To appear in *SIAM Journal on Computing* 

# ON THE DETERMINIZATION OF WEIGHTED FINITE AUTOMATA*

ADAM L. BUCHSBAUM†, RAFFAELE GIANCARLO‡, AND JEFFERY R. WESTBROOK§

**Abstract.** We study the problem of constructing the deterministic equivalent of a nondeterministic weighted finite-state automaton (WFA). Determinization of WFAs has important applications in automatic speech recognition (ASR). We provide the first polynomial-time algorithm to test for the *twins* property, which determines if a WFA admits a deterministic equivalent. We also give upper bounds on the size of the deterministic equivalent; the bound is tight in the case of acyclic WFAs. Previously, Mohri presented a super-polynomial time algorithm to test for the twins property, and he also gave an algorithm to determinize WFAs. He showed that the latter runs in time linear in the size of the output when a deterministic equivalent exists; otherwise, it does not terminate. Our bounds imply an upper bound on the running time of this algorithm.

Given that WFAs can expand exponentially in size when determinized, we explore why those that occur in ASR tend to *shrink* when determinized. According to ASR folklore, this phenomenon is attributable solely to the fact that ASR WFAs have simple topology, in particular, that they are acyclic and layered. We introduce a very simple class of WFAs with this structure, but we show that the expansion under determinization depends on the transition weights: some weightings cause them to shrink, while others, including random weightings, cause them to expand exponentially. We provide experimental evidence that ASR WFAs exhibit this weight dependence. That they shrink when determinized, therefore, is a result of favorable weightings in addition to special topology. These analyses and observations have been used to design a new, approximate WFA determinization algorithm, reported in a separate paper along with experimental results showing that it achieves significant WFA size reduction with negligible impact on ASR performance.

**Key words.** algorithms, rational functions and power series, speech recognition, weighted automata

**AMS subject classifications.** 20M35, 68Q25, 68Q45, 68T10

**PII.**

## 1. Introduction.

Finite-state machines and their relation to rational functions and power series have been extensively studied [2, 3, 13, 19] and widely applied in fields ranging from image compression [10, 11, 12, 17] to natural language processing [22, 20, 21, 28, 30]. A subclass of finite-state machines, the weighted finite-state automata (WFAs), has recently assumed new importance, because WFAs provide a powerful method for representing and manipulating models of human language in automatic speech recognition (ASR) systems [23, 24]. This new research direction also raises a number of challenging algorithmic questions [5].

A *weighted finite-state automaton* (WFA) is a nondeterministic finite automaton (NFA), $A$, that has both an alphabet symbol and a weight, from some set $K$, on each transition. Let $R = (K, \oplus, \otimes, \overline{0}, \overline{1})$ be a semiring. Then $A$ together with $R$ generates a partial function from strings to $K$: the value of an accepted string is the semiring sum over accepting paths of the semiring product of the weights along each accepting path. A partial function that can be generated this way is a *rational power series* [29]. An example important to ASR is the set of WFAs with the *min-sum semiring*,

* An extended abstract of this paper appears in *Proc. 25th Int'l. Conf. on Automata, Languages, and Programming*, 1998.

† AT&T Labs, Shannon Laboratory, 180 Park Avenue, Florham Park, NJ 07932, USA, alb@research.att.com.

‡ Dipartimento di Matematica ed Applicazioni, Universitá di Palermo, Via Archirafi 34, 90123 Palermo, Italy, raffaele@altair.math.unipa.it. Work supported by AT&T Labs.

§ 20th Century Television, Los Angeles, CA 90025, USA, jwestbrook@acm.org. Work completed while a member of AT&T Labs.

$(\Re^+ \cup \{0, \infty\}, \min, +, \infty, 0)$, which compute for each accepted string the minimum cost accepting path.

In this paper, we study problems related to the determinization of WFAs. A *deterministic*, or *sequential*, WFA has at most one transition with a given input symbol out of each state. Not all rational power series can be generated by deterministic WFAs. A *determinization algorithm* takes as input a WFA and produces a deterministic WFA that generates the same rational power series, if such a deterministic WFA exists. The importance of determinization to ASR is well established [20, 23, 24].

To the best of our knowledge, Mohri [20] presented the first determinization procedure for WFAs, extending the seminal ideas of Choffrut [8, 9] and Weber and Klemm [31] regarding string-to-string transducers. Mohri gives a determinization procedure with three phases. First, $A$ is converted to an equivalent unambiguous, trim WFA $A_t$, using an algorithm analogous to one for NFAs [13]. (We define *unambiguous* and *trim* below.) Mohri then gives an algorithm, **TT**, that determines if $A_t$ has the *twins property* (also defined below). If $A_t$ does not have the twins property, then there is no deterministic equivalent of $A$. If $A_t$ has the twins property, a second algorithm of Mohri's, **DTA**, can be applied to $A_t$ to yield $A'$, a deterministic equivalent of $A$. Algorithm **TT** runs in $O(m^{4n^2})$ time, where $m$ is the number of transitions and $n$ the number of states in $A_t$. Algorithm **DTA** runs in time linear in the size of $A'$. In practice, **DTA** is run directly on $A$, which is assumed to admit a deterministic equivalent; conversion to $A_t$ and testing for twins are theoretical steps needed to make the procedure well defined. Mohri observes that $A'$ can be exponentially larger than $A$, because WFAs include classical NFAs. He gives no upper bound on the worst-case state-space expansion, however, and because of the weights on transitions, the classical NFA upper bound does not apply. Finally, Mohri gives an algorithm that takes a deterministic WFA and outputs the minimum-size equivalent, deterministic WFA.

We present several results related to the determinization of WFAs. In Section 3 we give the first polynomial-time algorithm to test whether an unambiguous, trim WFA satisfies the twins property. It runs in $O(m^2n^6)$ time. We then provide a worst-case time complexity analysis of **DTA**. The number of states in the output deterministic WFA is at most $2^{n(2\log n + n^2 \log|\Sigma|+1)}$, where $\Sigma$ is the input alphabet. If the weights are rational, this bound becomes $2^{n(2\log n+1+\min(n^2\log|\Sigma|,\rho))}$, where $\rho$ is the maximum bit-size of a weight. When the input WFA is acyclic, the bound becomes $2^{n\log|\Sigma|}$. The acyclic bound holds for real weights, and it is tight (up to constant factors) for any alphabet size. It remains open whether there exists a polynomial-time procedure to determine whether an arbitrary WFA admits a deterministic equivalent, because the determinization process above requires converting a WFA to an unambiguous equivalent prior to testing for twins.

In Sections 4–6 we study questions motivated by the use of WFA determinization in ASR [23, 24]. Although determinization causes exponential state-space expansion in the worst case, in ASR systems the determinized WFAs are often *smaller* than the input WFAs [20], and they are seldom very large. This is fortunate, because the performance of ASR systems depends directly on WFA size [23, 24]. Folklore within the ASR community credits this phenomenon entirely to the special topology of ASR WFAs. (The topology of a WFA is its underlying directed graph and labeling by input symbols, ignoring weights.) ASR WFAs tend to be acyclic and layered. Such a WFA always admits a deterministic equivalent. The role that the transition weights might play in controlling expansion under determinization has not been considered.

In Section 4 we study the role of topology in expansion under determinization.

We exhibit a class of layered, acyclic WFAs whose minimum equivalent deterministic WFAs are exponentially larger regardless of weighting. The languages accepted by these WFAs are quite unnatural, however.

In Section 5 we study the role of transition weights in expansion under determinization. We introduce a class of nondeterministic WFAs, $RG$. Each WFA in this class has an extremely simple multi-partite, acyclic topology, accepts a very trivial language, and in the absence of weights (i.e., with all weights set to zero) has a smaller deterministic equivalent. We show, however, that for any $A \in RG$ and any $i \leq n$, there exists an assignment of weights to the transitions of $A$ such that the minimal equivalent deterministic WFA has $\Theta(2^{i \log |\Sigma|})$ states. This gives a lower bound to match the upper bounds of Section 3. Using ideas from universal hashing, we also show that similar results hold when the weights are random $i$-bit numbers.

This motivates us to examine experimentally the effect of varying weights on actual WFAs from ASR applications. In Section 6 we give the results of these experiments. We call a WFA *weight-dependent* if its expansion under determinization is strongly determined by its weights. Most of the examples from ASR were weight-dependent. These experimental results together with the theory we develop show that the folklore explanation is insufficient: ASR WFAs shrink under determinization because both the topology and weighting tend to be favorable.

Some of our results help explain the nature of WFAs from the algorithmic point of view, i.e., how weights assigned to the transitions of a WFA can affect the performance of algorithms manipulating it. Others relate directly to the theory of weighted automata. We have used our results to design an approximate variant of Mohri's determinization algorithm. We describe this algorithm separately [6], along with experimental results showing that it achieves size reductions in ASR language models that significantly exceed those of previous methods, with negligible effects on ASR performance (time and accuracy).

**2. Definitions and Terminology.** Given a semiring $(K, \oplus, \otimes, \overline{0}, \overline{1})$, a *weighted finite automaton* (WFA) is a tuple $G = (Q, \bar{q}, \Sigma, \delta, Q_f)$. $Q$ is the set of states, $\bar{q} \in Q$ is the initial state, $\Sigma$ is the set of symbols, $\delta \subseteq Q \times \Sigma \times K \times Q$ is the set of transitions, and $Q_f \subseteq Q$ is the set of final states. We assume that $|\Sigma| > 1$. A *deterministic*, or *sequential*, WFA has at most one transition $t = (q_1, \sigma, \nu, q_2)$ for any pair $(q_1, \sigma)$; a *nondeterministic* WFA can have multiple transitions on a pair $(q_1, \sigma)$, differing in target state $q_2$. The problems examined in this paper are motivated primarily by ASR applications, which work with the *min-sum semiring*, $(\Re^+ \cup \{0, \infty\}, \min, +, \infty, 0)$, and we therefore limit further discussion to the min-sum semiring. (Some of the algorithms considered use subtraction. To be well-defined, therefore, they require a skew field. The min-sum semiring is indeed embedded in a skew field [16].)

Let $\vec{t} = (t_1, \ldots, t_\ell)$ be some sequence of transitions, such that $t_i = (q_{i-1}, \sigma_i, \nu_i, q_i)$; $\vec{t}$ *induces* string $w = \sigma_1 \cdots \sigma_\ell$. String $w$ is *accepted by* $\vec{t}$ if $q_0 = \bar{q}$ and $q_\ell \in Q_f$; $w$ is *accepted by* $G$ if some $\vec{t}$ accepts $w$. Let $c(t_i) = \nu_i$ be the *weight* of $t_i$. Then the *weight of* $\vec{t}$ is

$$c(\vec{t}) = \sum_{i=1}^{\ell} c(t_i).$$

Let $T(w)$ be the set of all sequences of transitions that accept string $w$. Then the *weight of* $w$ is

$$c(w) = \min_{\vec{t} \in T(w)} c(\vec{t}).$$

The *weighted language* of $G$ is the set $L(G) = \{(w, c(w)) \mid w \text{ is accepted by } G\}$; i.e., the weighted strings accepted by $G$. Intuitively, the weight on a transition of $G$ can be seen as the "confidence" one has in taking that transition. The weights need not, however, satisfy stochastic constraints, as do the probabilistic automata introduced by Rabin [26].

Fix two states $q$ and $q'$ and a string $v \in \Sigma^*$. Let $c(q, v, q')$ be the minimum of $c(\vec{t})$, taken over all transition sequences $\vec{t}$ from $q$ to $q'$ inducing $v$. We refer to $c(q, v, q')$ as the *optimal cost* of inducing $v$ from $q$ to $q'$. We generally abuse notation so that $\delta(q, w)$ can represent the set of states reachable from state $q \in Q$ on string $w \in \Sigma^*$. We extend the function $\delta$ to strings in the usual way: $q' \in \delta(q, v), v \in \Sigma^+$, means that there is a sequence of transitions from $q$ to $q'$ inducing $v$.

The *topology* of $G$ is the projection $\pi_{Q\times\Sigma\times Q}(\delta)$: i.e., the transitions of $G$ without respect to the weights. We also refer to the topology of $G$ as the *graph* underlying $G$.

A WFA is *trim* if every state appears in an accepting path for some string and no transition is weighted $\overline{0}$ ($\infty$ in the min-sum semiring). A WFA is *unambiguous* if there is exactly one accepting path for each accepted string.

*Determinization* of $G$ is the problem of computing a deterministic WFA $G'$ such that $L(G') = L(G)$, if such a $G'$ exists. We denote the output of algorithm **DTA** by $dta(G)$. We denote the minimal deterministic WFA accepting $L(G)$ by $min(G)$, if one exists. We say that $G$ *expands* if $dta(G)$ has more states and/or transitions than $G$.

Let $n = |Q|$ and $m = |\delta|$, and let the *size* of $G$ be $|G| = n + m$. We also use $\#G$ to mean $|Q|$, the number of states of $G$. We assume that each transition is labeled with exactly one symbol, so $|\Sigma| \leq m$. Recall that the weights of $G$ are non-negative real numbers. Let $C$ be the maximum weight. In the *general case*, weights are incommensurable real numbers, requiring "infinite precision." In the *integer case*, weights can be represented with $\rho = \lceil \lg C \rceil$ bits. We denote the integral range $[a, b]$ by $[a, b]_Z$ . The integer case extends to the case in which the weights are rationals requiring $\rho$ bits. We assume that in the integer and rational cases, weights are normalized to remove excess least-significant zero bits.

For our analyses, we use the RAM model of computation as follows. In the general case, we charge constant time for each arithmetic-logic operation involving weights (which are real numbers). We refer to this model as the $\Re$-RAM [25]. The relevant parameters for our analyses are $n$, $m$, and $|\Sigma|$. In the integer case, we also use a RAM, except that each arithmetic-logic operation now takes $O(\rho)$ time. We refer to this model as the $\mathcal{CO}$-RAM[1]. The relevant parameters for the analyses are $n$, $m$, $|\Sigma|$, and $\rho$.

