International Journal of Foundations of Computer Science

STATE COMPLEXITY OF ADDITIVE WEIGHTED FINITE AUTOMATA*

KAI SALOMAA and PAUL SCHOFIELD
School of Computing, Queen's University, Kingston, Ontario K7L 3N6, Canada
{ksalomaa,schofiel}@cs.queensu.ca

Received (Day Month Year)
Accepted (Day Month Year)
Communicated by (xxxxxxxxxx)

It is known that the neighborhood of a regular language with respect to an additive distance is regular. We introduce an additive weighted finite automaton model that provides a conceptually simple way to reprove this result. We consider the state complexity of converting additive weighted finite automata to deterministic finite automata. As our main result we establish a tight upper bound for the state complexity of the conversion.

1. Introduction

Regularity preserving distances between words have been considered in [1] with applications to fault tolerant lexical analysis. A distance is said to be additive if it, in a certain sense, respects the factorizations of a word into subwords. The edit distance [10] is a standard example of an additive distance. Additivity of the distance is sufficient to guarantee that any neighborhood of a regular language is regular whereas, for example, finite distances do not necessarily have this property.

A weighted finite automaton (WFA) associates weights with transitions between states. Weighted finite automata have been used in many applications, see for example, [2, 3, 4, 5, 6, 11]. Here we consider an additive WFA model that provides a natural and conceptually simple way to recognize neighborhoods of regular languages with respect to an additive distance.

In image processing applications [3, 6] the weight of a path is obtained by multiplying together the weights of transitions on the path (and the values of the initial and final distribution). The weight of a word w is then the sum of the weights of the paths that spell out w. For the error detection application we have in mind it turns out to be useful that the weight of a path is defined to be the sum of the weights occurring on the path and the weight of a word w is the minimum weight

*Research supported in part by the Natural Sciences and Engineering Research Council of Canada, NSERC.

of any path from the start state to a final state that spells out w. Note that fuzzy finite automata [12] combine weights using a max–min strategy. In fuzzy automata the weight associated with a word indicates the degree of membership whereas in additive WFA the weight corresponds to the amount of error observed, that is, a larger weight can be viewed to indicate a smaller degree of membership.

For a given regular language L and an error radius r we construct an additive WFA that recognizes the neighborhood of L of radius r with respect to an additive distance. The construction uses r as a parameter, i.e., the same construction works for any radius up to the given upper bound. The construction gives a better upper bound for the state complexity of the neighborhood, that is the number of states of the minimal deterministic finite automaton (DFA) recognizing the neighborhood [7, 18], than would be obtained directly from the construction of [1].

We study the state complexity of converting additive WFAs with integer weights to DFAs and establish a tight upper bound for the state complexity. Our worst case examples that reach the upper bound use a variable size alphabet and it remains an open question whether the upper bound can be reached using a family of additive WFAs where the alphabet does not depend on the number of states.

To conclude the introduction, we mention some related work. Very efficient constructions of nondeterministic and deterministic automata that recognize the neighborhood of a single word with respect to the edit distance have been given in [15]. The complexity of computing the edit distance of a given word and a regular language was first considered in [16] and extensions of this problem have been addressed in [13]. Computing the edit distance of a regular language, that is, the smallest edit distance between two distinct words in the language, is shown to have polynomial time complexity in [9].

2. Preliminaries

For all unexplained notions concerning finite automata and regular languages we refer the reader, e.g., to [8, 17].

The cardinality of a finite set Q is denoted $|Q|$ and the power set of Q is $\mathcal{P}(Q)$. The symbol Σ denotes a finite alphabet, Σ^* is the set of words over Σ and ε is the empty word. When there is no confusion a singleton set $\{w\}$, $w \in \Sigma^*$, is denoted simply as w. The set of non-negative integers (respectively, non-negative rational numbers) is $\mathbb{N}_0$ (resp. $\mathbb{Q}_0$).

A nondeterministic finite automaton (NFA) is a tuple $A = (Q, \Sigma, \gamma, s, F)$ where Q is the finite set of states, Σ is the input alphabet, $\gamma : Q \times \Sigma \to \mathcal{P}(Q)$ defines the state transitions, $s \in Q$ is the start state and $F \subseteq Q$ is the set of accepting states. In the well known way γ is extended as a function $\hat{\gamma} : Q \times \Sigma^* \to \mathcal{P}(Q)$ and we denote also $\hat{\gamma}$ simply by γ. The language recognized by A is $L(A) = \{w \in \Sigma^* \mid \gamma(s, w) \cap F \neq \emptyset\}$.

