

1 ω -Automata

Berndt Farwer

Fachbereich Informatik
Universität Hamburg

1.1 Introduction and Notation

Automata on infinite words have gained a great deal of importance since their first definition some forty years ago. Apart from the interests from a theoretical point of view they have practical importance for the specification and verification of reactive systems that are not supposed to terminate at some point of time. Operating systems are an example of such systems, as they should be ready to process any user input as it is entered, without terminating after or during some task.

The main topic covered in this chapter is the question how to define acceptance of infinite words by finite automata. In contrast to the case of finite words, there are many possibilities, and it is a nontrivial problem to compare them with respect to expressive power.

First publications referring to ω -languages date back to the 1960's, at which time Büchi obtained a decision procedure for a restricted second-order theory of classical logic, the sequential calculus *S1S* (second order theory of one successor), by using finite automata with infinite inputs [17]. Muller [133] defined a similar concept in a totally different domain, namely in asynchronous switching network theory. Starting from these studies, a theory of automaton definable ω -languages (sets of infinite words) emerged. Connections were established with other specification formalisms, e.g. regular expressions, grammars, and logical systems. In this chapter, we confine ourselves to the automata theoretic view.

1.1.1 Notation

The symbol ω is used to denote the set of non-negative integers, i.e. $\omega := \{0, 1, 2, 3, \dots\}$.

By Σ we denote a finite alphabet. Symbols from a given alphabet are denoted by a, b, c, \dots . Σ^* is the set of finite words over Σ , while Σ^ω denotes the set of infinite words (or ω -words) over Σ (i.e. each word $\alpha \in \Sigma^\omega$ has length $|\alpha| = \omega$). Letters u, v, w, \dots denote finite words, infinite words are denoted by small greek letters $\alpha, \beta, \gamma, \dots$. We write $\alpha = \alpha(0)\alpha(1)\dots$ with $\alpha(i) \in \Sigma$. Often we indicate infinite runs of automata by ϱ, σ, \dots . A set of ω -words over a given alphabet is called an ω -language.

For words α and w , the number of occurrences of the letter a in α and w is denoted by $|\alpha|_a$ and $|w|_a$, respectively. Given an ω -word $\alpha \in \Sigma^\omega$, let

$$\text{Occ}(\alpha) = \{a \in \Sigma \mid \exists i \alpha(i) = a\}$$

be the (finite) set of letters occurring in α , and

$$\text{Inf}(\alpha) = \{a \in \Sigma \mid \forall i \exists j > i \alpha(j) = a\}$$

be the (finite) set of letters occurring infinitely often in α .

The powerset of a set M is denoted by 2^M and $|M|$ denotes the cardinality of M . The i -th projection of an ordered tuple or vector $\mathbf{a} = (a_1, \dots, a_k)$ is defined for $i \leq k$ and is written $\pi_i(\mathbf{a}) = a_i$.

The class of regular languages is denoted by REG.

1.2 ω -Automata

In classical formal language theory, the notion of acceptance of a word by an automaton is a well-known concept. One defines the notion of **finite computation** or **finite run** of an automaton on a given input word, specifies the configurations (by control states, or by control states and memory contents) which are considered to be “final”, and declares an input accepted if a run exists on the input which terminates in a final configuration.

In the present context we are interested only in the acceptance of words by automata (and not in generation of ω -words by grammars). Also we only consider finite automata. The definitions of acceptors and generators for context-free languages and more general language classes have also been adapted to suit the case of infinite words (see for example [113, 36, 37, 38]), or the survey [165]. In the remainder of this chapter we will use *ω -automaton* synonymously for *finite ω -automaton*.

The usual definitions of deterministic and nondeterministic automata are adapted to the case of ω -input-words by the introduction of new acceptance conditions. For this purpose one introduces an “acceptance component” in the specification of automata, which will arise in different formats.

Definition 1.1. An **ω -automaton** is a quintuple $(Q, \Sigma, \delta, q_I, Acc)$, where Q is a finite set of states, Σ is a finite alphabet, $\delta : Q \times \Sigma \rightarrow 2^Q$ is the state transition function, $q_I \in Q$ is the initial state, and Acc is the **acceptance component**. In a deterministic ω -automaton, a transition function $\delta : Q \times \Sigma \rightarrow Q$ is used.

The acceptance component can be given as a set of states, as a set of state-sets, or as a function from the set of states to a finite set of natural numbers. Instances of all these case will be presented below.

Definition 1.2. Let $\mathcal{A} = (Q, \Sigma, \delta, q_I, Acc)$ be an ω -automaton. A **run** of \mathcal{A} on an ω -word $\alpha = a_1 a_2 \dots \in \Sigma^\omega$ is an infinite state sequence $\varrho = \varrho(0)\varrho(1)\varrho(2)\dots \in Q^\omega$, such that the following conditions hold:

- (1) $\varrho(0) = q_I$,
- (2) $\varrho(i) \in \delta(\varrho(i-1), a_i)$ for $i \geq 1$ if \mathcal{A} is nondeterministic,
 $\varrho(i) = \delta(\varrho(i-1), a_i)$ for $i \geq 1$ if \mathcal{A} is deterministic.

With the different acceptance conditions defined in the following sections the question arises how they are related in expressive power, i.e. whether there exist transformations from one acceptance condition to another. If such transformations can be established another question naturally arises: what is the complexity for the respective translations?

The size of an automaton \mathcal{A} , denoted by $|\mathcal{A}|$, is measured by the number of its states, i.e. for $\mathcal{A} = (Q, \Sigma, \delta, q_I, Acc)$ the size is $|\mathcal{A}| = |Q|$. In addition to the number of states of an automaton the size of the acceptance condition is also of some importance for the efficiency of the transformation. This is usually measured by the number of designated sets or pairs of such. Details are given in the respective sections.

1.3 Nondeterministic Models

1.3.1 Büchi Acceptance

The Büchi acceptance condition has originally been introduced for nondeterministic ω -automata. In this case, the acceptance component is a set of states.

Definition 1.3. An ω -automaton $\mathcal{A} = (Q, \Sigma, \delta, q_I, F)$ with acceptance component $F \subseteq Q$ is called **Büchi automaton** if it is used with the following acceptance condition (**Büchi acceptance**): A word $\alpha \in \Sigma^\omega$ is accepted by \mathcal{A} iff there exists a run ϱ of \mathcal{A} on α satisfying the condition:

$$\text{Inf}(\varrho) \cap F \neq \emptyset$$

i.e. at least one of the states in F has to be visited infinitely often during the run. $L(\mathcal{A}) := \{\alpha \in \Sigma^\omega \mid \mathcal{A} \text{ accepts } \alpha\}$ is the ω -language recognized by \mathcal{A} .