## 3. Determinization of WFAs.

**3.1. An Algorithm for Testing the Twins Property.** DEFINITION 3.1. *Two states, $q$ and $q'$, of a WFA $G$ are* twins *if $\forall u, v \in \Sigma^*$ such that $q \in \delta(\bar{q}, u)$, $q' \in \delta(\bar{q}, u)$, $q \in \delta(q, v)$, and $q' \in \delta(q', v)$, the following holds: $c(q, v, q) = c(q', v, q')$. $G$ has the* twins property *if all pairs $q, q' \in Q$ are twins.*

That is, if states $q$ and $q'$ are reachable from $\bar{q}$ by a common string, then $q$ and $q'$ are twins only if any string that induces a cycle at each induces cycles of equal optimal cost. Note that two states having no cycle on a common string are twins. Mohri [20, Theorems 11 and 12] proves that any WFA $G$ over the min-sum semiring is determinizable if it has the twins property; furthermore, if $G$ is trim and unambiguous, the twins property becomes a necessary and sufficient condition. For an example of a non-determinizable WFA, see Figure 3.1.

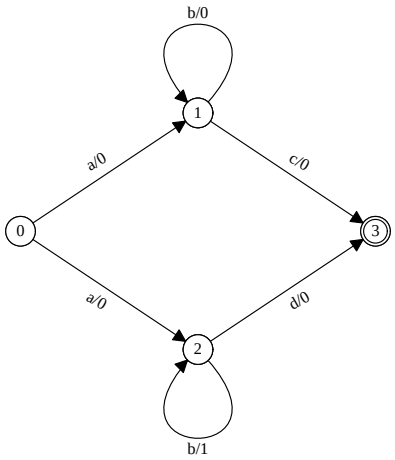


FIG. 3.1. *A nondeterministic, trim, unambiguous WFA $G$. Arcs labeled $\sigma/w$ correspond to transitions labeled $\sigma$ with weight $w$. For this and succeeding figures, the start state is the unique source, and final states are denoted by double circles. $G$ accepts the language $\{(ab^n c, 0),\ (ab^n d, n) \mid n \geq 0\}$. States 1 and 2 do not have the twins property: each is reachable from state 0 via string $a$, yet the costs of the cycles labeled $b$ at each differ. It is easily shown that no deterministic WFA can accept $L(G)$.*

The twins property for WFAs is analogous to that defined by Choffrut [8, 9] and (in different terms) by Weber and Klemm [31] to identify necessary and sufficient conditions for a string-to-string transducer to admit a sequential transducer realizing the same rational transduction. In spite of the strong analogy, the proof techniques used for WFAs differ from those used to obtain analogous results for string-to-string transducers. In particular, the efficient algorithm we derive here to test a WFA for twins is not related to the polynomial-time algorithm of Weber and Klemm [31] for testing twins in string-to-string transducers. We reduce the problem of testing the twins property to that of computing shortest paths on some suitably defined graphs, which we introduce next.

Let $T_{\bar{q},\bar{q}}$ be the multi-partite acyclic, labeled, weighted graph having $2n^2$ layers and inductively defined as follows. The root vertex $\hat{r}$ is at layer zero and corresponds to $(\bar{q}, \bar{q})$. The vertices at layer one correspond to a subset of $Q \times Q$ obtained as follows: $\hat{r}$ is connected to a vertex $u$, corresponding to $(q_1, q_2)$, if and only if there are two distinct transitions $t = (\bar{q}, a, c_1, q_1)$ and $t' = (\bar{q}, a, c_2, q_2)$ in $G$. The arc connecting $\hat{r}$ to $u$ is labeled with $a \in \Sigma$ and has cost $c = c_1 - c_2$. Assume that we have the vertices at layer $i - 1$. The vertices at layer $i$ are obtained as follows. Let $u$ be the vertex at layer $i - 1$ corresponding to $(q_1, q_2) \in Q \times Q$; $u$ is connected to $u'$, corresponding to $(q_1', q_2')$, at layer $i$ if and only if there are two distinct transitions $t = (q_1, a, c_1, q_1')$ and $t' = (q_2, a, c_2, q_2')$ in $G$. The arc connecting $u$ to $u'$ is labeled with $a \in \Sigma$ and has cost $c = c_1 - c_2$. This graph has $O(n^4)$ vertices and $O(m^2 n^4)$ arcs. Let $(q, q')_i$ denote the vertex corresponding to $(q, q') \in Q \times Q$ at layer $i$ of $T_{\bar{q},\bar{q}}$, if any. Let $RT \subseteq \{(q, q') \mid q \neq q'\}$ be the set of pairs of distinct states of $G$ that are reachable from $(\bar{q}, \bar{q})_0$ in $T_{\bar{q},\bar{q}}$. For each $(q, q') \in RT$, define $T_{q,q'}$ analogously to $T_{\bar{q},\bar{q}}$. Notice that $T_{q,q'}$ has $O(n^4)$ vertices and $O(m^2 n^4)$ arcs. We need the following.

LEMMA 3.2. *Fix two distinct states $q$ and $q'$ of $G$. They can be reached from the initial state $\bar{q}$ of $G$ by the same string $z \in \Sigma^+$ if and only if there exists some string*

*$z' \in \Sigma^i$, for some $1 \leq i \leq 2n^2 - 1$, such that $q$ and $q'$ are both reached from $\bar{q}$ using $z'$. In that case, there is at least one path in $T_{\bar{q},\bar{q}}$ that goes from $(\bar{q},\bar{q})_0$ to $(q,q')_i$.*

*Proof.* Fix a string $z \in \Sigma^+$, and assume that $q$ and $q'$ can be reached from $\bar{q}$ by $z$. Assume that $|z| > 2n^2 - 1$, or else we are done. Since there are only $n^2$ distinct pairs of states of $G$ and $|z| > 2n^2 - 1$, there must exist two states $q_1$ and $q_2$ and a string $v \in \Sigma^+$ such that (a) $z = xvu$; (b) $q_1$ (rsp., $q_2$) is on a path from $\bar{q}$ to $q$ (rsp., $q'$) inducing $z$; and (c) $q_1 \in \delta(q_1, v)$ (rsp., $q_2 \in \delta(q_2, v)$). But then, $z' = xu$ also reaches both $q$ and $q'$ from $\bar{q}$. If $|z'| \leq 2n^2 - 1$, we are done; otherwise we iterate the argument. The second part of the lemma follows by construction of $T_{\bar{q},\bar{q}}$. □

LEMMA 3.3. *Let $G$ be trim and unambiguous. Fix a string $y \in \Sigma^i$, $1 \leq i \leq 2n^2-1$, and two distinct states $q$ and $q'$ of $G$. Then $q \in \delta(q, y)$ and $q' \in \delta(q', y)$ if and only if there is exactly one path $\mathbf{p}$ in $T_{q,q'}$ that starts at $(q,q')_0$, ends at $(q,q')_{|y|}$, and induces $y$. Moreover, the cost of $\mathbf{p}$ is $c(q, y, q') - c(q', y, q')$.*

*Proof.* We prove the sufficient case; the necessary case should be clear from the construction of $T_{q,q'}$.

First observe that, since $G$ is trim and unambiguous, the following holds: for each string $y \in \Sigma^+$ such that $q \in \delta(q, y)$, there is exactly one cycle starting and ending at $q$ and inducing $y$.

Let $(q = q_0, q_1, q_2, \cdots, q_{|y|} = q)$ be the unique sequence of states of $G$ originating in $q$ and inducing $y$ in $G$. Therefore, $c(q, y, q)$ is the sum of the weights on the transitions in that sequence. Similarly define $(q' = q'_0, q'_1, q'_2, \cdots, q'_{|y|} = q')$. By the above construction, there exists a path $\mathbf{p}$ in $T_{q,q'}$, consisting of the vertices $((q,q')_0, (q_1,q'_1)_1, \cdots, (q,q')_{|y|})$ and inducing $y$. This path must be unique, and its cost is $c(q, y, q) - c(q', y, q')$. □

LEMMA 3.4 ([20, Lemma 2]). *Let $G$ be a trim, unambiguous WFA. $G$ has the twins property if and only if $\forall u, v \in \Sigma^*$ such that $|uv| \leq 2n^2 - 1$, the following holds: when there exist two states $q$ and $q'$ such that (i) $\{q, q'\} \subseteq \delta(\bar{q}, u)$, and (ii) $q \in \delta(q, v)$ and $q' \in \delta(q', v)$, then (iii) $c(q, v, q) = c(q', v, q')$ must follow.*

Fix two distinct states $q$ and $q'$ of $G$. Let $(q,q')_{i_1}, (q,q')_{i_2}, \ldots, (q,q')_{i_s}$, $0 < i_1 < i_2 < \cdots < i_s$, be all the occurrences of $(q,q')$ in $T_{q,q'}$, excluding $(q,q')_0$. This sequence may be empty. A symmetric sequence can be extracted from $T_{q',q}$. We refer to these sequences as the *common cycles sequences* of $(q,q')$. We say that $q$ and $q'$ satisfy the *local twins property* if and only if (1) their common cycles sequences are empty, or (2) zero is the cost of any shortest path from $(q,q')_0$ to $(q,q')_{i_j}$ in $T_{q,q'}$ and from $(q',q)_0$ to $(q',q)_{i_j}$ in $T_{q'q}$, for all $1 \leq j \leq s$.

LEMMA 3.5. *Let $G$ be a trim, unambiguous WFA. $G$ satisfies the twins property if and only if (i) $RT$ is empty or (ii) all $(q,q') \in RT$ satisfy the local twins property.*

*Proof.*

$\Rightarrow$) Assume that $G$ satisfies the twins property. If $RT$ is empty, we are done. Assume then that $RT \neq \emptyset$. The proof is by contradiction. Assume that some $(q,q') \in RT$ does not satisfy the local twins property. The common cycles sequences of $(q,q')$ cannot be empty, or else they would satisfy the local twins property. By assumption, there exists some $j$ for which the cost of some shortest path from $(q,q')_0$ to $(q,q')_{i_j}$ in $T_{q,q'}$ is not zero, while the cost of a shortest path from $(q',q)_0$ to $(q',q)_{i_j}$ in $T_{q'q}$ may be any value, including zero (or vice versa). Fix any such shortest path $\mathbf{p}$ in $T_{q,q'}$. According to Lemma 3.3, $\mathbf{p}$ corresponds to cycles around $q$ and $q'$ that each induce the same string $y$, for some $y \in \Sigma^{i_j}$. Moreover, we must have $c(q, y, q) - c(q', y, q') \neq 0$. By definition of $RT$, $q$ and $q'$ are each reachable by some string $u$ from the initial state of $G$. Therefore, $G$ does not satisfy the twins property, which is a contradiction.

$\Leftarrow$) Assume that $RT$ is empty. Then, by Lemma 3.2, no two distinct states $q, q'$ of $G$ can both be reached by some string $z \in \Sigma^+$ from the initial state $\bar{q}$. Therefore, $G$ satisfies the twins property. Assume now that $RT$ is not empty. We have two subcases.

*Subcase A)* Assume that all states in $RT$ satisfy the local twins property because their common cycles sequences are empty. This implies that all pairs of distinct states reachable from the initial state of $G$ through the same string $z \in \Sigma^+$ do not have any cycles in common inducing identical strings. Thus, $G$ satisfies the twins property.

*Subcase B)* Assume that some states in $RT$ satisfy the local twins property and their common cycles sequences are not empty. Let $RT'$ be such a set. Assume that $G$ does not satisfy the twins property. We derive a contradiction. Since $RT'$ is not empty, we have that the set of pairs of states for which $(i)$ and $(ii)$ are satisfied in Lemma 3.4 is not empty. But since $G$ does not satisfy the twins property, there must exist two distinct states $q$ and $q'$ and a string $uv \in \Sigma^*$, $|uv| \leq 2n^2 - 1$, such that $(i)$ both $q$ and $q'$ can be reached from the initial state of $G$ through string $u$; $(ii)$ $q \in \delta(q, v)$ and $q' \in \delta(q', v)$; and $(iii)$ $c(q, v, q) \neq c(q', v, q')$. We now argue that $(q, q')$ must be in $RT'$. Because $q \neq q'$ and $G$ has only one initial state, we have that $|u| \geq 1$. Thus, $1 \leq |u| \leq 2n^2 - 1$, implying that $(q, q') \in RT$. $v$ cannot be the empty string $\epsilon$ because $c(q, \epsilon, q) = c(q', \epsilon, q') = 0$. Since $|uv| \leq 2n^2 - 1$, we have that $1 \leq |v| \leq 2n^2 - 1$. But then, by Lemma 3.3 and $(ii)$ above, we have that $(q, q')_{|v|}$ can be reached from $(q, q')_0$ in $T_{q,q'}$ through the nonempty string $v$. Therefore, the common cycles sequences of $(q, q')$ cannot be empty, implying that $(q, q') \in RT'$. Without loss of generality, assume that $c(q, v, q) - c(q', v, q') < 0$. Since $1 \leq |v| \leq 2n^2 - 1$, we have by Lemma 3.3 that there is exactly one path $\mathbf{p}$ in $T_{q,q'}$ starting at $(q, q)_0$, ending in $(q, q')_{|v|}$, inducing $v$, and with cost $c(q, v, q) - c(q', v, q') < 0$. Since $\mathbf{p}$ has negative cost, the cost of the shortest path from $(q, q')_0$ to $(q, q')_{|v|}$ in $T_{q,q'}$ cannot be zero, which contradicts that $q$ and $q'$ satisfy the local twins property and have non-empty common cycles sequences. □

Our algorithm for testing whether a trim, unambiguous WFA has the twins property works as follows. First, compute $T_{\bar{q},\bar{q}}$ and the set $RT$. Then, for each pair of states $(q, q') \in RT$ that have not been processed yet: compute $T_{q,q'}$ and $T_{q',q}$, extract the common cycles sequences, and compute the single source (from the root) shortest paths to vertices in $T_{q,q'}$ and $T_{q',q}$.

THEOREM 3.6. *Let $G$ be a trim unambiguous WFA. In the general case, whether $G$ satisfies the twins property can be checked in $O(m^2n^6)$ time using the $\Re$-RAM model of computation. In the integer case, the bound becomes $O(\rho m^2 n^6)$ using the $\mathcal{CO}$-RAM model of computation.*

*Proof.* Lemma 3.5 implies correctness. We now analyze the algorithm, starting with the general case. Recall that each arithmetic-logic operation can be done in constant time. $T_{\bar{q},\bar{q}}$ can be easily obtained in $O(m^2n^4)$ time by visiting the automaton $G$. Now, visiting $T_{\bar{q},\bar{q}}$, we can obtain the set $RT$ in the same amount of time.

Fix a pair of distinct states $q$ and $q'$ of $G$. It is sufficient to discuss how to compute shortest paths from the root vertex of $T_{q,q'}$ to the other vertices in the graph. Notice that the edges of $T_{q,q'}$ may have negative cost. However, $T_{q,q'}$ is a multi-partite acyclic graph. In that case, it is a simple exercise to show how to perform the required computation in time linear in the size of $T_{q,q'}$, i.e., $O(m^2n^4)$ time. Since $|RT| = O(n^2)$, the total time of the algorithm is $O(m^2n^6)$.

