Equivalence of SDVFA of order (s, t) with DFA, VDFA, NFA and ϵ-NFA

A. Jain

Abstract. The notion of SDVFA of order (s, t) has alreday been introduced by the author [12]. In this paper, we show the equivalence of SDVFA of order (s, t) with DFA, VDFA, NFA and ϵ-NFA. The equivalence has been established by converting an SDVFA to DFA, VDFA and NFA (ϵ-NFA) and vice-versa.

AMS Subject Classification (2000): 68Q45

Keywords: Deterministic finite automaton, non-deterministic finite automaton, virtual deterministic finite automaton, formal languages, semi-deterministic virtual finite automaton

1. Introduction

The notion of SDVFA of order (s, t) has alreday been introduced by the author [12]. In this paper, we show the equivalence of SDVFA of order (s, t) with DFA, VDFA, NFA and ϵ-NFA. The equivalence has been established by converting an SDVFA to DFA, VDFA and NFA (ϵ-NFA) and vice-versa. We begin with the definition of SDVFA of order (s, t) [12].

Definition 1.1. A semi-deterministic virtual finite automaton (SDVFA) of order (s, t) is a finite automaton that can make atmost "s"($s \geq 1$) transitions on receiving a real input and atmost "t"($t \geq 0$) transitions on virtual input (or no input). (Zero transition means the automaton remains in the same state).

Remark 1.1. For an SDVFA having n states, we have the following:

(i) If $s = 1$ and $t = 0$, then an SDVFA of order $(1, 0)$ is simply a DFA [2,6,15].

(ii) If $s = 1$ and $t = n$, then an SDVFA of order $(1, n)$ is simply a VDFA [8].

(iii) If $s = n$ and $t = 0$, then an SDVFA of order $(n, 0)$ is simply an NFA [2,6,15].

(iv) If $s = n$ and $t = n$, then an SDVFA of order (n, n) is simply an ϵ-NFA [2,6,15].

We formally define a semi-deterministic virtual finite automaton (SDVFA) of order (s, t) as follows:

Definition 1.2. A semi-deterministic virtual finite automaton (SDVFA) of order (s, t) consists of

1. A finite set of states (including the dead state) often denoted by Q.
2. A finite set of input symbols including the empty string symbol ϵ. This is often denoted by $\Sigma \cup \{\epsilon\}$. Σ is called real alphabet.
3. A transition function $\delta_{(s,t)}$ that takes as arguments a state and an input symbol. On real input symbol i.e. if the symbol is a member of real alphabet Σ, $\delta_{(s,t)}$ returns a set of atmost "s" states while on virtual input ϵ, the transition function returns a set of atmost "t" states.
4. A start state S which is one of the states in Q.
5. A set of final or accepting states F. The set F is a subset of Q. Dead state is never an accepting state and it makes a transition to itself on every possible input symbol.

We can also denote an SDVFA of order (s, t) by a "five tuple" notation:

$$V = (Q, \Sigma \cup \{\epsilon\}, \delta_{(s,t)}, q_0, F)$$

where V is the name of the SDVFA, Q is the set of states, $\Sigma \cup \{\epsilon\}$ is the set of input symbols, $\delta_{(s,t)}$ is the transition function, q_0 is the start state and F is the set of accepting states.

We now define the meaning of equivalence of two finite automata:

Definition 1.3. Two finite automata $A1$ and A_2 are said to be equivalent if $L(A_1) = L(A_2)$ i.e. if they accept the same language.

In the following section, we show the equivalence of SDVFA of order (s, t) with DFA, VDFA, NFA and ϵ-NFA.

2. Conversion of an SDVFA of order (s,t) to DFA and vice-versa

Given an SDVFA V of order (s,t), we can find a DFA D that accepts the same language as that of V. The construction we use is very close to the subset construction, as the states of D are subsets of the states of V. The only difference is that we must incorporate ϵ-transitions of V, which we do through the mechanism of v-closure. Let $V = (Q_V, \Sigma \cup \{\epsilon\}, \delta_{(s,t)}, q_0, F_V)$ be an SDVFA of order (s,t). Then the equivalent DFA

$$D = (Q_D, \Sigma, \delta_D, q_D, F_D)$$

is defined as follows:

1. Q_D is the set of subsets of Q_V. More precisely, we shall find that the only accessible states of D are v-closed subsets of Q_V, that is, those sets $S \subseteq Q_V$ such that $S = v\text{-close}(S)$. Put another way, the v-closed sets of states S are those such that any ϵ-transitions out of one of the states in S lead to a state that is also in S. Note that $\emptyset$ is a v-closed set.

2. $q_D = v\text{-close}(q_0)$; that is, we get the start state of D by v-closing the start state of V.

3. F_D is those sets of states that contain at least one accepting state of V. That is, $F_D = \{S \mid S \text{ is in } Q_D \text{ and } S \cap F_V \neq \emptyset\}$.

4. $\delta_D(S,a)$ is computed, for all a in Σ and sets S in Q_D by:
   (a) Let $S = \{p_1, p_2, \cdots, p_k\}$.
   (b) compute $\bigcup_{i=1}^{k} \delta_{(s,t)}(p_i, a)$; let this set be $\{r_1, r_2, \cdots, r_m\}$.
   (c) Then $\delta_D(S,a) = \bigcup_{j=1}^{m} v\text{-close}\ (r_j)$.