Example 1.4. Consider the ω -language L over the alphabet $\{a, b\}$ defined by

$$L := \{\alpha \in \{a, b\}^\omega \mid \alpha \text{ ends with } a^\omega \text{ or } \alpha \text{ ends with } (ab)^\omega\}.$$

L is recognized by the nondeterministic Büchi automaton given by the state transition diagram from Figure 1.1. The states from F are drawn with a double circle.

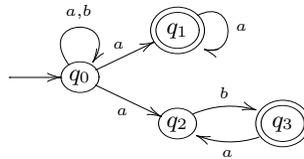


Fig. 1.1. A Büchi automaton accepting the words from $(a + b)^* a^\omega + (a + b)^* (ab)^\omega$

Consider a Büchi automaton $\mathcal{A} = (Q, \Sigma, \delta, q_I, F)$. Using this automaton with initial state p and final state q we obtain a regular language $W(p, q)$ of finite words. An ω -word α is accepted by \mathcal{A} iff some run of \mathcal{A} on α visits some final state $q \in F$ infinitely often. This is equivalent to $\alpha \in W(q_0, q) \cdot W(q, q)^\omega$. Taking the union over these sets for $q \in F$, we obtain the following representation result for Büchi recognizable ω -languages.

Theorem 1.5. *The Büchi recognizable ω -languages are the ω -languages of the form*

$$L = \bigcup_{i=1}^k U_i V_i^\omega \text{ with } k \in \omega \text{ and } U_i, V_i \in \text{REG for } i = 1, \dots, k$$

This family of ω -languages is also called the ω -Kleene closure of the class of regular languages.

From this remark one concludes immediately that each nonempty Büchi recognizable ω -language contains an ultimately periodic word.

Let us also note that the **emptiness problem** is decidable for Büchi automata, i.e. there exists an algorithm that decides whether the language recognized by an arbitrary (nondeterministic) Büchi automaton is empty. Given a Büchi automaton \mathcal{A} , one computes the set of reachable states, and for each reachable state q from F checks whether q is reachable from q by a nonempty path. Such a loop exists if and only if there exists an infinite word α and a run of \mathcal{A} on α such that q is a recurring state in this run.

1.3.2 Muller Acceptance

The Muller acceptance condition refers to an acceptance component which is a set of state sets $\mathcal{F} \subseteq 2^Q$.

Definition 1.6. An ω -automaton $\mathcal{A} = (Q, \Sigma, \delta, q_I, \mathcal{F})$ with acceptance component $\mathcal{F} \subseteq 2^Q$ is called **Muller automaton** when used with the following acceptance condition (**Muller acceptance**): A word $\alpha \in \Sigma^\omega$ is accepted by \mathcal{A} iff there exists a run ϱ of \mathcal{A} on α satisfying the condition:

$$\text{Inf}(\varrho) \in \mathcal{F}$$

i.e. the set of infinitely recurring states of ϱ is exactly one of the sets in \mathcal{F} .

Example 1.7. Consider again the ω -language L over $\{a, b\}$ consisting of the ω -words which end with a^ω or with $(ab)^\omega$. The deterministic Muller automaton of Figure 1.2 recognizes L , where the acceptance component consists of the two sets $\{q_a\}$ and $\{q_a, q_b\}$.

We now verify that nondeterministic Büchi automata and nondeterministic Muller automata are equivalent in expressive power.

One direction is straightforward: for a Büchi automaton $\mathcal{A} = (Q, \Sigma, \delta, q_I, F)$ define the family \mathcal{F} of sets of states by collecting all subsets of Q which contain a state from F .

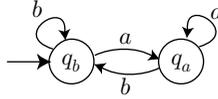


Fig. 1.2. A state transition diagram where the state q_a is reached after reading a and q_b after reading b .

Transformation 1.8. Let $\mathcal{A} = (Q, \Sigma, \delta, q_I, F)$ be a Büchi automaton. Define the Muller automaton $\mathcal{A}' = (Q, \Sigma, \delta, q_I, \mathcal{F})$ with $\mathcal{F} := \{G \in 2^Q \mid G \cap F \neq \emptyset\}$. Then $L(\mathcal{A}) = L(\mathcal{A}')$.

For the converse, a Muller automaton $\mathcal{A} = (Q, \Sigma, \delta, q_I, \mathcal{F})$ is given. The desired Büchi automaton \mathcal{A}' simulates \mathcal{A} and, in order to accept, it guesses the set $G \in \mathcal{F}$ which should turn out to be $\text{Inf}(\varrho)$ for the run ϱ to be pursued. For checking that the guess is correct, \mathcal{A}' makes another guess during the run, namely from which position onwards exactly the states from G will be seen again and again. This claim can be verified by accumulating the visited states in memory until the set G is complete, then resetting the memory to \emptyset and starting accumulating again, and so on. If this reset occurs again and again (and no state outside G is visited), the automaton \mathcal{A}' should accept. By declaring the “reset states” as accepting ones, we obtain the required Büchi automaton.

For an implementation of this idea, we work with the set Q of original states and introduce, for each set $G \in \mathcal{F}$, a separate copy of $Q \cap G$. We indicate such states with index G (and write q_G). The automaton \mathcal{A}' does the two guesses at the same moment, at which time it switches from a state p of Q to a state from $q_G \in G$ and initializes the accumulation component to \emptyset . So the new states for the accepting set G will be from $G \times 2^G$, where (q_G, R) codes that q is the current state of \mathcal{A} and R is the set of accumulated states since the last reset (where the R -value is \emptyset). So the set Q' of states of \mathcal{A}' is

$$Q' = Q \cup \bigcup_{G \in \mathcal{F}} (G \times 2^G)$$

and the set F' of final states of \mathcal{A}' consists of the states (q_G, \emptyset) for $G \in \mathcal{F}$. We do not give a formal definition of the transitions, which should be clear from the description above.

Transformation 1.9. Let $\mathcal{A} = (Q, \Sigma, \delta, q_I, \mathcal{F})$ be a Muller automaton. Define a Büchi automaton $\mathcal{A}' = (Q', \Sigma, \delta', q_I, F')$ with Q', δ', F' defined as described above. Then $L(\mathcal{A}) = L(\mathcal{A}')$.

If Q has n states and \mathcal{F} contains m sets then $|Q'|$ has at most $n + mn2^n = 2^{O(n)}$ states. Summarizing, we obtain the following result.