For the integer case, we multiply the above bound by $\rho$. □

We also mention, omitting the details, that the exponential-time algorithm for

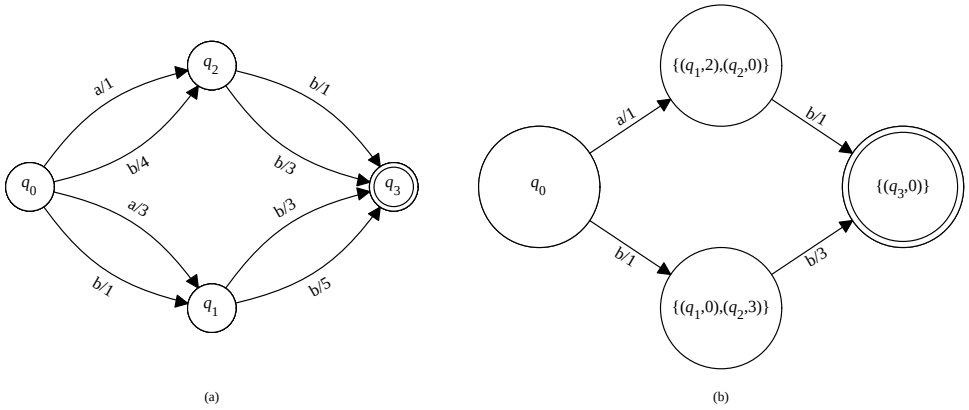


FIG. 3.2. *(a) A nondeterministic weighted automaton, $A$. Arcs labeled $\sigma/w$ correspond to transitions labeled $\sigma$ with weight $w$. (b) The result of applying* **DTA** *to $A$. This is derived from Figures 11 and 12 of Mohri [20].*

testing the twins property originally devised by Mohri [20] can be simplified and implemented to run in pseudo-polynomial time in the integer case. The algorithm we devise here is weakly polynomial in the integer case.

**3.2. The DTA Algorithm.** Mohri [20] describes a determinization algorithm for a finite-state automaton with weights drawn from a general semiring. What we refer to as **DTA** is that algorithm restricted to the min-sum semiring. **DTA** is a generalization of the classic power-set construction for finite automata. We describe the algorithm, starting with an example.

Consider the weighted automaton, $A$, in Figure 3.2(a). While $A$ is not unambiguous, it has the twins property, and so we can apply **DTA** directly to it, proceeding as follows. From the initial state $q_0$, we can reach states $q_1$ and $q_2$ using the input symbol $a$. Analogously to the determinization of finite-state automata, we establish a new state $\{q_1, q_2\}$ in $A'$, reachable from $q_0$ with input symbol $a$. The transitions to $q_1$ and $q_2$, however, have different weights in $A$. **DTA** selects the smaller weight to be the weight of the transition to $\{q_1, q_2\}$ and records the difference between the two weights in the new state. In the example, the weight of the $q_0 \to q_1$ transition is 3, and that of the $q_0 \to q_2$ transition is 1. Therefore, the new transition to $\{q_1, q_2\}$ gets weight 1, and the difference of $2 = 3 - 1$ is assigned as a *remainder* to component $q_1$. For completeness, a remainder of $0 = 1 - 1$ is assigned to $q_2$. The new state is thus encoded as $\{(q_1, 2), (q_2, 0)\}$ in $A'$. Similarly, from state $q_0$ in $A$, we can reach states $q_1$ and $q_2$ via symbol $b$. Again the minimum weight among these transitions is 1, so we assign this weight to the new arc and encode the remainder weights (0 and 3, respectively) in the new state $\{(q_1, 0), (q_2, 3)\}$ in $A'$.

In general, the states in $A'$ are of the form $\hat{q} = \{(q_{i_1}, r_{i_1}), \ldots, (q_{i_\ell}, r_{i_\ell})\}$. The $q_i$s are states from $A$, and the $r_i$s are called *remainders*. Each such $\hat{q}$ is interpreted as follows. Consider any string $w \in \Sigma^*$ such that there is a (single) path inducing $w$ from the start state, $q_0$, to $\hat{q}$. As in classical automata determinization, there is at least one path inducing $w$ from $q_0$ to each $q_{i_j}$ in the nondeterministic input, $A$. Let $c_j$ be the weight of the minimum weight path inducing $w$ from $q_0$ to $q_{i_j}$ in $A$. Let $c$ be the weight of the path from $q_0$ to $\hat{q}$ in $A'$. The remainders are constructed so that $r_{i_j} = c_j - c$. In this way, all necessary path length information is encoded into the

transition weights and remainders in $A'$.

Returning to the example, consider state $\{(q_1, 2), (q_2, 0)\}$ in $A'$ and the input symbol $b$. In $A$, we can reach state $q_3$ from both $q_1$ and $q_2$. Recalling the above discussion of remainders, we consider the sum of the weight of the transition in $A$ (3 for the $q_1 \rightarrow q_3$ transition and 1 for the $q_2 \rightarrow q_3$ transition) plus the remainder associated with the original source state encoded in state $\{(q_1, 2), (q_2, 0)\}$ in $A'$. That is, we consider the sums $3 + 2 = 5$ and $1 + 0 = 1$. We take the minimum among those values, i.e., 1, as the weight of the transition from $\{(q_1, 2), (q_2, 0)\}$ to $\{(q_3, r)\}$ ($r$ to be determined) in $A'$. Since there is only one destination state ($q_3$) in $A$, the remainder $r$ is 0, so we encode the new destination state as $\{(q_3, 0)\}$. Similarly, we construct an arc with weight 3 on symbol $b$ from $\{(q_1, 0), (q_2, 3)\}$ to $\{(q_3, 0)\}$. ($3+0 = 3$, $1+3 = 4$, and we take the minimum, which is 3.) The end result is shown in Figure 3.2(b).

Generalizing to an arbitrary WFA $G = (Q, \bar{q}, \Sigma, \delta, Q_f)$, the deterministic WFA $G'$ is obtained as follows. The start state of $G'$ is $\{(\bar{q}, 0)\}$, which forms an initial set $P$. While $P \neq \emptyset$, we remove any state $q = \{(q_1, r_1), \ldots, (q_n, r_n)\}$ from $P$, where $q_i \in Q$ and $r_i \in \Re^+ \cup \{0, \infty\}$. The remainders encode path length information, as described above. For each $\sigma \in \Sigma$, let $\{q'_1, \ldots, q'_m\}$ be the set of states reachable by $\sigma$-transitions out of all the $q_i$. For $1 \leq j \leq m$, let

$$\rho_j = \min_{1 \leq i \leq n; (q_i, \sigma, \nu, q'_j) \in \delta} \{r_i + \nu\}$$

be the minimum of the weights of $\sigma$-transitions into $q'_j$ from the $q_i$ plus the respective $r_i$. Let $\rho = \min_{1 \leq j \leq m}\{\rho_j\}$. Let $q' = \{(q'_1, s_1), \ldots, (q'_m, s_m)\}$, where $s_j = \rho_j - \rho$, for $1 \leq j \leq m$. If $q'$ is a new state, we add it to $P$.. We add transition $(q, \sigma, \rho, q')$ to $G'$. This is the only $\sigma$-transition out of state $q$, so $G'$ is deterministic.

Let $T_G(w)$ be the set of sequences of transitions in $G$ that accept a string $w \in \Sigma^*$; let $\vec{t}_{G'}(w)$ be the (one) sequence of transitions in $G'$ that accepts the same string. Mohri [20] shows that

$$c(\vec{t}_{G'}(w)) = \min_{\vec{t} \in T_G(w)} \{c(\vec{t})\},$$

and thus $L(G') = L(G)$. Moreover, let $T_G(w, q)$ be the set of sequences of transitions in $G$ from state $\bar{q}$ to state $q$ that induce string $w$. Again, let $\vec{t}_{G'}(w)$ be the (one) sequence of transitions in $G'$ that induces the same string; $\vec{t}_{G'}(w)$ ends at some state $\{(q_1, r_1), \ldots, (q_n, r_n)\}$ in $G'$ such that some $q_i = q$. Mohri [20] shows that

$$c(\vec{t}_{G'}(w)) + r_i = \min_{\vec{t} \in T_G(w,q)} \{c(\vec{t})\}.$$

Thus each remainder $r_i$ encodes the difference between the weight of the shortest path to some state that induces $w$ in $A$ and the weight of the path inducing $w$ in $A'$, as described above. Hence at least one remainder in each state is zero.

**3.3. An Analysis.** We first bound $\#dta(G)$, the number of states in $dta(G)$. The results of Section 5 show that our upper bound is tight to within polynomial factors.

LEMMA 3.7. *Assume that $G$ satisfies the twins property. Let $\tilde{R}$ be the set of remainders generated by* **DTA** *when computing $dta(G)$. Let $R$ be the set of remainders $r$ for which the following holds: $\exists w \in \Sigma^*$, $|w| \leq n^2 - 1$, and two states $q_1$ and $q_2$, such that $r = |c(\bar{q}, w, q_2) - c(\bar{q}, w, q_1)|$. Then $\tilde{R} \subseteq R$.*

*Proof.* Let $R'$ be the set of remainders $r'$ such that: $\exists w' \in \Sigma^*$ and two states $q_1$ and $q_2$ such that $r' = |c(\bar{q}, w', q_2) - c(\bar{q}, w', q_1)|$. Consider a state-remainder tuple in $dta(G)$ reached by $w'$ from the initial state, and assume that $q_1$ is the optimal state in that tuple, i.e., the one with zero remainder. Then the remainder associated to $q_2$ is $r'$. Thus, $\tilde{R} \subseteq R'$. We next show that $R = R'$.

Clearly $R \subseteq R'$. To prove the other inclusion we only need to show that the remainder $r$ generated by any string of length at least $n^2$ is generated by a string of length at most $n^2 - 1$. Let $\mathbf{p_1}$ and $\mathbf{p_2}$ be the paths of minimum cost in $G$, starting at $\bar{q}$, ending at $q_1$ and $q_2$ respectively, and each inducing $u$. Because $|u| \geq n^2$ and there are only $n^2$ distinct pairs of states in $G$, there exist two (not necessarily distinct) states, $q_1'$ and $q_2'$, in $\mathbf{p_1}$ and $\mathbf{p_2}$ respectively, and a partition of $u = xvz$, $v \in \Sigma^+$, such that $\{q_1', q_2'\} \subseteq \delta(\bar{q}, x)$, $q_1' \in \delta(q_1', v)$ and $q_2' \in \delta(q_2', v)$ (there are cycles at $q_1'$ and $q_2'$ inducing $v$), and, finally, $q_1 \in \delta(q_1', z)$ and $q_2 \in \delta(q_2', z)$. Since $q_1'$ and $q_2'$ are twins, we have that $|c(\bar{q}, u, q_1) - c(\bar{q}, u, q_2)| = |c(\bar{q}, \bar{u}, q_1) - c(\bar{q}, \bar{u}, q_2)|$, where $\bar{u} = xz$ is in $\Sigma^+$ and $|\bar{u}| < |u|$. If $\bar{u}$ is of the required length, we are finished; otherwise, we iterate the argument. □

THEOREM 3.8. *Let $G$ be a WFA satisfying the twins property. In the general case, $\#dta(G) < 2^{n(2\log n + n^2 \log|\Sigma| + 1)}$; in the integer (or rational) case, $\#dta(G) < 2^{n(2\log n + 1 + \min(n^2 \log|\Sigma|, \rho))}$; and if $G$ is acyclic, $\#dta(G) < 2^{n \log|\Sigma|}$ independent of any assumptions on weights. The acyclic bound is tight (up to constant factors) for any alphabet.*

*Proof.* Let $\tilde{R}$ be the set of remainders in $dta(G)$. Each state in $dta(G)$ is an $i$-tuple of states from $G$ with a corresponding $i$-tuple of remainders. In the worst case, each $i$-state tuple from $G$ will appear in $dta(G)$, and there are $|\tilde{R}|^i$ distinct $i$-tuples of remainders it can assume. (This over counts by including tuples without any zero remainders.) Therefore,

$$\#dta(G) \leq \sum_{i=1}^{n} \binom{n}{i} |\tilde{R}|^i \leq (2|\tilde{R}|)^n.$$

Let $R$ be the set of remainders $r$ for which the following holds: $\exists w \in \Sigma^*$, $|w| \leq n^2 - 1$, and two states $q_1$ and $q_2$, such that $r = |c(\bar{q}, w, q_2) - c(\bar{q}, w, q_1)|$. By Lemma 3.7, $\tilde{R} \subseteq R$, so we can bound $|\tilde{R}|$ in different settings by bounding $|R|$.

**General Case**: The weights on the transitions of $G$ are incommensurable real numbers, i.e., they require "infinite precision" as binary numbers. Since each string induced by $G$ corresponds to at least one path in $G$, we have by definition of $R$ that the cardinality of this set is bounded by the number of distinct pairs of paths of length at most $n^2 - 1$. There are at most $\sum_{i=1}^{n^2-1} m^i < m^{n^2}$ such paths, where $m$ is the number of edges in $G$. Therefore $|R| < m^{2n^2}$. On the other hand, the number of strings of length at most $n^2 - 1$ is bounded by $|\Sigma|^{n^2}$. Since each of those strings can reach a pair of (not necessarily distinct) states in $G$, we have that $|R| < n^2|\Sigma|^{n^2}$. But $|\Sigma| \leq m$, so $n^2|\Sigma|^{n^2}$ is a tighter bound on $|R|$. Our first estimate follows.

**Integer Case**: The weights are non-negative integers. Fix a state $q$ and a string $w$ that reaches $q$ from the initial state. Then $c(\bar{q}, w, q)$ is in $[0, (n^2 - 1)C]_Z$. Therefore, the remainders in $R$ must also be in that range. It follows that $(2|R|)^n < (2n^2C)^n = 2^{n(2\log n + \rho + 1)}$. Since the topological bound on $|R|$ we derived for the general case does not depend on the magnitude of weights and it holds also for the case we are considering, we have that $(2|R|)^n < 2^{n(2\log n + 1 + \min(n^2 \log|\Sigma|, \rho))}$. Our second estimate follows. Notice that this results also holds for the case in which the weights are

rational numbers represented by $\rho$ bits.