An NFA A as above is deterministic (a DFA) if for all $q \in Q$ and $a \in \Sigma$, $|\gamma(q, a)| = 1$. Here we assume that DFAs are complete, i.e., that all transitions are

defined. Both NFAs and DFAs recognize the regular languages.

For $L \subseteq \Sigma^*$ we define an equivalence relation $\equiv_L$ on Σ^* by setting, for $x, y \in \Sigma^*$,

$$x \equiv_L y \text{ iff } [(\forall z \in \Sigma^*)\ xz \in L \Leftrightarrow yz \in L].$$

The relation $\equiv_L$ is called the right invariant equivalence relation of L and the following result is well known.

Proposition 1. *For any regular language L the number of states of the minimal DFA recognizing L equals to the number of equivalence classes of $\equiv_L$.*

3. Distances and additive WFAs

First we recall some definitions and notation concerning distances between words. For more details and examples the reader is referred to [1].

A function $d : \Sigma^* \times \Sigma^* \to [0, \infty)$ is a *distance* if it satisfies the following three conditions:

(D1) $d(x, y) = 0$ if and only if $x = y$, $x, y \in \Sigma^*$,
(D2) $d(x, y) = d(y, x)$ for all $x, y \in \Sigma^*$,
(D3) $d(x, z) \leq d(x, y) + d(y, z)$ for all $x, y, z \in \Sigma^*$.

The function d defines a *quasi-distance* if it satisfies (D2) and (D3) and $d(x, x) = 0$ for all $x \in \Sigma^*$. A quasi-distance allows the possibility that $d(x, y) = 0$ for $x \neq y$.

In the following we restrict consideration to (quasi-)distances $d : \Sigma^* \times \Sigma^* \to \mathbb{Q}_0$ where the values are assumed to be non-negative rational numbers.

The neighborhood of $L \subseteq \Sigma^*$ of radius $r \geq 0$ is

$$E(L, d, r) = \{w \in \Sigma^* \mid (\exists u \in L)\ d(w, u) \leq r\}.$$

The distance d is said to be *finite* if for all $w \in \Sigma^*$ and $r \geq 0$, $E(L, d, r)$ is finite. The distance d is *additive* if for any decomposition $w = w_1 w_2$, $w_1, w_2 \in \Sigma^*$, and radius $r \geq 0$,

$$E(w, d, r) = \bigcup_{r_1 + r_2 = r} E(w_1, d, r_1) \cdot E(w_2, d, r_2).$$

We will use the following result:

Proposition 2. [1] *Every additive distance is finite.*

Next we introduce the weighted automaton model used here.

Definition 3. *An* additive weighted finite automaton *(additive WFA) is a tuple*

$$\widetilde{A} = (Q, \Sigma, \gamma, \beta, s, F), \tag{1}$$

where Q is the finite set of states, Σ is the alphabet of input symbols, $\gamma : Q \times \Sigma \to \mathcal{P}(Q)$ is the (nondeterministic) transition function, $\beta : Q \times \Sigma \times Q \to \mathbb{Q}_0$ is a partial function where $\beta(q_1, a, q_2)$ is defined iff $q_2 \in \gamma(q_1, a)$, $s \in Q$ is the start state and $F \subseteq Q$ is the set of accepting states.

The function β assigns non-negative rational weights to transitions. For possible applications we can restrict the weights to be rational. For notational convenience we include an explicit transition function γ although γ is naturally completely determined by the domain of the partial function β. If $\widetilde{A} = (Q, \Sigma, \gamma, \beta, s, F)$ is as in Definition 3, the contruct $A = (Q, \Sigma, \gamma, s, F)$ is a nondeterministic finite automaton.