Theorem 1.10. *A nondeterministic Büchi automaton with n states can be converted into an equivalent Muller automaton of equal size, and a nondeterministic Muller automaton with n states and m accepting sets can be transformed into an equivalent Büchi automaton with $\leq n + mn2^n$ states.*

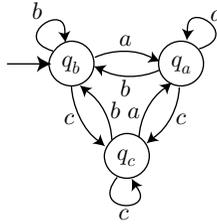


Fig. 1.3. A state diagram where q_x is reached after reading x .

The transformation sketched above transforms nondeterministic Büchi automata into nondeterministic Muller automata and conversely. For a given deterministic Büchi automaton the translation yields a deterministic Muller automaton. On the other hand, a deterministic Muller automaton is converted into a nondeterministic Büchi automaton. As we shall see later, this nondeterminism cannot in general be avoided.

1.3.3 Rabin and Streett Acceptance

The acceptance condition for Büchi automata is a positive condition on recurring states for the acceptance of ω -words. In Muller automata the specification by a set F is sharpened, because an accepting set F should contain precisely the recurring states (and not more). There are also formalisms specifying acceptance and rejection criteria separately. The Rabin condition – also called **pairs condition** – is such a condition.

The acceptance component is given by a finite family Ω of pairs (E_i, F_i) of designated state sets with the understanding that the sets E_i should be excluded from an accepting run after a finite initial segment, while at least one state in F_i has to be visited infinitely often.

Definition 1.11. An ω -automaton $\mathcal{A} = (Q, \Sigma, \delta, q_I, \Omega)$ with acceptance component $\Omega = \{(E_1, F_1), \dots, (E_k, F_k)\}$ with $E_i, F_i \subseteq Q$ is called **Rabin automaton** if it used with the following acceptance condition (**Rabin acceptance**): A word α is accepted by \mathcal{A} if there exists a run ϱ of \mathcal{A} on α such that

$$\exists(E, F) \in \Omega (\text{In}(\varrho) \cap E = \emptyset) \wedge (\text{Inf}(\varrho) \cap F \neq \emptyset).$$

Example 1.12. The Rabin automaton with state transition diagram from Figure 1.2 and Rabin condition $\Omega = \{(\{q_b\}, \{q_a\})\}$ accepts all words that consist of infinitely many a 's but only finitely many b 's.

To specify the language consisting of all words that contain infinitely many b 's only if they also contain infinitely many a 's with a Rabin automaton based on the state graph from Figure 1.2 we have to use $\Omega = \{(\emptyset, \{q_a\}), (\{q_a, q_b\}, \emptyset)\}$. This condition can be paraphrased by saying that each word in the accepted language has either infinitely many a 's or it has neither infinitely many a 's nor infinitely many b 's. It is clear that in the latter case no ω -word can be accepted

and the condition could be simplified to $\Omega = \{(\emptyset, \{q_a\})\}$. But in the presence of a third symbol and a third state as depicted in Figure 1.3 two pairs are needed, as the ω -word c^ω must be recognized: $\Omega = \{(\emptyset, \{q_a\}), (q_a, q_b), \{q_c\})\}$.

The Streett condition is dual to the Rabin condition. It is therefore sometimes called **complemented pair condition**. It can be viewed as a **fairness condition** meaning that for each pair $(E, F) \in \Omega$, if some state from F is visited infinitely often, there has to be a state from E that is also visited infinitely often during an accepting run.

Definition 1.13. An ω -automaton $\mathcal{A} = (Q, \Sigma, \delta, q_I, \Omega)$ with acceptance component $\Omega = \{(E_1, F_1), \dots, (E_k, F_k)\}$ with $E_i, F_i \subseteq Q$ is called **Streett automaton** if it is used with the following acceptance condition (**Streett acceptance**): A word α is accepted by \mathcal{A} if there exists a run ϱ of \mathcal{A} on α such that

$$\forall (E, F) \in \Omega. (\text{Inf}(\varrho) \cap E \neq \emptyset) \vee (\text{Inf}(\varrho) \cap F = \emptyset)$$

(equivalently: if $\text{Inf}(\varrho) \cap F \neq \emptyset$ then $\text{Inf}(\varrho) \cap E \neq \emptyset$).

Example 1.14. Let $\Sigma = \{a, b\}$. The language consisting of all words that contain infinitely many a 's if they contain infinitely many b 's can be recognized by a Streett automaton with the state graph from Figure 1.2. The condition can be paraphrased as $|\alpha|_b \neq \omega \vee |\alpha|_a = \omega$, i.e. $|\alpha|_b = \omega \Rightarrow |\alpha|_a = \omega$. In the automaton of Figure 1.2 the two states q_a and q_b indicate that respectively symbol a or b has been read in the previous step. The appropriate Streett automaton is obtained by taking as acceptance component the set $\Omega = \{(\{q_a\}, \{q_b\})\}$.

Rabin automata and Streett automata are transformed into Muller automata by simply gathering all state sets that satisfy the Rabin condition, respectively Streett condition, into a Muller acceptance set.

Transformation 1.15. Let $\mathcal{A} = (Q, \Sigma, \delta, q_I, \Omega)$ be a Rabin automaton, respectively Streett automaton. Define a Muller automaton $\mathcal{A}' = (Q, \Sigma, \delta, q_I, \mathcal{F})$ with $\mathcal{F} := \{G \in 2^Q \mid \exists (E, F) \in \Omega. G \cap E = \emptyset \wedge G \cap F \neq \emptyset\}$, respectively with $\mathcal{F} := \{G \in 2^Q \mid \forall (E, F) \in \Omega. G \cap E \neq \emptyset \vee G \cap F = \emptyset\}$. Then $L(\mathcal{A}) = L(\mathcal{A}')$.

For the converse it suffices to invoke the transformation of Muller automata into Büchi automata, as in the preceding subsection, and to observe that Büchi acceptance can be viewed as a special case of Rabin acceptance (for the set F of final states take $\Omega = \{(\emptyset, F)\}$), as well as a special case of Streett condition (for the set F of final states take $\Omega = \{(F, Q)\}$).

1.3.4 The Parity Condition

The parity condition amounts to the Rabin condition for the special case where the accepting pairs $(E_1, F_1), \dots, (E_m, F_m)$ form a chain with respect to set inclusion. We consider the case of an increasing chain $E_1 \subset F_1 \subset E_2 \subset \dots \subset E_m \subset F_m$.

Let us associate indices (called colours) with states as follows: states of E_1 receive colour 1, states of $F_1 \setminus E_1$ receive colour 2, and so on with the rule that states of $E_i \setminus F_{i-1}$ have colour $2i - 1$ and states of $F_i \setminus E_i$ have colour $2i$. An ω -word α is then accepted by the Rabin automaton iff the least colour occurring infinitely often in a run on α is even (hence the term “parity condition”).