**Acyclic Case**: The graph underlying $G$ is acyclic. Thus, each string induced by $G$ is of length at most $n-1$. There are $|\Sigma|^n = 2^{n \log |\Sigma|}$ such strings. Each of the strings induced by $G$ will reach exactly one state in $dta(G)$ (which is a deterministic automaton). Therefore, the number of states of $dta(G)$ is bounded by $2^{n \log |\Sigma|}$. Tightness follows from Theorem 5.10. □

Processing each tuple of state-remainders generated by **DTA** takes $O(|\Sigma|(n+m))$ time, excluding the cost of arithmetic and min operations involving two weights and/or remainders, yielding the following.

THEOREM 3.9. *Let $G$ be a WFA satisfying the twins property.* **DTA** *takes $O(|\Sigma|(n+m) \cdot \#dta(G))$ time using the $\Re$-RAM and $O(\rho|\Sigma|(n+m) \cdot \#dta(G))$ time using the $\mathcal{CO}$-RAM. For the general case, using the $\Re$-RAM, the time is $O(|\Sigma|(n+m)2^{n(2\log n + n^2 \log |\Sigma| + 1)})$. For the (rational or) integer case, using the $\mathcal{CO}$-RAM, the time is $O(\rho|\Sigma|(n+m)2^{n(2\log n + 1 + \min(n^2 \log |\Sigma|, \rho))})$. For the acyclic case, the time is $O(|\Sigma|(n+m)2^{n \log |\Sigma|})$ using the $\Re$-RAM and $O(\rho|\Sigma|(n+m)2^{n \log |\Sigma|})$ using the $\mathcal{CO}$-RAM.*

Theorems 3.8 and 3.9 do not require $G$ to be unambiguous. **DTA** terminates within the stated resource bounds on any WFA that has the twins property. Consider in the integer case the interplay between the growth of $G$ when determinized, the time complexity of the algorithm, and the magnitude of the weights.

In the acyclic case first, we have that $\#G \leq \mathcal{S} \leq 2^{n \log |\Sigma|}$, where $\mathcal{S}$ is the number of distinct strings accepted by $G$. In some sense, $\mathcal{S}$ gives the "expressive power" of $G$, i.e., how much information is compactly stored in $G$ with the aid of nondeterminism. For small weights, i.e., $\rho \leq n \log |\Sigma|$, the worst-case time complexity of the algorithm is dominated by the number of strings accepted by $G$. Therefore, we can actually "uncompact" some or all of the information contained in $G$ by eliminating nondeterminism. On the other hand, when $\rho > n \log |\Sigma|$, the bigger weights add no information and actually slow down the algorithm to the point that, for very large weights, the arithmetic and logic operations dominate the cost of the entire algorithm.

For the cyclic case, the situation is analogous, with weights playing an even more prominent role. Let $\rho_{max} = n^2 \log |\Sigma|$. For $\rho < \rho_{max}$, the estimate of $\#dta(G)$ depends on $\rho$, although we do not know how tight that estimate is. For $\rho \gg \rho_{max}$, the expansion of $G$ depends only on its topology, but the large weights slow down the algorithm.

**3.4. Computing a Worst-Case Weighting.** The results of Section 3.3 can be used to generate hard instances for any determinization algorithm. Let $G$ be a WFA. A *reweighting function* (or simply *reweighting*) $f$ is such that, when applied to $G$, it preserves the topology and labeling of $G$, but possibly changes the weights on its transitions. We want to determine a reweighting $f$ such that $min(f(G))$ exists and $\#min(f(G))$ is maximized among reweightings for which $min(f(G))$ exists. We restrict attention to the integer case and, without loss of generality, we assume that $G$ is trim and unambiguous.

Theorem 3.8 shows that for weights to have an effect on the growth of $dta(G)$, it must be that $\rho \leq n^2 \log |\Sigma|$. Set $\rho_{max} = n^2 \log |\Sigma|$. To find the required reweighting, we simply consider all possible weight assignments to $G$ satisfying the twins property and requiring at most $\rho_{max}$ bits, choosing the one that leads to the minimum deterministic equivalent of maximum size. There are $(2^{\rho_{max}})^m = 2^{m\rho_{max}}$ possible reweightings, and it takes $2^{O(n(2\log n + (n^2 \log |\Sigma|)))}$ time to compute the size expansion or decide that the resulting machine cannot be determinized. The total time is thus

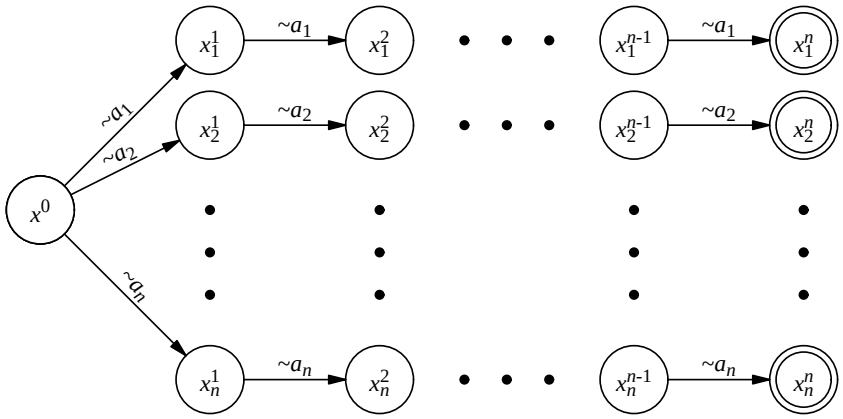


FIG. 4.1. *A nondeterministic finite-state automaton accepting language* $L = \bigcup_{i=1}^{n} (\Sigma - \{a_i\})^n$. *Arcs labeled* $\tilde{}a_i$ *denote transitions on all symbols in* $\Sigma - \{a_i\}$.

bounded by $2^{O(n(2\log n + (n^2 \log |\Sigma|)) + m\rho_{max})}$.

**4. Hot Automata.** This section provides a family of acyclic, multi-partite WFAs that are *hot*: when determinized, they expand independently of the weights on their transitions. Given some alphabet $\Sigma = \{a_1, \ldots, a_n\}$, consider the language

$$L = \bigcup_{i=1}^{n} (\Sigma - \{a_i\})^n ;$$

i.e., the set of all $n$-length strings that do not include all symbols from $\Sigma$. It is simple to obtain an acyclic, multi-partite NFA $H$ of poly($n$) size that accepts $L$. (See Figure 4.1.) One can also show that the minimal DFA accepting $L$ has $\Theta(2^{n+\log n})$ states. Furthermore, we can construct $H$ so that these bounds hold for a *binary* alphabet: encode the symbols in $\Sigma$ as binary strings of length $\log n$, and replace arcs in the above NFA with $n$-vertex, $(\log n)$-depth trees appropriately. $H$ corresponds to a WFA with all arcs weighted identically. Since acyclic WFAs satisfy the twins property, they can always be determinized. Altering the weights can only increase the expansion.

Continuing, Kintala and Wotschke [18] provide a set of NFAs that produces a hierarchy of expansion factors when determinized. Consider the set of languages

$$L_{h,k} = \{x1y \mid x, y \in \{0,1\}^*; |x| \leq k-1; |y| = k; x \text{ has at most } h \text{ 1's in it}\}$$

for $k \geq 1$, $h < k$. They show that for each $L_{h,k}$, there is an $O(k^2)$-state acyclic (but not multi-partite) NFA that accepts $L_{h,k}$, yet any DFA accepting $L_{h,k}$ must have at least $\sum_{i=0}^{2\log h} \binom{k}{i}$ states. These provide additional examples of hot WFAs.

**5. Weight-Dependent Automata.** In this section we address the effect of weights on the size of the deterministic equivalent of an input WFA. We study a simple family of WFAs with multi-partite, acyclic topology. When the arcs are all weighted zero, all WFAs in this family shrink when determinized. We show, however, that even though the topology is by itself very benign, certain weightings can cause exponential increases in size when the WFA is determinized. This study is related in spirit to works that measure amounts of nondeterminism and ambiguity in finite automata [14, 15,

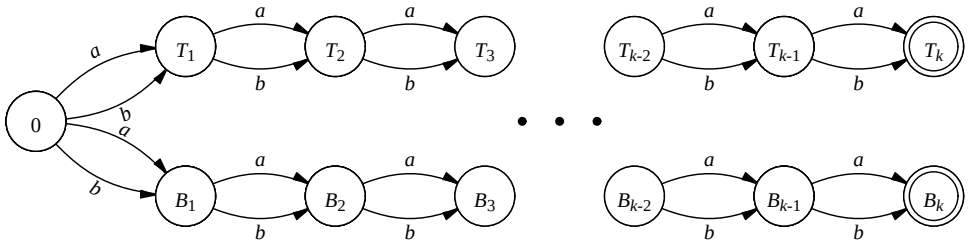


FIG. 5.1. *Topology of the k-layer rail graph.*

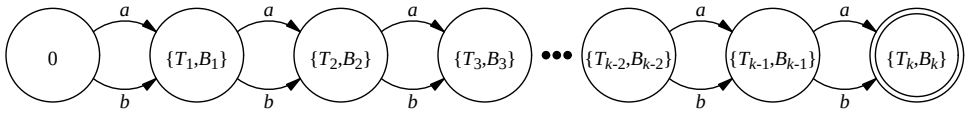


FIG. 5.2. *The result of determinizing $RG(k)$ when all arc weights are 0. In the result, all arcs are again weighted 0, and the remainders in the vertices are all 0; these values are omitted from the figure.*

18]. We first discuss the case of a binary alphabet and then generalize to arbitrary alphabets. In this section, we use the term automaton and graph interchangeably.

**5.1. The Rail Graph.** We denote by $RG(k)$ the *k-layer rail graph*. See Figure 5.1. $RG(k)$ has $2k+1$ vertices, which we denote by $\{0, T_1, B_1, \ldots, T_k, B_k\}$. There are arcs $(0, T_1, a)$, $(0, T_1, b)$ $(0, B_1, a)$, $(0, B_1, b)$, and then, for $1 \leq i < k$, arcs $(T_i, T_{i+1}, a)$, $(T_i, T_{i+1}, b)$, $(B_i, B_{i+1}, a)$, and $(B_i, B_{i+1}, b)$. (It should be clear from Figure 5.1 that $T$ stands for "top" and $B$ for "bottom.")

Note that $RG(k)$ is $(k+1)$-partite and also has fixed in- and out-degrees. (All vertices have in- and out-degrees 2, except the root, which has in-degree 0 and out-degree 4, and the vertices $T_k$ and $B_k$, which have out-degree 0.) If we consider the strings induced by paths from 0 to either $T_k$ or $B_k$, then the language of $RG(k)$ is the set of strings $L_{RG}(k) = \{a, b\}^k$. The only nondeterministic choice is at the state 0, where either the top or bottom rail may be selected. Hence a string $w$ can be accepted by one of two paths, one following the top rail and the other the bottom rail.

Technically, the rail graph is ambiguous. We can easily disambiguate $RG(k)$ by adding transitions from $T_k$ and $B_k$, each on a distinct symbol, to a new final state. Our results extend to this case. For clarity of presentation, however, we discuss the ambiguous rail graph.

The rail graph is weight-dependent. In Section 5.2 we provide weightings such that **DTA** produces a trivial $(k+1)$-vertex series-parallel graph. (See Figure 5.2 for an example.) On the other hand, in Section 5.3 we exhibit weightings for the rail graph such that, when input to **DTA**, we get an exponential increase in the number of states. (See Figures 5.3 and 5.4 for an example). Notice that we cannot get more than $2^k$ vertices, one per string in $L_{RG}(k)$, in the last layer of the determinized graph, and thus the weighting in Figure 5.3 is in some sense worst case. In that section we also explore the relationship between the magnitude of the weights and the amount of expansion that is possible. In Section 5.4, we show that random weightings induce the behavior of worst-case weightings. We discuss variants of the rail graph in Section 5.5, and finally, in Section 5.6 we generalize the rail graph to arbitrary alphabets.

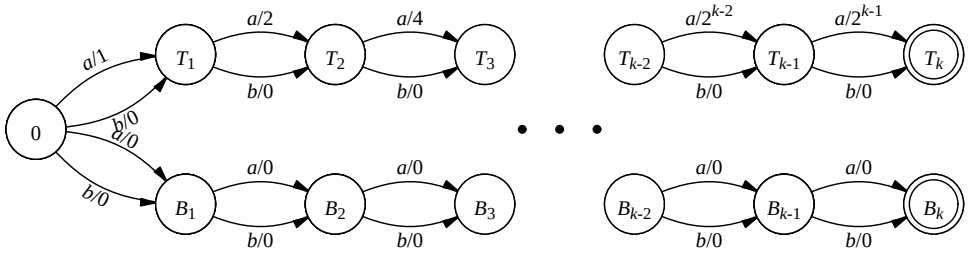


FIG. 5.3. *"Worst-case" weighting of $RG(k)$. Arc label $\sigma/w$ means the arc is labeled with symbol $\sigma$ and has weight $w$.*

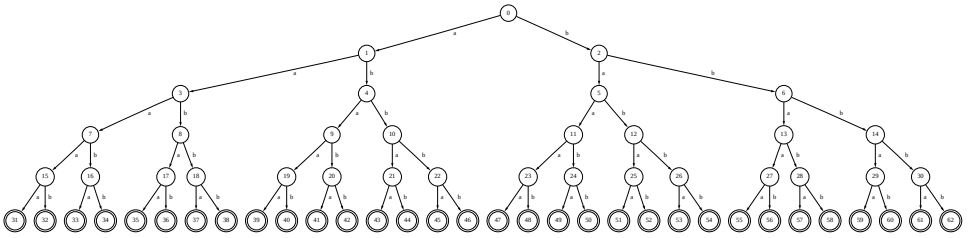

FIG. 5.4. *Result of determinizing $RG(5)$, weighted as in Figure 5.3. States have been renamed. All arcs are weighted 0. The remainders are not shown.*

**5.2. A Framework for Examining Weightings of $RG(k)$.** Consider determinizing $RG(k)$ with **DTA**. The set of states reachable on any string $w = \sigma_1 \cdots \sigma_j$ of length $j \leq k$ is $\{T_j, B_j\}$. For a given weighting function $c$, let $c_T(w)$ denote the cost of accepting string $w$ if the top path is taken; i.e.,

$$c_T(w) = c(0, \sigma_1, T_1) + \sum_{i=1}^{j-1} c(T_i, \sigma_{i+1}, T_{i+1}).$$

Analogously define $c_B(w)$ to be the corresponding cost along the bottom path. Let $R(w)$ be the *remainder vector* for $w$, which is a pair of the form $(0, c_B(w) - c_T(w))$ or $(c_T(w) - c_B(w), 0)$. A state at layer $0 < i \leq k$ in the determinized WFA is labeled $(\{T_i, B_i\}/R(w))$ for any string $w$ leading to that state; i.e., all strings leading to a particular state induce the same remainder vector. Two strings $w_1$ and $w_2$ of identical length lead to distinct states in the determinized version of the rail graph if and only if $R(w_1) \neq R(w_2)$.