Let $\widetilde{A}$ be as in (1), $w = a_1 \cdots a_m$, $a_i \in \Sigma$, $i = 1, \ldots, m$, $m \geq 0$, and $p_1, p_2 \in Q$. By a *computation path* of $\widetilde{A}$ along w from state p_1 to p_2 we mean an element of $(Q \times \Sigma \times Q)^*$,

$$\alpha = (q_0, a_1, q_1) \cdot \ldots \cdot (q_{m-1}, a_m, q_m), \tag{2}$$

where $p_1 = q_0$ and $p_2 = q_m$, and $q_i \in \gamma(q_{i-1}, a_i)$, $i = 1, \ldots, m$. The *weight* of a computation path α as in (2) is

$$\beta(\alpha) = \sum_{i=1}^{m} \beta((q_{i-1}, a_i, q_i)).$$

If $m = 0$ and $p_1 = p_2$, α as in (2) is interpreted to be the empty sequence and in this case we set $\beta(\alpha) = 0$. The set of all computation paths along w from state p_1 to state p_2 is denoted $\Theta(p_1, w, p_2)$.

Now the language recognized by $\widetilde{A}$, as in (1), within weight bound $r \geq 0$ is defined as

$$L(\widetilde{A}, r) = \{w \in \Sigma^* \mid (\exists f \in F)(\exists \alpha \in \Theta(s, w, f)), \beta(\alpha) \leq r\}.$$

The language $L(\widetilde{A}, r)$ consists of all words w that take the start state of $\widetilde{A}$ to an accepting state along some path having cumulative weight at most r.

The usefulness of the model of additive WFA for error recognition is based on the following result.

Theorem 4. *Let B be an NFA, d an additive distance and $r_0 \geq 0$. We can construct an additive WFA $\widetilde{A}$ such that for any $0 \leq r \leq r_0$,*

$$L(\widetilde{A}, r) = E(L(B), d, r).$$

Furthermore, if d and r_0 are fixed, based on the description of the NFA B the WFA $\widetilde{A}$ can be constructed in square time.

Proof. Due to length restrictions we only sketch the construction here. The WFA $\widetilde{A}$ has the same set of states Q as the NFA B. Between any pair of states q_1, q_2 the WFA $\widetilde{A}$ will have a transition labeled by $b \in \Sigma$ if and only if, in the NFA B, q_2 is reachable from q_1 along a path spelled by some word $w \in \Sigma^*$ where $d(b, w) \leq r_0$. The weight of the transition (q_1, b, q_2) is defined to be the minimum of all values $d(b, w)$ with the above property.

Using induction on the length of w (and additivity of d) we can show that w spells out some path in $\widetilde{A}$ from the start state s to a state q with cumulative weight $r \leq r_0$ only if some word u such that $d(w, u) \leq r$ takes B from the start state to the same state q.

Conversely, given any word $w \in E(L(B), d, r)$ we find a word w' such that w' takes the NFA B to an accepting state q_f and $d(w, w') \leq r$. Again using the additivity of d, there exists an accepting computation path of $\widetilde{A}$ with cumulative weight at most $d(w, w')$. A detailed proof for the correctness of the construction is given in [14].

Finally, to verify the time bound we note that when d and r_0 are fixed Proposition 2 implies that, for each pair of states of B, we need to add a constant number of weighted transitions (and the weight can be found from a constant number of candidate values). ■

It is easy to see that $L(\widetilde{A}, r)$ is always regular, and this follows also from the explicit WFA-to-DFA construction described in the next section when dealing with state complexity. Thus, Theorem 4 gives a new proof for the result [1] that the neighborhoods of regular languages with respect to an additive distance are regular. The proof of Theorem 4 is conceptually simpler than the original proof, and it has the advantage that the same construction works for all neighborhoods having a radius within some upper bound.

Theorem 4 combined with the state complexity upper bound in the next section (Theorem 5) gives a better upper bound for the state complexity of the neighborhood of a regular language than the bound obtained by first constructing an NFA as in [1] and then converting it to a DFA.

Also additive quasi-distances are known to preserve regularity [1]. However, the analogy of Proposition 2 does not hold for additive quasi-distances and it remains an open question whether one can use a WFA construction analogous to Theorem 4.

4. State Complexity

We first give a construction of a DFA that recognizes the language of an arbitrary n state WFA within a given weight bound r and then show that the construction is optimal in terms of the number of states. The state complexity bound can be viewed as an extension of the well known tight upper bound for the NFA–to–DFA conversion.

In the following we assume that all transition weights of WFAs are integers, this does not lose generality since we can, if necessary, multiply all transition weights and the weight bound by an arbitrary integer.