Definition 1.16. An ω -automaton $\mathcal{A} = (Q, \Sigma, \delta, q_I, c)$ with acceptance component $c : Q \rightarrow \{1, \dots, k\}$ (where $k \in \omega$) is called **parity automaton** if it is used with the following acceptance condition (**parity condition**): An ω -word $\alpha \in \Sigma^\omega$ is accepted by \mathcal{A} iff there exists a run ϱ of \mathcal{A} on α with

$$\min\{c(q) \mid q \in \text{Inf}(\varrho)\} \text{ is even}$$

Sometimes it is more convenient to work with the condition that the maximal colour occurring infinitely often in the run under consideration is even. This applies to some constructions in later chapters of this book.

Example 1.17. Consider the parity automaton from Figure 1.4 with colouring function c defined by $c(q_i) = i$.

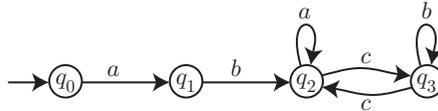


Fig. 1.4. Another ω -automaton

It accepts the ω -words with start with ab , continue by a finite sequence of segments in a^*cb^*c , and end with a^ω ; so $L(\mathcal{A}) = ab(a^*cb^*c)^*a^\omega$. For the parity automaton \mathcal{A}' with the same transition graph but colouring c' defined by $c'(q_i) = i + 1$ we obtain $L(\mathcal{A}') = ab(a^*cb^*c)^*b^\omega \vee ab(a^*cb^*c)^\omega$.

It is obvious how a parity condition is cast into the form of a Rabin condition.

Transformation 1.18. Let $\mathcal{A} = (Q, \Sigma, \delta, q_I, c)$ be an ω -automaton be a parity automaton with $c : Q \rightarrow \{0, \dots, k\}$. An equivalent Rabin automaton $\mathcal{A}' = (Q, \Sigma, \delta, q_I, \Omega)$ has the acceptance component $\Omega := \{(E_0, F_0), \dots, (E_r, F_r)\}$ with $r := \lfloor \frac{k}{2} \rfloor$, $E_i := \{q \in Q \mid c(q) < 2i\}$ and $F_i := \{q \in Q \mid c(q) \leq 2i\}$.

1.3.5 Discussion

The equivalence results obtained above can be summarized as follows:

Theorem 1.19. (1) *Nondeterministic Büchi automata, Muller automata, Rabin automata, Streett automata, and parity automata are all equivalent in expressive power, i.e. they recognize the same ω -languages.*

- (2) *The ω -languages recognized by these ω -automata form the class $\omega\text{-KC(REG)}$, i.e. the ω -Kleene closure of the class of regular languages.*

The ω -languages in this class are commonly referred to as the regular ω -languages, denoted by $\omega\text{-REG}$.

At this point two fundamental questions arise.

- Are there types of *deterministic* ω -automata which recognize precisely the ω -languages in $\omega\text{-REG}$?
- Is the class $\omega\text{-REG}$ closed under complementation?

Both questions can be answered affirmatively; and both involve tedious work.

The complementation problem can be attacked via several approaches (see Chapter 4 below). One possibility is to work with deterministic ω -automata and thus use a reduction to the determinization problem.

1.4 Deterministic Models

In Chapter 3 below, it will be shown that deterministic Muller automata recognize precisely the regular ω -languages. In the present section, we discuss the relation between deterministic Muller automata and other deterministic ω -automata, and also give some remarks on the complementation problem. We shall see that deterministic Muller automata, Rabin automata, Streett automata, and parity automata are all equivalent in expressive power. Note that the equivalence proof given above for the nondeterministic case cannot be copied: We proceeded via nondeterministic Büchi automata and thus, even from deterministic Muller automata, would obtain nondeterministic Rabin, Streett, and parity automata. As we now verify, we cannot in general sharpen the construction of a Büchi automaton to obtain a deterministic one.

1.4.1 The Büchi Condition for Deterministic ω -Automata

Let us see that Büchi automata are too weak to recognize even very simple ω -languages from $\omega\text{-REG}$. The Büchi automaton depicted in Figure 1.5 with $F = \{q_1\}$ accepts those ω -words over the alphabet $\{a, b\}$ that have only finitely many b 's.

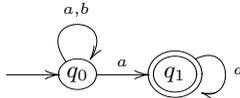


Fig. 1.5. An automaton recognizing $(a + b)a^\omega$

It is easy to provide an equivalent deterministic Muller automaton, using two states q_a, q_b which are visited after reading a, b , respectively, and declaring

$\mathcal{F} = \{\{q_a\}\}$ as acceptance component (see Figure 1.2). Then a run is accepting iff it ends by a sequence consisting of state q_a only, which means that the input word ends with a^ω .

If one would work with the Büchi acceptance condition, using a set F of accepting states, then one has a specification of states which should be visited infinitely often, but it is not directly possible to specify which states should be seen only finitely often.

The argument which shows that deterministic Büchi automata are too weak for recognizing $L = (a + b)^*b^\omega$ works by contradiction: Assuming that the deterministic Büchi automaton \mathcal{A} with final state set F recognizes L , it will on input b^ω visit an F -state after a finite prefix, say after the n_0 -th letter. It will also accept $b^{n_0}ab^\omega$, visiting F -states infinitely often and hence after the a , say when finishing the prefix $b^{n_0}ab^{n_1}$. Continuing this construction the ω -word $b^{n_0}ab^{n_1}ab^{n_2}a \dots$ is generated which causes \mathcal{A} to pass through an F -state before each letter a but which should of course be rejected.

1.4.2 Transforming Muller Automata to Rabin Automata

Let us now show that deterministic Muller automata, Rabin automata, Streett automata, and parity automata all have the same expressive power. We show first the crucial step, namely that deterministic Muller automata can be transformed into deterministic Rabin automata.

We use a technique called **latest appearance record** (LAR). The idea is to use permutations of the states of the given Muller automaton as new states, extended by a **hit position**. So the memory of the new automaton stores *lists* of states from the original automaton; this is in contrast to the construction of Theorem 1.10 which produced a nondeterministic Büchi automaton from a Muller automaton; in that case we stored *sets* of states of the original automaton in the memory of the constructed one.

In a list of (distinct) states, we use the last entry for the current state in the run on the given Muller automaton. The hit position (the position of the marker \natural) indicates where the last change occurred in the record. For every transition from one state p to q in the original automaton, the state q is moved to the last position of the record while the symbols which were to the right of q are shifted one position to the left (so the previous place of q is filled again). The marker is inserted in front of the position where q was taken from. So the positions before the marker are untouched by the transition under consideration.

