a Romeo strategy τ be such that the play $\Pi(\sigma_{\mathcal{A}_{\delta(q,\widehat{a})}}, \tau, w) = (K_0, K_1, \dots)$ contains a configuration K_n with $\mathcal{A}_{\delta(q,\widehat{a})}$-state q and current symbol a. Such w and τ exist by Claim 5.3. For $i = 0, \dots, n$, let $\overline{u_i} \in \overline{\Sigma}^*$ and $v_i \in \Sigma^+$ be such that $K_i = (\overline{u_i}, v_i)$. Thus, $\overline{u_0} = \varepsilon$, $v_0 = w$, $\delta^*(q, \widehat{a}\overline{u_n}) = q$ and $v_n = av$ for some $v \in \Sigma^*$. Let τ' be any Romeo strategy

with $\tau'((\widehat{a\overline{u_n}})^k, a) = w = v_0$ and $\tau'((\widehat{a\overline{u_n}})^k\widehat{a\overline{u_i}}, b) = \tau(\overline{u_i}, b)$ for each $k \in \mathbb{N}_0$, $b \in \Sigma_f$ and $i = 0, \ldots, n-1$. We claim that $\Pi(\sigma_{\mathcal{A}_q}, \tau', a)$ is the infinite sequence

$$
\begin{aligned}
&((\widehat{a\overline{u_n}})^0, av^0), ((\widehat{a\overline{u_n}})^0\widehat{a\overline{u_0}}, v_0v^0), ((\widehat{a\overline{u_n}})^0\widehat{a\overline{u_1}}, v_1v^0), \ldots, ((\widehat{a\overline{u_n}})^0\widehat{a\overline{u_{n-1}}}, v_{n-1}v^0),\\
&((\widehat{a\overline{u_n}})^1, av^1), ((\widehat{a\overline{u_n}})^1\widehat{a\overline{u_0}}, v_0v^1), ((\widehat{a\overline{u_n}})^1\widehat{a\overline{u_1}}, v_1v^1), \ldots, ((\widehat{a\overline{u_n}})^1\widehat{a\overline{u_{n-1}}}, v_{n-1}v^1),\\
&((\widehat{a\overline{u_n}})^2, av^2), \ldots.
\end{aligned}
$$

The initial configuration is correct, since $((\widehat{a\overline{u_n}})^0, av^0) = (\varepsilon, a)$. By induction on $k \in \mathbb{N}_0$ it is clear that $\delta^*(q, (\widehat{a\overline{u_n}})^k) = q$. Therefore,

$$\delta^*(q, (\widehat{a\overline{u_n}})^k a) = \delta(\delta^*(q, (\widehat{a\overline{u_n}})^k), a) = \delta(q, a) = \bot,$$

so $\sigma_{\mathcal{A}_q}$ plays Call in a configuration $((\widehat{a\overline{u_n}})^k, av^k)$. The next configuration is therefore

$$((\widehat{a\overline{u_n}})^k\widehat{a}, \tau'((\widehat{a\overline{u_n}})^k, a)v^k) = ((\widehat{a\overline{u_n}})^k\widehat{a}, wv^k) = ((\widehat{a\overline{u_n}})^k\widehat{a\overline{u_0}}, v_0v^k),$$

which conforms to the sequence above.

Consider now a configuration $K = ((\widehat{a\overline{u_n}})^k\widehat{a\overline{u_i}}, v_iv^k)$ for $k \in \mathbb{N}_0$ and $i \in \{0, \ldots, n-1\}$. We can write $v_i = b_iv_i'$ where $b_i \in \Sigma$ and $v_i' \in \Sigma^*$. The strategy $\sigma_{\mathcal{A}_q}$ plays Call if $\delta^*(q, (\widehat{a\overline{u_n}})^k\widehat{a\overline{u_i}}b_i) = \bot$, i. e. if $\delta^*(\delta(q, \widehat{a}), \overline{u_i}b_i) = \bot$, which is equivalent to $\sigma_{\mathcal{A}_{\delta(q,\widehat{a})}}$ playing Call in the configuration $K_i = (\overline{u_i}, b_iv_i')$. In that case, $\overline{u_{i+1}} = \overline{u_i}\widehat{b_i}$ and $v_{i+1} = \tau(\overline{u_i}, b_i)v_i' = \tau'((\widehat{a\overline{u_n}})^k\widehat{a\overline{u_i}}, b_i)v_i'$. The configuration after K is then $((\widehat{a\overline{u_n}})^k\widehat{a\overline{u_{i+1}}}, v_{i+1}v^k)$, as above. Note that this also conforms to the sequence for $i = n-1$ since $v_n = av$. The other case is that $\sigma_{\mathcal{A}_q}$ plays Read and in K and $\sigma_{\mathcal{A}_{\delta(q,\widehat{a})}}$ plays Read in K_i. This case is analogous. We have shown that if condition (i) from the definition of *Inf* holds, then $\Pi(\sigma_{\mathcal{A}_q}, \tau', a)$ is the infinite sequence from above. This finishes the base case of the structural induction.

The inductive cases are considerably easier. If $r = a \in \Sigma$, $\delta(q, a) = \bot$ and $(\delta(q, \widehat{a}), R_a) \in \mathit{Inf}$, then by the induction hypothesis there exist $w \in L(R_a)$ and a Romeo strategy τ such that $\Pi(\sigma_{\mathcal{A}_{\delta(q,\widehat{a})}}, \tau, w)$ is infinite. Consider a Romeo strategy τ' with $\tau'(\varepsilon, a) = w$ and $\tau'(\widehat{a\overline{u}}, b) = \tau(\overline{u}, b)$ for $\overline{u} \in \overline{\Sigma}^*$ and $b \in \Sigma_f$. The play $\Pi(\sigma_{\mathcal{A}_q}, \tau', a)$ is clearly infinite. In the other inductive cases, the second statement follows similarly from the induction hypothesis.

"$\Longleftarrow$": Let $w_1 \in L(r)$ and a Romeo strategy τ_1 be such that $\Pi(\sigma_{\mathcal{A}_q}, \tau_1, w_1)$ is infinite. The idea is to abuse that there must exist a pair $(q_m, a_m) \in Q \times \Sigma$ such that $\Pi(\sigma_{\mathcal{A}_q}, \tau_1, w_1)$ contains infinitely many configurations with $\mathcal{A}_q$-state q_n and current symbol a_n.

Let $u_1 \in \Sigma^*$ be the longest prefix of w_1 such that $\Pi(\sigma_{\mathcal{A}_q}, \tau_1, u_1)$ is finite. Let $\overline{u_1} \in \overline{\Sigma}^*$ be the final history string of $\Pi(\sigma_{\mathcal{A}_q}, \tau_1, u_1)$ and let $a_1 \in \Sigma$ and $v_1 \in \Sigma^*$ be such that $w_1 = u_1a_1v_1$. This means that $\Pi(\sigma_{\mathcal{A}_q}, \tau_1, w_1)$ contains the configuration $(\overline{u_1}, a_1v_1)$ with $\mathcal{A}_q$-state $q_1 = \delta^*(q, \overline{u_1})$ and, by maximality of u_1, the play $\Pi(\sigma_{\mathcal{A}_{q_1}}, \tau_1[\overline{u_1}], a_1)$ is infinite. Thus, $\delta(q_1, a_1) = \bot$ and, letting $\tau_2 = \tau_1[\overline{u_1}\widehat{a_1}]$ and $w_2 = \tau_1(\overline{u_1}, a_1)$, the play $\Pi(\sigma_{\mathcal{A}_{\delta(q_1,\widehat{a_1})}}, \tau_2, w_2)$ must also be infinite. By repeating the argument we obtain infinite sequences $(w_i), (u_i), (v_i) \subseteq \Sigma^*$, $(a_i) \subseteq \Sigma$, $(\overline{u_i}) \subseteq \overline{\Sigma}^*$, $(q_i) \subseteq Q$ and (τ_i) of Romeo strategies such that

- $w_{i+1} = \tau_i(\overline{u_i}, a_i) = u_{i+1}a_{i+1}v_{i+1}$,

- $\tau_{i+1} = \tau_i[\overline{u_i}\widehat{a_i}]$,
- $\Pi(\sigma_{\mathcal{A}_{\delta(q_i,\widehat{a_i})}}, \tau_{i+1}, u_{i+1})$ is finite with final history string $\overline{u_{i+1}}$,
- $q_{i+1} = \delta^*(q_i, \widehat{a_i}\overline{u_{i+1}})$,
- $\Pi(\sigma_{\mathcal{A}_{\delta(q_i,\widehat{a_i})}}, \tau_{i+1}, w_{i+1})$ is infinite and contains, for each $j > i$, a configuration with $\mathcal{A}_{\delta(q_i,\widehat{a_i})}$-state q_j and current symbol a_j,
- $\delta(q_i, a_i) = \bot$

for each $i \in \mathbb{N}$.

Since Q and Σ are finite but the sequences are infinite, there must exist $m < n$ with $(q_m, a_m) = (q_n, a_n)$. We choose these such that (n,m) is lexicographically minimal and prove the implication by induction on $|r| + \sum_{i=1}^{n-1} |R_{a_i}|$. It is clear that $r \neq \varepsilon$ since a play on ε is never infinite. There are several possible cases.

If $r = a \in \Sigma$, then $w_1 = a = a_1$, $q_1 = q$, $\delta(q,a) = \bot$ and $\Pi(\sigma_{\mathcal{A}_{\delta(q,\widehat{a})}}, \tau_2, w_2)$ is infinite and contains a configuration with $\mathcal{A}_{\delta(q,\widehat{a})}$-state q_n and current symbol a_n. There are two subcases.

- If $m = 1$ then $q_n = q$ and $a_n = a$. By Claim 5.3 this means that $(\delta(q,\widehat{a}), R_a, q, a) \in \mathit{Next}$. Thus, condition (i) of the definition of Inf is satisfied and $(q,r) \in \mathit{Inf}$.
- If $m > 1$ then the induction hypothesis can be applied because the sequences (a_i) and (q_i) for the play $\Pi(\sigma_{\mathcal{A}_{\delta(q,\widehat{a})}}, \tau_2, w_2)$ are $(a_2, a_3, \dots)$ and $(q_2, q_3, \dots)$. Thus, $(\delta(q,\widehat{a}), R_a) \in \mathit{Inf}$ and condition (ii) of the definition yields $(q,r) \in \mathit{Inf}$.

If $r = r_1 r_2$, $r = r_1 + r_2$ or $r = r_1^*$, then $(q,r) \in \mathit{Inf}$ follows similarly from the induction hypothesis. □

This completes the proof of Theorem 5.1. □

Of course it is desirable to have not just a UFA but a DFA that recognizes $L(\sigma_{\mathcal{A}})$. Since UFAs can be transformed into DFAs with an exponential blow-up, the previous theorem implies that such a DFA can be constructed in exponential time. It was shown by Abiteboul et al. [3, Proposition 3.2] that this is also a lower bound on the running time in the worst case, since there exist strategy automata $\mathcal{A}$ such that any DFA for $L(\sigma_{\mathcal{A}})$ must have a number of states exponential in the size of $\mathcal{A}$.

We will use the possibility of constructing a UFA recognizing $L(\sigma_{\mathcal{A}})$ in polynomial time for our complexity results in the next chapter.

6. Complexity of decision problems

We will investigate the complexity of three (types of) problems. The first is to test for a given game, input word and strategy whether the strategy wins on the input word. The second is similar, but the strategy is not part of the input: The question in this case is whether a strategy exists that wins on a given input word in a given game. The third problem is relevant in the context of determining optimum or optimal strategies: Given a game and two strategies, is one strategy better than the other? In the cases where a strategy is part of the input, we have to restrict to automaton strategies, since strategies of type 1 and 3 do not in general have a finite encoding (since there are uncountably many of them).

6.1. Testing if a strategy wins

For a set $\mathcal{G}$ of games and $X \in \{2,4,5\}$ we consider the following decision problem:

IsWinning($\mathcal{G}, X$)
Given: A game $G = (\Sigma, R, T) \in \mathcal{G}$, a word $w \in \Sigma^*$ and a type X automaton $\mathcal{A}$ for G
Question: Is $w \in L(\sigma_{\mathcal{A}})$?

For a target language given by a DFA, the problem is tractable:

6.1 Theorem. *Let $X \in \{2,4,5\}$. The problem* IsWinning($\mathcal{G}, X$) *is* **P***-complete if $\mathcal{G}$ is*

- *the set of games $G = (\Sigma, R, T)$ where T is a DFA or*
- *the set of non-recursive games $G = (\Sigma, R, T)$ where T is a DFA and replacement languages are finite and self-delimiting.*

Proof. The upper bound follows easily from our result of the last chapter: By Theorem 5.1 (and since each type 5 automaton is also a type 4 automaton), a UFA $\mathcal{U}$ recognizing $L(\sigma_{\mathcal{A}})$ can be constructed in polynomial time. Then, the set of states of $\mathcal{U}$ that can be reached at the end of a run on w can be computed in polynomial time by simulating the power set automaton. If all of these states are accepting, then $\sigma_{\mathcal{A}}$ wins on w, otherwise $\sigma_{\mathcal{A}}$ does not win on w.

To prove the lower bound, we give a logarithmic space reduction from **P**-complete emptiness problem for context-free grammars, i. e. the problem of deciding whether the language generated by a given context-free grammar is empty (see [8, Corollary 11] and [6, Theorem 5.6.1]).

Let $C = (V, \Gamma, P, S)$ be a context-free grammar, where V is the set of variables, Γ the set of terminals, $P \subset V \times (V \cup \Gamma)^*$ the set of production rules and $S \in V$ the start variable. We construct a

(a) Target language automaton T (b) Type 5 automaton $\mathcal{A}$

Figure 6.1.: The automata T and $\mathcal{A}$ constructed in the reduction in the proof of Theorem 6.1

non-recursive game $G = (\Sigma, R, T)$ with finite and self-delimiting replacement languages, a word $w \in \Sigma^*$ and a type 5 automaton $\mathcal{A}$ such that $w \in L(\sigma_{\mathcal{A}})$ if and only if the language $L(C)$ generated by C is empty. By definition, the type 5 automaton $\mathcal{A}$ is also a type 4 automaton. Furthermore, it can be transformed into a type 2 automaton by adding a loop for $\widehat{a} \in \Sigma_f$ to each state that has no transition for a. Therefore, it suffices to prove the lower bound for the case $X = 5$.