Theorem 5. *Let $\widetilde{A}$ as in (1) be an additive WFA where all transition weights are integers and let $r \in \mathbb{N}_0$. The language $L(\widetilde{A}, r)$ can be recognized by a DFA B having $(r+2)^n$ states.*

Proof. Denote the set of states of $\widetilde{A}$ as $\{q_1, \ldots, q_n\}$ where q_1 is the start state. The states of B are tuples of integers $(i_1, \ldots, i_n)$, $0 \leq i_j \leq r+1$, $j = 1, \ldots, n$. Intuitively, a state $(i_1, \ldots, i_n)$ is used to indicate that the value i_j, where $0 \leq i_j \leq r$, is the smallest cumulative weight of any path in $\widetilde{A}$ that can reach the state q_j with the

input processed so far. A value $i_j = r+1$ indicates that there is no path spelled out by the current input that reaches state q_j from the start state with a cumulative weight at most r.

The start state of B is $s_B = (0, r+1, \ldots, r+1)$ and a state $(i_1, \ldots, i_n)$ is accepting if $i_j \leq r$ for some j such that q_j is an accepting state of $\widetilde{A}$. On input symbol b, the DFA B changes state from $(i_1, \ldots, i_n)$ to $(j_1, \ldots, j_n)$ where for $t = 1, \ldots n$,

$$j_t = \min(\{r+1\} \cup \{s \mid q_t \in \gamma(q_k, b), s = i_k + \beta((q_k, b, q_t)), 1 \leq k \leq n\}).$$

We denote the (deterministic) transition function of B by π_B.

If either the minimum weight path of $\widetilde{A}$ from s to q_k spelling out the word w, $1 \leq k \leq n$, has weight at least $r+1$ or $\widetilde{A}$ does not have a path along w from s to q_k, we say that the minimal path from s to q_k along w has weight $r+1$. With this convention and using induction on the length of the input word w we can show that the minimum weight path in $\widetilde{A}$ that spells out w from s to q_k has weight i_k, $k = 1, \ldots, n$, if and only if the transition function π_B takes s_B to $(i_1, \ldots, i_n)$ along w. The details of the proof are given in [14]. ■

Next we present a construction of a WFA where the state complexity of the minimal equivalent DFA reaches the upper bound given by Theorem 5. The construction below uses $2n-1$ alphabet symbols for a WFA with n states. Let $n \geq 1$ and let

$$\widetilde{A}_n = (Q, \Sigma, \gamma, \beta, s, F) \tag{3}$$

be an additive WFA where $Q = \{1, \ldots, n\}$, $\Sigma = \{a_1, \ldots, a_{n-1}, b_1, \ldots, b_n\}$, $s = 1$, $F = \{n\}$, and the functions γ and β are defined as follows.

The function γ is determined by setting

- $\gamma(i, a_i) = \{i, i+1\}$, $i = 1, \ldots, n-1$;
- $\gamma(i, a_j) = \{i\}$, $i = 1, \ldots n-2$, $j = i+1, \ldots n-1$;
- $\gamma(i, b_j) = \{i\}$, $i = 1, \ldots, n$, $i-1 \leq j \leq n$;
- for all cases not included in the above, the transition is undefined.

The transition weights are assigned by β as follows:

- $\beta((i, b_i, i)) = 1$, $i = 1, \ldots, n$
- β assigns the weight zero to all the other transitions of $\widetilde{A}_n$.

The only transitions of $\widetilde{A}_n$ that change the state are transitions from state i to $i+1$ with input a_i, $i = 1, \ldots, n-1$. All other transitions of $\widetilde{A}_n$ are self-loops. The self-loops (i, b_i, i), $i = 1, \ldots, n$, have weight one. All other transitions of $\widetilde{A}_n$ have weight zero.

Figure 1 represents the additive WFA $\widetilde{A}_5$ constructed as described above. In the figure, note that all transitions where the weight is not marked have a weight of zero.