Transformation 1.20. Let $\mathcal{A} = (\Sigma, Q, \delta, q_I, \mathcal{F})$ be a deterministic Muller automaton. Assume w.l.o.g. that $Q = \{1, \dots, k\}$ and $q_I = 1$. Let \natural be a new symbol, i.e. $\natural \notin Q$.

An equivalent Rabin automaton \mathcal{A}' is given by the following definition:

- \tilde{Q} is the set of all order vector words with hit position over Q , i.e.

$$\tilde{Q} := \{w \in (Q \cup \{\natural\})^* \mid \forall q \in Q \cup \{\natural\}. |w|_q = 1\}$$

- The initial state is $q'_1 := \natural k \dots 1$.
- The transition function δ' is constructed as follows: Assume $i, i' \in Q$, $a \in \Sigma$, and $\delta(i, a) = i'$. Then δ' is defined for any word $m_1 \dots m_r \natural m_{r+1} \dots m_k \in \tilde{Q}$ with $m_k = i$. Supposing that $i' = m_s$, define

$$\delta'(m_1 \dots m_r \natural m_{r+1} \dots m_k, a) := (m_1 \dots m_{s-1} \natural m_{s+1} \dots m_k i').$$

- The acceptance component is given by $\Omega = \{(E_1, F_1), \dots, (E_k, F_k)\}$, defined as follows:
 - $E_j := \{u \natural v \mid |u| < j\}$
 - $F_j := \{u \natural v \mid |u| < j\} \cup \{u \natural v \mid |u| = j \wedge \{m \in Q \mid m \sqsubseteq v\} \in \mathcal{F}\}$.

Here the infix relation $m \sqsubseteq v$ should be read as “ m occurs in v ”, since m is a single letter.

Consider a run of the Muller automaton \mathcal{A} , where the set of infinitely often visited states is, say, $J = \{m_1, \dots, m_j\}$. This means that in the corresponding run of the Rabin automaton \mathcal{A}' , the states of $Q \setminus J$ will eventually reach the first positions and then stay indefinitely in front of the marker. So finally the \mathcal{A}' -states will be of the form $u \natural v$ where the $(Q \setminus J)$ -elements occur at the beginning of u (or constitute the whole word u). Hence, eventually we will constantly have $|u| \geq |Q \setminus J|$, in other words $|v| \leq |J| = j$. Clearly infinitely often we have $|v| = |J| = j$, since otherwise, from some point onwards we would have $|v| < j$ and thus less than j states would be visited infinitely often.

So infinitely often a state $u \natural v$ with $|v| = j$ is seen but only finitely often a state with $v > j$. Moreover, the states which constitute the word v in the first case $|v| = j$ form precisely the set J .

We can summarize this as follows:

Lemma 1.21. *Let ϱ be an infinite run of the deterministic Muller automaton \mathcal{A} with state set $Q = \{1, \dots, k\}$ and let $u_0 \natural v_0, u_1 \natural v_1, u_2 \natural v_2, \dots$ be the corresponding sequence of order vectors with hit, according to Transformation 1.20. Then $\text{Inf}(\varrho) = J$ with $|J| = j$ iff the following conditions hold:*

- for only finitely many i we have $|v_i| > j$ (and hence $|u_i| \leq k - j$)
- for infinitely many i we have $|v_i| = j$ (and hence $|u_i| = k - j$) and $J = \{m \in Q \mid m \sqsubseteq v_i\}$.

The Muller automaton \mathcal{A} accepts by the run ϱ if the set J considered in the Lemma belongs to \mathcal{F} . This means that the run will infinitely often visit a state in the defined set F_{k-j} but only finitely often visit states $u \natural v$ with $|u| < k - j$, i.e. states from E_{k-j} . So the Rabin condition of \mathcal{A}' is satisfied and \mathcal{A}' accepts in this case. The converse implication (“if \mathcal{A}' accepts an input word, then \mathcal{A} does”) is shown analogously.

From the definition of the sets E_j, F_j we see that they are arranged in a chain: $E_1 \subseteq F_1 \subseteq E_2 \dots \subseteq E_k \subseteq F_k$. We can shorten the chain by admitting only pairs where $E_j \neq F_j$, without altering the set of accepting runs. Then we are left with a strictly increasing chain of sets, and thus have defined an ω -automaton which is presentable as a parity automaton.

Altogether we obtain the following result:

Theorem 1.22. *By Transformation 1.20, a deterministic Muller automaton with n states is transformed into a deterministic Rabin automaton with $n \cdot n!$ states and n accepting pairs, and also into a deterministic parity automaton with $n \cdot n!$ states and $2n$ colours.*

Transformation 1.20 is given here for deterministic automata, but it works analogously for nondeterministic automata.

In order to cover also Streett automata it is useful to look at the complementation of ω -languages. Note that the negation of the Rabin acceptance condition

$$(*) \quad \exists(E, F) \in \Omega \quad (\text{Inf}(\varrho) \cap E = \emptyset) \wedge (\text{Inf}(\varrho) \cap F \neq \emptyset).$$

is equivalent to the Streett condition:

$$(**) \quad \forall(E, F) \in \Omega \quad (\text{Inf}(\varrho) \cap E \neq \emptyset) \vee (\text{Inf}(\varrho) \cap F = \emptyset)$$

Hence, when we transform a deterministic Rabin automaton recognizing L into a Streett automaton by keeping all its components, including the acceptance component, but using it in the form $(**)$ instead of $(*)$, then the resulting Streett automaton recognizes the complement of L .

We can transform a deterministic Rabin automaton into an equivalent Streett automaton by a detour through Muller automata. Namely, the complement of an ω -language recognized by a deterministic Muller automaton is accepted by the same automaton up to the set of designated state sets; this set \mathcal{F} has to be replaced by its complement w.r.t. the set of states Q of the automaton.

Transformation 1.23. Let $\mathcal{A} = (Q, \Sigma, \delta, q_I, \mathcal{F})$ be a deterministic Muller automaton. Then the Muller automaton $\mathcal{A}' := (Q, \Sigma, \delta, q_I, 2^Q \setminus \mathcal{F})$ recognizes the complement of $L(\mathcal{A})$.

Now we can transform a deterministic Rabin automaton \mathcal{A} into a deterministic Streett automaton as follows: From \mathcal{A} construct an equivalent Muller automaton, by copying Transformation 1.15 for the deterministic case. Complement the Muller automaton, and then apply Transformation 1.20 to obtain a Rabin automaton \mathcal{A}' recognizing the complement of L . Used as a Streett automaton, \mathcal{A}' recognizes L , as desired.

The converse transformation from Streett to Rabin automata works analogously.