We will argue first that we can assume without loss of generality that C is non-recursive and that replacement languages (i. e. the sets of replacement words over $V \cup \Gamma$ that can be derived from a variable in one step) are self-delimiting and do not contain ε. The assumption that C is non-recursive will guarantee that each play of the game we construct is finite. The assumption that there are no ε-rules will ensure that the game we construct also has no ε-rules. This is necessary because replacement strings in context-free games have to be non-empty by definition.

The reduction to the emptiness problem for context-free grammars presented by Gurari [6, Theorem 5.6.1] already constructs a non-recursive grammar with self-delimiting replacement languages, so the problem is **P**-hard even with these restrictions. Alternatively, consider Lemma A.3 in the appendix where it is shown that the emptiness problem for non-recursive context-free grammars with self-delimiting replacement languages is **P**-hard.

To avoid ε-rules, any rule of the form $A \to \varepsilon$ can be replaced by $A \to x$ for some terminal x. Clearly this only requires logarithmic space and does not affect whether the language generated by the grammar is empty or not.

Henceforth, we can assume that $C = (V, \Gamma, P, S)$ is already non-recursive, has self-delimiting replacement languages and no ε-rules. We construct $G = (\Sigma, R, T)$, $\mathcal{A}$ and w as follows.

The alphabet of G is $\Sigma = V \cup \Gamma$ and the set of rules is $R = P$. The target language automaton T is displayed in Figure 6.1a. It has a non-accepting initial state q_0 and an accepting state q_1. The transitions in q_0 for a variable $A \in V$ lead to q_1. All other transitions are loops. The type 5 automaton $\mathcal{A}$ is obtained from T by removing the transitions from q_0 to q_1 for those variables that actually appear on the left hand side of a production rule (see Figure 6.1b). The input word is the start variable, i. e. $w = S$.

Clearly, G, $\mathcal{A}$ and w can be constructed using only logarithmic space. It remains to show that $L(C) = \emptyset$ if and only if $w \in L(\sigma_{\mathcal{A}})$. We show that $L(C) \neq \emptyset$ if and only if $w \notin L(\sigma_{\mathcal{A}})$.

If $L(C) \neq \emptyset$, then there exists a word $w' \in \Gamma^*$ that can be derived from S. Note that the strategy $\sigma_{\mathcal{A}}$ plays Call whenever possible until a variable occurs that has no production rule. Consider a play of $\sigma_{\mathcal{A}}$ on S. If Romeo chooses his replacement words according to a leftmost derivation of w', then all occurring symbols are either terminals or variables that have a production rule. Therefore, the T-state q_0 is never left and Juliet loses, i. e. $w = S \notin L(\sigma_{\mathcal{A}})$.

Conversely, if $w \notin L(\sigma_{\mathcal{A}})$ then there exists a play Π of $\sigma_{\mathcal{A}}$ on w that is not winning. Hence Π is losing because infinite plays are impossible in non-recursive games. The T-state of the last configuration of Π must be q_0, hence the final string w' of Π is in Γ^*. Since w' was derived from S by applying rules of P, it holds that $w' \in L(C)$. Thus, $L(C) \neq \emptyset$. □

Theorem 6.1 also implies that the problem to decide whether some automaton strategy wins on a word is tractable under *data complexity*, i. e. if the game G is fixed and not part of the input. This is because for a fixed game, the target language automaton can always be transformed into a DFA beforehand.

As the next theorem shows, the problem (with G as part of the input) becomes **PSPACE**-complete if the target language automaton is given as an NFA. It should be noted that for the case of type 5 automata this scenario is somewhat artificial: In practice one would expect that the type 5 automaton is constructed *from* the minimal DFA for the target language. Therefore, a DFA for the target language is usually known when a type 5 automaton is known. In fact, the **PSPACE**-hardness in the case of type 5 automata is also artificial. It stems only from the hardness of deciding whether the strategy automaton given in the input is really a type 5 automaton, i. e. whether it can be obtained from the minimal DFA for the target language by removing transitions (intuitively because this requires computing the minimal DFA of the target language). In the case of Theorem 6.1 this was not a problem because even if the target language DFA is not minimal, it can be minimized in polynomial time.

If one could trust that the input of IsWinning$(\mathcal{G}, 5)$ were always encoded correctly (i. e. the given strategy automaton is indeed a type 5 automaton) then the target language could be ignored because its essential parts are implicit in the type 5 automaton; then the problem would be **P**-complete by Theorem 6.1.

6.2 Theorem. *Let $X \in \{2,4,5\}$. The problem* IsWinning$(\mathcal{G}, X)$ *is* **PSPACE***-complete if $\mathcal{G}$ is*

- *the set of all context-free games or*
- *the set of non-recursive games with finite and self-delimiting replacement languages.*

Proof. We first prove the upper bound. Let $G = (\Sigma, R, T)$ be a game with target language NFA $T = (Q, \Sigma, \delta, q_0, F)$, let $\mathcal{A}$ be a strategy automaton and $w \in \Sigma^*$. As before we can assume that $\mathcal{A} = (Q_{\mathcal{A}}, \overline{\Sigma}, \delta_{\mathcal{A}}, q_0', F_{\mathcal{A}})$ is a type 2 automaton because type 4 automata can easily be transformed into a type 2 automata.

Since **PSPACE** is closed under complementation and **PSPACE** = **NPSPACE**, we can prove the upper bound by giving an algorithm that uses polynomial space and accepts non-deterministically if $\sigma_{\mathcal{A}}$ does *not* win on w. The idea of this algorithm is to simulate a play of $\sigma_{\mathcal{A}}$ on w and guessing Romeo's replacement words along the way.

There are two challenges that we have to be careful about. First, the only plays of $\sigma_{\mathcal{A}}$ on w that are not winning might be infinite and our simulation has to detect this in finite time. Second, if R is infinite, then Romeo's replacement words can be arbitrarily long. If a word is too long, then it does not fit in the store of polynomial size at once, so the algorithm will guess replacement words

symbol by symbol. But at some point Romeo needs to be stopped, to prevent him from producing a replacement word of infinite length.

We will describe how we can overcome both of these challenges by proving several claims. The issue with infinite plays is taken care of by the following claim:

6.3 Claim. *If a play of $\sigma_{\mathcal{A}}$ on w with depth greater than $|Q_{\mathcal{A}}| \cdot |\Sigma_f|$ exists, then there exists an infinite play of $\sigma_{\mathcal{A}}$ on w.*

Proof. Let Π be a play of $\sigma_{\mathcal{A}}$ on w with depth greater than $|Q_{\mathcal{A}}| \cdot \Sigma_f$. Then there exist $q \in Q_{\mathcal{A}}$ and $a \in \Sigma_f$ such that $\mathcal{T}(\Pi)$ has a path starting at the root that contains (at least) two configurations with $\mathcal{A}$-state q and current symbol a. Let K be the upper of these two configurations. Romeo can force infinite play by repeating – whenever a configuration with $\mathcal{A}$-state q and current symbol a is reached – the moves that he would choose after reaching the configuration K. The details are very similar to the proof of the implication "$\Longrightarrow$" in the proof of Claim 5.4. □

Restricting the length of replacement words by Romeo takes a bit more effort. Let us first introduce some notation.

For $a \in \Sigma_f$ let $(Q_a, \Sigma, \delta_a, q_a^0, F_a)$ be an NFA that recognizes the language $L(R_a)$. Such an NFA can be constructed from R_a in polynomial time by Glushkov's algorithm [5]. For a state $q \in Q_{\mathcal{A}}$ we denote by $\mathcal{A}_q$ the automaton obtained from $\mathcal{A}$ by changing its initial state to q. Abusing notation we denote by $\delta^* \colon \mathcal{P}(Q) \times \Sigma^* \to \mathcal{P}(Q)$ be the extended transition function of the powerset automaton of the NFA T, given by

$$\delta^*(Q', v) = \begin{cases} Q', & \text{if } v = \varepsilon, \\ \bigcup\limits_{q \in \delta^*(Q', u)} \delta(q, a), & \text{if } v = ua \text{ for } u \in \Sigma^* \text{ and } a \in \Sigma. \end{cases}$$

The following claim solves the issue of arbitrarily long replacement word at least for the case of finite plays on input words of length 1.

6.4 Claim. *Let $Q^0 \subseteq Q$ be a set of states of T. Let $q \in Q_{\mathcal{A}}$, a Romeo strategy τ and $a \in \Sigma$ be such that $\Pi(\sigma_{\mathcal{A}_q}, \tau, a)$ is finite with some final history string $\overline{u}$. Then there exists a Romeo strategy τ' with $|\tau'(\overline{v}, b)| \leq |Q_b| \cdot |Q_{\mathcal{A}}| \cdot 2^{|Q|}$ for each $\overline{v} \in \overline{\Sigma}^*$, $b \in \Sigma_f$ such that $\Pi(\sigma_{\mathcal{A}_q}, \tau', a)$ is finite with some final history string $\overline{u}' \in H_a$ for which it holds that $\delta^*(Q^0, \natural\overline{u}) = \delta^*(Q^0, \natural\overline{u}')$.*

Proof. The proof is by induction on the depth of $\Pi(\sigma_{\mathcal{A}_q}, \tau, a)$. For $b \in \Sigma_f$ let $x_b \in L(R_b)$ be a word of length at most $|Q_b|$. Clearly such a word exists.

If the depth of the play $\Pi(\sigma_{\mathcal{A}_q}, \tau, a)$ is 0 then the play consists of a single move by Juliet, which is Read, and we can simply define τ' via $\tau'(\overline{v}, b) = x_b$ for $\overline{v} \in \overline{\Sigma}^*$ and $b \in \Sigma_f$.

If the depth of $\Pi(\sigma_{\mathcal{A}_q}, \tau, a)$ is positive, let $a_1 \dots a_n = \tau(\varepsilon, a) \in L(R_a)$ be the first replacement word that Romeo chooses in this play, where $a_i \in \Sigma$. Let $q_{\mathcal{A}}^0 = \delta_{\mathcal{A}}(q, \widehat{a})$ and for $i = 1 \dots, n$, let

- $\overline{a_i} \in H_{a_i}$ be such that $(\widehat{a}\overline{a_1} \dots \overline{a_i}, a_{i+1} \dots a_n)$ is the first configuration in $\Pi(\sigma_{\mathcal{A}_q}, \tau, a)$ after the initial Call that has remaining string $a_{i+1} \dots a_n$,
- $q_a^i \in \delta_a(q_a^{i-1}, a_i)$ be such that $q_a^n \in F_a$,

- $q_{\mathcal{A}}^i = \delta_{\mathcal{A}}^*(q_{\mathcal{A}}^{i-1}, \overline{a_i})$ and
- $Q^i = \delta^*(Q^{i-1}, \natural\overline{a_i})$.

If $n > |Q_a| \cdot |Q_{\mathcal{A}}| \cdot 2^{|Q|}$, then there exist $0 \leq i < j \leq n$ such that $(q_a^i, q_{\mathcal{A}}^i, Q^i) = (q_a^j, q_{\mathcal{A}}^j, Q^j)$. In this case, $a_1 \dots a_i a_{j+1} \dots a_n \in L(R_a)$ and the Romeo strategy $\widetilde{\tau}$ given by

$$\widetilde{\tau}(\overline{v}, b) = \begin{cases} a_1 \dots a_i a_{j+1} \dots a_n, & \text{if } (\overline{v}, b) = (\varepsilon, a), \\ \tau(\widehat{a}\overline{a_1} \dots \overline{a_j v_2}, b), & \text{if } \overline{v} = \overline{a_1} \dots \overline{a_i v_2} \text{ for some } \overline{v_2} \in \overline{\Sigma}^*, \\ \tau(\overline{v}, b), & \text{otherwise,} \end{cases}$$

also satisfies that $\Pi(\sigma_{\mathcal{A}_q}, \widetilde{\tau}, a)$ is finite and, if $\overline{u}'$ is its final history string, then $\delta^*(Q^0, \natural\overline{u}) = \delta^*(Q^0, \natural\overline{u}')$. This allows us to gradually shorten Romeo's replacement word for the initial a and we can therefore assume that $n \leq |Q_a| \cdot |Q_{\mathcal{A}}| \cdot 2^{|Q|}$.

By the induction hypothesis, there exist Romeo strategies τ_i for $i = 1, \dots, n$ with $|\tau_i(\overline{v}, b)| \leq |Q_b| \cdot |Q_{\mathcal{A}}| \cdot 2^{|Q|}$ for each $\overline{v} \in \overline{\Sigma}^*$ and $b \in \Sigma_f$ such that $\Pi(\sigma_{\mathcal{A}_{q_{\mathcal{A}}^{i-1}}}, \tau_i, a_i)$ is finite with some final history string $\overline{u_i} \in H_{a_i}$ for which it holds that $\delta^*(Q^{i-1}, \natural\overline{u_i}) = Q^i$.

The Romeo strategy τ' given by

$$\tau'(\overline{v}, b) = \begin{cases} a_1 \dots a_n, & \text{if } (\overline{v}, b) = (\varepsilon, a), \\ (\tau_1\langle a_1\rangle\tau_2\langle a_2\rangle \dots \langle a_{n-1}\rangle\tau_n)(\overline{v_2}, b), & \text{if } \overline{v} = \widehat{a}\overline{v_2} \text{ for some } \overline{v_2} \in \overline{\Sigma}^*, \\ x_b, & \text{otherwise,} \end{cases}$$

has all the properties stated in the claim. □

Finally, we will overcome both challenges at once with the following claim.