In the following we show that if $\widetilde{A}_n$, $n \geq 1$, is defined as in (3), then for any integer $r \geq 0$, the minimal DFA for the language $L(\widetilde{A}_n, r)$ has at least $(r+2)^n$

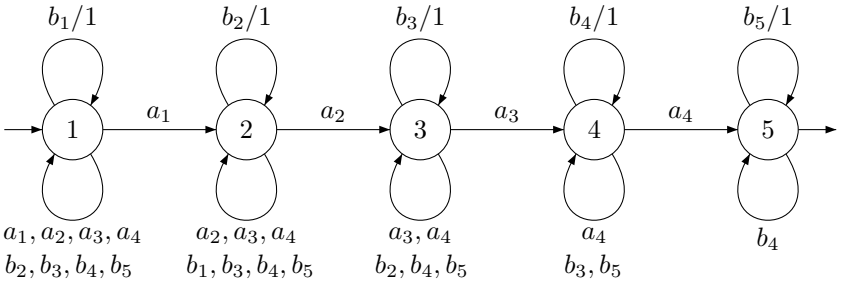

Fig. 1. The weighted finite automaton $\widetilde{A}_5$.

states. By Proposition 1 it is sufficient to find $(r+2)^n$ words that are all pairwise distinguishable with respect to the the language $L(\widetilde{A}_n, r)$, that is, all belong to different equivalence classes of $\equiv_{L(\widetilde{A}_n,r)}$.

We define the following words over the alphabet Σ:

$$w(k_1, \ldots, k_n) = a_1 b_1^{k_1} a_2 b_2^{k_2} \cdots a_{n-1} b_{n-1}^{k_{n-1}} b_n^{k_n}, \quad 0 \le k_i \le r+1, \;\; i = 1, \ldots, n. \quad (4)$$

Note that there are exactly $(r+2)^n$ words of the form (4).

The WFA $\widetilde{A}_n$ is nondeterministic since $\gamma(i, a_i)$ always has two choices to continue the computation, $i = 1, \ldots, n-1$. However, in the following lemma we show that on an input as in (4), the automaton $\widetilde{A}_n$ can reach the state i, $1 \le i \le n$, only with a computation that has cumulative weight exactly k_i.

Lemma 6. *Assume that $\widetilde{A}_n$ reaches state i, $1 \le i \le n$, along a computation path $\alpha \in \Theta(s, w(k_1, \ldots, k_n), i)$ that consumes input word $w(k_1, \ldots, k_n)$. Then the weight of α is k_i. Furthermore, A_n can reach the state i after reading any word of the form $w(k_1, \ldots, k_n)$ along a path with weight k_i.*

Proof. Recall that the only transitions of $\widetilde{A}_n$ that are not self-loops are transitions of the form $(j, a_j, j+1)$. Since α ends in state i, it follows that when reading the word $w(k_1, \ldots, k_n)$ each symbol a_j, $1 \le j < i$, changes the state from j to $j+1$. Note that otherwise the computation would get stuck in some state $i' < i$.

The above means that when reading the word $w(k_1, \ldots, k_n)$, $\widetilde{A}_n$ reads the k_1 symbols b_1 in state 2, the k_2 symbols b_2 in state 3, and continuing in this way, it reads the k_{i-1} symbols b_{i-1} in state i. Since the computation ends in state i, the next symbol a_i has to be read using a self-loop (i, a_i, i). After this the input word has k_i symbols b_i and each of the corresponding self-loops in state i has weight one.

After the above the remaining suffix of the input is

$$w_{\text{suffix}} = a_{i+1} b_{i+1}^{k_{i+1}} \cdots a_{n-1} b_{n-1}^{k_{n-1}} b_n^{k_n}.$$

All symbols occurring in w_{suffix} are processed deterministically in state i with self-loops having weight zero. The total weight of all the transitions used in the computation is $k_i \cdot 1 = k_i$.

The second claim was shown also above since we constructed a computation on input $w(k_1, \ldots, k_n)$ with cumulative weight k_i that reaches the state i, $1 \leq i \leq n$. ■

Using the above lemma we can now establish that all words as in (4) are pairwise in distinct equivalence classes of $\equiv_{L(\widetilde{A}_n,r)}$ and this gives the desired lower bound.

Theorem 7. *Let $r \geq 0$ be an arbitrary radius. The minimal DFA for $L(\widetilde{A}_n, r)$ has $(r+2)^n$ states.*

Proof. Let $B_{n,r}$ be the minimal DFA for $L(\widetilde{A}_n, r)$. From Theorem 5 we know that $B_{n,r}$ has at most $(r+2)^n$ states.

