- Proof that DBW are less expressive than NBW (p. 3)
- Presentation of Safra's construction (NBW —> DMW)
- Optimality of Safra's construction (proof that the Safra construction produces at most 2^O(n log n) states and that no other Büchi determinisation algorithm can do better)

3 Determinization of Büchi-Automata

Markus Roggenbach

Bremen Institute for Safe Systems
Bremen University

For Bene

Introduction

To determinize Büchi automata it is necessary to switch to another class of ω-automata, e.g. Muller or Rabin automata. The reason is that there exist languages which are accepted by some nondeterministic Büchi-automaton, but not by any deterministic Büchi-automaton (c.f. section 3.1).

The history of constructions for determinizing Büchi automata is long: it starts in 1963 with a faulty construction [133]. In 1966 McNaughton showed, that a Büchi automaton can be transformed effectively into an equivalent deterministic Muller automaton [125]. Safra's construction [158] of 1988 leads to deterministic Rabin or Muller automata (c.f. section 3.2): given a nondeterministic Büchi automaton with n states, the equivalent deterministic automaton has $2^{O(n \log n)}$ states. For Rabin automata, Safra's construction is optimal (c.f. section 3.3). The question whether it can be improved for Muller automata is open. Safra's construction is often felt to be difficult. Thus, in 1995 Muller and Schupp [137] presented a 'more intuitive' alternative, which is also optimal for Rabin automata.

Although Safra's construction is optimal for Rabin automata, the resulting automata often contain equivalent states, which can be eliminated. An example for this effect is presented in Exercise3.6. As it is completely open how to minimize ω-automata, it would be quite interesting to develop procedures for 'fine tuning' Safra's construction. Some ideas in this direction can be found e.g. in [153].

Considering the languages recognizable by different classes of automata we obtain the following picture:

nondeterministic Büchi	$\Leftrightarrow$	nondeterministic Muller	$\Leftrightarrow$	nondeterministic Rabin
$\Uparrow$		$\Updownarrow$		$\Updownarrow$
deterministic Büchi	$\Rightarrow$	deterministic Muller	$\Leftrightarrow$	deterministic Rabin

These relations hold thanks to the following results:

- Obviously the deterministic variant of a class of automata is included in the nondeterministic variant of this class.
- Theorem 3.2 shows that the inclusion of the deterministic variant in the nondeterministic variant is strict for Büchi automata.

E. Grädel et al. (Eds.): Automata, Logics, and Infinite Games, LNCS 2500, pp. 43-60, 2002.

- Safra's construction implies that the class of nondeterministic Büchi automata is included in the class of deterministic Muller automata as well as in the class of deterministic Rabin automata (Theorem 3.6).
- Section 1.3.2 describes how to transform a nondeterministic Muller automaton into an equivalent nondeterministic Büchi automaton.
- Transformation 1.15 of section 1.3.3 constructs an equivalent nondeterministic Muller automaton from a given nondeterministic Rabin automaton.

The above picture also shows how determinization can be used for complementation: given a nondeterministic Büchi automaton accepting a language L, use Safra's construction to obtain an equivalent deterministic Muller automaton with state set Q and system $\mathcal{F}$ of final state sets. With $2^Q \backslash \mathcal{F}$ as system of final state sets, this automaton accepts the complement of L. Applying the construction of section 1.3.2 results in a Büchi automaton for the complement of L.

Another application of Safra's construction can be found in Klarlund, Mukund and Sohoni [99]: they generalize the construction to asynchronous finite automata accepting infinite Mazurkiewicz traces.

This chapter is organized as follows: section 3.1 shows that the inclusion of the deterministic variant in the nondeterministic variant is strict for Büchi automata. Then Safra's construction is discussed and proven to be correct in section 3.2. Finally, section 3.3 deals with the optimality of Safra's construction.

3.1 Deterministic versus Nondeterministic Büchi-Automata

Safra's construction is a refinement of the classical powerset construction as used in the determinization of automata over finite words (c.f. [87]): given an automaton with states Q, the powerset construction uses sets of states from Q – which we call *macrostates* here – as states of the desired deterministic automaton. In order to understand Safra's modifications, it is quite instructive to study why the original construction fails for automata over infinite words.

Example 3.1. Consider the Büchi automaton $\mathcal{A} = (\{q_\mathrm{I}, f\}, \{a, b\}, \Delta, q_\mathrm{I}, \{f\})$, where

$$\Delta = \{(q_\mathrm{I}, a, q_\mathrm{I}), (q_\mathrm{I}, b, q_\mathrm{I}), (q_\mathrm{I}, a, f), (f, a, f)\}$$

(c.f. Figure 3.1[1]). This automaton $\mathcal{A}$ accepts the language

$$L := \{\alpha \in \{a, b\}^\omega \mid \sharp_b(\alpha) < \infty\},$$

where $\sharp_b(\alpha)$ denotes the number of 'b's occurring in word α.

The powerset construction for finite automata leads to the deterministic automaton shown in Figure 3.2. The only reasonable Büchi condition would be

[1] We follow the convention to depict the initial state by an incoming arc without source, and recurring states of a Büchi automaton by double circles.

$F = \{\{q_I, f\}\}$, which would also accept the word $(ab)^\omega \notin L$. The problem with the corresponding run $\varrho = \{q_I\}\{q_I, f\}\{q_I\}\{q_I, f\}\ldots$ is that – although there are infinitely many macrostates containing f – we cannot extract a run of $\mathcal{A}$ from ϱ exhibiting infinitely many f: at any time when we could choose f from a macrostate in ϱ, there is no transition with label b available in $\mathcal{A}$.

Note that with

- $\mathcal{F} = \{\{\{q_I, f\}\}\}$ as Muller condition, and with
- $(\{\{q_I\}\}, \{\{q_I, f\}\})$ as Rabin condition,

the automaton of Figure 3.2 accepts the desired language L. □

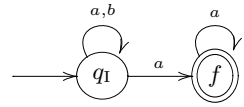

Fig. 3.1. A nondeterministic Büchi automaton.