6.5 Claim. *The following are equivalent:*

- *The strategy $\sigma_{\mathcal{A}}$ does not win on w.*
- *There exists a Romeo strategy τ with $|\tau(\overline{u}, a)| \leq |Q_a| \cdot |Q_{\mathcal{A}}| \cdot 2^{|Q|}$ for each $\overline{u} \in \overline{\Sigma}^*$, $a \in \Sigma_f$ such that $\Pi(\sigma_{\mathcal{A}}, \tau, w)$ is losing or has depth greater than $|Q_{\mathcal{A}}| \cdot |\Sigma_f|$.*

Proof. The implication from the second to the first statement follows from Claim 6.3.

Now suppose the first statement holds. We first consider the case that there exists a losing play of $\sigma_{\mathcal{A}}$ on w. Let $w = a_1 \dots a_n$, where $a_i \in \Sigma$. If we define $\tau_1, \dots, \tau_n$ like in the proof of the previous claim, then the second statement is true for $\tau = \tau_1\langle a_1\rangle\tau_2\langle a_2\rangle \dots \langle a_{n-1}\rangle\tau_n$.

Otherwise, there exists an infinite play of $\sigma_{\mathcal{A}}$ on w. By König's lemma, all infinite plays have infinite depth, thus depth greater than $|Q_{\mathcal{A}}| \cdot |\Sigma_f|$. Suppose the second statement does not hold. Let τ be a Romeo strategy such that $\Pi(\sigma_{\mathcal{A}}, \tau, w)$ is infinite and

$$m = \min\left\{ |\overline{u}| \;\middle|\; |\tau(\overline{u}, a)| > |Q_a| \cdot |Q_{\mathcal{A}}| \cdot 2^{|Q|}, \overline{u} \in \overline{\Sigma}^*, a \in \Sigma_f \right\}$$

is maximal. We can assume that $|\tau(\overline{u}, a)| \leq |Q_a|$ for all combinations of $\overline{u}$ and a that do not occur as the combination of history string and current symbol of a configuration in $\Pi(\sigma_{\mathcal{A}}, \tau, w)$ where Juliet

plays Call. Thus, a configuration $(\overline{u}, av)$ with $\overline{u} \in \overline{\Sigma}^*$, $a \in \Sigma_f$, $v \in \Sigma^*$, $m = |\overline{u}|$, $\sigma_{\mathcal{A}}(\overline{u}, a) = \mathit{Call}$ and $|\tau(\overline{u}, a)| > |Q_a| \cdot |Q_{\mathcal{A}}| \cdot 2^{|Q|}$ occurs in $\Pi(\sigma_{\mathcal{A}}, \tau, w)$. If the play $\Pi(\sigma_{\mathcal{A}}[\overline{u}], \tau[\overline{u}], a)$ is finite, then – using Claim 6.4 – we could adjust τ so as to obtain a contradiction to the maximality of m. Therefore, $\Pi(\sigma_{\mathcal{A}}[\overline{u}], \tau[\overline{u}], a)$ must be infinite. But then we can again obtain a contradiction to the maximality of m: Let $\tau(\overline{u}, a) = a_1 \dots a_n bx$ where and $a_1, \dots, a_n \in \Sigma$, $b \in \Sigma_f$ and $x \in \Sigma^*$ are such that $a_1 \dots a_n$ is the longest prefix of $\tau(\overline{u}, a)$ for which $\Pi(\sigma_{\mathcal{A}}[\overline{u\widehat{a}}], \tau[\overline{u\widehat{a}}], a_1 \dots a_n)$ is finite. Let $q^0_{\mathcal{A}} = \delta_{\mathcal{A}}(q'_0, \overline{u\widehat{a}})$ and for $i = 1 \dots, n$, let

- $\overline{a_i} \in H_{a_i}$ be such that $(\overline{a_1} \dots \overline{a_i}, a_{i+1} \dots a_n)$ is the configuration of $\Pi(\sigma_{\mathcal{A}}[\overline{u\widehat{a}}], \tau[\overline{u\widehat{a}}], a_1 \dots a_n)$ where $a_1 \dots a_i$ is completely processed,
- $q^i_a \in \delta_a(q^{i-1}_a, a_i)$ be such that a state in F_a is reachable from state q^n_a by reading bx,
- $q^i_{\mathcal{A}} = \delta^*_{\mathcal{A}}(q^{i-1}_{\mathcal{A}}, \overline{a_i})$ and

If $n > |Q_a| \cdot |Q_{\mathcal{A}}|$, then there exist $0 \leq i < j \leq n$ with $(q^i_a, q^i_{\mathcal{A}}) = (q^j_a, q^j_{\mathcal{A}})$. Like in the proof of Claim 6.4 it can be argued that the string $a_1 \dots a_n$ could be shortened. Therefore, we can assume without loss of generality that $n \leq |Q_a| \cdot |Q_{\mathcal{A}}|$. We can also assume that $|x| \leq |Q_a| - 1$ because x just needs to be any string for which a state in F_a is reachable from q^n_a by reading bx. But then we would have that

$$|\tau(\overline{u}, a)| = |a_1 \dots a_n bx| \leq |Q_a| \cdot |Q_{\mathcal{A}}| + |Q_a| \leq |Q_a| \cdot |Q_{\mathcal{A}}| \cdot 2^{|Q|},$$

a contradiction. □

Thanks to Claim 6.5 it is now easy to give a polynomial space algorithm that decides the complement of IsWinning$(\mathcal{G}, 2)$ non-deterministically. The algorithm simply tries to prove that the second statement of Claim 6.5 holds by simulating a play of $\sigma_{\mathcal{A}}$ on w. At all times, the $\mathcal{A}$-state and the set of T-states of the configuration up to which the play has been simulated so far are stored. When $\sigma_{\mathcal{A}}$ plays Call at some point when the current symbol is a, Romeo's replacement word $x \in L(R_a)$ is guessed symbol by symbol. To ensure that the guessed word x is in $L(R_a)$, a run of the Glushkov-NFA for $L(R_a)$ on x is guessed at the same time, i. e. a current state $q \in Q_a$ is stored as well. A counter guarantees that the replacement word is not longer than $|Q_a| \cdot |Q_{\mathcal{A}}| \cdot 2^{|Q|}$. If a nested Call happens, q and the counter are pushed on a stack and pulled back once the play has been simulated up to the end of the nested replacement word. Another counter keeps track of the size of the stack. If it exceeds $|Q_{\mathcal{A}}| \cdot |\Sigma_f|$, then the algorithm accepts because there exists a play of $\sigma_{\mathcal{A}}$ on w of depth greater than $|Q_{\mathcal{A}}| \cdot |\Sigma_f|$. If the simulated play is finite and the set of T-states of the last configuration is disjoint from F, then the algorithm also accepts because a losing play has been found. Otherwise the algorithm rejects.

Due to the bounds on the length of replacement words and the size of the stack, the algorithm is guaranteed to terminate. Correctness of the algorithm and existence of a polynomial space bound are straightforward. Thus, the complement of IsWinning$(\mathcal{G}, 2)$ is in **NPSPACE** and therefore IsWinning$(\mathcal{G}, X) \in$ **PSPACE** for $X \in \{2, 4\}$.

We still have to prove that IsWinning$(\mathcal{G}, X)$ is also **PSPACE**-hard. We show the hardness for $X = 4$ and $X = 5$ simultaneously. The case $X = 2$ follows easily from the case $X = 4$. Our reduction

is from the universality problem for NFAs, i. e. the problem of deciding whether a given NFA over the alphabet $\{0,1\}$ recognizes the entire language $\{0,1\}^*$. This problem is well-known to be **PSPACE**-complete (see [9, Lemma 2.3] and [7, Proposition 2.4]).

Let $\mathcal{N}$ be an NFA over $\{0,1\}$. We construct a non-recursive game $G = (\Sigma, R, T)$ with finite and self-delimiting replacement languages, a string $w \in \Sigma^*$ and a type 4 automaton $\mathcal{A}$ such that $w \in L(\sigma_{\mathcal{A}})$ if and only if $L(\mathcal{N}) = \{0,1\}^*$.

If the initial state of $\mathcal{N}$ is non-accepting, then we choose a dummy instance of $(G, w, \mathcal{A})$ such that that $w \notin L(\sigma_{\mathcal{A}})$. Otherwise, let n be the number of states of $\mathcal{N}$. We define the alphabet Σ of G as $\Sigma = \{0, 1, a_0, \ldots, a_n\}$ and the set of rules R via

$$\begin{aligned} a_i &\to a_{i+1}a_{i+1} + 0 + 1 \text{ for } i = 0, \ldots, n-1 \\ a_n &\to 0 + 1 \end{aligned}$$

The target language automaton T is the same automaton as $\mathcal{N}$ except that $a_0, \ldots, a_n$ are added to its alphabet (but there are no transitions for $a_0, \ldots, a_n$). Let $\mathcal{A}$ be the type 4 automaton consisting of a single state with loop transitions for 0 and 1 and no transition for $a_0, \ldots, a_n$. For the reduction for the case $X = 5$ we choose this state as accepting; therefore, $\mathcal{A}$ is a type 5 automaton for G if and only if $L(\mathcal{N}) = \{0,1\}^*$. The input word is $w = a_0$. Obviously, the construction takes only polynomial time, G is non-recursive and replacement languages are finite and self-delimiting.

It remains to show that $w \in L(\sigma_{\mathcal{A}})$ if and only if $L(\mathcal{N}) = \{0,1\}^*$.

If $L(\mathcal{N}) = \{0,1\}^*$, then the initial state of $\mathcal{N}$ is accepting, so $(G, w, \mathcal{A})$ is not the dummy instance. Since every symbol a_i is called in a play of $\sigma_{\mathcal{A}}$, the final string of any such play is in $L(T) = L(\mathcal{N}) = \{0,1\}^*$. Therefore, $w \in L(\sigma_{\mathcal{A}})$.

Conversely, if $w \in L(\sigma_{\mathcal{A}})$, then $(G, w, \mathcal{A})$ is again not the dummy instance, so the initial state of $\mathcal{N}$ is accepting and therefore $\varepsilon \in L(\mathcal{N})$. Romeo can choose his replacement words in a play of $\sigma_{\mathcal{A}}$ on a_0 such that an arbitrary string in $\{0,1\}^{\leq 2^n} \setminus \{\varepsilon\}$ is the final string of the play. So since $a_0 \in L(\sigma_{\mathcal{A}})$ it follows that $\{0,1\}^{\leq 2^n} \subseteq L(T) = L(\mathcal{N})$. Suppose $L(\mathcal{N}) \neq \{0,1\}^*$. Let $u \in \{0,1\}^* \setminus L(\mathcal{N})$ be of minimal length and let $q_0, q_1, \ldots, q_{|u|}$ be the sequence of states of the minimal DFA of $L(\mathcal{N})$ visited in a run on u. These states must be pairwise distinct because the shortest directed path from q_i to a non-accepting state has length $|u| - i$. Therefore, the minimal DFA of $L(\mathcal{N})$ has at least $|u| + 1$ states, and $|u| > 2^n$. But the minimal DFA of $L(\mathcal{N})$ has at most as many states as the powerset automaton of $\mathcal{N}$, i. e. at most 2^n states. Contradiction! This finishes the proof of the lower bound and therefore the proof of the theorem. □

The only reason why we treated the case that the initial state of $\mathcal{N}$ is non-accepting separately is that otherwise we would have had to define R_a as $(0+1)^*$, but replacement languages must not contain the empty string by definition.

Rather than testing whether a *given* strategy wins, we will next study the complexity of deciding whether a winning strategy *exists*.

6.2. Existence of a winning strategy

We define for a set $\mathcal{G}$ of games and $X \in \{1,2,3,4,5\}$ the following decision problem:

EXISTSWINNING($\mathcal{G}, X$)
Given: A game $G = (\Sigma, R, T) \in \mathcal{G}$ and a word $w \in \Sigma^*$
Question: Does a type X strategy σ for G exist such that $w \in L(\sigma)$?

Observe that if type Y is a restriction of type X and the implication $X \Longrightarrow Y$ for winning strategies holds for all games in $\mathcal{G}$, then EXISTSWINNING($\mathcal{G}, X$) = EXISTSWINNING($\mathcal{G}, Y$). For instance, EXISTSWINNING($\mathcal{G}, 3$) = EXISTSWINNING($\mathcal{G}, 4$) if $\mathcal{G}$ is the set of games with a finite target language.

Our first theorem in this section is the counterpart of Theorem 6.1 where we fix $X = 5$ and replace ISWINNING by EXISTSWINNING. Unsurprisingly, the complexity rises from **P**-completeness to **NP**-completeness.

6.6 Theorem. *The problem* EXISTSWINNING($\mathcal{G}, 5$) *is* **NP***-complete if* $\mathcal{G}$ *is*

- *the set of games* $G = (\Sigma, R, T)$ *where* T *is a DFA or*
- *the set of non-recursive games* $G = (\Sigma, R, T)$ *where* T *is a DFA and replacement languages are finite and self-delimiting.*

Proof. The problem EXISTSWINNING($\mathcal{G}, 5$) can be solved non-deterministically in polynomial time as follows. Let $G = (\Sigma, R, T)$ and $w \in \Sigma^*$ be the input, where T is a DFA. First, minimize the DFA T in polynomial time. Second, guess a type 5 automaton $\mathcal{A}$ by removing transitions from the minimal DFA for $L(T)$. Third, call the polynomial time algorithm for ISWINNING($\mathcal{G}, 5$), which exists due to Theorem 6.1, with input G, $\mathcal{A}$ and w. This algorithm shows that EXISTSWINNING($\mathcal{G}, 5$)$\in$ **NP**.

The lower bound, we give a reduction from an arbitrary problem $K \subseteq \{0,1\}^*$ that is in **NP**. The reduction is inspired by (but more involved than) a reduction presented by Gurari [6, Theorem 5.6.1] to prove **P**-hardness of the emptiness problem for context-free grammars (i. e. the problem that we used in the reduction of Theorem 6.1).