As a consequence of the previous constructions we note the following:

Theorem 1.24. *Deterministic Muller automata, Rabin automata, Streett automata, and parity automata recognize the same ω -languages, and the class of ω -languages recognized by any of these types of ω -automata is closed under complementation.*

In this result, the complementation of parity automata would work as follows: Write the parity condition as a Rabin condition, define the complement by reading it as a Streett condition, pass to an equivalent Muller automaton, and obtain from it an equivalent Rabin automaton by Transformation 1.20. This is simplified considerably by the direct approach, which applies the idea of exchanging even and odd colours.

For showing that the complement of a language accepted by an ω -automaton with parity condition is also acceptable by a parity automaton, the colour function has to be modified such that henceforth every word previously not accepted has even parity in its minimal colour value and uneven parity for all previously accepted words.

Transformation 1.25. Let $\mathcal{A} = (Q, \Sigma, \delta, q_I, c)$ be a deterministic ω -automaton with parity condition. Then the complement of $L(\mathcal{A})$ is recognized by the parity automaton $\mathcal{A}' := (Q, \Sigma, \delta, q_I, c')$ where $c'(q) = c(q) + 1$.

So the complementation process is easy (and does not affect the number of states of the automata) if we deal with deterministic Muller or parity automata. For Rabin and Streett automata, the constructions above involve a blow-up of $2^{O(n \log n)}$ (the growth-rate of $n \cdot n!$ as it appears in the LAR construction of Transformation 1.20). The same applies to the transformation of Rabin into Streett automata and conversely. In the next section we will see that this blow-up is not avoidable.

Before turning to these lower bound results, we note a fact about accepting runs of Rabin and Streett automata which will be used there.

Lemma 1.26. *Let $\mathcal{A} = (Q, \Sigma, \delta, q_I, \Omega)$ be an ω -automaton with Rabin condition, and assume ϱ_1, ϱ_2 are two nonaccepting runs. Then any run ϱ with $\text{Inf}(\varrho) = \text{Inf}(\varrho_1) \cup \text{Inf}(\varrho_2)$ is also non-accepting.*

For the proof assume that ϱ_1, ϱ_2 are non-accepting but ϱ with $\text{Inf}(\varrho) = \text{Inf}(\varrho_1) \cup \text{Inf}(\varrho_2)$ is accepting. Then for some accepting pair (E, F) we have $\text{Inf}(\varrho) \cap E = \emptyset$ and $\text{Inf}(\varrho) \cap F \neq \emptyset$. By $\text{Inf}(\varrho) = \text{Inf}(\varrho_1) \cup \text{Inf}(\varrho_2)$ we must have $\text{Inf}(\varrho_1) \cap E = \text{Inf}(\varrho_2) \cap E = \emptyset$, and also $\text{Inf}(\varrho_1) \cap F \neq \emptyset$ or $\text{Inf}(\varrho_2) \cap F \neq \emptyset$. So one of the two runs ϱ_i would be accepting, contradicting the assumption.

By duality, we obtain the following:

Lemma 1.27. *Let $\mathcal{A} = (Q, \Sigma, \delta, q_I, \Omega)$ be a Streett automaton, and assume ϱ_1, ϱ_2 are two accepting runs. Then any run ϱ with $\text{Inf}(\varrho) = \text{Inf}(\varrho_1) \cup \text{Inf}(\varrho_2)$ is also accepting.*

1.5 Two Lower Bounds

In this section we establish two lower bounds of rate $2^{O(n \log n)}$ for the transformation of ω -automata:

- (1) from nondeterministic Büchi automata to deterministic Rabin automata,
- (2) from deterministic Streett to deterministic Rabin automata.

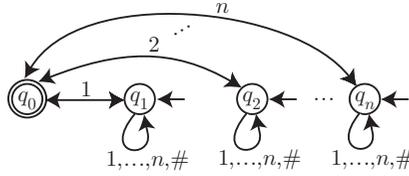


Fig. 1.6. Nondeterministic Büchi automaton \mathcal{A}_n

The first lower bound will be useful in Chapter 3, where a transformation from Büchi automata to deterministic Rabin automata is presented, using the construction of Safra [158]. The lower bound will show that Safra’s construction is optimal.

The second lower bound is of interest in connection with the conversion of Streett automata into Rabin automata (or conversely) presented above. The lower bound result will be taken up again in Chapter 5, where Streett automata are studied in more depth.

1.5.1 From Büchi Acceptance to Rabin Acceptance

The proof idea of the present section is due to Michel [128]. We follow the presentation as given by Löding in [114].

In order to keep the representation of nondeterministic automata small, a set of initial states is used in the examples that follow. It is obvious that the automata can be presented in the usual format by adding just one state and adding arcs from this new state for each arc leaving an initial state of the given automaton.

Example 1.28. Consider the family of Büchi automata from Figure 1.6. This family of automata $(\mathcal{A}_n)_{n \geq 2}$ is defined over the alphabets $\{1, \dots, n, \#\}$ respectively. (The constraint $n \geq 2$ is introduced for the proof of Lemma 1.29 where two different permutations of symbols from $\{1, \dots, n\}$ are assumed to exist.)

The languages L_n accepted by these automata can be characterised by the condition: A word α is accepted by \mathcal{A}_n iff there exists k and $i_1, \dots, i_k \in \{1, \dots, n\}$ such that each pair $i_j i_{j+1}$ for $j < k$ and $i_k i_1$ appears infinitely often in α .

We encode the symbols $1, \dots, n$ by words over $\{0, 1\}^*$ such that

$$i \text{ is encoded by } \begin{cases} 0^i 1 & \text{if } i < n, \\ 0^i 0^* 1 & \text{if } i = n \end{cases}$$

furthermore we keep $\#$ unaltered. Now we can specify the same family of languages w.r.t. the encoding by the family of automata $(\mathcal{A}'_n)_{n \geq 2}$ over the fixed alphabet $\{0, 1, \#\}$. The size of \mathcal{A}_n (in either of the two versions) is $\mathcal{O}(n)$.

The family of automata from the previous example can be used to prove the following lemma.

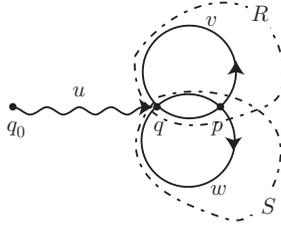


Fig. 1.7. An accepting run of \mathcal{A}'

Lemma 1.29. *There exists a family of languages $(L_n)_{n \geq 2}$ over the alphabet $\{0, 1, \#\}$ recognizable by nondeterministic Büchi automata of size $O(n)$ such that any nondeterministic Streett automaton accepting the complement language of L_n has at least $n!$ states.*

Proof ([114]). Let $n \in \omega$ and $(i_1, \dots, i_n), (j_1, \dots, j_n)$ be different permutations of $\{1, \dots, n\}$. It is clear from the definition of the Büchi automaton \mathcal{A}_n from the previous example that the words $\alpha := (i_1 \dots i_n \#)^\omega$ and $\beta := (j_1 \dots j_n \#)^\omega$ are not accepted by \mathcal{A}_n . Hence, α and β belong to the complement language $L' := \{1, \dots, n, \#\}^\omega \setminus L(\mathcal{A}_n)$.