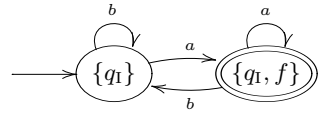

Fig. 3.2. A deterministic Büchi automaton.

It is no accident that in the above example the powerset construction fails in case of Büchi automata. The considered language cannot be accepted by any deterministic Büchi automaton:

Theorem 3.2 (Deterministic versus Nondeterministic Büchi Automata)**.** *There exist languages which are accepted by some nondeterministic Büchi-automaton but not by any deterministic Büchi-automaton.*

only finite number of b

Proof. Let $\Sigma := \{a, b\}$. Consider again the language $L := \{\alpha \in \Sigma^\omega \mid \sharp_b(\alpha) < \infty\}$. As shown in the above example, L is accepted by a *nondeterministic* Büchi automaton.

Assume that there is a *deterministic* Büchi automaton $\mathcal{A} = (Q, \Sigma, \delta, q_I, F)$ accepting L. This automaton accepts all words of the form σa^ω, where $\sigma \in \Sigma^*$.

Consider a reachable state $q \in Q$. Any finite word σ_q leading in $\mathcal{A}$ from the initial state q_I to q can be extended to an infinite word $\sigma_q a^\omega \in L$, i.e. some state $f \in F$ occurs infinitely often in its run on $\mathcal{A}$. Thus there must be a *finite* sequence of a-transitions from q to a recurring state.

Let m be the maximal number of a-transitions which are needed in $\mathcal{A}$ to get from a reachable state to a recurring state. Then $\mathcal{A}$ also accepts the word $\alpha = (ba^m)^\omega \notin L$: As $\mathcal{A}$ is deterministic, there exists a run ϱ of $\mathcal{A}$ on $(ba^m)^\omega$. By construction of α, the automaton $\mathcal{A}$ reaches, after each 'b', a recurring state within the next m a-transitions. Thus ϱ includes infinitely many occurrences of recurring states, and α is accepted. □

3.2 Safra's Construction

The results of the previous section demonstrate that it is indeed necessary to switch to another class of ω-automata in order to find an equivalent deterministic automaton for a given Büchi automaton. But example 3.1 leaves open the question whether the powerset construction might be sufficient to obtain an equivalent Rabin or Muller automaton. This is not the case for Rabin automata:

Example 3.3. Consider the Büchi automaton $\mathcal{A} = (\{q_\mathrm{I}, q_1, q_\sharp\}, \{1, \sharp\}, \Delta, q_\mathrm{I}, \{q_I\})$ of Figure 3.3. □

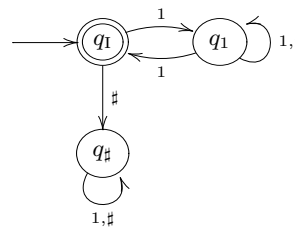

Fig. 3.3. A nondeterministic Büchi automaton.

Exercise 3.1. Apply the powerset construction to the nondeterministic Büchi automaton of Figure 3.3. Prove that there exists no Rabin condition, that allows to accept the same language with the resulting automaton.

Hint: Assume that there exists such a Rabin condition and consider the pair (E, F) which is necessary to accept the word $1(11\sharp)^\omega$.

The weakness of the powerset construction is that the resulting automaton allows for too many 'accepting' runs: given a run of this automaton with infinitely many macrostates containing a recurring state it might be impossible to 'extract' a sequence of states out of this run forming an accepting run of the original nondeterministic Büchi automaton. Safra's key idea is to modify the classical powerset construction in such a way that it allows for such an operation.

3.2.1 Safra's Tricks

Before presenting the construction in detail, we discuss Safra's tricks for extending the powerset construction in an informal way:

state of the output automaton

Trick 1: *Initialize new runs of macrostates starting from recurring states.*
Whenever recurring states occur in a macrostate, say M, the successor state of M for some input $a \in \Sigma$ gets an extra component. This component consists of all states which can be reached under a from $F \cap M$, c.f. Figure 3.4. This leads to new concept of states replacing the old macrostates by 'sets' of macrostates. The modified powerset construction can be applied componentwise on these sets of macrostates. This trick corresponds to Step 2 in Safra's construction, c.f. section 3.2.3.
This idea has the effect that every state included in an extra component has – in the original nondeterministic Büchi automaton – a recurring state as predecessor. Using this information in a clever way allows for constructing an accepting run on the nondeterministic Büchi automaton from an accepting run of the automaton obtained by the improved powerset construction, c.f. Lemma 3.9. See Exercise3.2 for an example illustrating this trick.
Safra organizes these sets of macrostates as ordered trees with macrostates as labels, the so-called Safra trees. Applying Trick 1 to a leaf gives rise to a son, which increases the *height* of a Safra tree. Applying Trick 1 to a node, which already has a son, gives rise to a younger brother of this son, which might increase the *width* of a Safra tree. In order to obtain a *finite* set of states in the automaton to be constructed this growth in height and width has to be restricted: Trick 2 does so for width, while Trick 3 controls the height.

Exercise 3.2. Apply Trick 1 on the Büchi automaton of example 3.1. How does it prevent the run of $(ab)^\omega$ to be accepting?

Keep the width finite

Trick 2: *Keep track of joining runs of the nondeterministic Büchi automaton just once.*
To illustrate this trick we consider two finite runs

$$q_1 q_2 \dots f q_i \dots q_{j-1} q_j \dots q_n q_{n+1} \text{ and } q_1 q'_2 \dots q'_{i-1} q'_i \dots f' q'_j \dots q'_n q_{n+1}$$

of a nondeterministic Büchi automaton on the same finite word $a_1 \dots a_n$, where both runs start in state q_1, end in state q_{n+1}, and visit recurring states f and f', respectively. Figure 3.5 shows, how Trick 1 leads to extra components: the recurring states f and f' give rise to components $\{q_i\}$ and $\{q'_j\}$, respectively. As both runs join in state q_{n+1}, the extra components of the last state are identical and hold the same 'information': state q_{n+1} has – in the original nondeterministic Büchi automaton – a recurring state as predecessor. As there is no need to store this information twice, the second component can be removed. On Safra trees this operation is called 'horizontal merge'. It corresponds to Step 4 in Safra's construction, c.f. section 3.2.3.