Let M be a Turing machine for K that accepts or rejects after at most $p(|x|)$ steps for an input $x\#y$ with $x, y \in \{0,1\}^*$, where p is a polynomial. Given $x \in \{0,1\}^*$, we show how to construct in polynomial time a game $G = (\Sigma, R, T)$ and an input word $w \in \Sigma^*$ such that a type 5 strategy $\mathcal{A}$ for G with $w \in L(\sigma_{\mathcal{A}})$ exists if and only if there exists $y \in \{0,1\}^*$ such that M accepts $x\#y$. Intuitively, Juliet's role is to specify some $y \in \{0,1\}^*$ and Romeo's goal is to prove that M rejects $x\#y$. Juliet (i. e. the strategy $\sigma_{\mathcal{A}}$) wins if and only if Romeo fails to prove his claim.

Let S be the set of states of M, Γ the tape alphabet with $\{0,1,\#,\triangleright\} \subseteq \Gamma$, Δ the transition function and $s_0 \in S$ the initial state of M. The symbol $\triangleright \in \Gamma$ marks the left boundary of the tape. We assume that M has a single accepting state $+ \in S$ and a single rejecting state $- \in S$ and M halts as soon as one of these states is reached. Let $x = x_1 x_2 \ldots x_n$ with $x_i \in \{0,1\}$ and let $t = p(n)$. The initial string of the Turing machine is $\triangleright x\#y$ for some $y \in \{0,1\}^*$ and the head starts at position 0, pointing at $\triangleright$.

We can assume without loss of generality that $t \geq n+2$ and y has length exactly $t-n-1$, i. e. the bits of y occupy the tape positions $n+2$ to t.

The alphabet Σ of G is defined as

$$\Sigma = \{0,1,a,b,C,c,D,d,E\} \cup \{a_j \mid j = n+2,\ldots,t\} \cup \{A_{i,j,W} \mid i,j = 0,\ldots,t,\ W \in \Gamma \cup (\Gamma \times S)\}.$$

The symbols E and $A_{i,j,W}$ represent statements about a run of M on $x\#y$ for some $y \in \{0,1\}^{t-n-1}$. The symbol E represents the statement that the run is rejecting. This is what Romeo will try to prove via his choice of replacement words. A symbol $A_{i,j,z}$ with $z \in \Gamma$ represents the statement that after i steps, the jth symbol on the tape is z and the head does *not* point at position j. A symbol $A_{i,j,(z,q)}$ with $(z,q) \in \Gamma \times S$ represents that after i steps, the jth symbol is z, the current state is q and the head points at position j. The remaining symbols will be used to check the validity of the initial configuration of M and to force Juliet to choose a type 5 strategy that represents $y \in \{0,1\}^{t-n-1}$ in a well-defined way (as we will see later).

The set R of rules of G is given by

$0 \to b$

$1 \to b$

$C \to c1$

$D \to d1d$

$E \to A_{i,j,(z,-)}$ for $i,j = 0,\ldots,t$, $z \in \Gamma$ and the rejecting state "$-$" of M

$A_{0,0,(\triangleright,s_0)} \to a$

$A_{0,j,x_j} \to a$ for $j = 1,\ldots,n$

$A_{0,n+1,\#} \to a$

$A_{0,j,0} \to a_j 0$ for $j = n+2,\ldots,t$

$A_{0,j,1} \to a_j 1$ for $j = n+2,\ldots,t$

$A_{i+1,j,W} \to A_{i,j-1,X}A_{i,j,Y}A_{i,j+1,Z}$ for $i = 0,\ldots,t-1,\ j = 0,\ldots,t$ and $W,X,Y,Z \in \Gamma \cup (\Gamma \times S)$ if – according to the transition function Δ – the statements represented by $A_{i,j-1,X}$, $A_{i,j,Y}$ and $A_{i,j+1,Z}$ imply the statement represented by $A_{i+1,j,W}$. For the boundaries, $A_{i,-1,X}$ and $A_{i,t+1,Z}$ shall denote the empty string.

The rules of the form $A_{i+1,j,W} \to A_{i,j-1,X}A_{i,j,Y}A_{i,j+1,Z}$ are defined only based on the transition function Δ and regardless of whether there exists a run of M on $x\#y$ for some y for which the represented statements can be true. We do not even forbid that the statements of $A_{i,j-1,X}$, $A_{i,j,Y}$ and $A_{i,j+1,Z}$ contradict each other (e. g. by saying that the head is at two positions at once), but the target language will be such that Romeo is better off not to choose a self-contradictory replacement string. Note that the replacement languages are finite and self-delimiting and G is non-recursive.

The target language automaton $T = (Q,\Sigma,\delta,q_0,F)$ of G contains components T_j for $j = n+2,\ldots,t$, containing states j, f_j, ℓ_j and other states and transitions as specified by Figure 6.2. The

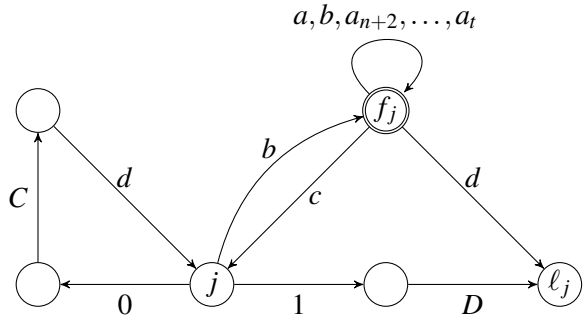


Figure 6.2.: Component T_j of the automaton T in the proof of Theorem 6.6

component T_j corresponds to the jth symbol of the initial tape content, i. e. the $(j-n-1)$st symbol of y. Additional to the states in the components T_j there exists a non-accepting error state q_e in T. The set of accepting states is $F = \{f_{n+2}, \ldots, f_t\}$. The construction of T is completed by adding the following transitions:

- (ℓ_{j-1}, a_j, j) for $j = n+3, \ldots, t$ (connecting T_{j-1} and T_j)
- (q, a_j, j) for each $q \in Q \setminus (F \cup \{q_e, \ell_{n+2}, \ell_{n+3}, \ldots, \ell_{t-1}\})$ and $j = n+2, \ldots, t$
- $(q, A_{i,j,W}, f_t)$ for each $q \in Q \setminus \{q_e\}$ and i, j, W such that $A_{i,j,W} \in \Sigma \setminus \Sigma_f$
- (q, a, q) for each $q \in Q \setminus F$
- All remaining transitions lead to the error state q_e.

The initial state of T is the state $n+2$ of T_{n+2}.

The automaton T is the minimal automaton for $L(T)$ because all states are reachable from the initial state $n+2$ and the states are pairwise non-equivalent:

- The error state is not equivalent to any other state because it is the only state from which no accepting state is reachable.
- For any two states p and q with distinct roles within their components (p and q may belong to the same component T_j or to two distinct components T_j and T_k with $j \neq k$) it is easy to find some symbol $e \in \Sigma$ such that $\delta(p, e) = q_e$ and $\delta(q, e) \neq q_e$ or vice versa.
- Two states q_j and q_k with the same role in their respective components T_j and T_k with $j < k$ are not equivalent: Let $u \in \{1, c, C, d, D, a_{k+1}, \ldots, a_t\}^*$ be of minimal length such that $\delta^*(q_k, u) = \ell_t$. Then $\delta^*(q_k, ua_{n+2}) = n+2$ and $\delta^*(q_j, ua_{n+2}) = q_e$.

Thus, the type 5 automata for G are exactly the automata that can be obtained from T by removing transitions for symbols from Σ_f.

The input word is defined as $w = 0CDa_{n+3}0CDa_{n+4}0CD\ldots a_t0CDE$. Since t is polynomial in n, the game G and the word w can be constructed in time polynomial in n. It remains to show the following equivalence:

$$\begin{aligned} &\text{There exists a type 5 automaton } \mathcal{A} \text{ for } G \text{ such that } w \in L(\sigma_{\mathcal{A}}). \\ \iff &\text{There exists } y \in \{0,1\}^{t-n-1} \text{ such that } M \text{ accepts } x\#y. \end{aligned}$$

"$\Longrightarrow$": Let $\mathcal{A}$ be a type 5 automaton for G with $w \in L(\sigma_{\mathcal{A}})$. We will show for each $j = n+2,\ldots,t$ that either the 0-transition of T_j or the 1-transition of T_j exists in $\mathcal{A}$ (but not both). Then we will choose the bits of y according to the existent transition and show that M accepts $x\#y$.

Consider a play of $\sigma_{\mathcal{A}}$ on w. We make a case-by-case analysis depending on whether $\mathcal{A}$ has the 0-transition leaving its initial state $n+2$ or not.

If $\mathcal{A}$ has the 0-transition leaving $n+2$, then Juliet reads the first symbol 0 of w. Juliet also reads the subsequent C because replacing it by $c1$ would mean that the error state q_e is reached upon reading c and the play of $\sigma_{\mathcal{A}}$ on w would be losing. The transition for D to q_e cannot exist – for the same reason. So D will be called and replaced by $d1d$. Juliet can only read d, since it is not a function symbol, leading back to state $n+2$. Next, she calls 1 because otherwise the subsequent d would lead to the error state. So 1 gets replaced by b, Juliet reads b and d and the $\mathcal{A}$-state after the prefix $0CD$ of w is completely processed is ℓ_{n+2}. In this case, the 0-transition of T_{n+2} exists in $\mathcal{A}$ and the 1-transition of T_{n+2} does not exist in $\mathcal{A}$.

If $\mathcal{A}$ does not have the 0-transition leaving $n+2$, then the initial 0 is called and replaced by b, which will be read, and then C will be called and replaced by $c1$. After reading c and coming back to state $n+2$, the remaining string starts with $1D$. If Juliet calls 1 (getting b as replacement), then she would have to call D as well (which gets replaced by $d1d$), but then there would be no way to prevent reaching the error state because both the 1- and the b-transition of ℓ_{n+2} lead to the error state. So Juliet reads 1 and then she has to read D in order to preserve her chance to win, reaching ℓ_{n+2} like in the other case. In this case, the 1-transition of T_{n+2} exists in $\mathcal{A}$ whereas the 0-transition is missing. In both cases, the $\mathcal{A}$-state when $0CD$ is completely processed is ℓ_{n+2}.

The next symbol of the input string is a_{n+3}, which will lead to state $n+3$ of T_{n+3}, and the next symbols of w are again $0CD$. By repeating the same arguments for the other components T_j it follows that for each $j = n+2,\ldots,t$, either the 0-transition or the 1-transition of T_j exists in $\mathcal{A}$. We define $y_j = 0$ if $\mathcal{A}$ has the 0-transition of T_j and $y_j = 1$ if it has the 1-transition of T_j. We want to show that M accepts $x\#y$.

Suppose M rejects $x\#y$. Since $w \in L(\sigma_{\mathcal{A}})$ by the premise, the $\mathcal{A}$-state (which is the same as the T-state) has to be accepting at the end of the play. However, we will show that Romeo can choose replacement words such that the invariant holds that every configuration after the prefix $0CDa_{m+3}0CDa_{m+4}0CD\ldots a_t0CD$ of w is completely processed has a non-accepting $\mathcal{A}$-state and a remaining suffix of the form uv where v only consists of symbols representing true statements about the run of M on $x\#y$ and u satisfies one of the following properties:

- $u = \varepsilon$,

- $u = a$,
- $u = a_j k$ for $j \in \{n+2, \ldots, t\}$ and $k \in \{0,1\}$ and Juliet's last move was to call $A_{0,j,k}$, or
- $u = k$ for $k \in \{0,1\}$ and Juliet's last two moves were to call $A_{0,j,k}$ and read a_j for some $j \in \{m+2, \ldots, t\}$.

The invariant holds initially after the prefix $0CDa_{m+3}0CDa_{m+4}0CD \ldots a_t 0CD$ of w is completely processed because the $\mathcal{A}$-state is ℓ_t and the remaining string is E, which represents the true statement (by assumption) that the run of M on $x\#y$ is rejecting.

Suppose the statement holds at some point when the remaining string uv is non-empty (i. e. before the end of the play).

If $u = \varepsilon$ then the first symbol H of v represents a true statement about the run of M on $x\#y$ on u and is therefore a non-terminal. Thus, its transition in T leads to the error state, so this transition cannot exist in $\mathcal{A}$ and Juliet calls H. In all possible cases for H, Romeo can choose a replacement so that the invariant holds again.

If $u = a$, Juliet will read a and loop in the current state. The invariant now holds with $u = \varepsilon$.

If $u = a_j k$ with $k \in \{0,1\}$ and Juliet's last move was to call $A_{0,j,k}$, then Juliet will read a_j and the invariant holds again.

If $u = k \in \{0,1\}$ and Juliet's last two moves were to call $A_{0,j,k}$ and read a_j for some $j \in \{m+2, \ldots, t\}$, then the $\mathcal{A}$-state is j. Since the invariant also held before Juliet's last two moves, before, $A_{0,j,k}$ represents a true statement about the run of M on $x\#y$, i. e. $y_j = k$. By definition of y_j this means that the k-transition in T_j exists in $\mathcal{A}$. Thus, Juliet reads k and the invariant holds again, completing the proof of the invariant.

The invariant implies that Romeo can choose the replacement words so that also the final T-state is non-accepting. But we assumed that $\sigma_{\mathcal{A}}$ wins on w, a contradiction.

"$\Longleftarrow$": Let $y = y_{n+2} \ldots y_t$ with $y_i \in \{0,1\}$ be such that M accepts $x\#y$. We define $\mathcal{A}$ as the automaton obtained from T by removing

- for $j = n+2, \ldots, t$ and $k \in \{0,1\}$ the k-transition from T_j if and only if $k \neq y_j$,
- all transitions for function symbols leaving an accepting state and
- all transitions for function symbols in $\Sigma_f \setminus \{0,1,C,D\}$ leaving a non-accepting state.

Suppose $w \notin L(\sigma_{\mathcal{A}})$, then there exists a Romeo strategy τ such that $\Pi(\sigma_{\mathcal{A}}, \tau, w)$ is losing (it cannot be infinite since G is non-recursive). We will obtain a contradiction by showing that M rejects $x\#y$.