This means that for any Streett automaton \mathcal{A}' accepting $L(\mathcal{A}') = L'$ there have to exist accepting runs ϱ_α and ϱ_β with $R := \text{Inf} \varrho_\alpha$ and $S := \text{Inf} \varrho_\beta$. Due to the Streett condition of \mathcal{A}' it is sufficient to show that $R \cap S = \emptyset$, as there are $n!$ permutations of $\{1, \dots, n\}$, thus, leading to an automaton with no less than $n!$ states.

Now, assume on the contrary that there is some state $q \in R \cap S$. Then there has to exist an accepting run ϱ_γ of \mathcal{A}' on a word $\gamma = u(vw)^\omega$ such that u is a subword read on some path from the initial state of \mathcal{A}' to the state q and v and w are words read on paths from q to q cycling only through states from R and S respectively. Suppose the infix v of α is given by $i_0, \dots, i_k \in \{1, \dots, n\}$ and similarly $w = j_0, \dots, j_l \in \{1, \dots, n\}$. This situation is depicted in Figure 1.7.

Since $\alpha \neq \beta$ there has to exist an index in which the two words differ. Let m be the least of such indices, i.e. $\forall x. x < m \rightarrow i_x = j_x$ and $i_m \neq j_m$. But now there have to exist indices $k', l' > m$ such that $j_m = i_{k'}$ and $i_m = j_{l'}$. This leads to a sequence $i_m, \dots, i_{m+1}, \dots, i_{m'-1}, i_{m'} j_m, j_{m+1}, \dots, j_{l'-1}, j_{l'}$ satisfying the characterisation of the words in $L(\mathcal{A}_n)$. So $\gamma \in L(\mathcal{A}_n)$.

We now show that \mathcal{A}' also accepts γ , which contradicts the assumption $L(\mathcal{A}') = \{1, \dots, n, \#\}^\omega \setminus L(\mathcal{A}_n)$. Namely, for the run ρ_γ we know that $\text{Inf}(\rho_\gamma) = \text{Inf}(\rho_\alpha) \cup \text{Inf}(\rho_\beta)$. Hence, by Lemma 1.27, the \mathcal{A}' -run ρ_γ is accepting.

By the duality of Rabin and Streett conditions it is obvious that if there exists an ω -automaton of size less than $n!$ with Rabin condition that accepts L_n then there also exists a deterministic Streett automaton that accepts the complement language $\Sigma_n^\omega \setminus L_n$ with less than $n!$ states. Thus from Lemma 1.29 we conclude the following theorem.

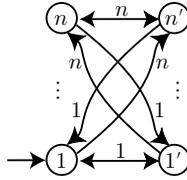


Fig. 1.8. Deterministic Streett automaton \mathcal{A}_n

Theorem 1.30. *There exists a family of languages $(L_n)_{n \geq 2}$ over the alphabet $\{0, 1, \#\}$ recognizable by nondeterministic Büchi automata of size $O(n)$ such that any equivalent deterministic Rabin automaton must be of size $n!$ or larger.*

1.5.2 A Lower Bound for the Transformation of Deterministic Streett Automata to Deterministic Rabin Automata

The technique of latest appearance records is used for the transformation of various automata into parity automata. Two variants are studied in the literature: state appearance records and index appearance records.

State appearance records have been introduced in Section 1.4.2 for the transformation of Muller automata into Rabin automata. Löding [114] shows that any transformation of a deterministic Streett automaton of size n with r pairs of designated sets into a deterministic Rabin automaton will result in an automaton where the number of states is a factorial in $\min(n, r)$, and by the fact that parity automata are special cases of Rabin automata, a transformation to a parity condition will result in an automaton with at least $\min(n, r)!$ states and $O(r)$ colours. Since the automata used in the proof consist of n states and n pairs, this also proves the optimality of the best known transformation from Muller automata to automata with parity condition.

Due to the duality of Rabin and Streett conditions the result is transferrable to the case with Rabin condition and Streett condition interchanged.

Index appearance records (IAR) are used for example by Safra [159] for the transformation of nondeterministic Streett automata into deterministic Rabin automata. The transformation, to be presented in full detail in Chapter 5 below, takes a deterministic Streett automaton of size n with an acceptance condition consisting of r pairs of designated sets to an equivalent deterministic Rabin automaton of size $nO(r)!$ that uses $O(r)$ accepting pairs.

By Theorem 1.32 we obtain the optimality of the IAR construction. The following example gives the family of automata on which the proof is based.

Example 1.31. Consider the family of deterministic Streett automata $(\mathcal{A}_n)_{n \geq 2}$ from Figure 1.8 with pairs of designated state sets $\Omega_n = \{(E_1, F_1), \dots, (E_n, F_n)\}$ and $E_i = \{i\}$, $F_i = \{i'\}$.

The language accepted by the automaton \mathcal{A}_n can be characterised by the symbols occurring in odd and even positions of the accepted words. Each word

α in $L(\mathcal{A}_n)$ satisfies the condition that each symbol occurring infinitely often in an odd position must also occur infinitely often in an even position of α .

This family of automata $(\mathcal{A}_n)_{n \geq 2}$ is defined over the alphabets $\{1, \dots, n\}$, respectively. By encoding the symbols $1, \dots, n$ by words over $\{0, 1\}^*$ such that

$$i \text{ is encoded by } \begin{cases} 0^i 1 & \text{if } i < n, \\ 0^i 0^* 1 & \text{if } i = n \end{cases}$$

we can specify the same family of languages w.r.t. the encoding by the family of automata $(\mathcal{A}'_n)_{n \geq 2}$ over the fixed alphabet $\{0, 1\}$. The construction is similar to that in Section 1.5.1.

Theorem 1.32 ([114]). *There exists a family of languages $(L_n)_{n \geq 2}$ over the alphabet $\{0, 1\}$ recognizable by deterministic Streett automata with $O(n)$ states and $O(n)$ pairs of designated state sets such that any deterministic Rabin automaton accepting L_n requires at least $n!$ states.*

Proof. The idea for proving Theorem 1.32 is motivated by the observation that for any finite word $u \in \{1, \dots, n\}^*$ of even length, the word $u\alpha$ is accepted by \mathcal{A}_n iff α is accepted by \mathcal{A}_n . It can be shown by induction over n that any deterministic Rabin automaton accepting $L(\mathcal{A}_n)$ must have at least $n!$ states.