It is convenient simply to write $R(w) = c_T(w) - c_B(w)$. The sign of $R(w)$ then determines which of the two forms $(0, x)$ or $(x, 0)$ of the remainder vector occurs.

Suppose that $w$ has length $j$ and can be written $w'\sigma$, where $\sigma \in \{a, b\}$. Let $r_i^T(\sigma)$ denote the weight on the (top) arc labeled $\sigma$ into vertex $T_i$ and $r_i^B(\sigma)$ denote the weight on the (bottom) arc labeled $\sigma$ into vertex $B_i$. Then we can write $R(w) = R(w') + r_j^T(\sigma) - r_j^B(\sigma)$. Abbreviating $r_i^T(\sigma) - r_i^B(\sigma)$ by $\delta_i(\sigma)$, we have

$$R(w) = \sum_{i=1}^{j} \delta_i(\sigma_i).$$

The rail graph with a specific weighting can also be regarded as a function that hashes a $k$-bit string $w$ into a number $R(w)$. Define symbol $a$ to be 0 and symbol $b$ to

be 1, so that a string $w$ can be viewed as a sequence of bits $b_1, \ldots, b_k$. We can write

$$R(b_1, \ldots, b_k) = R(b_1, \ldots, b_{k-1}) + \delta_k(b_k).$$

Also, we can write $\delta_k(b_k) = b_k \cdot \delta_k(1) + (1 - b_k)\delta_k(0)$. Rearranging gives $\delta_k(b_k) = \delta_k(0) + b_k(\delta_k(1) - \delta_k(0))$. Summing over all $i$ gives

$$R(w) = \sum_{i=1}^{k} (\delta_i(0) + b_i(\delta_i(1) - \delta_i(0))).$$

Alternatively,

$$R(w) = R_a + \sum_{i=1}^{k} b_i(\delta_i(1) - \delta_i(0)), \tag{5.1}$$

where $R_a = \sum_{i=1}^{k} \delta_k(0)$ is fixed for a given weighting function on $RG(k)$.

THEOREM 5.1. *There is a reweighting $f$ such that both dta($f(RG(k))$) and min($f(RG(k))$) realize the topology of the $(k+1)$-vertex trivial series-parallel graph (exemplified in Figure 5.2).*

*Proof.* Any weighting in which $\delta_i(a) = \delta_i(b)$ for $i = 1$ to $k$ suffices, since in this case $R(w_1) = R(w_2)$ for all pairs of strings $\{w_1, w_2\}$. In particular, giving zero weights suffices. □

**5.3. Worst-Case Weightings of $RG(k)$.** See Figures 5.3 and 5.4.

THEOREM 5.2. *For any $j \in [0, k]_Z$ there exists a reweighting $f$ such that dta($f(RG(k))$) has the following form: Layers 0 through $j$ form the complete binary tree on $2^{j+1} - 1$ vertices, and the remaining layers $j+1$ through $k$ consist of trivial series-parallel graphs, each rooted at a leaf of the tree.*

*Proof.* Choose any weighting such that $\delta_i(a) = 2^{i-1}$ and $\delta_i(b) = 0$ for $1 \le i \le j$, and let $\delta_i(a) = \delta_i(b) = 0$ for $j < i \le k$. Consider a pair of strings $w_1, w_2$ of identical length that differ in position $i' \le j$. Let $\sigma_i^1 = 1$ if the $i$th symbol of $w_1$ is $a$ and $\sigma_i^1 = 0$ otherwise; similarly define $\sigma_i^2$ with respect to $w_2$. Then we can write $R(w_1) = \sum_{i=1}^{j} \sigma_i^1 2^{i-1}$ and $R(w_2) = \sum_{i=1}^{j} \sigma_i^2 2^{i-1}$. Since $\sigma_{i'}^1 \neq \sigma_{i'}^2$, $R(w_1)$ must differ from $R(w_2)$. Hence the two strings must lead to different states. If on the other hand they differ only in positions $i' > j$ they will lead to the same state. There are $2^i$ strings that differ in positions 1 through $i$; thus for $i \le j$, there are $2^i$ distinct vertices in the $i$th layer of the graph. Since each vertex has out-degree 2, one arc for each symbol, the graph must have the desired form. □

Note that if we set all weights on the bottom rail to zero, and the weights $r_i^T(a) = 2^{i-1}$ and $r_i^T(b) = 0$ for all $1 \le i \le k$, we get a weighting that yields a complete binary tree of depth $k$ when **DTA** is applied. It is easy to show that the *minimum* deterministic graph preserving shortest paths, however, consists of a trivial series-parallel graph in which all edges have weight zero, corresponding to the lower rail. We can remedy this by choosing weights more judiciously.

THEOREM 5.3. *For any $j \in [0, k]_Z$ there is a reweighting $f$ such that both dta($f(RG(k))$) and min($f(RG(k))$) have the following form: Layers 0 through $j-1$ form the complete binary tree on $2^j - 1$ vertices, and the remaining layers $j$ through $k$ form a trivial series-parallel graph with incoming arcs to the layer-$j$ vertex from each vertex at layer $j-1$.*

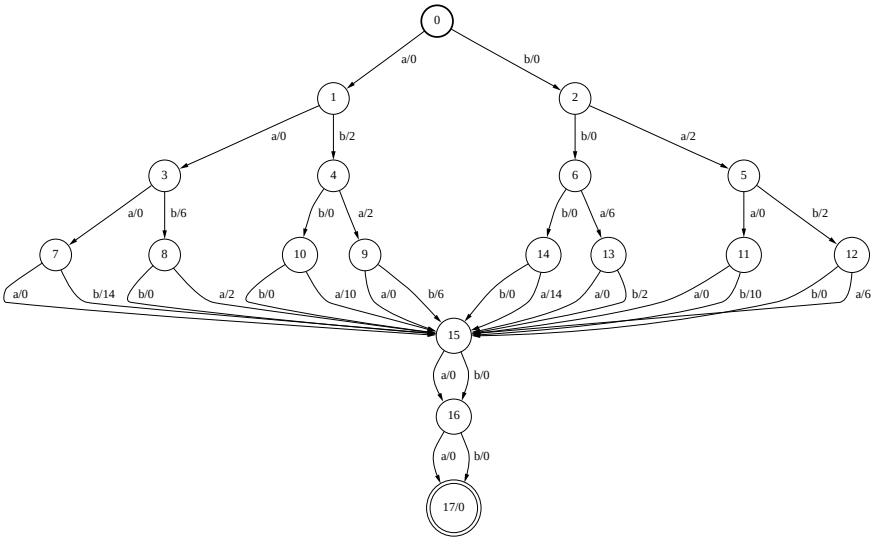


FIG. 5.5. *Result of minimizing dta(RG(6)), weighting RG(6) as follows:* $r_i^T(a) = r_i^B(b) = 2^i$ *for* $1 \le i \le 4$*; all other weights were 0. States have been renamed, and remainders are not shown.*

See Figure 5.5. Theorem 5.3 is generalized by Theorem 5.10, and therefore we omit its proof here. Theorems 5.2 and 5.3 show that the bound obtained in Theorem 3.8 for the acyclic case is tight for binary alphabets.

We now address the sensitivity of the size expansion to the magnitude of the weights. Note that $RG(k)$ has $2k+1$ vertices and $4k$ arcs, but we use $\Theta(k)$ bits to encode the weight of each arc in the proofs of Theorems 5.2 and 5.3; the input size of the WFA is thus $n = \Theta(k^2)$ bits. The determinized WFA has $2^{k+1} - 1$ states and $2^{k+1} - 2$ transitions. Again we need $\Theta(k)$ bits to encode the weight of each transition, so the bit size of the determinized WFA is $\Theta(k2^k)$, or $2^{\Theta(\sqrt{n})}$ bits. So while the determinized WFA has exponentially more states than the original WFA, the size expansion in bits, while superpolynomial, is not exponential. We argue that exponential state-space expansion requires exponentially big weights for the rail graph.

THEOREM 5.4. *Let $f$ be a reweighting. If $\#dta(f(RG(k))) = \Omega(2^k)$, then $\Omega(k^2)$ bits are required to represent $f(RG(k))$.*

*Proof.* Consider the $\Omega(2^k)$ vertices at depth $k$ in the determinized graph. Each such state is labeled by a distinct $R(w)$ for some string $w = \sigma_1 \cdots \sigma_k$. Hence if there are $\Omega(2^k)$ states in $dta(f(RG(k)))$, there must be $\Omega(2^k)$ distinct values of $R(w)$. In addition, there must be $\Omega(2^k)$ distinct values of the absolute value of $R(w)$.

Recalling the formulation of $R(w)$ from Equation (5.1), there can be at most $2^{k'}$ distinct values of $R(w)$, where $k'$ is the number of distinct values of $(\delta_i(1) - \delta_i(0))$. Each value may be included in the sum or not, and at best a choice of inclusions and exclusions will lead to a unique sum. Therefore, the assumption of $\Omega(2^k)$ distinct remainders implies there must be $\Omega(k)$ distinct values of the $(\delta_i(1) - \delta_i(0))$.

Now ignore the $\lfloor \frac{k}{2} \rfloor$ low-order bits of the absolute values of the remainders, and consider only the remaining high-order bits. There must be $\Omega(2^{\lceil \frac{k}{2} \rceil})$ distinct values induced by the high-order bits, or else there cannot be $\Omega(2^k)$ distinct values overall. By the same argument as above, there must be $\Omega(\lceil \frac{k}{2} \rceil)$ distinct values of the $(\delta_i(1) - \delta_i(0))$ that affect the high-order bits of a *large* remainder, i.e., one with one of its high-order

$\lceil \frac{k}{2} \rceil$ bits set to 1.

For a particular $(\delta_i(1) - \delta_i(0))$ to affect a high-order bit of a large remainder, $(\delta_i(1) - \delta_i(0))$ must have a non-zero bit at least as high as position $\lceil \frac{k}{2} \rceil - \log k$. This is only true if one of the four arc weights for layer $i$ has a non-zero bit at least that high. Therefore, $\Omega(k)$ arc weights require some non-zero bit at least as high as position $\lceil \frac{k}{2} \rceil - \log k$. Hence $\Omega(k^2)$ bits are required to represent all the arc weights. □

COROLLARY 5.5. *Let $f$ be a reweighting. If $\#min(f(RG(k))) = \Omega(2^k)$, then $\Omega(k^2)$ bits are required to represent $f(RG(k))$.*

*Proof.* Theorem 5.4 applies, because $\#min(f(RG(k))) = \Omega(2^k)$ implies that $\#dta(f(RG(k))) = \Omega(2^k)$. □

Finally, consider the following analogy between the hot graphs in Section 4 and the rail graph. Observe that the hot graphs in Section 4 contain some nondeterministic choices that cannot be resolved until the end of the input. This causes the respective deterministic expansions. In those graphs, these choices are part of the strings being accepted. The rail graph manifests this same phenomenon, but in terms of weights rather than strings. The weighted variants of the rail graph that expand when determinized do so because it is not clear until the end of the expansion which rail will provide the shorter path: at any point, the choice of top or bottom rail depends on the symbols that follow. Therefore, the determinization must maintain enough state information to provide for all possible outcomes. Furthermore, in the non-minimizable cases, whereas the language $L_{RG}(k)$ itself could be accepted by a $(k+1)$-state DFA, the weights on $RG(k)$ necessitate an exponential number of states and arcs in any deterministic WFA that induces all the appropriate path lengths.

**5.4. Random Weightings of $RG(k)$.** An *$i$-bit reweighting function* (or simply *$i$-bit reweighting*) is a reweighting function $f$ such that the weights on the arcs of $f(G)$ are constrained to be in $[0, 2^i - 1]_Z$. A function $f^R$ is a *random reweighting function* (or simply *random reweighting*) if and only if it chooses the weights to assign to the transitions of $G$ uniformly and independently at random from $\Re^+ \cup \{0, \infty\}$. Finally, let $x \in_R Y$ denote that $x$ is selected uniformly and independently at random from set $Y$, and let $E[X]$ denote the expected value of some random variable $X$. We need the following technical claim.

CLAIM 5.6. *Let $X, Y, U, V \in_R [0, 2^k - 1]$. Then*

$$\max_{-2^{k+1}+1 < i < 2^{k+1}-1} \Pr(X - Y - (U - V) = i) \leq \frac{2}{3 \cdot 2^k} + O\left(1/4^k\right).$$

*Proof.* See Appendix A.1. □

THEOREM 5.7. *Let $f^R$ be a random $k$-bit reweighting. $E[\#dta(f^R(RG(k)))] = \Theta(2^k)$.*

*Proof.* As before, let $R(w)$ be the remainder induced by string $w$ of length $k$; i.e., the difference between the cost of the upper path and the cost of the lower path that respectively induce $w$. Let $\delta_i(\sigma)$ be the cost of the (top) arc labeled $\sigma$ into vertex $T_i$ minus the cost of the (bottom) arc labeled $\sigma$ into vertex $B_i$. Recall from Equation (5.1) that the rail graph with a specific weighting can be regarded as a function that hashes a $k$-bit string $w = \sigma_1 \cdots \sigma_k$ into a number $R(w)$.

Suppose $w_1 \neq w_2$. We can adapt a standard analysis from the theory of universal hash functions [7] to calculate the probability that $R(w_1) = R(w_2)$. Let $w_1 = \alpha_1 \cdots \alpha_k$ and $w_2 = \beta_1 \cdots \beta_k$. Without loss of generality, assume $\alpha_k \neq \beta_k$. (The strings must

differ somewhere.) Suppose $R(w_1) = R(w_2)$. Then by definition (with some manipulation)

$$(\alpha_k - \beta_k)(\delta_k(1) - \delta_k(0)) = \sum_{i=1}^{k-1}(\beta_i - \alpha_i)(\delta_i(1) - \delta_i(0)).$$

Fix a set of weights on the arcs in the first $k-1$ layers, so the right hand side of the equation is a constant. The value $(\alpha_k - \beta_k)$ is either $-1$ or 1, by assumption. Hence, for the given set of weights on the first $k-1$ layers, there is exactly one value of $(\delta_k(1) - \delta_k(0))$ that makes $R(w_1) = R(w_2)$. If we assign the weights randomly in the $k$th layer, the probability that $(\delta_k(1)-\delta_k(0))$ has the necessary value is upper-bounded by

$$\max_{-2^{k+1}+1<C<2^{k+1}-1} \Pr[X_1 - X_2 - (X_3 - X_4) = C],$$

where the $X_i$s are uniform random variables in the range $[0, 2^k - 1]_Z$. (It is strictly upper-bounded, since the value of $(\delta_k(1) - \delta_k(0))$ lies in $[-2^{k+1} + 2, 2^{k+1} - 2]$, which is in general smaller than the range of values that the right hand side of the equation can assume.)