Keep the height finite

Trick 3: *If all states in a macrostate have a recurring state as predecessor, delete the corresponding components.*
Figure 3.6 illustrates this situation: starting in a macrostate M, a finite run leads via Trick 1 to a situation where we have a macrostate M' and extra components $K'_1, \ldots, K'_k$. If

$$M' = K'_1 \cup \cdots \cup K'_k,$$

then all states collected in M' have a recurring state as predecessor. Encoding this situation by marking macrostate M' with a special sign, say '!', all extra components can be removed. On Safra trees this operation is called 'vertical merge'. It corresponds to step Step 6 in Safra's construction, c.f. section 3.2.3.

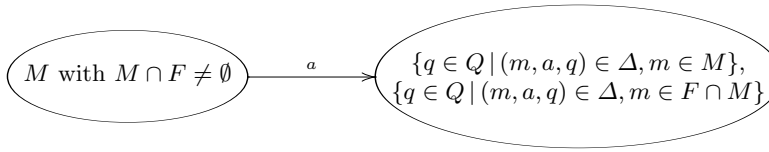

Fig. 3.4. Illustration for Trick 1.

3.2.2 Safra Trees

A Safra tree represents one state of the output automaton

Given some fixed set of states Q, Safra trees are ordered, directed trees over some vocabulary V of node names, where the nodes have *nonempty* macrostates, i.e. subsets of Q, as labels. Additionally to its macrostate, a node can be marked with the special symbol '!'. Safra trees satisfy the following two conditions:

Condition 1: The union of brother macrostates is a proper subset of their parent macrostate.
Condition 2: Brother macrostates are disjoint.

This has as a consequence that the number of nodes in a Safra tree is bounded by $|Q|$. We prove this claim by induction on the height of Safra trees over Q : For the empty tree with height 0, and also for a tree with height 1, which consists just of a root node, the claim holds trivially. In the induction step observe that the sons of the root define Safra trees of lower height over disjoint subsets Q_i of states (Condition 2). Thus, by induction hypothesis, the number of nodes in the whole tree is bounded by $(\sum_i |Q_i|) + 1$. By Condition 1 we have $\sum_i |Q_i| < |Q|$, and we finally obtain $(\sum_i |Q_i|) + 1 \leq |Q|$.

Interpreting this result in terms of height and branching of a Safra tree we obtain:

- The height of a Safra tree is at most $|Q|$.
- Safra trees are finitely branching, a node has at most $|Q| - 1$ sons.

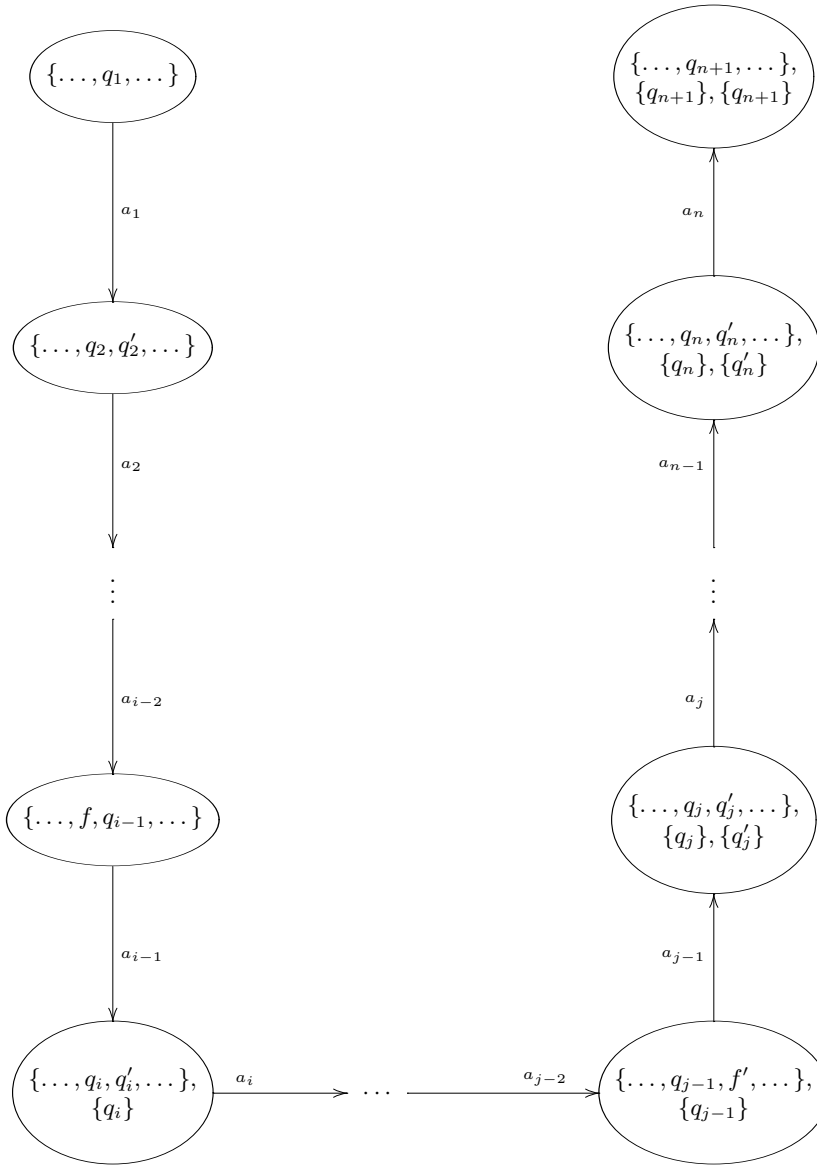

Fig. 3.5. Illustration for Trick 2.