Let $K = (\overline{u}, E)$ be the configuration of $\Pi(\sigma_{\mathcal{A}}, \tau, w)$ when $0CDa_{m+3}0CDa_{m+4}0CD \ldots a_t 0CD$ is completely processed. It is straightforward to check (like above) that the T-state of K is ℓ_t. Observe that the T-state of all future configurations must be non-accepting because the only transitions leading from an accepting to a non-accepting state that have not been removed in $\mathcal{A}$ are the transitions for c and d, but these symbols cannot be derived from E. Also, the

T-state of all future configurations cannot be in $\{\ell_{n+2},\ldots,\ell_{t-1}\}$ because the only transitions leading to these states are for D and d, which cannot be derived from E.

Let $K' = (\overline{uv}, Hv)$ with $\overline{v} \in \overline{\Sigma}^*$, $H \in \Sigma$ and $v \in \Sigma^*$ be a configuration of $\Pi(\sigma_\mathcal{A}, \tau, w)$ occurring not before K and with a T-state different from q_e. We prove that H is either a symbol representing a true statement about the run of M on $x\#y$ or $H \in \{0,1,a,a_{n+2},\ldots,a_t\}$. This suffices to finish the proof, because the T-state of K is ℓ_t and E itself represents the statement that M rejects $x\#y$, which would be the desired contradiction. The proof is by induction on the depth of $\Pi(\sigma_\mathcal{A}[\overline{uv}], \tau[\overline{uv}], H)$, i. e. the depth of the subtree of $\mathcal{T}(\Pi(\sigma_\mathcal{A}, \tau, w))$ rooted at K'.

The base of induction is that the symbol H is read. The only transitions for function symbols that exist in $\mathcal{A}$ are transitions for 0, 1, C and D. So any symbol that is read must be 0, 1, C, D or in $\Sigma \setminus \Sigma_f$. But H cannot be C, D, c or d because these symbols cannot be derived from E. Further, H cannot $A_{i,j,W} \in \Sigma \setminus \Sigma_f$ because these transitions would lead to the accepting state f_t, since the $\mathcal{A}$-state of K' is different from q_e by assumption. The only remaining possibility for H to be a symbol in $\Sigma \setminus \Sigma_f$ but not in $\{a, a_{n+2},\ldots,a_t\}$ would be that $H = b$. However, if $H = b$ then the symbol that was read before was a_j for some j. Since the T-state of K' is not in $F \cup \{q_e, \ell_{n+2},\ldots,\ell_{t-1}\}$, These symbols lead to the state j, but then reading b would lead to the accepting state f_j. So H cannot be b and it follows that $H \in \{0,1,a,a_{n+2},\ldots,a_t\}$.

For the induction step, suppose H is called. By the induction hypothesis, the first symbol of the replacement string for H is either a symbol representing a true statement about the run of M on $x\#y$ or in $\{0,1,a,a_{n+2},\ldots,a_t\}$. Therefore, H can only be E or of the form $A_{i,j,W}$.

If $H = E$, then $A_{i,j,(z,-)}$ represents a true statement for some i, j and z (by the induction hypothesis), and this statement implies that the run of M is rejecting, so E also represents a true statement.

If $H = A_{0,j,W}$ with $j \neq n+1$, then H represents a true statement about the initial configuration of M given the input $x\#y$ because otherwise $A_{0,j,W}$ could not have been called.

If $H = A_{0,j,k}$ for $k \in \{0,1\}$ and $j \in \{m+2,\ldots,t\}$, then Romeo replaces it by $a_j k$. Subsequently, $\mathcal{A}$ will enter the state j upon reading a_j. The k-edge leaving j must exist because otherwise k would be replaced by b and the T-state after reading b would be the accepting state f_j. Thus, $k = y_j$ by construction of $\mathcal{A}$. Hence, $H = A_{0,j,k}$ represents a true statement.

Finally, if $H = A_{i+1,j,W}$, then Romeo chooses some replacement string $A_{i,j-1,X}A_{i,j,Y}A_{i,j+1,Z}$. By the induction assumption, $A_{i,j-1,X}$ represents a true statement. The configurations reached when the current symbol is $A_{i-1,j,Y}$ and $A_{i-1,j+1,Z}$ respectively will also have a T-state different from q_e because all symbols read until then are a or a symbol a_j followed by 0 or 1 – none of which are possibilities to reach q_e. Therefore, the induction hypothesis can be applied to those symbols as well, meaning that the statements represented by $A_{i,j-1,X}$, $A_{i,j,Y}$ and $A_{i,j+1,Z}$ are all true. By definition of R, this implies that the statement represented by $A_{i+1,j,W}$ is also true. This finishes the proof by induction and it follows that E represents a true statement about the run of M on $x\#y$. Contradiction!

Since we have shown that EXISTSWINNING($\mathcal{G}$,5) is **NP**-hard and in **NP**, the theorem follows. □

The situation is more difficult if the target language is specified by an NFA. We have the following lower and upper bounds.

6.7 Proposition. *The problem* EXISTSWINNING($\mathcal{G}$,5) *is* **PSPACE***-hard and in* **NEXPTIME** *if* $\mathcal{G}$ *is*

- *the set of all context-free games or*
- *the set of non-recursive games with finite and self-delimiting replacement languages.*

Proof. The problem can then be decided by applying the **NP**-algorithm from Theorem 6.6 to the exponentially large input obtained by transforming the target language automaton into a DFA. This shows that EXISTSWINNING($\mathcal{G}$,5)$\in$ **NEXPTIME**.

The lower bound follows from the same reduction that was used in Theorem 6.2 except that $\mathcal{A}$ is not constructed. For the game constructed in the proof, a strategy that ever plays Read on some symbol a_i in a play on a_0 is bound to lose. Therefore a type 5 strategy that wins on a_0 exists if and only the strategy constructed in the proof of Theorem 6.2 wins on a_0. Hence, the reduction property still holds. □

It is not clear whether one of these bounds is tight. Note that if it is really possible to decide the problem using polynomial space, then this would mean that it can be solved without ever storing the type 5 automaton completely; but whenever the automaton is in some state q its behavior should be the same; if the algorithm consists of simulating a play, then this would need to be achieved without storing in the memory what the behavior in state q is. However, finding a stronger lower bound also seems challenging. The reduction from Theorem 6.6 cannot be naïvely adjusted to Turing machines that do not run in polynomial time because then the set of rules would already be super-polynomially large. Another challenge in proving a stronger lower bound is that the reduction would have to construct an NFA, but then the argumentation would be about the existence of strategies obtained from the corresponding exponentially larger minimal DFA. We did not have this difficulty in the proof of Theorem 6.6 because the $\mathcal{A}$-state was always the same as the T-state.

For $X \in \{1,2,3,4\}$, it is not even known whether EXISTSWINNING($\mathcal{G}$,X) is decidable if $\mathcal{G}$ is the set of all context-free games. The problem here is that type 2 and 4 automata can be arbitrarily large, so we cannot simply guess one as easily as in the case of type 5 automata, of which there exist only finitely many. However, there are some interesting cases, where we can bound the necessary number of strategy automata. The next theorem states upper bounds on the complexity for several such cases.

6.8 Theorem. *For* $X \in \{1,2\}$*, the problem* EXISTSWINNING($\mathcal{G}$,X) *is in*

- 3**NEXPTIME** *if* $\mathcal{G}$ *is the set of games with self-delimiting replacement languages,*
- 2**NEXPTIME** *if* $\mathcal{G}$ *is the set of games with self-delimiting replacement languages where the target language automaton is a DFA.*

Further, for $X \in \{1,2,3,4\}$*, the problem* EXISTSWINNING($\mathcal{G}$,X) *is in*

- 2**NEXPTIME** *if $\mathcal{G}$ is the set of non-recursive games with finitely many rules,*
- **NEXPTIME** *if $\mathcal{G}$ is the set of non-recursive games with a finite target language*

Proof. Let $G = (\Sigma, R, T) \in \mathcal{G}$ be a game and $w \in \Sigma^*$ a word. The approach is the same in all cases: We show that if a strategy exists that wins on w, then there exists a strategy automaton $\mathcal{A}$ of k-fold exponential size such that $w \in L(\sigma_{\mathcal{A}})$. A k**NEXPTIME**-algorithm then works as follows:

1. Guess $\mathcal{A}$ in k-fold exponential time.
2. Construct a DFA T' recognizing $L(T)$ in exponential time.
3. Test whether $w \in L(\sigma_{\mathcal{A}})$ in time polynomial in the size of (Σ, R, T'), w and $\mathcal{A}$, i. e. in time k-fold exponential in the size of G and w.

The time bound on the 3rd step follows from Theorem 6.1. For the rest of this proof, we only need to show that if a type X strategy exists that wins on w, then there exists a strategy automaton $\mathcal{A}$ as stated above – with $\mathcal{A}$ being of type 2 if $X \in \{1,2\}$ and of type 4 if $X \in \{3,4\}$. Note that the converse of the last statement is trivial, since each type 2 strategy is also a type 1 strategy and each type 4 strategy is also a type 3 strategy.

For the two cases where all games in $\mathcal{G}$ have self-delimiting replacement languages, the existence of such $\mathcal{A}$ of 3-fold and 2-fold exponential size respectively is immediate from Theorem 4.20.

We consider now the cases that G is non-recursive. Then any play tree of a play on w has depth at most $|\Sigma_f| + 1$. The root has $|w|$ children.

If R is finite, then the number of children of a non-root node in a play tree is bounded by the maximal length of a replacement word; this is bounded by $\max_{a \in \Sigma_f} |R_a|$. Thus, the number of nodes of a play tree in the case that R is finite is bounded by

$$B = |w| \cdot \max_{a \in \Sigma_f} |R_a|^{|\Sigma_f|+1} + 1.$$

The set of states of the type 2 and 4 automata in the proof of Theorem 4.19 is a subset of $\overline{\Sigma}^{\leq B}$. Clearly this is 2-fold exponential in the size of the input (with the bottleneck being the maximal recursion depth $|\Sigma_f|$).

If $L(T)$ is finite, then the length of a word in $L(T)$ is less than $|Q|$, i. e. the number of states of T. Therefore, a strategy σ that wins on w plays less than $|Q|$ Reads in a play on w. Thus, the play tree of a play of σ on w has less than $|Q|$ leaves. So altogether, such a play tree has at most

$$B = (|\Sigma_f| + 1) \cdot |Q|$$

nodes. Therefore, the size of the type 2 and 4 automata in the proof of Theorem 4.19 is bounded exponentially in the size of G (independently of w). □

It should be noted that the *data complexity* of these problems is much lower, i. e. the complexity if the game is fixed and the input consists only of w. In the case of self-delimiting replacement languages, the upper bound drops to **NP** under data complexity, and for non-recursive games with

finitely many rules it drops to **NEXPTIME**. This can be seen easily by analyzing the last proof. For games with a finite target language the problem can even be solved in constant time because the set of words for which a winning strategy exists is finite.

One could argue that the problems studied in this and the last section violate the idea of one-pass strategies, since they require an input string as part of their input. In the one-pass setting, Juliet has to choose a one-pass strategy without knowledge of the input word. In the next section, we will study a problem that requires as input only a game and two strategy automata.

6.3. Comparing two strategies

Perhaps even more interesting than the complexity of determining whether a winning strategy for a single word exists is the complexity of determining whether an optimal/optimum strategy exists or whether a given strategy is optimal/optimum. For this, it is helpful to determine which of two strategies is better. For a set $\mathcal{G}$ of context-free games and $X \in \{2,4,5\}$, we define the comparison problem as follows.

COMPARE$(\mathcal{G},X)$
Given: A game $G \in \mathcal{G}$ and type X automata $\mathcal{A}_1$ and $\mathcal{A}_2$ for G
Question: Is $L(\sigma_{\mathcal{A}_1}) \subseteq L(\sigma_{\mathcal{A}_2})$?

The main theorem of this section states that the comparison problem is **PSPACE**-complete for games with a target language DFA – even if restrictions are imposed that are a lot stronger than non-recursiveness and finite, self-delimiting replacement languages.

6.9 Theorem. *For $X \in \{2,4,5\}$, the problem* COMPARE$(\mathcal{G},X)$ *is* **PSPACE***-complete if $\mathcal{G}$ is*

- *the set of games $G = (\Sigma,R,T)$ where T is a DFA or*
- *the set of games $G = (\Sigma,R,T)$ where T is a DFA, all replacement words have length* 1 *and G is non-recursive such that all plays have depth at most* 2

Proof. We first show the upper bound. Let $G \in \mathcal{G}$ be a game and let $\mathcal{A}_1$ and $\mathcal{A}_2$ be type X automata for G. By Theorem 5.1, UFAs $\mathcal{U}_1$ and $\mathcal{U}_2$ recognizing $L(\sigma_{\mathcal{A}_1})$ and $L(\sigma_{\mathcal{A}_2})$ respectively can be constructed in polynomial time. We need an algorithm that tests whether $L(\mathcal{U}_1) \subseteq L(\mathcal{U}_2)$ using polynomial space. If $\mathcal{N}_1$ and $\mathcal{N}_2$ are NFAs obtained from $\mathcal{U}_1$ and $\mathcal{U}_2$ by swapping accepting and non-accepting states, then a word is accepted by $\mathcal{U}_i$ (under UFA semantics) if and only if it is *not* accepted by $\mathcal{N}_i$ (under NFA semantics). Therefore, $L(\mathcal{U}_1) \subseteq L(\mathcal{U}_2)$ is equivalent to $L(\mathcal{N}_2) \subseteq L(\mathcal{N}_1)$. This can be tested using polynomial space, since the containment problem for NFAs is **PSPACE**-complete [7]. Therefore, COMPARE$(\mathcal{G},X) \in$ **PSPACE**.

We prove the lower bound by a reduction from the universality problem for NFAs (given an NFA $\mathcal{N}$ over the alphabet $\{0,1\}$, does $L(\mathcal{N}) = \{0,1\}^*$?). As mentioned in the proof of Theorem 6.2, this problem is **PSPACE**-complete [9, 7]. Like in earlier proofs it suffices to show the hardness for the case $X = 5$.