We show that all of the $(r+2)^n$ words as defined in (4) are pairwise distinguishable with respect to the language $L(\widetilde{A}_n, r)$. Proposition 1 then implies that $B_{n,r}$ has at least $(r+2)^n$ states.

Consider two distinct words $w(k_1, \ldots, k_n)$ and $w(k'_1, \ldots, k'_n)$ as in (4), $0 \leq k_i \leq r+1$, $0 \leq k'_i \leq r+1$, $i = 1, \ldots n$. Thus there exists an index j such that $k_j \neq k'_j$. Without loss of generality we assume that

$$k_j < k'_j \tag{5}$$

since the other possibility is symmetric.

Choose

$$z = b_j^{r-k_j} a_j a_{j+1} \cdots a_{n-1}.$$

Note that since $k_j < k'_j \leq r+1$, it follows that $r - k_j \geq 0$ and z is a well-defined word. We claim that

$$w(k_1, \ldots, k_n) \cdot z \in L(\widetilde{A}_n, r) \text{ and } w(k'_1, \ldots, k'_n) \cdot z \notin L(\widetilde{A}_n, r). \tag{6}$$

By Lemma 6, $\widetilde{A}_n$ has a computation on input $w(k_1, \ldots, k_n)$ that ends in state j and has cumulative weight k_j. In state j, $\widetilde{A}_n$ can read the first $r - k_j$ symbols b_j of z, and after this the total weight is $k_j + (r - k_j) = r$. Finally the zero weight transitions on the suffix $a_j a_{j+1} \cdots a_{n-1}$ take the automaton from state j to the accepting state n.

Now we show the second part of (6). First we consider the question from which states q the WFA $\widetilde{A}_n$ can reach the only accepting state n on input z — here for the time being we do not consider weights of transitions. On any state of $\widetilde{A}_n$, the symbol b_j either defines a self-loop or the transition on b_j is undefined. Thus, $\widetilde{A}_n$ can reach the accepting state from q on input z only if $\widetilde{A}_n$ reaches the accepting state from q on input $a_j a_{j+1} \cdots a_{n-1}$. Since for any $j' > j$ the transition on a_j from state j' is undefined, the only state from which $\widetilde{A}_n$ reaches the final state on input $a_j a_{j+1} \cdots a_{n-1}$ is j. Note that from a state $j' < j$, $\widetilde{A}_n$ cannot reach the final state since the given input does not contain the symbol $a_{j'}$.

Thus, the only possibility for $\widetilde{A}_n$ to accept $w(k'_1, \ldots, k'_n) \cdot z$ would be that the computation has to reach state j on the prefix $w(k'_1, \ldots, k'_n)$. By Lemma 6, the

weight of this computation can only be k'_j. Again, when continuing the computation on z from state j, $\widetilde{A}_n$ has to read the first $r - k_j$ symbols b_j each with a self-loop transition having weight one. After this the cumulative weight of the computation will be $k'_j + r - k_j$ which is, by (5), greater than r. Since the cumulative weight of any possible accepting computation path exceeds r, it follows that $w(k'_1, \ldots, k'_n) \cdot z \notin L(\widetilde{A}_n, r)$.

We have shown that $\equiv_{L(\widetilde{A}_n,r)}$ has at least $(r+2)^n$ equivalence classes. ■

As a direct consequence of the upper bound given in Theorem 5 and the lower bound given in Theorem 7, we state the following corollary.

Corollary 8. *Let n be the number of states of a WFA $\widetilde{A}_n$ and $r \geq 0$ be an integer. The tight upper bound for the number of states of the minimal DFA for $L(\widetilde{A}_n, r)$ is $(r+2)^n$.*

In the construction used for Theorem 7, the size of the alphabet is $2n-1$ for a WFA having n states. The WFA–to–DFA conversion has been implemented in [14] and using the software we have found examples, where at least for small values of n, an alphabet of size $n+1$, is sufficient to reach the upper bound of $(r+2)^n$. In Theorem 7 we have used the slightly larger alphabet, in order to make the proof more transparent, because also the more complicated examples require a variable size alphabet.