The base case for the induction is obvious: Any (Rabin) automaton accepting a proper subset of the infinite words over a 2-letter alphabet with some word having occurrences of both letters needs at least two states.

The induction step relies on the fact that any given deterministic Rabin automaton \mathcal{A} accepting $L(\mathcal{A}_n)$ can be modified to a deterministic automaton over $\{1, \dots, n\} \setminus \{i\}$ for any $i \in \{1, \dots, n\}$ by simply removing all arcs labelled by i . Setting the initial state of the modified automaton to any q that is reachable in \mathcal{A}_n by an even number of state transitions we obtain a deterministic Rabin automaton \mathcal{A}_i^q .

Because of the characterisation of $L(\mathcal{A}_n)$ given above, it is clear that \mathcal{A}_i^q accepts a language isomorphic up to the renaming of symbols to $L(\mathcal{A}_{n-1})$. The induction hypothesis requires the automaton \mathcal{A}_i^q to have at least $(n-1)!$ states.

For a complete proof the reader is referred to [114].

1.6 Weak Acceptance Conditions

In the previous sections we have defined acceptance by a reference to those states in a run which occur infinitely often. For some purposes a “weak acceptance condition” is appropriate. This is a condition on the set of states that occur at least once (but maybe only finitely often) in a run.

Recall that

$$\text{Occ}(\varrho) := \{q \in Q \mid |\varrho^{-1}(q)| \geq 1\}$$

is the set of states that occur at least once in the run ϱ . Let $\mathcal{A} = (Q, \Sigma, \delta, q_I, \text{Acc})$ be an ω -automaton.

There are different possibilities to use the set $\text{Occ}(\varrho)$ for acceptance. The analogue to the Muller condition, introduced by Staiger and Wagner [166], uses a family \mathcal{F} of state sets and declares the run ϱ accepting if

$$\text{Occ}(\varrho) \in \mathcal{F}.$$

Other acceptance modes refer to a set F of designated states and require

$$\text{Occ}(\varrho) \cap F \neq \emptyset,$$

(also called 1-acceptance, following [110]), and

$$\text{Occ}(\varrho) \subseteq F,$$

also called 1'-acceptance.

These acceptance modes are special cases of Staiger-Wagner acceptance. In the first case one collects in \mathcal{F} all sets X with $X \cap F \neq \emptyset$, in the second case the sets X with $X \subseteq F$.

Example 1.33. To accept the ω -words over the alphabet $\{a, b\}$ that have at least one symbol a , we take an automaton $\mathcal{A} = (\{q_a, q_b\}, \{a, b\}, \delta, q_b, F)$, where $F = \{q_a\}$, δ is defined according to the state transition graph of Figure 1.2, and 1-acceptance is used.

The requirement that only the word b^ω should be accepted can be specified with the same transition graph, now with 1'-acceptance using the set $F = \{q_b\}$. i.e. the only state that may be visited in any successful run is q_b .

In later chapters of the book also the parity condition will be used in the weak sense. The requirement for acceptance is that the minimal (or maximal) colour occurring in a run is even.

We show that acceptance by an occurrence set can be simulated by Büchi acceptance. The idea is to simulate \mathcal{A} and to accumulate the visited states in a separate component of the state, signalling acceptance whenever this component is a set from \mathcal{F} .

Transformation 1.34. Let $\mathcal{A} = (Q, \Sigma, \delta, q_I, \mathcal{F})$. The language $L(\mathcal{A})$ recognized by \mathcal{A} with the Staiger-Wagner acceptance condition is recognized by a Büchi automaton $\mathcal{A}' = (Q \times 2^Q, \Sigma, \delta', (q_I, \{q_I\}), F')$ where $\delta'((p, P), a)$ contains all states (p', P') with $p' \in \delta(p)$ and $P' = P \cup \{p'\}$, and where F' contains all states (p, P) with $P \in \mathcal{F}$.

The exponential blow-up can be avoided if only 1-acceptance or 1'-acceptance are involved. In order to capture 1-acceptance via a set F by Büchi acceptance, one introduces a transition from each F -state to a new state q_f , with a transition back to q_f , which serves as only final state in the Büchi automaton. For 1'-acceptance, it suffices to take the given automaton and use it as a Büchi automaton (with the same set of designated states).

The reverse transformations are not possible; it should be obvious that an infinity condition in the definition of an ω -language cannot in general be simulated

by an occurrence condition. For example, the set L of ω -words over $\{a, b\}$ with infinitely many b is not recognizable by an ω -automaton with Staiger-Wagner acceptance. Assuming such an automaton which recognizes L , say with n states, one would consider an accepting run on the input word $(a^{n+1}b)^\omega$. After some finite prefix, say after $(a^{n+1}b)^k$, the run would have visited the states which are visited at all. In the succeeding block a^{n+1} the automaton assumes a loop, which can be repeated if the input is changed to $(a^{n+1}b)^k a^\omega$. So over this input the same states would be visited as in the considered run over $(a^{n+1}b)^\omega$. Hence $(a^{n+1}b)^k a^\omega$ would be accepted, a contradiction.

1.7 Conclusion

We have shown the expressive equivalence of

- nondeterministic Büchi, Muller, Rabin, Streett, and parity automata
- deterministic Muller, Rabin, Streett, and parity automata

The missing link will be provided in Chapter 3 below: Nondeterministic Büchi automata accept the same ω -languages as deterministic Muller automata.

Figure 1.9 gives an overview; it shows the dependencies and known bounds for transformations between different models (including results that are shown in Part II of the book).

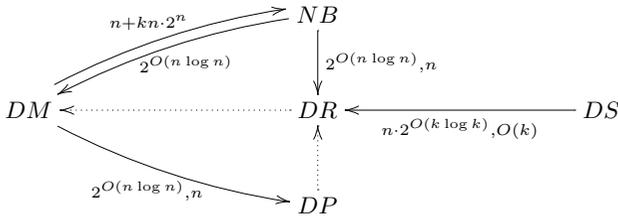


Fig. 1.9. An overview of transformation bounds for ω -automata.

We indicate by D and N the deterministic, respectively, nondeterministic versions and write B, M, R, S for Büchi, Muller, Rabin, Streett, respectively. The noted complexity bounds are given as pairs (n', k') where n' is the size of the constructed automaton and k' the size of the acceptance component, relative to the original sizes (n is the original number of states and k the size of the original acceptance component). Dotted arrows are used for trivial transformations.