Let $\mathcal{N} = (Q_\mathcal{N}, \{0,1\}, \delta_\mathcal{N}, q_0, F_\mathcal{N})$ be an NFA. We construct a game $G = (\Sigma, R, T)$ and type 5 automata $\mathcal{A}_1$, $\mathcal{A}_2$ for G such that $L(\mathcal{N}) = \{0,1\}^*$ if and only if $L(\sigma_{\mathcal{A}_1}) \subseteq L(\sigma_{\mathcal{A}_2})$. If $q_0 \notin F_\mathcal{N}$, then $\varepsilon \notin L(\mathcal{N})$ and $(G, \mathcal{A}_1, \mathcal{A}_2)$ can be chosen as a dummy No-instance. We therefore assume in the following that $q_0 \in F_\mathcal{N}$. We also assume without loss of generality that all states of $\mathcal{N}$ are reachable from the initial state q_0 (otherwise, unreachable states can be removed).

We construct $G = (\Sigma, R, T)$ with $\{0,1\} \subset \Sigma$ and automata $\mathcal{A}_1$ and $\mathcal{A}_2$ such that

(i) $L(\mathcal{N}) = \{0,1\}^* \setminus L(\sigma_{\mathcal{A}_1})$, i. e. $\mathcal{N}$ recognizes the language of words in $\{0,1\}^*$ on which $\sigma_{\mathcal{A}_1}$ does not win.

(ii) $L(\sigma_{\mathcal{A}_2}) = \Sigma^* \setminus \{0,1\}^*$, i. e. $\sigma_{\mathcal{A}_2}$ loses on exactly the words in $\{0,1\}^*$.

Note that this implies the reduction property: It holds that $L(\mathcal{N}) = \{0,1\}^*$ if and only if $L(\sigma_{\mathcal{A}_1}) \subseteq \Sigma^* \setminus \{0,1\} = L(\sigma_{\mathcal{A}_2})$. So it suffices to show that we can construct G, $\mathcal{A}_1$ and $\mathcal{A}_2$ with (i) and (ii) in polynomial time. An example of the subsequently described construction is shown in Figure 6.3.

For $k \in \{0,1\}$ let

$$n_k = \max_{q \in Q_\mathcal{N}} |\delta_\mathcal{N}(q,k)|$$

be the maximum number of k-transitions leaving some state of $\mathcal{N}$. We define the alphabet of G as

$$\Sigma = \{0,1\} \cup \{k_i \mid k \in \{0,1\}, i = 1, \ldots, n_k\} \cup \{\$_q \mid q \in Q_\mathcal{N}\}.$$

The set R of rules of G is given by

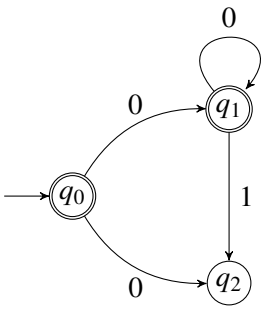
$$\begin{aligned}
k &\to k_1 + \cdots + k_{n_k} && \text{for } k \in \{0,1\} \\
k_i &\to \$_{q_0} && \text{for } k \in \{0,1\} \text{ and } i = 1, \ldots, n_k \\
\$_q &\to \$_{q_0} && \text{for } q \in Q_\mathcal{N} \setminus \{q_0\}.
\end{aligned}$$

Note that G is non-recursive with these rules and all plays have depth at most 2.

The target language automaton of G is $T = (Q, \Sigma, \delta, q_0, F)$, where $Q = Q_\mathcal{N} \dot\cup \{q_f\}$ for some new state $q_f \notin Q_\mathcal{N}$, the set $F = \{q_f\} \dot\cup Q_\mathcal{N} \setminus F_\mathcal{N}$ of accepting states consists besides q_f of the non-accepting states of $\mathcal{N}$ and the transition function δ is given as follows:

- For $q \in Q_\mathcal{N}$ and $k \in \{0,1\}$, the transitions for the symbols $k_1, \ldots, k_{|\delta_\mathcal{N}(q,k)|}$ leaving q are distributed so that each of them reaches a distinct state of $\delta_\mathcal{N}(q,k)$,
- $\delta(q,k) = q$ for $q \in Q_\mathcal{N}$ and $k \in \{0,1\}$,
- $\delta(q, \$_q) = q_0$ for $q \in Q_\mathcal{U} \setminus \{q_0\}$,
- all remaining transitions lead to the state q_f.

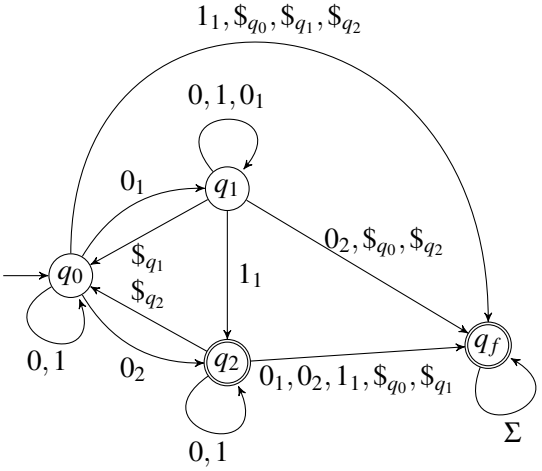
The automaton T is in fact the minimal DFA for $L(T)$ because its states are reachable (by assumption on $\mathcal{N}$) and pairwise non-equivalent: The two states q_0 and q_f are non-equivalent because q_f is accepting and q_0 is non-accepting (since q_0 is accepting in $\mathcal{N}$ by assumption). Each remaining state $q \in Q_\mathcal{U} \setminus \{q_0\}$ is not equivalent to any other state of T because it is the only state from which reading $\$_q$ leads to a non-accepting state (namely q_0).



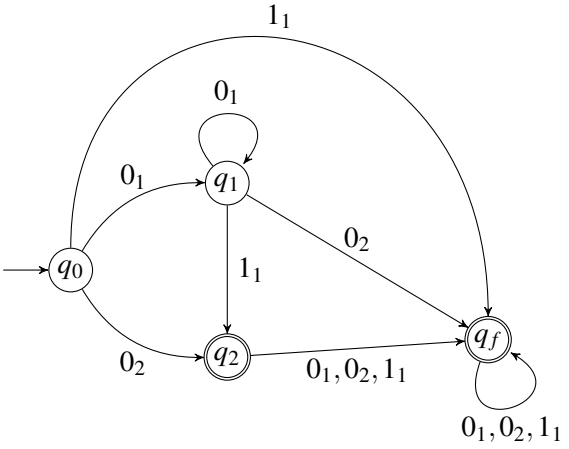
(a) NFA $\mathcal{N}$

$$
\begin{aligned}
0 &\to 0_1 + 0_2 \\
1 &\to 1_1 \\
0_1 &\to \$_{q_0} \\
0_2 &\to \$_{q_0} \\
1_1 &\to \$_{q_0} \\
\$_{q_1} &\to \$_{q_0} \\
\$_{q_2} &\to \$_{q_0}
\end{aligned}
$$

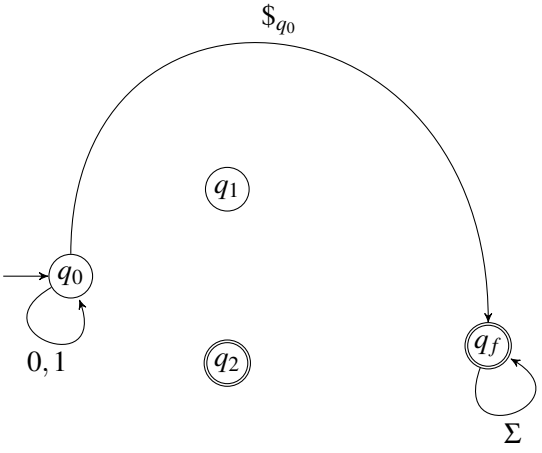
(b) Set of rules R



(c) Target language DFA T



(d) Type 5 automaton $\mathcal{A}_1$



(e) Type 5 automaton $\mathcal{A}_2$

Figure 6.3.: Example of an NFA and the rules, target language DFA, and strategy automata constructed from it in the proof of Theorem 6.9

So, since T is minimal, we need to define the type 5 automata $\mathcal{A}_1$ and $\mathcal{A}_2$ like T except that some edges are missing. The idea to achieve property (i) is that Romeo's choice of replacement words in a play of $\sigma_{\mathcal{A}_1}$ on an input word $w \in \{0,1\}^*$ shall correspond to the choice of a run of $\mathcal{N}$ on w; Romeo shall have a chance to win on w if and only if there exists an accepting run of $\mathcal{N}$ on w.

To this end, we define $\mathcal{A}_1$ as the automaton obtained from T by keeping only the k_i-edges (for $k \in \{0,1\}$ and $i = 1,\ldots,n_k$). The automaton $\mathcal{A}_2$ is obtained from T by keeping only the transitions for 0, 1 and $\$_{q_0}$ leaving q_0 and the loop transitions of q_f.

The construction of $G = (\Sigma, R, T)$, $\mathcal{A}_1$ and $\mathcal{A}_2$ clearly requires only polynomial time. It only remains to show that the properties (i) and (ii) hold.

Proof of (i): For the inclusion $L(\mathcal{N}) \subseteq \{0,1\}^* \setminus L(\sigma_{\mathcal{A}_1})$ let w be a word in $L(\mathcal{N})$. Since $\{0,1\}$ is the alphabet of $\mathcal{N}$, clearly $w \in \{0,1\}^*$. Since $w \in L(\mathcal{N})$, there exists an accepting run of $\mathcal{N}$ on w. Let $q_0,\ldots,q_{|w|}$ be the sequence of states of this run. The strategy $\sigma_{\mathcal{A}_1}$ plays Call on each symbol $k \in \{0,1\}^*$ from the input word, followed by a Read on the replacement symbol k_i. If k is the jth symbol of w, then induction on j shows that Romeo can choose k_i so that the T-state after the following Read is q_j. So the T-state of the last configuration is $q_{|w|}$. Since that state is accepting in $\mathcal{N}$, it is non-accepting in T by definition. Therefore, such a play is losing for Juliet, hence $w \notin L(\sigma_{\mathcal{A}_1})$.

To see the other inclusion, let $w \in \{0,1\}^* \setminus L(\sigma_{\mathcal{A}_1})$. There must exist a losing play of $\sigma_{\mathcal{A}_1}$ on w. None of the configurations of this play can have T-state q_f because q_f is accepting and there are no transitions leaving q_f. Using the arguments from above in reverse direction, the sequence of T-states of this losing play induces an accepting run of $\mathcal{N}$ on w. Hence, $w \in L(\mathcal{N})$.

Proof of (ii): Consider a play of $\sigma_{\mathcal{A}_2}$ on a word $w \in \Sigma^*$. If $w \in \{0,1\}^*$, then the strategy $\sigma_{\mathcal{A}_2}$ plays only Read moves and the $\mathcal{A}_2$-state is q_0 all the time. Since, as mentioned above, we could assume that $q_0 \in F_{\mathcal{N}}$, it follows that $q_0 \notin F$ and therefore $\mathcal{A}_2$ loses on words from $\{0,1\}^*$. This shows that $L(\sigma_{\mathcal{A}_2}) \subseteq \Sigma^* \setminus \{0,1\}^*$.

Conversely, if $w \in \Sigma^* \setminus \{0,1\}^*$, let $x \in \Sigma \setminus \{0,1\}$ be the first symbol in w that is not in $\{0,1\}$. If $x \neq \$_{q_0}$ then $\sigma_{\mathcal{A}_2}$ calls x and x gets replaced by $\$_{q_0}$. If $x = \$_{q_0}$ or once x has been replaced by $\$_{q_0}$, the strategy $\sigma_{\mathcal{A}_2}$ reads $\$_{q_0}$ and the new T-state is q_f. This state cannot be left any more, and since it is accepting, the play is winning. Thus, $w \in L(\sigma_{\mathcal{A}_2})$, which proves the other inclusion. □

For the case that the target language is specified by an NFA, the previous result implies an **EXPSPACE** upper bound for the comparison problem; if the strategy automaton is of type 5, then the problem remains **PSPACE**-complete:

6.10 Corollary. *Let $\mathcal{G}$ be the set of all context-free games. The problem* COMPARE($\mathcal{G}, X$) *is* **PSPACE***-complete for $X = 5$ and in* **EXPSPACE** *for $X \in \{2,4\}$.*

Proof. The **EXPSPACE** upper bound holds because the target language automaton can be transformed into a DFA of exponential size and then the **PSPACE**-algorithm from Theorem 6.9 can be applied.

For the case $X = 5$, the lower bound is immediate from Theorem 6.9. For the upper bound it suffices to show that the problem is in **NPSPACE**. Let $G = (\Sigma, R, T)$, $\mathcal{A}_1$ and $\mathcal{A}_2$ be an input for the problem. An algorithm can guess which transitions to insert into $\mathcal{A}_1$ so as to obtain a standard DFA

T'. Then it can be checked using polynomial space whether T' is the minimal DFA for the target language [7] and whether also $\mathcal{A}_2$ can be obtained from T' be removing transitions. If that is the case, the **PSPACE**-algorithm from Theorem 6.9 is applied with input $G' = (\Sigma, R, T')$, $\mathcal{A}_1$ and $\mathcal{A}_2$. Otherwise the algorithm rejects. It is obvious that this algorithm accepts non-deterministically if and only if $\mathcal{A}_1$ and $\mathcal{A}_2$ are type 5 automata for G and $L(\sigma_{\mathcal{A}_1}) \subseteq L(\sigma_{\mathcal{A}_2})$. □

Interesting open questions that are connected to the comparison problem are concerned with the complexity of finding an optimal/optimum strategy. Like in the case of winning strategies, it is not even clear whether it is decidable whether an optimum strategy exists. Even testing for a given strategy whether it is optimal or optimum seems difficult. The natural approach would be to guess an automaton of a strategy that is strictly better or incomparable to the given strategy; but the problem is – again – that an upper bound on the necessary size of the guessed strategy automaton would be needed.

7. Summary

We studied several questions concerning the existence of optimum, optimal and winning strategies of five types of one-pass strategies as well as the complexity of related problems. This chapter contains a summary of the results and open questions, followed by a conclusion.