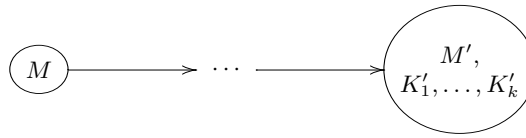

Fig. 3.6. Illustration for Trick 3.

Safra's construction builds the output automaton in a similar way as the subset construction: starting with an initial state and then adding successor states. In the end choosing an acceptance condition. —> Safra's construction is a modified subset construction

3.2.3 The Construction

Let $\mathcal{B} = (Q, \Sigma, q_I, \Delta, F)$ be a nondeterministic Büchi automaton. Safra's construction yields a Safra tree as an initial state q'_I, a set of Safra trees as set of states Q', and a transition function $\delta : Q' \times \Sigma \to Q'$ for the alphabet Σ. To complete the construction a suitable accepting component has to be chosen: either a system of final state sets $\mathcal{F}$ to obtain a deterministic Muller automaton $\mathcal{M} = (Q', \Sigma, q'_I, \delta, \mathcal{F})$, or a set of accepting pairs $\Omega = \{(E_1, F_1), \ldots, (E_k, F_k)\}$ to obtain a Rabin automaton $\mathcal{R} = (Q', \Sigma, q'_I, \delta, \Omega)$, that both accept the same language as the original nondeterministic Büchi automaton $\mathcal{B}$.

Choose $V := \{1, 2, \ldots, 2\,|Q|\}$ as vocabulary for denoting nodes of Safra trees. This is sufficient, as the number of nodes in Safra trees is bounded by $|Q| = n$, and the computation of a successor of a Safra tree introduces at most n new nodes at intermediate stages (c.f. Step 2).

(1) The *initial state* q'_I is the Safra tree consisting of the single node 1 labelled with macrostate $\{q_I\}$.
(2) The value of the *transition function* $\delta(T, a)$ for a given input $a \in \Sigma$ and a Safra tree T with a set N of nodes is computed as follows:
   Step 1: Remove all marks '!' in the Safra tree T.
   Step 2: For every node v with macrostate M such that $M \cap F \neq \emptyset$, create a new node $v' \in (V \backslash N)$, such that v' becomes the youngest son of v and carries the macrostate $M \cap F$.
   Step 3: Apply the powerset construction on every node v, i.e. replace its macrostate M by $\{q \in Q \mid \exists (m, a, q) \in \Delta : m \in M\}$.
   Step 4 (horizontal merge): For every node v with macrostate M and state $q \in M$, such that q also belongs to an older brother of v, remove q from M.
   Step 5: Remove all nodes with empty macrostates.
   Step 6 (vertical merge): For every node whose label is equal to the union of the labels of its sons, remove all the descendants of v and mark v with '!'.
(3) The set of *states* Q' consists of all reachable Safra trees.

A *Muller automaton* is obtained by choosing the acceptance component as follows: A set $S \subseteq Q'$ of Safra trees is in the system $\mathcal{F}$ of final state sets if for some node $v \in V$ the following holds:

Muller 1: v appears in all Safra trees of S, and
Muller 2: v is marked at least once in S.

To obtain a *Rabin automaton,* one takes all pairs $(E_v, F_v), v \in V$, as acceptance component, where

Rabin 1: E_v consists of all Safra trees without a node v, and
Rabin 2: F_v consists of all Safra trees with node v marked '!'.

We should check first, that – given a Safra tree T and an input symbol a – the transition function δ computes indeed a Safra tree. This ensures that Q' consists of Safra trees, as the initial state q'_{I} is obviously a Safra tree.

Removing the marks '!' from all nodes of a Safra tree T does not violate Condition 1 or Condition 2, and as all macrostates are nonempty in T, they are also nonempty after Step 1. Thus Step 1 preserves the Safra tree properties.

Applying Step 2 on a Safra tree T with a node v carrying a macrostate $M \subseteq F$, yields a tree violating Condition 1, as v and its youngest son carry afterwards the same label M. Computing new macrostates for all nodes of a tree in Step 3 may lead to even more trouble:

(1) Afterwards, brother macrostates might share a state $q \in Q$, violating Condition 2.
(2) The new computed macrostate can be the empty set.
(3) It might also happen, that Condition 1 is violated, i.e. the union of brother macrostates equals the parent macrostate. This happens for example if in Step 2 a node carries a macrostate $M \subseteq F$.

Step 4, Step 5, and Step 6 deal with these problems, resp.: Step 4 ensures Condition 2 by horizontal merge of brother macrostates. Step 5 removes nodes with empty macrostates. By vertical merge Step 6 fixes situations where Condition 1 is violated. Thus, we finally obtain after all six steps a Safra tree.

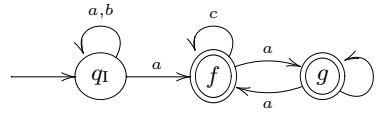

Fig. 3.7. A nondeterministic Büchi automaton.

Example 3.4 *(Applying Safra's construction).*

We apply Safra's construction to the nondeterministic Büchi automaton shown in Figure 3.7. Figure 3.8 and Figure 3.9 give some examples how to compute the transition function δ: they present the resulting tree after executing a certain step. If a step alters the tree, its name is typed bold. Note that the empty tree ϵ may arise as a result of Safra's construction. The resulting automaton is depicted in Figure 3.10.

To obtain a Rabin automaton, we choose – according to the above described construction – two accepting pairs (E_1, F_1) and (E_2, F_2), where

- $E_1 = \{\epsilon\}$, $F_1 = \{1 - \{f\}-!, 1 - \{g\}-!, 1 - \{f, g\}-!\}$, and
- $E_2 = \{1-\{q_{\mathrm{I}}\}, \epsilon, 1-\{q_{\mathrm{I}}, f\}, 1-\{f\}-!, 1-\{g\}-!\}$, $F_2 = \left\{ \begin{array}{c} 1 - \{q_{\mathrm{I}}, f, g\} \\ \downarrow \\ 2 - \{g, f\}-! \end{array} \right\}$.