By Claim 5.6, therefore, for any fixed assignment of the first $k-1$ values, the probability that $R(w_1) = R(w_2)$ is bounded above by $\frac{2}{3\cdot 2^k} + O(1/4^k)$. Hence, summing over all possible assignments to the the first $k-1$ values, the total probability that $R(w_1) = R(w_2)$ is also bounded above by $\frac{2}{3\cdot 2^k} + O(1/4^k)$. We will use this result to compute the expected number of vertices at layer $k$ in the determinized graph.

First, we compute the expected number of *collisions*: the number of strings of length $k$ that have the same remainder. Since the number of strings is $2^k$, this is simply

$$\sum_{w_1,w_2} \left( \frac{2}{3 \cdot 2^k} + O(1/4^k) \right) = \frac{2}{3} 2^k + O(1).$$

Suppose $x_1, \ldots, x_\ell$ form the set of layer-$k$ vertices in the deterministic graph. Let $S_i$ be the set of strings that map to vertex $x_i$, and let random variable $s_i$ be $|S_i|$. The expected number of collisions can thus be written as $E[\sum_{i=1}^{\ell} \binom{s_i}{2}]$. Since the expected value is $\frac{2}{3}2^k + O(1)$, it must be the case that in one-half of the weight assignments, the actual value of the sum is at most $\frac{4}{3}2^k + O(1)$.

We now consider the minimum number of sets so that there is an assignment of strings to sets that satisfies

$$\sum_{i=1}^{\ell} \binom{s_i}{2} \leq \frac{4}{3} 2^k + O(1)$$

and

$$\sum_{i=1}^{\ell} s_i = 2^k.$$

Elementary calculus shows that the binomial sum is minimized when the $s_i$s are all equal, which implies that the minimal value of $\ell$ is $\Omega(2^k)$.

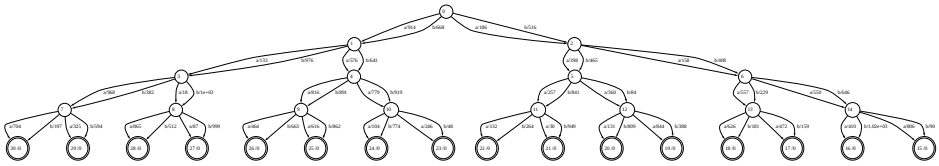

FIG. 5.6. *A rail tree of depth four, with ten-bit random weights.*

Therefore, in at least half the weight assignments, there are $\Omega(2^k)$ distinct remainders and so $\Omega(2^k)$ distinct states in the determinized graph. Hence the expected number of states in the determinized graph is also $\Omega(2^k)$. □

Since the arcs weights of ASR WFAs are (negated) log probabilities, we consider the behavior of determinization when the arcs are weighted with *log-random* numbers, i.e., logarithms of uniform random variables. On the rail graph, we find the same behavior with log-random weights as with random weights, including the increasing expansion as more bits are employed. As the following claim shows, retaining enough bits of the log-random weights drives the collision probability (as discussed in the proof of Theorem 5.7) low enough to ensure this behavior.

CLAIM 5.8. *Let* $X, Y, V, Z \in_R [1, 2^k - 1]$. *Let* $R = \lfloor 2^b \log X \rfloor$, $S = \lfloor 2^b \log Y \rfloor$, $T = \lfloor 2^b \log V \rfloor$, *and* $U = \lfloor 2^b \log Z \rfloor$, *for some* $b \geq k - O(1)$. *Then*

$$\max_{-k2^{b+1}+1<i<k2^{b+1}-1} \Pr(R - S - (T - U) = i) = \frac{1}{4 \cdot 2^k} + O\left(1/4^k\right).$$

*Proof.* See Appendix A.2. □

Thus we derive the analogue to Theorem 5.7 for log-random weights.

THEOREM 5.9. *Let* $G$ *be* $RG(k)$ *weighted with logarithms of numbers chosen independently and uniformly at random from* $[1, 2^k - 1]_Z$. *Then* $E[\#dta(G)] = \Theta(2^k)$.

*Proof.* Apply Claim 5.8 instead of Claim 5.6 in the proof of Theorem 5.7, and alter the ensuing discussion appropriately to account for the respective constants. □

**5.5. Variations of the Rail Graph.** We remark that all of the above results extend even when we simplify the rail graph as follows.

1. Eliminate the $b$-arcs out of state 0.
2. Allow the symbols on the arcs at any layer $i$ to differ from $a$ and $b$. Enforce only that the symbols on the top rail at layer $i$ be the same as those on the bottom rail at layer $i$.

Using the original definition of $RG(k)$ simplifies the presentation of the theorems and proofs.

Although the rail graph allows us to investigate the relationship between deterministic expansion and weighting in WFAs, the general problem of characterizing this relationship seems difficult. Consider, for example, a *rail tree*, as in Figure 5.6, in which the rail graph is extended into a tree in a straightforward manner. As shown in Figure 5.7, when determinized, the resulting tree has the same number of vertices and half the arcs of the original. This is because the language of the original tree contains only $2^4$ distinct strings, and so the determinized tree can have only $2^4$ leaves. So while many individual rail graphs are embedded in the rail tree, when determinized, it is as if only one expands, at the expense of the others.

**5.6. Extending $RG(k)$ to Arbitrary Alphabets.** We can extend the rail graph to arbitrary alphabets, defining $RG(r, k)$, the *k-layer r-rail graph*, as follows.

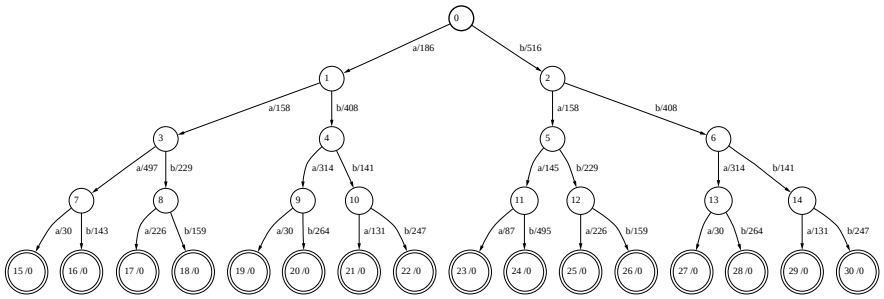


FIG. 5.7. *The result of determinizing the rail tree of Figure 5.7.*

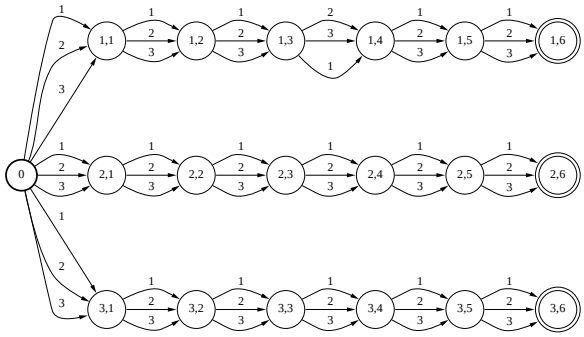


FIG. 5.8. *Topology of* $RG(3,6)$.

$RG(r,k)$ has $rk+1$ vertices: vertex 0 and, for $1 \le i \le r$ and $1 \le j \le k$, vertex $v_j^i$. Assume the alphabet is $\{1,\ldots,r\}$. $RG(r,k)$ has arcs $(0, v_1^i, s)$ for all $1 \le i, s \le r$ and also arcs $(v_j^i, v_{j+1}^i, s)$ for all $1 \le i, s \le r$ and $1 \le j < k$. See Figure 5.8.

The subgraph induced by vertex 0 and vertices $v_j^i$ for some $i$ and all $1 \le j \le k$ comprises *rail* $i$ of $RG(r,k)$. The subgraph induced by vertices $v_j^i$ for all $1 \le i \le r$ and some $j$ comprises *layer* $j$ of $RG(r,k)$. Vertex 0 comprises *layer* 0 of $RG(r,k)$. Thus, $RG(2,k)$ is the $k$-layer rail graph, $RG(k)$, defined in Section 5.1.

Denote by $c(i,j,s)$ the weight of the arc labeled $s$ into vertex $v_j^i$. Theorem 5.3 generalizes to $RG(r,k)$ as follows. (See Figure 5.9.)

THEOREM 5.10. *For any* $j \in [0,k]_Z$ *there is an assignment of weights such that the minimal deterministic realization of* $RG(r,k)$ *has the following form: Layers* 0 *through* $j-1$ *form the complete* $r$*-ary tree on* $\frac{r^j-1}{r-1}$ *vertices, and the remaining layers* $j$ *through* $k$ *form a trivial series-parallel graph with incoming arcs to the layer-*$j$ *vertex from each vertex at layer* $j-1$.

*Proof.* Choose the following weighting. Set $c(i,\ell,s) = [(i+s) \mod r] \cdot r^\ell$ for all $1 \le i, s \le r$ and $1 \le \ell \le j$. Set $c(i,\ell,s) = 0$ for all $1 \le i, s \le r$ and $j < \ell \le k$.

We show that there are no two strings of length less than $j$ that lead to the same vertex in *any* deterministic realization of the graph that preserves shortest paths. Hence the number of vertices at layer $\ell < j$ is equal to the number of strings of length $\ell$, which is $r^\ell$. The proof is by contradiction. Suppose that strings $w_1 \ne w_2$ lead to the same vertex $x$. Let $\ell < j$ be the length of strings $w_1$ and $w_2$. Now consider all

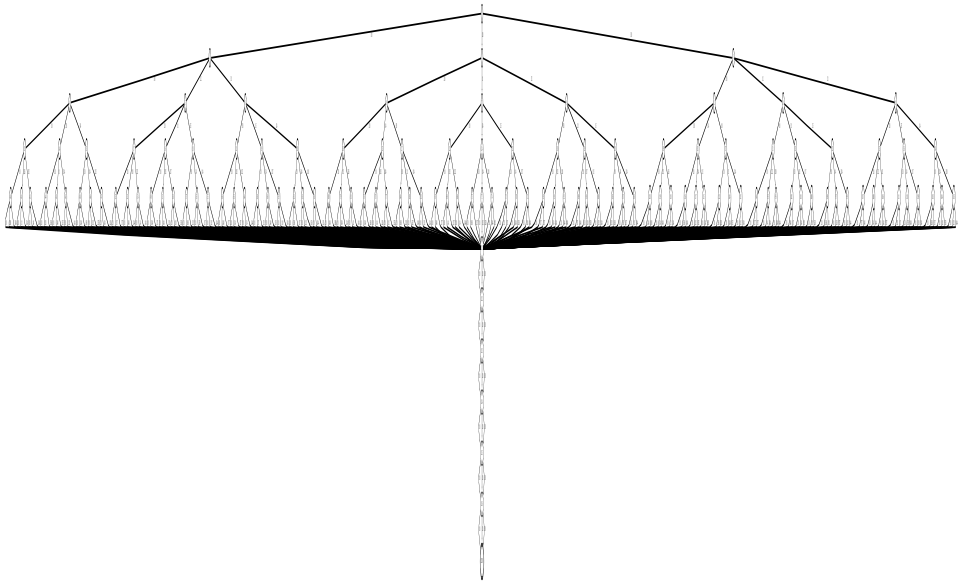

FIG. 5.9. *Result of minimizing dta*$(RG(3,10))$*, weighting* $RG(3,10)$ *as follows.* $c(i,\ell,s) = [(i+s) \mod r] \cdot r^\ell$ *for all* $1 \le i,s \le 3$ *and* $1 \le \ell \le 4$. $c(i,\ell,s) = 0$ *for all* $1 \le i,s \le 3$ *and* $4 < \ell \le 10$. *Only topology is shown.*

strings that have $w_1$ or $w_2$ as a prefix. Let $\{w'_1, w'_2, \ldots, w'_m\}$ be the set of possible suffixes, of length $k-\ell$, that can follow any of these prefixes. (In our case, this is simply the set $\{1,\ldots,r\}^{k-\ell}$.)

Let $c(w_1w'_i)$ and $c(w_2w'_i)$ denote the costs of strings $w_1w'_i$ and $w_2w'_i$, respectively, in the nondeterministic graph, i.e., the sums of the costs of the arcs along the shortest paths inducing the respective strings in $RG(r,k)$. The deterministic graph must satisfy $c(w_1w'_i) = c_d(w_1) + c_d(w'_i|x)$ and $c(w_2w'_i) = c_d(w_2) + c_d(w'_i|x)$, for all $1 \le i \le m$, where $c_d$ denotes the cost function in the deterministic graph, and $c_d(w'_i|x)$ denotes the cost of string $w'_i$ starting from vertex $x$. Hence we have that for all $i$, $c(w_1w'_i) - c(w_2w'_i) = c_d(w_1) - c_d(w_2)$. The right hand side of this equation is a fixed value, say $\Delta$.

Now consider the pair of suffixes $x\sigma_1 y$ and $x\sigma_2 y$, such that $\sigma_1 \ne \sigma_2$ and, for $i \in \{1,2\}$, $|\sigma_i| = 1$, $|x\sigma_i| = j-\ell$, and $|x\sigma_i y| = k-\ell$; i.e., $x\sigma_i y$ appended to $w_1$ or $w_2$ results in a string of length $k$ with $\sigma_i$ as the $j$th symbol. We claim that there exists some choice of $\sigma_1, \sigma_2 \in \{1,\ldots,r\}$ such that $c(w_1x\sigma_1y) - c(w_2x\sigma_1y) \ne c(w_1x\sigma_2y) - c(w_2x\sigma_2y)$. This contradicts the requirement that both differences should be equal to $\Delta$ and hence proves the theorem.