7.1. Results and open questions

The results regarding implications for optimum strategies are summarized in Figure 7.1. We showed that in non-recursive games, games with a finite target language and games with at most finitely many indistinguishability classes of one-pass strategies, the existence of an optimum one-pass strategy guarantees the existence of an optimum strategy that is of type 5. However, if none of these three conditions is satisfied, then the implications $1 \Longrightarrow 3$, $2 \Longrightarrow 4$ and $4 \Longrightarrow 5$ do not in general hold – even if all replacement languages are singletons. It remains open whether the existence of an optimum type 1 (or 3) strategy implies the existence of an optimum type 2 (or 4) strategy. We could show that if replacement languages are self-delimiting, then this is the case for the implication $1 \Longrightarrow 2$.

We have shown that optimal strategies always exist in non-recursive games, games with a finite target language and games with self-delimiting replacement languages. More generally, they exist in games that have the bounded depth property. Perhaps one of the most important open questions to which an answer could considerably improve the understanding of one-pass strategies is whether every context-free game has this property. If this were the case, a direct consequence would be that there always exists an optimal one-pass strategy; indirectly, this might also entail further results that, so far, we were able to show only in the case of non-recursiveness or self-delimiting replacement languages. In the case of finite target languages, even an optimal type 2 strategy is guaranteed to exist because the implication $1 \Longrightarrow 2$ holds for these games; also the implication $3 \Longrightarrow 4$ holds for games with a finite target language. It is open whether these two implications hold more generally. Especially for the implication $1 \Longrightarrow 2$ there is hope that it holds at least for games with self-delimiting replacement languages, and it might be possible to show this by adjusting the arguments due to which this implication holds for optimum and winning strategies. A positive answer would imply that every game with self-delimiting replacement languages has an optimal type 2 strategy. The remaining implications for optimal strategies, however, do not in general hold even if the conditions are satisfied under which they hold for optimum strategies. An overview of our results regarding the existence of optimal strategies of the five types is given in Figure 7.2.

Finally, we have considered implications for winning strategies, i. e. whether for each input word the existence of a winning strategy of one type implies the existence of a winning strategy of a restricted type. The results are similar to those for optimal strategies, but additionally we

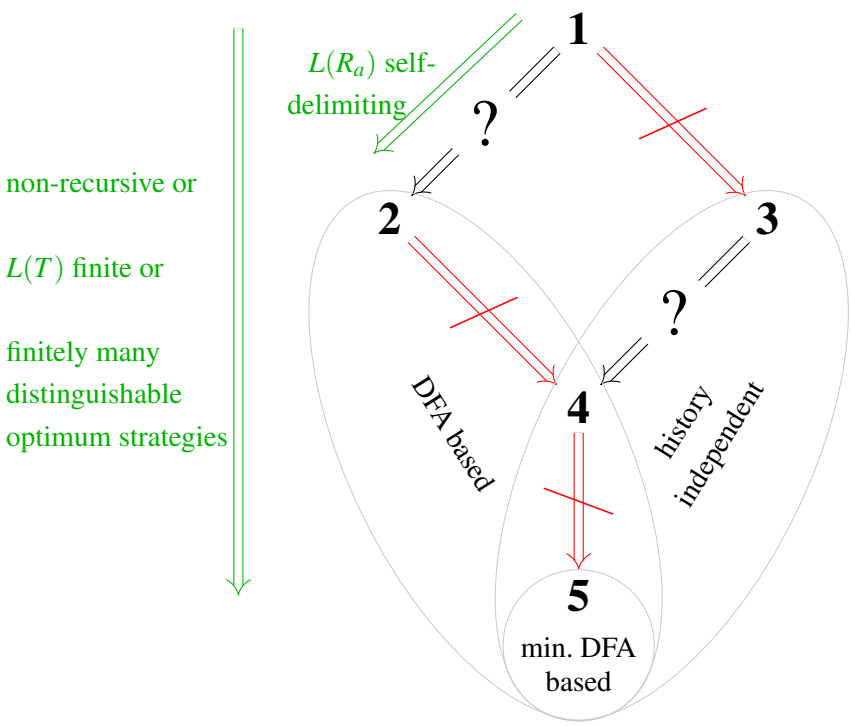


Figure 7.1.: Implications for optimum strategies

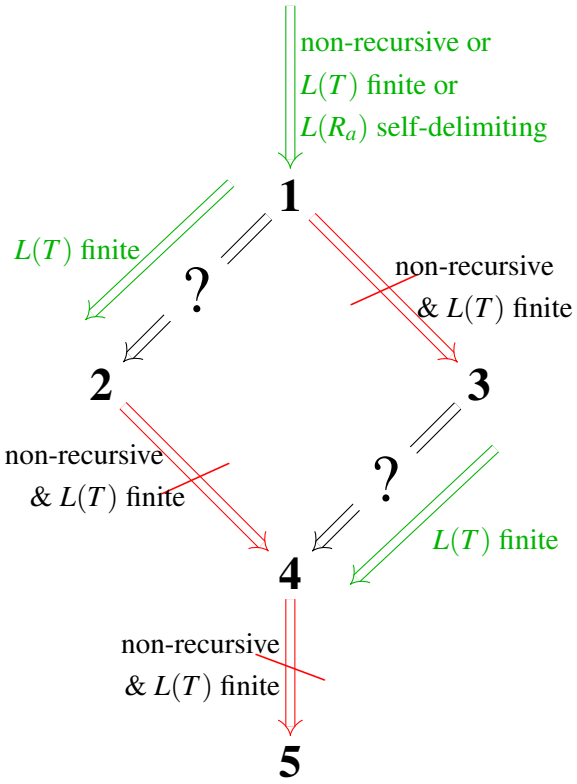


Figure 7.2.: Implications for optimal strategies

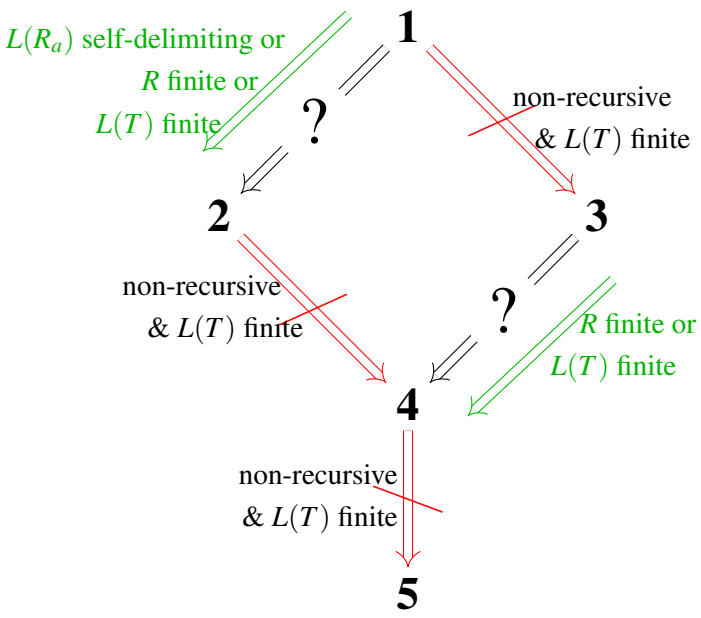


Figure 7.3.: Implications for winning strategies

could show that the implications $1 \Longrightarrow 2$ and $3 \Longrightarrow 4$ hold in the case of finitely many rules, and the implication $1 \Longrightarrow 2$ also holds in the case of self-delimiting replacement languages. The implications for winning strategies are summarized in Figure 7.3.

An overview of most of our complexity results is given in Table 7.1. We showed that deciding whether an automaton strategy wins on an input word in a game is, in general, **PSPACE**-complete. The problem is tractable, namely **P**-complete, if the target language is specified by a DFA. The upper bound in this case follows from the possibility of constructing in polynomial time a universal automaton recognizing the (regular) language of words on which the automaton strategy wins. The lower bounds in both cases hold even if the game is restricted to be non-recursive and have self-delimiting and finite replacement languages.

Further, deciding whether a type 5 strategy exists that wins on a given word in a given game is **NP**-complete if the target language is specified by a DFA. If it is specified by an NFA, then the problem becomes harder (presuming standard assumptions about distinctness of complexity classes) and the complexity lies between **PSPACE**-hardness and **NEXPTIME**. For the case of strategies of type 1, 2, 3 and 4 it is not clear whether the according problem is even decidable in the general case. It is decidable, but presumably of high complexity, if conditions are satisfied that guarantee that the implications $1 \Longrightarrow 2$ and $3 \Longrightarrow 4$ respectively for winning strategies hold and that allow an upper bound on the necessary size of a strategy automaton. A result not listed in Table 7.1 is that EXISTSWINNING$(\mathcal{G},X)$ is in **NEXPTIME** if $\mathcal{G}$ is the set of non-recursive games with a finite target language.

Towards determining optimum or optimal strategies, it is useful to decide which of two automaton strategies is better. Therefore, we defined the comparison problem. It is **PSPACE**-complete for the case of type 5 automata or if the target language is given by a DFA. It seems likely that the problem is harder in the case of type 2 and 4 automata if the target language is given by an NFA.

	T NFA			T DFA		
	general	self-delimiting	R finite, non-rec.	general	self-delimiting	R finite, non-rec.
IsWinning$(\mathcal{G},2)$ IsWinning$(\mathcal{G},4)$ IsWinning$(\mathcal{G},5)$	**PSPACE**-complete			**P**-complete		
ExistsWinning$(\mathcal{G},1)$ ExistsWinning$(\mathcal{G},2)$		$\in$ 3**NEXP**	$\in$ 2**NEXP**		$\in$ 2**NEXP**	$\in$ 2**NEXP**
ExistsWinning$(\mathcal{G},3)$ ExistsWinning$(\mathcal{G},4)$			$\in$ 2**NEXP**			$\in$ 2**NEXP**
ExistsWinning$(\mathcal{G},5)$	**PSPACE**-hard, $\in$ **NEXP**			**NP**-complete		
Compare$(\mathcal{G},2)$ Compare$(\mathcal{G},4)$	$\in$ **EXPSPACE**			**PSPACE**-complete		
Compare$(\mathcal{G},5)$	**PSPACE**-complete					

Table 7.1.: Complexity results

Further interesting open questions are concerned with the decidability and complexity of whether an optimum one-pass strategy exists for a given game or whether a given strategy is optimum or optimal. While some upper bounds for restricted cases can be deduced from the results above, the general case as well as lower bounds remain open. One could soften the definitions of optimum and optimal by saying that a strategy is type-5-optimum/optimal if it is optimum/optimal only among all type 5 strategies (and not among all one-pass strategies). Since there exist only finitely many type 5 strategies for a game, a type-5-optimal strategy must exist. For the case that the target language is given as a DFA, a **PSPACE** upper bound for determining whether a type-5-optim*um* strategy exists or whether a given type 5 strategy is type-5-optimum/optimal follows easily from our results for Compare$(\mathcal{G},5)$; it is unclear whether this bound is tight.

7.2. Conclusion

We have seen that in several cases it is possible to use a one-pass strategy that is based on an automaton and there is hope that this is possible in general. However, we can negate the hope that one could restrict to strategy automata as simple as type 5 automata without losing strategic power. The advantage of automaton strategies is, as mentioned in [3], that their moves can be computed very efficiently. On the other hand, however, we have also seen that it is not generally possible to *find* good strategies efficiently.

An issue with the original, unrestricted strategies for context-free games defined in [12] is that it is undecidable whether a winning unrestricted strategy for a given word exists. This problem is eliminated by left-to-right strategy, but it might arise again for one-pass strategies due to the new aspect of incomplete information. This issue is certainly avoided if a game has self-delimiting replacement languages. Indeed, self-delimiting replacement languages seem to be a realistic

solution in practice because they can be achieved easily and with almost no overhead by suffixing replacement words with an end-of-file symbol. Another setting in which many positive properties hold is if the target language is finite. While finiteness of the target language may seem like a very strong restriction, this is often the case in practice according to the discussion in [12].

A. Appendix

We prove some further results that were mentioned above.

A.1 Theorem. *There exists a game G that is neither left- nor right-recursive for which the implications* $1 \Longrightarrow 3$ *and* $2 \Longrightarrow 4$ *for optimum strategies do not hold.*

Proof. Consider the game $G = (\Sigma, R, T)$ with $\Sigma = \{a, b\}$, the only rule being $a \to bab$ and the target language $L(T) = \{uabv \mid u, v \in \Sigma^*\}$ consisting of all strings with substring ab.

A type 2 automaton specifying an optimal strategy for G is given in Figure A.1. It wins on all words containing the symbol a.

However, there exists no optimum type 3 strategy for G. For the sake of contradiction, suppose σ were an optimum type 3 strategy for G. There must exist $k \in \mathbb{N}_0$ such that $\sigma((\widehat{ab})^k, a) = \mathit{Read}$ because otherwise the (unique) play of σ on a would be infinite. Since σ is a type 3 strategy, this implies that $\sigma(b^k, a) = \mathit{Read}$. But then σ loses on $b^k a$, and therefore σ is not optimum. □

A.2 Theorem. *There exists a non-recursive game $G = (\Sigma, R, T)$ with $|L(R_a)| = 1$ for each $a \in \Sigma_f$ such that the implication* $4 \Longrightarrow 5$ *for optimal strategies does not semi-hold for G.*

Proof. Consider the game $G = (\Sigma, R, T)$ over the alphabet $\Sigma = \{a, b, c, d\}$ with rules given by

$$a \to dbcbd$$
$$b \to d$$

and target language automaton T as per Figure A.2a. Like in the proof of Theorem 4.17 it suffices to show that the implication $4 \Longrightarrow 5$ does not *hold* for G, since G is non-recursive.

Since Romeo can never choose from more than one replacement word, a play is uniquely determined by a one-pass strategy and the input word. Note that the state q_2 of T is an error state, so if it is entered during some play, then Juliet will surely lose.

The rest of this proof consists of two parts. First, we show that there exists an optimal type 4 strategy for G. Then we show that there exists no optimal type 5 strategy for G.