Computing $\delta(1 - \{q_I\}, a)$: {q1} has successors q1 and f on c

Step 1	Step 2	**Step 3**
$1 - \{q_I\}$	$1 - \{q_I\}$	$1 - \{q_I, f\}$

each final form of a Safra tree is a state of the output automaton

Computing $\delta(1 - \{q_I\}, c)$: {q1} has no successors on c

Step 1	Step 2	**Step 3**	Step 4	**Step 5**
$1 - \{q_I\}$	$1 - \{q_I\}$	$1 - \emptyset$	$1 - \emptyset$	ϵ

Computing $\delta(1 - \{q_I, f\}, c)$: {q1} has successor f on c

Step 1	**Step 2**	**Step 3**	Step 4	Step 5	**Step 6**
$1 - \{q_I, f\}$	$1 - \{q_I, f\}$ ↓ $2 - \{f\}$	$1 - \{f\}$ ↓ $2 - \{f\}$	$1 - \{f\}$ ↓ $2 - \{f\}$	$1 - \{f\}$ ↓ $2 - \{f\}$	$1 - \{f\}$ - !

Fig. 3.8. Steps from Safra's Construction.

Note that we need indeed a 'true' Rabin condition: While it is possible to choose E_1 as the empty set, this is not the case for $E_2 : E_2 = \emptyset$ allows to accept the word $(aaab)^\omega \notin L$.

The construction and discussion of the Muller condition is left as Exercise 3.4. □

As the above example indicates, the constructed Rabin and Muller conditions are not 'minimal'. For Muller conditions the following restriction might lead to a smaller a system of final state sets (the proof is left for Exercise 3.5):

Remark 3.5 *(Refinement of the Muller Condition).*

Restricting the system of final state sets $\mathcal{F}$ obtained by conditions Muller 1 and Muller 2 to those sets which form a strongly connected component in the automaton leads to an equivalent Muller automaton. □

Exercise 3.3. Apply Safra's construction to the nondeterministic Büchi automaton of Figure 3.3.

Computing $\delta(1-\{q_I, f, g\}, a)$: {q1, f, g} has successors {q1, f, g} on a

$\downarrow$

$2-\{g, f\}$-!

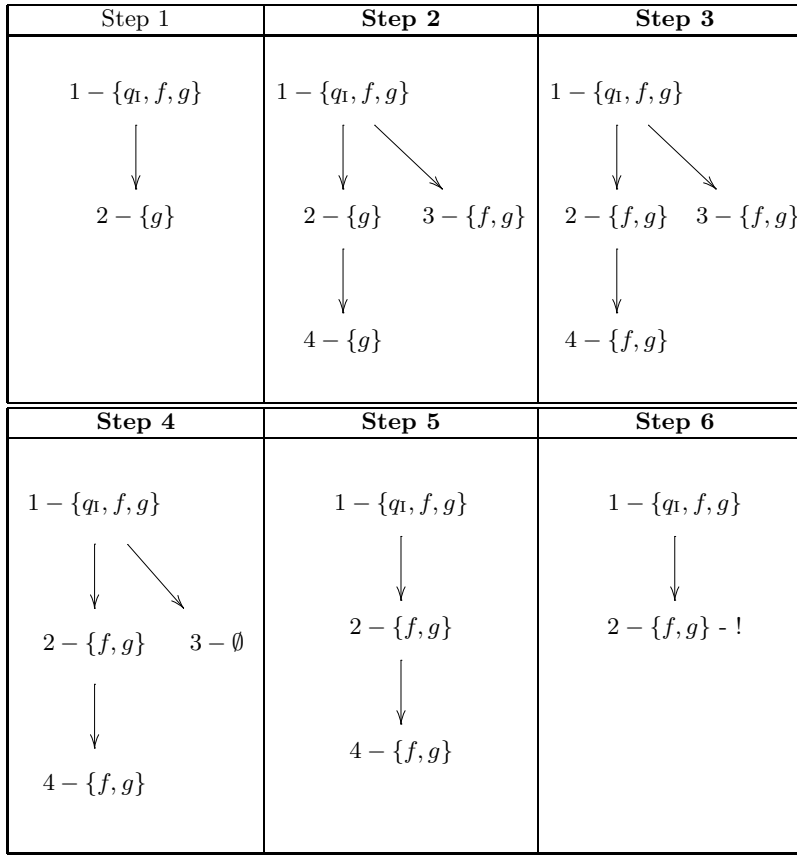

Fig. 3.9. Some Steps from Safra's Construction.

Exercise 3.4. Consider the nondeterministic Büchi automaton shown in Figure 3.7. Figure 3.10 shows the result of Safra's construction. Compute a system of final state sets $\mathcal{F}$ to obtain a deterministic Muller automaton $\mathcal{M} = (Q', \Sigma, q_I', \delta, \mathcal{F})$.

Argue, why this system $\mathcal{F}$ may be restricted to include only sets, which are strongly connected components.

Exercise 3.5. Prove Remark 3.5. Is it possible to generalize this result to arbitrary Muller conditions?

Theorem 3.6 (Correctness). *Let $\mathcal{B} = (Q, \Sigma, q_I, \Delta, F)$ be a nondeterministic Büchi automaton. Let $\mathcal{M} = (Q', \Sigma, q_I', \delta, \mathcal{F})$ be the deterministic Muller and*