Note that by choosing the weight bound to be $r = 0$ in Corollary 8, the language recognized by a WFA $\widetilde{A}$ reduces to the "crisp" language consisting of all words accepted by the subautomaton of $\widetilde{A}$ that has only the transitions of weight zero. In this case the result reduces to the well known 2^n bound for the state complexity of the NFA–to–DFA transformation.

5. Conclusion

The main open problem concerning the state complexity of additive WFAs is whether the upper bound of Theorem 5 can be reached using automata defined over a fixed size alphabet. We have experimental results that indicate that the size of the alphabet can be reduced from $2n-1$.

Since additive distances preserve regularity, an interesting question would also be to consider the state complexity of neighborhoods of regular languages with respect to additive distances. For example, when L has (non)deterministic state complexity n what is the worst case size, as a function of n and r, of the minimal DFA that recognizes the neighborhood of L of radius r with respect to the edit distance? The question could be naturally extended for arbitrary additive distances. Theorems 4 and 5 give an upper bound for the state complexity of additive neighborhoods.

Theorem 4 indicates that additive WFAs are a useful model for recognizing (additive) errors in regular languages. A natural topic for further research is to consider whether similar techniques can be used for error correction, for example, by employing a weighted finite transducer model.

References

[1] C. Calude, K. Salomaa and S. Yu, Additive distances and quasi-distances between words, *J. Universal Computer Sci.* **8** (2002) 141–152.

[2] K. Culik II and J. Karhumäki, Finite automata computing real functions, *SIAM J. Comput.* **23** (1994) 789–914.

[3] K. Culik II and J. Kari, Digital images and formal languages, in *Handbook of Formal Languages, Vol. 3,* eds. G. Rozenberg and A. Salomaa, (Springer-Verlag, 1997), pp. 599–616.

[4] D. Derencourt, J. Karhumäki, M. Latteaux and A. Terlutte, On the computational power of weighted finite automata, *Fund. Inf.* **25** (1996) 285–293.

[5] M. Droste and P. Gastin, Weighted automata and weighted logics, in *Proc. of ICALP 2005,* Lecture Notes in Computer Science 3580, (Springer, 2005) pp. 513–525.

[6] M. Eramian, Efficient simulation of nondeterministic weighted finite automata, *J. Automata, Languages and Combinatorics* **9** (2004) 257–267.

[7] J. Goldstine, M. Kappes, C.M.R. Kintala, H. Leung, A. Malcher and D. Wotschke, Descriptional complexity of machines with limited resources, *J. Universal Computer Sci.* **8** (2002) 193–234.

[8] J.E. Hopcroft and J.D. Ullman, *Introduction to Automata Theory, Languages, and Computation,* Addison Wesley, Reading, Mass., 1979.

[9] S. Konstantinidis, Computing the edit distance of a regular language, in *Proc. of IEEE Information Theory Workshop on Coding and Complexity,* Roturoa, New Zealand, Aug. 29 – Sep. 1, 2005, pp. 113–116.

[10] V.I. Levenshtein, Binary codes capable of correcting deletions, insertions and reversals, *Soviet Physics Dokl.* **10** (1966) 707–710.

[11] C. Martin-Vide, V. Mitrana and R. Stiebe, Weighted grammars and automata with threshold interpretation, *J. Automata, Languages and Combinatorics* **8** (2003) 303–318.

[12] A. Mateescu, A. Salomaa, K. Salomaa and S. Yu, Lexical analysis with a simple finite fuzzy-automaton model, *J. Universal Computer Sci.* **1** (1995) 288–307.

[13] G. Pighizzini, How hard is computing the edit distance? *Inform. Computation* **165** (2001) 1–13.

[14] P. Schofield, Error quantification and recognition using weighted finite automata, M.Sc. Thesis, School of Computing, Queen's University, Canada, 2006.

[15] K.U. Schulz and S. Mihov, Fast string correction with Levenshtein automata, *Internat. J. Document Analysis and Recognition* **5** (2002) 67–85.

[16] R.A. Wagner, Order-n correction for regular languages, *Comm. of the ACM* **17** (1974) 265–268.

[17] S. Yu, Regular languages, in *Handbook of Formal Languages, Vol. 1,* eds. G. Rozenberg and A. Salomaa, (Springer-Verlag, 1997), pp. 41–110.

[18] S. Yu, State complexity of regular languages, *J. Automata, Languages and Combinatorics* **6** (2001) 221–234.