To see the claim, observe that the given weighting on $RG(r,k)$ forces the minimum cost path for any string with some symbol $\sigma$ in position $j$ to follow rail $(r-\sigma)$. Consider any position $i \le \ell$ in which $w_1$ and $w_2$ differ. Denote the $i$th symbol of $w_1$ (rsp., $w_2$) by $w_1(i)$ (rsp., $w_2(i)$). Then that position contributes $[(w_1(i)+r-\sigma_1) \mod r - (w_2(i)+r-\sigma_1) \mod r] \cdot r^i$ to the difference $\Delta_1 = c(w_1x\sigma_1y) - c(w_2x\sigma_1y)$ and $[(w_1(i)+r-\sigma_2) \mod r - (w_2(i)+r-\sigma_2) \mod r] \cdot r^i$ to the difference $\Delta_2 = c(w_1x\sigma_2y) - c(w_2x\sigma_2y)$. Picking $\sigma_1 = w_1(i)$ and $\sigma_2 = w_2(i)$ implies that position $i$ contributes $\delta_1 = -[(w_2(i) - w_1(i)) \mod r] \cdot r^i$ to $\Delta_1$ and $\delta_2 = [(w_1(i) - w_2(i)) \mod r] \cdot r^i$ to $\Delta_2$. $\Delta_1$ and $\Delta_2$ are sums of the form $\sum_{i=1}^{j} c_i r^i$ where $-r < c_i < r$. For

some $1 \leq c_1, c_2 < r$, $\delta_1 = -c_1 r^i$ and $\delta_2 = c_2 r^i$ are the only terms involving $r^i$ in $\Delta_1$ and $\Delta_2$, respectively. It follows that $\Delta_1 \neq \Delta_2$, thereby proving the claim. □

We point out that Theorem 5.10 shows that the bounds in Theorem 3.8 for the acyclic case are tight for general alphabets. We also point out that Theorems 5.1 and 5.2 generalize easily to the $k$-layer $r$-rail graphs. As for the remaining results stated for binary alphabets in Sections 5.3 and 5.4, it would be of some interest to extend them to arbitrary alphabets: while those extensions would not change the "nature" of the lower bound stated in Theorem 5.4, they might shed some more light on the relationship between hashing and **DTA**.

**6. Experimental Observations on ASR WFAs.** In this section, we study whether the WFAs that are generated by ASR systems are weight-dependent. This possibility had not previously been considered by the ASR community, and if true, it suggests that the weights of ASR WFAs, which are generated from training sets with considerable manual intervention, must be carefully monitored to preserve their favorable characteristics.

**6.1. Data.** We experimented on 100 WFAs generated by the AT&T North American Business speech recognizer [27], using a grammar for the Air Travel Information System (ATIS), a standard 5000-word vocabulary DARPA test bed [4]. Each transition was labeled with a word from the ATIS vocabulary and weighted by the recognizer with the negated log of the probability of realizing that transition out of the source state; we refer to these weights as *speech weights*.

**6.2. Method.** For each input WFA, we varied the weights on the given topology. We determinized each with its speech weights, with zero weights, and with weights assigned independently and uniformly at random from the integral range $[0, 2^i - 1]$ (for each $0 \leq i \leq 8$). One WFA could not be determinized with speech weights due to computational limitations, and it is omitted from the data. The experiments were run on an SGI R4400 processor with 1 GB of main memory and an SGI R10000 processor with 1.5 GB of main memory.

Figure 6.1 shows how many WFAs expanded when determinized with different weightings. Figure 6.2 classifies the 63 WFAs that expanded with at least one weighting. For each WFA, we took the weighting that produced maximal expansion. This was usually the 8-bit random weighting, although due to computational limitations we were unable to determinize some WFAs with large random weightings. The $x$-axis indicates the open intervals within which the value $\log(|dta(G)|/|G|)$ falls.

Figure 6.3 provides additional classification of the eighteen WFAs that expanded when determinized with speech weights. The $x$-axis indicates the open intervals within which the ratio $|dta(G)|/|G|$ falls when $G$ is determinized with its speech weights.

Since the utility of determinization in ASR includes the reduction in size achieved with actual speech weights, we provide some data on how much the WFAs shrink. In our sample, 82 WFAs shrank when determinized. For each of these WFAs, we computed the value $\log(|G|/|dta(G)|)$. In Figure 6.4, we plot the number of WFAs whose corresponding values fell in various ranges.

Figure 6.5 demonstrates the relationship between the (log of the) expansion ratio $|dta(G)|/|G|$ and the number of bits used in the random weights, for the ten WFAs with highest final expansion value. For reference the functions $i^2$, $2^{\sqrt{i}}$, and $2^i$ are plotted, where $i$ is the number of bits. Most of the WFAs exhibit subexponential growth as the number of bits increases, although some, like `q0t063` have increased by 128 times even with four random bits.

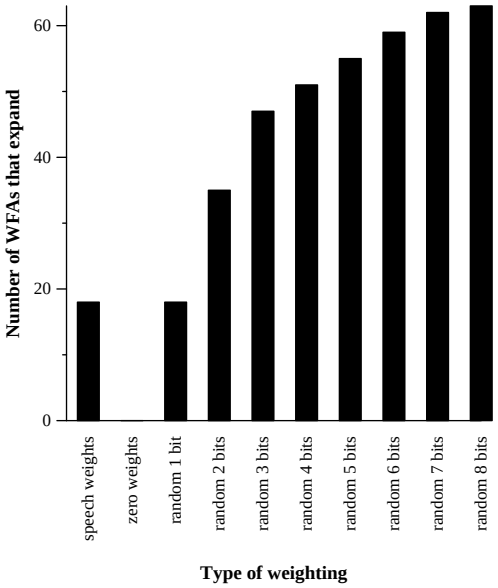


FIG. 6.1. *Number of WFAs that expanded when determinized with various weightings.*

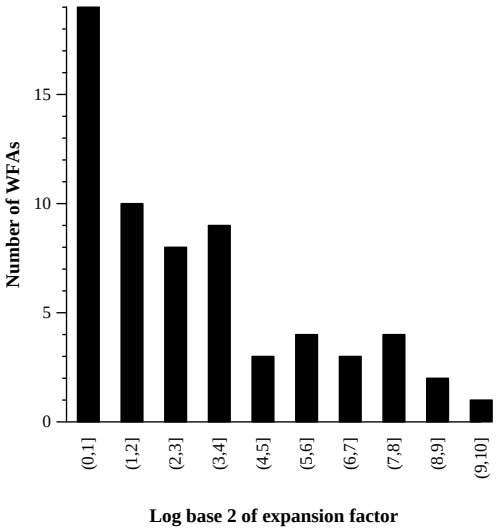


FIG. 6.2. *Histogram of expansion factors.*

**6.3. Discussion.** The WFA that could not be determinized with speech weights was "slightly hot," in that the determinized zero-weighted variant had 2.7% more arcs than the original WFA. The remaining ninety-nine WFAs shrank with zero weights: none was hot. If one expanded, it did so due to weights rather than topology.

Figure 6.1 indicates that many of the WFAs have some degree of weight dependence, since 63 expanded when sufficiently large random weights were used. Finally, Figure 6.5 suggests that random weights are a good way to estimate the degree to which a WFA is weight dependent. Note that the expansion factor is some superlinear, possibly exponential, function of the number of random bits. This suggests that using

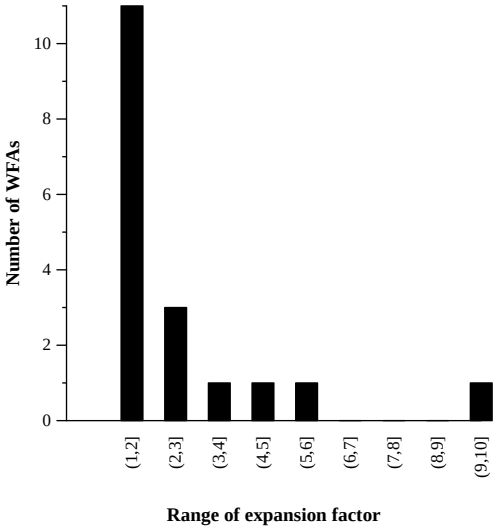


FIG. 6.3. *Expansion factors for WFAs that expanded when determinized with speech weights.*

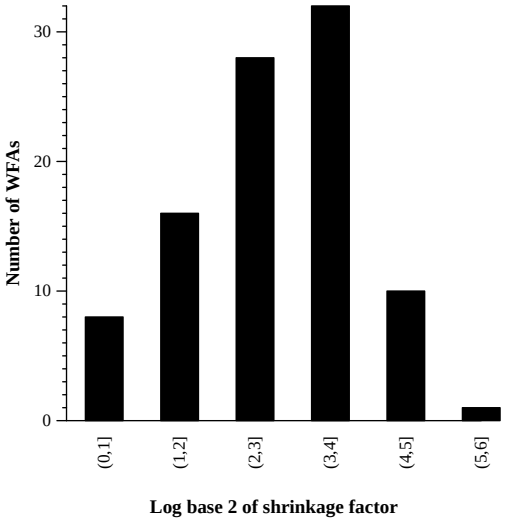


FIG. 6.4. *Shrinkage factors for WFAs that shrank when determinized with at least one weighting.*

large random weights, 32 bits for example, should cause expansion if anything will. Analogous experiments on the minimized determinized WFAs yield results that are qualitatively the same, although fewer WFAs still expand after minimization. Hence weight dependence seems to be a fundamental property of these WFAs rather than an artifact of this particular determinization algorithm.

We have used the analyses and experimental observations above to design an *approximate* WFA determinization algorithm: a variant of Mohri's algorithm that unifies state tuples whose remainders differ within a specified relative factor. Using this algorithm, we achieve size reductions in ASR language models that significantly exceed those of previous methods, with negligible effects on ASR performance (time and accuracy). We detail the algorithm and these results separately [6].

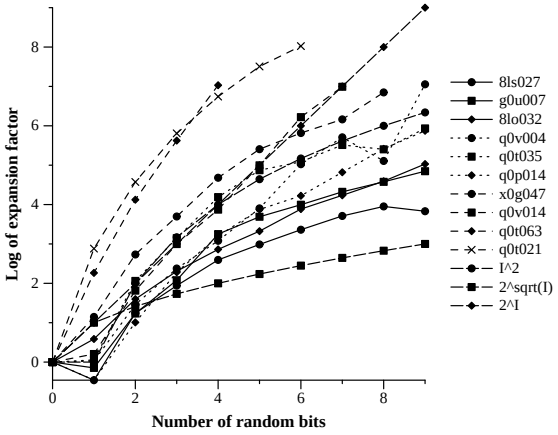


FIG. 6.5. *Relationship between expansion ratio and number of bits.*

The experimental results show that ASR WFAs are generally not hot. This suggests the following open problem: characterize a class of WFAs that are not hot and to which many of the ASR examples belong. Such a characterization might be useful in generating ASR WFAs, so that hot WFAs are guaranteed not to occur.

**Acknowledgements.** We thank Mehryar Mohri, Fernando Pereira, and Antonio Restivo for fruitful discussions. Additionally, Fernando provided us with the WFAs we used in our experiments.

## REFERENCES

[1] R. K. AHUJA, T. L. MAGNANTI, AND J. B. ORLIN, *Network Flows: Theory, Algorithms, and Applications*, Prentice-Hall, 1993.

[2] J. BERSTEL, *Transduction and Context-Free Languages*, vol. 38 of Leitfaden der angewandten Mathematik und Mechanik LAMM, Springer-Verlag, 1979.

[3] J. BERSTEL AND C. REUTENAUER, *Rational Series and Their Languages*, vol. 12 of EATCS Monographs on Theoretical Computer Science, Springer-Verlag, 1988.

[4] E. BOCCHIERI, G. RICCARDI, AND J. ANANTHARAMAN, *The 1994 AT&T ATIS CHRONUS recognizer*, in Proc. ARPA Spoken Language Technology Workshop, 1995, pp. 265–8.

[5] A. L. BUCHSBAUM AND R. GIANCARLO, *Algorithmic aspects in speech recognition: An introduction*, ACM Journal of Experimental Algorithmics, 2 (1997). `http://www.jea.acm.org`.

[6] A. L. BUCHSBAUM, R. GIANCARLO, AND J. R. WESTBROOK, *Shrinking language models by robust approximation*, in Proc. IEEE Int'l. Conf. on Acoustics, Speech, and Signal Processing '98, vol. 2, 1998, pp. 685–8. To appear in *Algorithmica* as, "An Approximate Determinization Algorithm for Weighted Finite-State Automata".

[7] J. L. CARTER AND M. N. WEGMAN, *Universal classes of hash functions*, Journal of Computer and System Sciences, 18 (1979), pp. 143–54.

[8] C. CHOFFRUT, *Une caracterisation des fonctions sequentielles et des fonctions sous-sequentielles en tant que relations rationnelles*, Theoretical Computer Science, 5 (1977), pp. 325–37.

[9] ———, *Contributions á l'étude de quelques familles remarquables de function rationnelles*, PhD thesis, LITP-Université Paris 7, Paris, France, 1978.

[10] K. CULIK II AND J. KARHUMÄKI, *Finite automata computing real functions*, SIAM Journal on Computing, 23 (1994), pp. 789–814.

[11] K. CULIK II AND P. RAJČÁNI, *Iterative weighted finite transductions*, Acta Informatica, 32 (1995), pp. 681–703.

[12] D. DERENCOURT, J. KARHUMÄKI, M. LATTEUX, AND A. TERLUTTE, *On computational power of weighted finite automata*, Fundamenta Informaticae, 25 (1996), pp. 285–93.

[13] S. EILENBERG, *Automata, Languages, and Machines*, vol. A, Academic Press, San Diego, 1974.