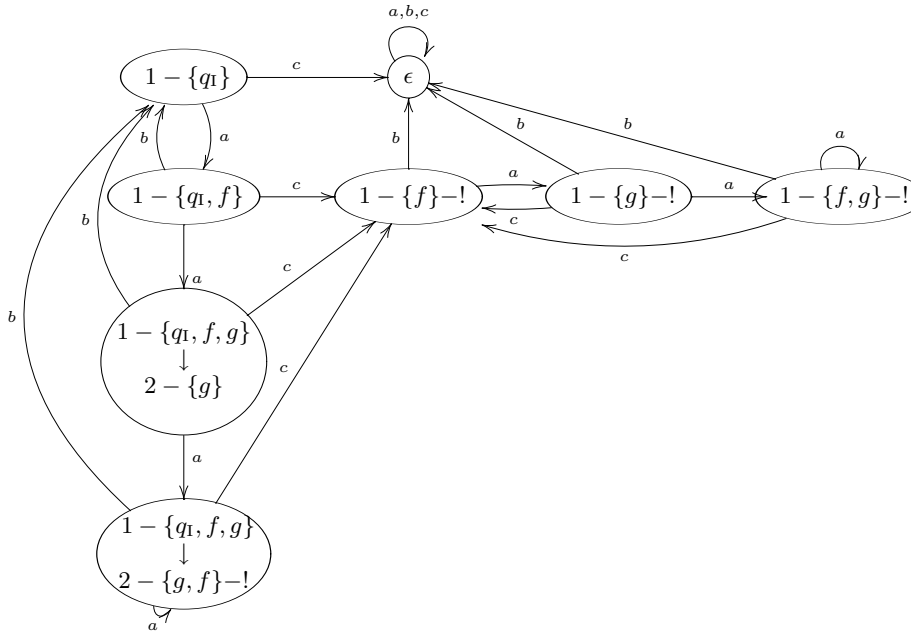

Fig. 3.10. Safra's construction applied on the automaton of Figure 3.7.
A Rabin or Muller acceptance condition can now be added to this automaton

$\mathcal{R} = (Q', \Sigma, q'_{\mathrm{I}}, \delta, \Omega)$ *be the deterministic Rabin automaton obtained by Safra's construction. Then*

$$L(\mathcal{B}) = L(\mathcal{M}) = L(\mathcal{R}).$$

Proof. By Lemma 3.7 and Lemma 3.9. □

Lemma 3.7 (Completeness)**.** *For the nondeterministic Büchi automaton* $\mathcal{B}$*, the deterministic Muller automaton* $\mathcal{M}$*, and the deterministic Rabin automaton* $\mathcal{R}$ *of Theorem 3.6 we have:* output automaton accepts ALL the words it must

$$L(\mathcal{B}) \subseteq L(\mathcal{M}) \text{ and } L(\mathcal{B}) \subseteq L(\mathcal{R}).$$

Proof. Let $\alpha \in L(\mathcal{B})$. As $\mathcal{M}$ and $\mathcal{R}$ are deterministic and have the same initial state and transition relation there exists *one* run ϱ' on α of both automata. We claim that there is at least one node v in the Safra trees of ϱ' such that

Claim 1: v – from a certain point on – is a node of all Safra trees in ϱ' and
Claim 2: v is marked '!' infinitely often.

Concerning the *Muller condition* this proves that $\mathrm{Inf}(\varrho')$ equals a set in the system $\mathcal{F}$ of final states: condition Muller 1 is true because a Safra tree of ϱ' not including v is not in $\mathrm{Inf}(\varrho')$. As v is marked '!' infinitely often in ϱ' and Q' is a finite set, there exists some Safra tree in $\mathrm{Inf}(\varrho')$ with v marked '!'. Therefore also

condition Muller 2 is fulfilled. Thus ϱ' is an accepting run of the deterministic Muller automaton $\mathcal{M}$, and we obtain $\alpha \in L(\mathcal{M})$.

The *Rabin condition* holds trivially for the accepting pair $(E_v, F_v) : \mathrm{Inf}(\varrho') \cap E_v = \emptyset$ is true thanks to Claim 1, Claim 2 implies $\mathrm{Inf}(\varrho') \cap F_v \neq \emptyset$. Thus ϱ' is an accepting run of the deterministic Rabin automaton $\mathcal{R}$, and we obtain $\alpha \in L(\mathcal{R})$.

Claim 1 holds for the root node: As $\alpha \in L(\mathcal{B})$, there exists an accepting run ϱ in the nondeterministic Büchi automaton $\mathcal{B}$. Thus the root of all Safra trees occurring in the run ϱ' is nonempty: the root macrostate of the i-th Safra tree in ϱ' includes $\varrho(i)$ and therefore cannot be removed in Step 5 of the Safra construction. If the root is marked '!' infinitely often, also Claim 2 holds, and we are done.

If the root is not marked '!' infinitely often we need to consider another candidate for v. As the run ϱ of the nondeterministic Büchi automaton $\mathcal{B}$ is accepting, there exists a state $q \in \mathrm{Inf}(\varrho) \cap F$, which occurs infinitely often in the root macrostate of the Safra trees in the run ϱ'. Consider the first occurrence of q in ϱ' after the last occurrence of the mark '!' at the root (if marks existed at all). As $q \in F$, q will be put into the macrostate of the youngest son of the root (Step 2 of Safra's construction). From this point onwards, the states of the run ϱ appear in the macrostate of this son, or (due to the horizontal merge operation in Step 4 of Safra's construction) get associated to older brothers of this son. Such a shift to an older brother can happen only a finite number of times: Due to Condition 2 on Safra trees any node has only finitely many brothers, especially only finitely many older brothers. Step 4 of Safra's construction moves the state of ϱ to an *older* brother, while Step 2 of Safra's construction leads to brothers which are *younger.* Thus eventually the states of the run ϱ remain in some fixed son of the root.

This son is our new candidate for v. It cannot be removed by Step 5 of Safra's construction, as it carries the states of the run ϱ and is therefore nonempty. Furthermore, it cannot be removed by Step 6 of Safra's construction, as the root is no more marked '!'. Thus Claim 1 holds. If this son is marked marked '!' infinitely often, also Claim 2 is true, and we are done.

Otherwise, proceed with this son (in which q occurs infinitely often) in the same way as with the root above. Continuing this way, Claim 2 must hold eventually, since the depth of Safra trees is finite (Condition 1 on Safra trees). □

The following lemma makes use of a result which is known as König's Infinity Lemma (for a proof and further discussion see e.g. [62]).