Existence of an optimal type 4 strategy:

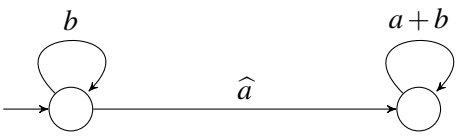
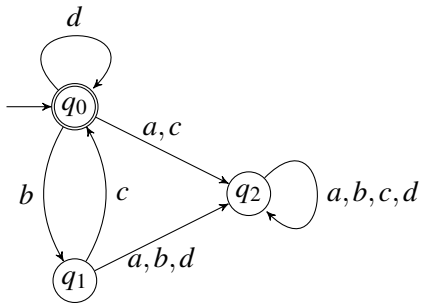
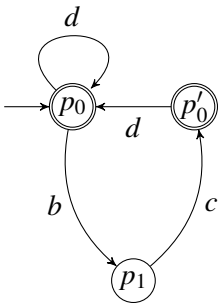


Figure A.1.: Automaton of an optimum type 2 strategy for the game G in the proof of Theorem A.1



(a) Target language automaton T



(b) Type 4 automaton $\mathcal{A}$. For clarity, transitions for c and d to an (error) state p_2 are not displayed.

Figure A.2.: Automata T and $\mathcal{A}$ for the proof of Theorem A.2

Consider the type 4 automaton $\mathcal{A}$ depicted in Figure A.2b. Note the correspondence between states of $\mathcal{A}$ and states of T: When $\mathcal{A}$ is in state p_i or p_i' during a play of $\sigma_\mathcal{A}$, then T is in its state q_i.

We claim that $\sigma_\mathcal{A}$ is optimal. Suppose it is not, then there exists a type 1 strategy σ with $L(\sigma_\mathcal{A}) \subsetneq L(\sigma)$. Let $w \in L(\sigma) \setminus L(\sigma_\mathcal{A})$ and let Π be the losing play of $\sigma_\mathcal{A}$ on w. Let $K = (\overline{u}, ev)$ with $\overline{u} \in \overline{\Sigma}^*$, $e \in \Sigma_f = \{a, b\}$ and $v \in \Sigma^*$ be the last configuration of Π up to which Π could also be a play of σ, i. e. the first configuration with $\sigma(\overline{u}, e) \neq \sigma_\mathcal{A}(\overline{u}, e)$. There are several possibilities:

- If $e = a$, then σ would read a next, since $\sigma_\mathcal{A}$ always calls a. But then the error state q_2 of T is entered immediately and σ would lose on w. Contradiction.
- If $e = b$ and the $\mathcal{A}$-state of K is p_1, then σ would lose again by reading b because the q_2 is entered. Contradiction.
- If $e = b$ and the $\mathcal{A}$-state of K is p_0', then the symbol that was read last was c. The current symbol b is either the b in forth position in the replacement string $dbcbd$ of a or from the input word. In the former case, the next symbol of the current string is d. The strategy σ would enter q_1 upon reading b and then the error state q_2 upon reading d, so σ would lose on w. If the current b is from the input word w, then the invisible suffix of K is also from the input word, i. e. $w = ubv$ for some $u \in \Sigma^*$. Consider the (unique) plays of σ and $\sigma_\mathcal{A}$ on ub. Both will have the configuration $(\overline{u}, b)$ with T-state q_0. But σ loses on ub since the T-state after reading b will be the non-accepting state q_1, whereas $\sigma_\mathcal{A}$ wins on ub because the last configuration will be $(\widehat{\overline{u}bd}, \varepsilon)$ with T-state q_0. Hence $ub \in L(\sigma_\mathcal{A}) \setminus L(\sigma)$, in contradiction to $L(\sigma_\mathcal{A}) \subset L(\sigma)$.
- If $e = b$ and the $\mathcal{A}$-state of K is p_0, then the current symbol b is either the b in second position in the replacement string $dbcbd$ of a or from the input word, i. e. $w = ubv$ for some $u \in \Sigma^*$. In the former case, σ would lose because the error state q_2 is reached after calling b and reading d and c. In the latter case, it follows similarly to above that $ubc \in L(\sigma_\mathcal{A}) \setminus L(\sigma)$, in contradiction to $L(\sigma_\mathcal{A}) \subset L(\sigma)$.

So we have a contradiction in each case and therefore $\sigma_\mathcal{A}$ is optimal.

Nonexistence of an optimal type 5 strategy:

Let $\mathcal{B}$ be an arbitrary type 5 automaton for G. Since T is clearly the minimal DFA for $L(T)$, the automaton $\mathcal{B}$ is obtained from T by removing edges. We construct a strategy σ that plays like $\sigma_{\mathcal{B}}$ on input words not starting with a but differently on input words starting with a. The construction will guarantee that σ wins on all words on which $\sigma_{\mathcal{B}}$ wins and additionally on a (and some other words starting with a).

We first show that $\sigma_{\mathcal{B}}$ does not win on any word that starts with the symbol a. For the sake of contradiction suppose $\sigma_{\mathcal{B}}$ does win on a word au with $u \in \Sigma^*$. Like before, $\sigma_{\mathcal{B}}$ must avoid entering q_2 in order to have a chance of winning. The strategy $\sigma_{\mathcal{B}}$ must first call a because otherwise q_2 would be entered immediately. The d at the start of the replacement string $dbcbd$ can only be read. The configuration hereafter is $(\widehat{a}d, bcbdu)$ with T-state q_0. Here, $\sigma_{\mathcal{B}}$ must read b to avoid entering q_2 upon reading c later on. After reading b and then reading c, the new configuration is $(\widehat{a}dbc, bdu)$ and the T-state is again q_0. Since $\sigma_{\mathcal{B}}$ is a type 5 strategy and $\sigma_{\mathcal{B}}$ played Read the last time when the T-state were q_0 and the current symbol was b, it must play Read again. But then the T-state will be the error state q_2 when the configuration $(\widehat{a}dbcbd, u)$ is reached. Thus, $\sigma_{\mathcal{B}}$ does not win on au in contradiction to out assumption.

A strategy σ that is strictly better than $\sigma_{\mathcal{B}}$ can be defined informally as follows:

- If the input word starts with a, then σ first calls a (so it gets replaced by $dbcbd$), then reads d, b and c, calls b (so it gets replaced by d) and reads all remaining symbols (the first two of which are both d).
- If the input word does not start with a, then σ plays like $\sigma_{\mathcal{B}}$.

Since σ plays like $\sigma_{\mathcal{B}}$ on all input words not starting with a, so in particular on all strings in $L(\sigma_{\mathcal{B}})$, it also wins on all strings in $L(\sigma_{\mathcal{B}})$. However, σ also wins on input a (and on some other strings starting with a). So we have $L(\sigma_{\mathcal{B}}) \subsetneq L(\sigma_{\mathcal{B}}) \dot{\cup} \{a\} \subseteq L(\sigma)$, which shows that $\sigma_{\mathcal{B}}$ is not optimal. □

A.3 Lemma (cf. [6, Theorem 5.6.1]). *The emptiness problems for context-free grammars is* **P***-hard even if grammars are restricted to be non-recursive and replacement languages self-delimiting (i. e. for each variable A, the language of strings that appear on the right hand side of rules for A is self-delimiting).*

Proof. We give a reduction from the **P**-hard emptiness problem for context-free grammars without the stated restrictions (see [8, Corollary 11] and [6, Theorem 5.6.1]).

Let $C = (V, \Gamma, P, S)$, where V is the set of variables, Γ the set of terminals, $P \subset V \times (V \cup \Gamma)^*$ the set of production rules and $S \in V$ the start variable. We will construct a non-recursive grammar with self-delimiting replacement languages that describes the empty language if and only if C describes the empty language.

We can achieve self-delimiting replacement languages as follows. If C contains two $A \to u$ and $A \to uv$ for some $A \in V$, $u \in (V \cup \Gamma)^*$ and $v \in (V \cup \Gamma)^+$, then we remove the rule $A \to uv$. Clearly, if the language of C is empty, the language of the new grammar is still empty because there exist

fewer rules. Conversely, if it is possible to derive a word from S using the rules of C, then it is also possible to derive a word from S without making use of the rule $A \to uv$. Hence, this adjustment does not affect whether the language described by the context-free grammar is non-empty.

A Turing machine only requires logarithmic space to compute the context-free grammar that is obtained by perform this adjustment repeatedly until replacement languages are self-delimiting: The Turing machine iterates pairs of positions in the encoding of the context-free grammar. If both positions denote the beginning of an encoding of a rule for the same variable A, the Turing machine compares the two right hand sides symbol by symbol to check if one is a proper prefix of the other. If that is the case, the rule with the longer right hand side needs to be removed. For this, the Turing machine outputs the context-free grammar consisting of only those rules $A \to u$ for which there exists no rule $A \to uv$ with a strictly longer right hand side.

To achieve non-recursiveness, observe that if $L(C)$ is non-empty, then it also contains a string that has a derivation tree of depth at most $|V|$. Indeed, in a derivation tree of depth greater than $|V|$, some variable must occur twice (or more) on a path from the root to a leaf. This path can be shortened by replacing the subtree rooted at the upper occurrence of the variable by the subtree rooted at the lower occurrence, leading to a derivation tree for some other word. After repeating this often enough, a derivation tree of depth at most $|V|$ for a word in $L(C)$ is obtained. A non-recursive grammar $C' = (V', \Gamma', P', S')$ with $L(C') = \emptyset$ if and only if $L(C) = \emptyset$ can be constructed as follows:

- Define $V' = \{A_i \mid A \in V, i \in \{1, \ldots, |V|\}\}$.
- For each rule $A \to w$ of P and all $i = 1, \ldots, |V| - 1$, introduce for P' the rule $A_i \to w_{i+1}$, where w_{i+1} is the string obtained from w by replacing each variable B in w by B_{i+1}. If w consists of only terminals, also introduce the rule $A_{|V|} \to w$.
- Define $\Gamma' = \Gamma$ and $S' = S_1$.

It is straightforward to check, due to the arguments above, that $L(C') = \emptyset$ if and only if $L(C) = \emptyset$. Further, the construction of C' requires only logarithmic space.

The statement of the lemma follows, since the concatenation of two logarithmic space computable functions can be computed using logarithmic space. □

Bibliography

[1] S. Abiteboul, O. Benjelloun, and T. Milo. The Active XML project: an overview. *The VLDB Journal*, 17(5):1019–1040, 2008.

[2] S. Abiteboul, A. Bonifati, G. Cobéna, I. Manolescu, and T. Milo. Dynamic XML documents with distribution and replication. In *Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data*, pages 527–538. ACM, 2003.

[3] S. Abiteboul, T. Milo, and O. Benjelloun. Regular rewriting of Active XML and unambiguity. In *PODS*, pages 295–303. ACM, 2005.

[4] H. Björklund, M. Schuster, T. Schwentick, and J. Kulbatzki. On optimum left-to-right strategies for active context-free games. In *Joint 2013 EDBT/ICDT Conferences, ICDT '13 Proceedings*, pages 105–116, 2013.

[5] V. M. Glushkov. The abstract theory of automata. *Russian Mathematical Surveys*, 16(5):1–53, 1961.

[6] E. Gurari. *An introduction to the theory of computation*. Computer Science Press, New York, NY, USA, 1989.

[7] H. B. Hunt, D. J. Rosenkrantz, and T. G. Szymanski. On the equivalence, containment, and covering problems for the regular and context-free languages. *Journal of Computer and System Sciences*, 12(2):222–268, 1976.

[8] N. D. Jones and W. T. Laaser. Complete problems for deterministic polynomial time. *Theoretical Computer Science*, 3(1):105–117, 1976.

[9] A. R. Meyer and L. J. Stockmeyer. The equivalence problem for regular expressions with squaring requires exponential space. In *Proceedings of the 13th Annual Symposium on Switching and Automata Theory*, SWAT '72, pages 125–129. IEEE Computer Society, 1972.

[10] T. Milo, S. Abiteboul, B. Amann, O. Benjelloun, and F. D. Ngoc. Exchanging intensional XML data. *ACM Transactions on Database Systems*, 30(1):1–40, 2005.

[11] A. Muscholl, T. Schwentick, and L. Segoufin. Active context-free games. In *STACS 2004, 21st Annual Symposium on Theoretical Aspects of Computer Science*, pages 452–464, 2004.

[12] A. Muscholl, T. Schwentick, and L. Segoufin. Active context-free games. *Theory of Computing Systems*, 39(1):237–276, 2006.

[13] M. Schuster and T. Schwentick. Games for Active XML revisited. In *18th International Conference on Database Theory, ICDT 2015*, pages 60–75, 2015.

Eidesstattliche Versicherung

_____ _____

Name, Vorname Matr.-Nr.

Ich versichere hiermit an Eides statt, dass ich die vorliegende Bachelorarbeit/Masterarbeit* mit dem Titel

_____

_____

_____

selbstständig und ohne unzulässige fremde Hilfe erbracht habe. Ich habe keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie wörtliche und sinngemäße Zitate kenntlich gemacht. Die Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

_____ _____

Ort, Datum Unterschrift

*Nichtzutreffendes bitte streichen

Belehrung:

Wer vorsätzlich gegen eine die Täuschung über Prüfungsleistungen betreffende Regelung einer Hochschulprüfungsordnung verstößt, handelt ordnungswidrig. Die Ordnungswidrigkeit kann mit einer Geldbuße von bis zu 50.000,00 € geahndet werden. Zuständige Verwaltungsbehörde für die Verfolgung und Ahndung von Ordnungswidrigkeiten ist der Kanzler/die Kanzlerin der Technischen Universität Dortmund. Im Falle eines mehrfachen oder sonstigen schwerwiegenden Täuschungsversuches kann der Prüfling zudem exmatrikuliert werden. (§ 63 Abs. 5 Hochschulgesetz - HG -)

Die Abgabe einer falschen Versicherung an Eides statt wird mit Freiheitsstrafe bis zu 3 Jahren oder mit Geldstrafe bestraft.

Die Technische Universität Dortmund wird gfls. elektronische Vergleichswerkzeuge (wie z.B. die Software „turnitin“) zur Überprüfung von Ordnungswidrigkeiten in Prüfungsverfahren nutzen.

Die oben stehende Belehrung habe ich zur Kenntnis genommen:

_____ _____

Ort, Datum Unterschrift