[14] J. Goldstine, C. M. R. Kintala, and D. Wotschke, *On measuring nondeterminism in regular languages*, Information and Computation, 86 (1990), pp. 179–94.
[15] J. Goldstine, H. Leung, and D. Wotschke, *On the relation between ambiguity and nondetermism in finite automata*, Information and Computation, 100 (1992), pp. 261–70.
[16] L. Grove, *Algebra*, Academic Press, 1983.
[17] J. Kari and P. Fränti, *Arithmetic coding of weighted finite automata*, RAIRO Informatique Théorique et Applications, 28 (1994), pp. 343–60.
[18] C. M. R. Kintala and D. Wotschke, *Amounts of nondeterminism in finite automata*, Acta Informatica, 13 (1980), pp. 199–204.
[19] W. Kuich and A. Salomaa, *Semirings, Automata, Languages*, vol. 5 of EATCS Monographs on Theoretical Computer Science, Springer-Verlag, 1986.
[20] M. Mohri, *Finite-state transducers in language and speech processing*, Computational Linguistics, 23 (1997), pp. 269–311.
[21] ———, *On the use of sequential transducers in natural language processing*, in Finite-State Language Processing, MIT Press, 1997.
[22] ———, *Minimization algorithms for sequential transducers*, Theoretical Computer Science, 234 (2000), pp. 177–201.
[23] F. Pereira and M. Riley, *Speech recognition by composition of weighted finite automata*, in Finite-State Language Processing, MIT Press, 1997.
[24] F. Pereira, M. Riley, and R. Sproat, *Weighted rational transductions and their application to human language processing*, in Proc. ARPA Human Language Technology Conf., 1994, pp. 249–54.
[25] F. P. Preparata and M. I. Shamos, *Computational Geometry: An Introduction*, Texts and Monographs in Computer Science, Springer-Verlag, New York, 1988.
[26] M. O. Rabin, *Probabilistic automata*, Information and Control, 6 (1963), pp. 230–45.
[27] M. D. Riley, A. Ljolje, D. Hindle, and F. C. N. Pereira, *The AT&T 60,000 word speech-to-text system*, in Proc. 4th Euro. Conf. on Speech Communication and Technology, vol. 1, 1995, pp. 207–210.
[28] E. Roche, *Analyse Syntaxique Transformationelle du Francais par Transducteurs et Lexique-Grammaire*, PhD thesis, LITP-Université Paris 7, Paris, France, 1993.
[29] A. Salomaa and M. Soittola, *Automata-Theoretic Aspects of Formal Power Series*, Springer-Verlag, 1978.
[30] M. Silberztein, *Dictionnaires électroniques et analise automatique de textes: le systéme INTEX*, PhD thesis, Masson, Paris, France., 1993.
[31] A. Weber and R. Klemm, *Economy of description for single-valued transducers*, Information and Computation, 118 (1995), pp. 327–40.

## Appendix A. Proofs of Claims.

Lemma A.1. *Let $Y_1$ and $Y_2$ be random variables from the same probability distribution. Let $X$ be a random variable such that*

$$\Pr(X = i) = \sum_{j=B+|i|}^{T} \Pr(Y_1 = j) \Pr(Y_2 = j - |i|)$$

*for any $i$ and some fixed $B$ and $T$. Then $\Pr(X = i) \leq \Pr(X = 0)$ for any $i$.*

*Proof.* Using the fact $ab \leq \frac{1}{2}\left(a^2 + b^2\right)$, we derive

$$\begin{aligned}
\Pr(X = i) &= \sum_{j=B+|i|}^{T} \Pr(Y_1 = j) \Pr(Y_2 = j - |i|) \\
&\leq \frac{1}{2} \sum_{j=B+|i|}^{T} \left(\Pr(Y_1 = j)^2 + \Pr(Y_2 = j - |i|)^2\right) \\
&= \frac{1}{2} \sum_{j=B+|i|}^{T} \Pr(Y_1 = j)^2 + \frac{1}{2} \sum_{j=B+|i|}^{T} \Pr(Y_2 = j - |i|)^2. \qquad \text{(A.1)}
\end{aligned}$$

By definition, $\Pr(X=0) = \sum_{j=B}^{T} \Pr(Y_1 = j)\Pr(Y_2 = j)$. By symmetry, therefore,

$$\Pr(X=0) = \sum_{j=B}^{T} \Pr(Y_1 = j)^2 = \sum_{j=B}^{T} \Pr(Y_2 = j)^2,$$

which, combined with Equation (A.1), completes the proof. □

**A.1. Proof of Claim 5.6.** *Let* $X, Y, U, V \in_R [0, 2^k - 1]$. *Then*

$$\max_{-2^{k+1}+1<i<2^{k+1}-1} \Pr(X - Y - (U - V) = i) \leq \frac{2}{3 \cdot 2^k} + O\left(1/4^k\right).$$

*Proof.* It should be clear that, for $-(2^k - 1) \leq i \leq 2^k - 1$,

$$\Pr(X - Y = i) = \Pr(U - V = i) = \left(\frac{1}{2^k}\right)^2 \left(2^k - |i|\right).$$

Consider $\Pr(X - Y - (U - V) = i)$ and the following cases.

$$\Pr(X - Y - (U - V) = i \mid i \geq 0) = \sum_{j=-2^k+1+i}^{2^k-1} \Pr(X - Y = j)\Pr(U - V = j - i).$$

$$\Pr(X - Y - (U - V) = i \mid i < 0) = \sum_{j=-2^k+1-i}^{2^k-1} \Pr(X - Y = j + i)\Pr(U - V = j).$$

Unifying the cases and exploiting symmetry yields

$$\Pr(X - Y - (U - V) = i) = \sum_{j=-2^k+1+|i|}^{2^k-1} \Pr\left(X - Y = j\right)\Pr\left(U - V = j - |i|\right).$$

By Lemma A.1, it suffices to solve for $i = 0$.

$$\begin{aligned}
\Pr(X - Y - (U - V) = 0) &= \sum_{j=-2^k+1}^{2^k-1} \Pr(X - Y = j)\Pr(U - V = j) \\
&= \sum_{j=-2^k+1}^{2^k-1} \left(\frac{1}{2^k}\right)^4 \left(2^k - |j|\right)^2 \\
&= \left(\frac{1}{2^k}\right)^4 \left[\left(2^k\right)^2 + 2\sum_{j=1}^{2^k-1} \left(2^k - j\right)^2\right] \\
&= \left(\frac{1}{2^k}\right)^4 \left[\left(2^k\right)^2 + 2\sum_{j=1}^{2^k-1} j^2\right] \\
&= \left(\frac{1}{2^k}\right)^4 \left[\left(2^k\right)^2 + 2\left(\frac{\left(2^k - 1\right)\left(2^k\right)\left(2^{k+1} + 1\right)}{6}\right)\right] \\
&\approx \left(\frac{1}{2^k}\right)^4 \left[\left(2^k\right)^2 + \frac{2}{3}\left(2^k\right)^3\right] \\
&= \frac{2}{3 \cdot 2^k} + O\left(1/4^k\right).
\end{aligned}$$

□

**A.2. Proof of Claim 5.8.** *Let $X, Y, V, Z \in_R [1, 2^k - 1]$. Let $R = \lfloor 2^b \log X \rfloor$, $S = \lfloor 2^b \log Y \rfloor$, $T = \lfloor 2^b \log V \rfloor$, and $U = \lfloor 2^b \log Z \rfloor$, for some $b \geq k - O(1)$. Then*

$$\max_{-k2^{b+1}+1<i<k2^{b+1}-1} \Pr(R - S - (T - U) = i) = \frac{1}{4 \cdot 2^k} + O\left(1/4^k\right).$$

*Proof.* For $0 \leq i \leq k2^b - 1$, $\Pr(R = i) = \Pr(S = i) = \Pr(T = i) = \Pr(U = i)$, and

$$\begin{aligned}
\Pr(R = i) &= \Pr\left(i \leq 2^b \log Z < i + 1\right) \\
&= \Pr\left(2^{i/2^b} \leq Z < 2^{(i+1)/2^b}\right) \\
&= \frac{1}{2^k - 1}\left[2^{i/2^b}\left(2^{1/2^b} - 1\right)\right].
\end{aligned}$$

Now consider $\Pr(R - S = i)$ (which equals $\Pr(T - U = i)$). Exploiting symmetry as before, we derive that, for $-k2^b + 1 \leq i \leq k2^b - 1$,

$$\begin{aligned}
\Pr(R - S = i) &= \sum_{j=|i|}^{k2^b-1} \Pr(R = j) \Pr(S = j - |i|) \\
&= \sum_{j=|i|}^{k2^b-1} \left[\frac{1}{2^k - 1}\left(2^{1/2^b} - 1\right)\right]^2 2^{j/2^b} 2^{(j-|i|)/2^b} \\
&= \left[\frac{1}{2^k - 1}\left(2^{1/2^b} - 1\right)\right]^2 \frac{1}{2^{|i|/2^b}} \sum_{j=|i|}^{k2^b-1} 2^{2j/2^b} \\
&= \left[\frac{1}{2^k - 1}\left(2^{1/2^b} - 1\right)\right]^2 \frac{1}{2^{|i|/2^b}} \left[\frac{\left(2^{2/2^b}\right)^{k2^b} - 1}{2^{2/2^b} - 1} - \frac{\left(2^{2/2^b}\right)^{|i|} - 1}{2^{2/2^b} - 1}\right] \\
&= \left[\frac{1}{2^k - 1}\left(2^{1/2^b} - 1\right)\right]^2 \frac{1}{2^{|i|/2^b}} \left(\frac{2^{2k} - 2^{2|i|/2^b}}{2^{2/2^b} - 1}\right) \\
&= \frac{1}{(2^k - 1)^2} \cdot \frac{2^{2k} - 2^{2|i|/2^b}}{2^{|i|/2^b}} \cdot \frac{2^{1/2^b} - 1}{2^{1/2^b} + 1}.
\end{aligned}$$

By Lemma A.1, $\Pr(R - S = i)$ maximizes at $i = 0$.

$$\Pr(R - S = 0) = \frac{\left(2^k + 1\right)\left(2^{1/2^b} - 1\right)}{\left(2^k - 1\right)\left(2^{1/2^b} + 1\right)}.$$

As $k$ gets large, $\frac{2^k+1}{2^k-1}$ tends to 1. We also have $2 < 2^{1/2^b} + 1 \leq 3$. Finally, $2^{1/2^b} - 1 \leq 1/2^k$ as long as $b \geq k - O(1)$. Thus,

$$\max_{-k2^b+1 \leq i \leq k2^b-1} \Pr(R - S = i) \approx \frac{1}{2 \cdot 2^k}.$$

Now consider $\Pr(R - S - (T - U) = i)$. As before, by symmetry we derive that

$$\Pr(R - S - (T - U) = i) = \sum_{j=-k2^b+1+|i|}^{k2^b-1} \Pr(R - S = j) \Pr(T - U = j - |i|)$$

for $-k2^{b+1} + 1 < i < k2^{b+1} - 1$. By Lemma A.1, it suffices to solve for $i = 0$.

$$\begin{aligned}
&\Pr(R - S - (T - U) = 0) \\
&= \left[\frac{2^{1/2^b} - 1}{(2^k - 1)^2 \left(2^{1/2^b} + 1\right)}\right]^2 \sum_{j=-k2^b+1}^{k2^b-1} \left(\frac{2^{2k} - 2^{2|j|/2^b}}{2^{|j|/2^b}}\right)^2 \\
&< \left[\frac{2^{1/2^b} - 1}{(2^k - 1)^2 \left(2^{1/2^b} + 1\right)}\right]^2 \left[\left(2^{2k} - 1\right)^2 + 2\sum_{j=1}^{k2^b-1} \left(\frac{2^{4k}}{2^{2j/2^b}} + 2^{2j/2^b}\right)\right] \\
&= \frac{\left(2^{1/2^b} - 1\right)^2 \left(2^{2k} - 1\right)^2}{(2^k - 1)^4 \left(2^{1/2^b} + 1\right)^2} + \frac{2\left(2^{1/2^b} - 1\right)^2}{(2^k - 1)^4 \left(2^{1/2^b} + 1\right)^2} \left[2^{4k} \sum_{j=1}^{k2^b-1} \frac{1}{2^{2j/2^b}} + \sum_{j=1}^{k2^b-1} 2^{2j/2^b}\right].
\end{aligned} \tag{A.2}$$

$$\begin{aligned}
2^{4k} \sum_{j=1}^{k2^b-1} \frac{1}{2^{2j/2^b}} &= 2^{4k} \left[\frac{\left(\frac{1}{2^{2/2^b}}\right)^{k2^b} - 1}{\frac{1}{2^{2/2^b}} - 1} - 1\right] \\
&= 2^{4k} \left[\frac{1/2^{2k} - 1}{\frac{1}{2^{2/2^b}} - 1} - 1\right] < \frac{2^{4k}}{\left(2^{1/2^b} + 1\right)\left(2^{1/2^b} - 1\right)}.
\end{aligned} \tag{A.3}$$

$$\begin{aligned}
\sum_{j=1}^{k2^b-1} 2^{2j/2^b} &< \frac{\left(2^{2/2^b}\right)^{k2^b} - 1}{2^{2/2^b} - 1} \\
&= \frac{2^{2k} - 1}{2^{2/2^b} - 1} = \frac{\left(2^k + 1\right)\left(2^k - 1\right)}{\left(2^{1/2^b} + 1\right)\left(2^{1/2^b} - 1\right)}.
\end{aligned} \tag{A.4}$$

Combining Equations (A.2)–(A.4) yields

$$\begin{aligned}
&\Pr(R - S - (T - U) = 0) < \frac{\left(2^{1/2^b} - 1\right)^2 \left(2^{2k} - 1\right)^2}{(2^k - 1)^4 \left(2^{1/2^b} + 1\right)^2} + \\
&\qquad \frac{2\left(2^{1/2^b} - 1\right)^2}{(2^k - 1)^4 \left(2^{1/2^b} + 1\right)^2} \left[\frac{2^{4k}}{\left(2^{1/2^b} + 1\right)\left(2^{1/2^b} - 1\right)} + \frac{\left(2^k + 1\right)\left(2^k - 1\right)}{\left(2^{1/2^b} + 1\right)\left(2^{1/2^b} - 1\right)}\right] \\
&= \underbrace{\frac{\left(2^{1/2^b} - 1\right)^2 \left(2^{2k} - 1\right)^2}{(2^k - 1)^4 \left(2^{1/2^b} + 1\right)^2}}_{A} + \underbrace{\frac{2 \cdot 2^{4k}\left(2^{1/2^b} - 1\right)}{(2^k - 1)^4 \left(2^{1/2^b} + 1\right)^3}}_{B} + \underbrace{\frac{2\left(2^k + 1\right)\left(2^{1/2^b} - 1\right)}{(2^k - 1)^3 \left(2^{1/2^b} + 1\right)^3}}_{C}.
\end{aligned}$$

Recall $2 \leq 2^{1/2^b} + 1 < 3$, and $2^{1/2^b} - 1 \leq 1/2^k$ as long as $b \geq k - O(1)$. Thus,

$$\max\{A\} \approx \frac{1}{4}\left(\frac{1}{2^k}\right)^2;$$
$$\max\{B\} \approx \frac{1}{4} \cdot \frac{1}{2^k};$$
$$\max\{C\} \approx \frac{1}{4}\left(\frac{1}{2^k}\right)^3.$$

□