Theorem 3.8 (König's Infinity Lemma)**.**
An infinite rooted tree which is finitely branching (i.e., where each node has only finitely many sons) has an infinite path.

Lemma 3.9 (Soundness)**.** *For the nondeterministic Büchi automaton $\mathcal{B}$, the deterministic Muller automaton $\mathcal{M}$, and the deterministic Rabin automaton $\mathcal{R}$ of Theorem 3.6 we have:* output automaton accepts ONLY the words it must

$$L(\mathcal{M}) \subseteq L(\mathcal{B}) \text{ and } L(\mathcal{R}) \subseteq L(\mathcal{B}).$$

Proof. Let $\alpha \in L(\mathcal{M})$. Then there exists an accepting run ϱ' of the *deterministic Muller automaton* $\mathcal{M}$ on α, i.e. $\mathrm{Inf}(\varrho') \in \mathcal{F}$. Thus there exists some node v such that

- v appears in all Safra trees of $\mathrm{Inf}(\varrho')$, and
- v is marked at least once in $\mathrm{Inf}(\varrho')$.

This has as consequences that

- v – from a certain point on – is a node of all Safra trees in ϱ', and
- in ϱ' Safra trees T_i occur infinitely often with node v marked '!', i.e. ϱ' has the form

$$q'_{\mathrm{I}} \ldots T_1 \ldots T_2 \ldots T_3 \ldots$$

The same situation is achieved if we consider a word $\alpha \in L(\mathcal{R})$: Then there exists an accepting run ϱ'' of the *deterministic Rabin automaton* $\mathcal{R}$ on α, i.e. there exist a node v and an accepting pair (E_v, F_v) such that $\mathrm{Inf}(\varrho'') \cap E_v = \emptyset$ and $\mathrm{Inf}(\varrho'') \cap F_v \neq \emptyset$. By construction E_v consists of all Safra trees without node v (Rabin 1), i.e. v – from a certain point on – is a node of all Safra trees in ϱ''. As F_v consists of all Safra trees with node v marked '!' (Rabin 2), infinitely many Safra trees T_i with node v marked '!' occur in ϱ''.

Thus we can proceed with the proof independently of the automaton under consideration, taking a run ϱ' on a word α, which is accepted either by the Muller automaton $\mathcal{M}$ or by the Rabin automaton $\mathcal{R}$.

In order to mark the node v with '!' in Step 6 of Safra's construction, it is necessary that – at least during the computation of the transition function δ – v has to be a parent. To become a parent, Step 2 is the only possibility in Safra's construction. Thus in run ϱ' the node v carries *before* any occurrence of a Safra tree T_i a macrostate containing a recurring state $f \in F$ of the nondeterministic Büchi automaton $\mathcal{B}$.

We consider a subrun of ϱ' after the point, where v occurs in all Safra trees, in more detail: Let T and U be Safra trees of ϱ' with node v marked '!', such that in no Safra tree between T and U node v is marked '!'. Let B be a Safra tree between T and U such that v carries a macrostate with $Q \cap F \neq \emptyset$, i.e.

$$T \ldots B \ldots U,$$

say $\varrho'(i) = T$, $\varrho'(j) = B$ and $\varrho'(k) = U$, for some $0 \leq i \leq j < k$. Note that T and B might be identical. Let P, H, R be the macrostate of v in T, B, U, resp.

For the sake of simplicity assume for the moment that B is the only Safra tree between T and U, where v carries a macrostate including a recurring state. Later we will also deal with the general situation.

As ϱ' is a run on α, there exist subwords

$$\alpha[i,j) := \alpha(i)\alpha(i+1)\ldots\alpha(j-1) \text{ and } \alpha[j,k) := \alpha(j)\alpha(j+1)\ldots\alpha(k-1)$$

of α corresponding to the finite subruns $T \ldots B$ and $B \ldots U$ of ϱ'.

Consider the computation of the successor state of B and the computation of state U from some predecessor state, say X (which might be identical with B), at certain points in Safra's construction of the transition function δ :

Point 1: During the computation of $\delta(B, \alpha(j-1))$ we obtain in Step 2 of Safra's construction a node w with macrostate $H \cap F$ as son of v. This node w remains in all Safra trees before U, as no vertical merge takes place before the computation of U.

Point 2: During the computation of $U = \delta(X, \alpha(k-1))$, the condition of Step 6 of Safra's construction becomes true, i.e., before Step 6 the nodes v and it's son w carry the same macrostate R.

The following picture shows the macrostates of v and w at these points, adds the macrostate of v in T, and shows also the subwords corresponding to the subruns:

node	in T		at Point 1		at Point 2
v	P	$\overset{\alpha[i,j)}{\Longrightarrow}$	H	$\overset{\alpha[j,k)}{\Longrightarrow}$	R
			$\cup\mid$		$\parallel$
w			$H \cap F$	$\overset{\alpha[j,k)}{\Longrightarrow}$	R

As new macrostates on a node are computed in Step 3 by the classical powerset construction, the lower row can be read: for all $r \in R$, there exists a $h \in H \cap F$ and a finite run $h \dots r$ of the nondeterministic Büchi automaton $\mathcal{B}$ on the subword $\alpha[j,k)$. This run can be completed by the upper row: for all $h \in H \cap F$, there exists a $p \in P$ and a finite run $p \dots h$ of the nondeterministic Büchi automaton $\mathcal{B}$ on the subword $\alpha[i,j)$. I.e., for all $r \in R$, there exists a $p \in P$ and a run of $\mathcal{B}$ on $\alpha[i,k)$ which leads from p to r while visiting a recurring state. Note that

- there might exist several such run segments, and
- that for any $r \in R$, there exists some predecessor $p \in P$ – but not vice versa.

In general, there might occur several Safra trees between T and U, in which v carries a macrostate including a recurring state. This changes our picture in the way that we have to deal with several 'Point 1'-situations, which might lead to several sons of v. At Point 2 we take the union of all son macrostates. Looking now for the run of $\mathcal{B}$ on $\alpha[i,k)$ ending in some $r \in R$, we take the first suitable 'Point 1'-situation to switch from the lower to the upper row. This situation arises, when the predecessor of some state $r' \in Q$ is a recurring state. As all states in the macrostates of the sons of v stem from recurring states, such a situation will always arise.

It remains to combine these finite run segments to a complete infinite run of $\mathcal{B}$: Let $0 < i_1 < i_2 < \dots$ be the positions of ϱ' at which v is marked '!'. Let $S_0 := \{q_\mathrm{I}\}$ and S_j be the macrostate of v at position i_j. Now we construct a directed tree with

- pairs (q, j) as nodes, where $q \in S_j$, $j \geq 0$, and
- as parent of a node $(r, j+1)$ we pick *one* of the pairs (p, j), such that $p \in S_j$ and there exists a subrun from p to r as described above.

Obviously, this is a well formed tree with $(q_\mathrm{I}, 0)$ as root. It has infinitely many nodes and is finitely branching. Thus, by König's Lemma, c.f. Theorem 3.8, there exists an infinite path $(q_\mathrm{I}, 0)(q_1, 1)\dots$ in the tree. Collecting all subruns along

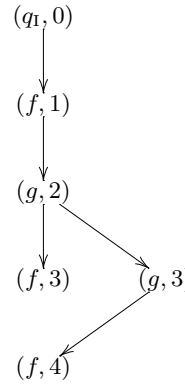

Fig. 3.11. Tree construction for $ac(aac)^\omega$.

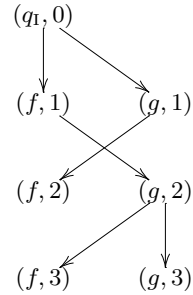

Fig. 3.12. Tree construction for a^ω.

this path, we obtain a run ϱ of $\mathcal{B}$ on α, which visits infinitely often recurring states. □

Example 3.10 *(Illustrating the Tree Construction for König's Lemma).*

We consider again the nondeterministic Büchi automaton shown in Figure 3.7 and the resulting automaton from Safra's construction depicted in Figure 3.10.

The word $ac(aac)^\omega$ leads to the following sequence of macrostates:

- $S_0 = \{q_I\}$,
- $S_{3i+1} = \{f\}$, $i \geq 0$,
- $S_{3i+2} = \{g\}$, $i \geq 0$, and
- $S_{3i+3} = \{f, g\}$, $i \geq 0$.

Figure 3.11 illustrates that we need indeed Königs Lemma to obtain an infinite path in the tree constructed: The pair $(f, 3)$ has no son, i.e., in node $(f, 3)$ ends a *finite* path of the tree.

The word a^ω demonstrates that there might be indeed a choice between different parent nodes. Here we have as sequence of macrostates

- $S_0 = \{q_I\}$ and
- $S_i = \{f, g\}$, $i \geq 1$.

In Figure 3.12 one can see that the tree constructed is not uniquely determined: as a parent for $(g, j+1)$ we have the choice between (f, j) and (g, j). For $(g, 2)$ we choose $(f, 1)$, while $(g, 3)$ has $(g, 2)$ as parent. □

Exercise 3.6. Apply Safra's construction to the nondeterministic Büchi automaton of Figure 3.1. Compare the result with the automaton of Figure 3.2 – which states of the automaton obtained by Safra's construction are equivalent?

3.3 Safra's Construction Is Optimal

In section 3.2.2 we showed that – given some fixed set of states Q – the number of nodes in a Safra tree is bounded by $|Q|$. Now we refine this result to obtain an upper bound for the number of states necessary in Safra's construction.

Theorem 3.11 (Complexity of Safra's Construction)**.** *Safra's construction converts a nondeterministic Büchi automaton with n states into a deterministic Muller automaton or into a deterministic Rabin automaton with $2^{O(n*log(n))}$ states.*

Proof. We have already seen that Safra trees consist of at most n nodes, and that it is sufficient for Safra's construction to have a vocabulary of $2n$ elements.

To compute a bound on the number of Safra trees, we describe them in terms of functions:

- Let $\{q_1, \ldots, q_n\}$ be the states of the nondeterministic Büchi automaton. To describe the macrostate labels at all nodes of a Safra tree, it is sufficient to know for any q_i the node v with q_i in its macrostate, which has the greatest hight: by Condition 1 all ancestors of v carry also this state. Due to Condition 2, q_i can not be an element of any other node's macrostate. Thus the macrostate labelling can be captured by a function of type $\{q_1, \ldots, q_n\} \to \{0, 1, \ldots, 2n\}$, where the value 0 is used for the case, that a state q_i is not in the Safra tree.
- The parent relation of a Safra tree can be encoded by a function of type $\{1, \ldots, 2n\} \to \{0, 1, \ldots, 2n\}$, where the value 0 is used for the case, that a node v has no parent in the Safra tree.
- The next-older brother relation can also be captured by a function of type $\{1, \ldots, 2n\} \to \{0, 1, \ldots, 2n\}$, where the value 0 is used for the case, that a node v has no next-older brother in the Safra tree.
- The marks '!' can be encoded by a function of type $\{1, \ldots, 2n\} \to \{0, 1\}$, where 1 stands for 'is marked'. For sake of similarity, we take here a function of the same type as above, i.e. $\{1, \ldots, 2n\} \to \{0, 1, \ldots, 2n\}$.

The number of combinations of such maps (and hence the number of possible Safra trees) is bounded by $(2n+1)^{n+3*2n} = (2n+1)^{7n} = 2^{\log((2n+1)^{7n})} = 2^{7n \log(2n+1)} \in 2^{O(n \log(n))}$. □

This complexity bound is optimal in the following sense:

Corollary 3.12 (Optimality of Safra's Construction)**.** *There is no conversion of Büchi automata with n states into deterministic Rabin automata with $2^{O(n)}$ states.*

Proof. We refer to Theorem 1.30. □

Note that this result holds for *Rabin automata,* and that it is open whether Safra's construction can be improved for *Muller automata.*