We now give an example to illustrate the above procedure:

Example 2.1. Consider the SDVFA V of order $(1,1)$ in Fig. 2.1

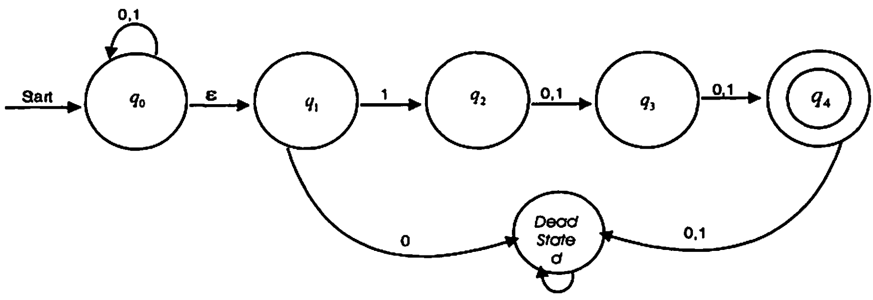

Fig. 2.1: SDVFA of order (1, 1) that accepts all strings of 0's and 1's such that 3rd symbol from the end is 1.

We convert the SDVFA V of Fig. 2.1 to DFA. From V, we construct a DFA D which is shown in Fig. 2.2.

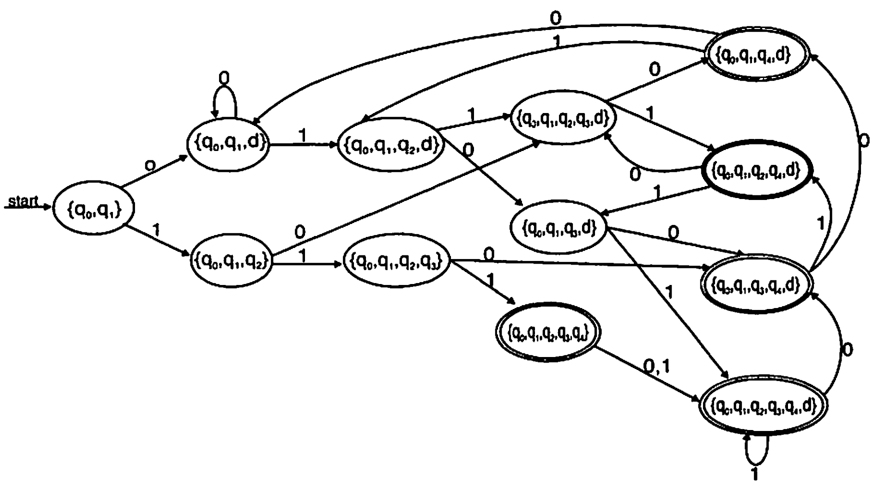

Fig. 2.2: The DFA constructed from the SDVFA of Fig. 2.1.

Since the start state of V in Fig. 4.3 is q_0, the start state of D is v-close (q_0) which is $\{q_0, q_1\}$. Our first job is to find the successors of q_0, q_1 on input symbols 0 and 1. Since q_0 goes to q_0, q_1 goes to dead state on input 0. Thus to compute $\delta_D(\{q_0, q_1\}, 0)$ we compute v-close $(q_0) \cup v$-close $(d) = \{q_0, q_1, d\}$

i.e. $\delta_D(\{q_0, q_1, d\}, 0) = \{q_0, q_1, d\}$. Similarly, $\delta_D(\{q_0, q_1\}, 1) = \{q_0, q_1, q_2\}$. We have explained the arcs out of $\{q_0, q_1\}$ in Fig. 2.2. The other transitions are computed similarly. Since q_4 is the only accepting state of V, the accepting states of D are those accessible states that contain q_4. We see that the five sets $\{q_0, q_1, q_2, q_3, q_4\}, \{q_0, q_1, q_4, d\}, \{q_0, q_1, q_2, q_4, d\}, \{q_0, q_1, q_3, q_4, d\}$ and $\{q_0, q_1, q_2, q_3, q_4, d\}$ indicated by double circle in Fig. 2.2 are accepting states of D.

Now, we prove a theorem establishing the equivalence between SDVFA and DFA:

Theorem 2.1. *A language L is accepted by some SDVFA if and only if L is accepted by some DFA.*

Proof. (If) This direction is easy: Suppose $L = L(D)$ for some DFA. Turn D into an SDVFA A by adding transitions $\delta_{(1,0)}(q, \epsilon) = q$ for all states q of D. Thus, the transitions of V and D are the same, but V explicitely states that there are no transitions out of any state on ϵ.

(Only-if) Let $V = (Q_V, \Sigma \cup \{\epsilon\}, \delta_{(s,t)}, q_0, F_V)$ be an SDVFA of order (s,t). Eliminating multiple transitions on real input and all transitions on virtual input ϵ from this SDVFA by the procedure discussed above, we get a DFA D where

$$D = (Q_D, \Sigma, \delta_D, q_D, F_D).$$

We need to show that $L(D) = L(A)$, and we do so by showing that the extended transition functions of V and D are the same. Formally, we show $\hat{\delta}_{(s,t)}(q_0, w) = \hat{\delta_D}(q_D, w)$ by induction on the length of w.

Basis. If $|w| = 0$, then $w = \epsilon$. we know $\hat{\delta}_{(s,t)}(q_0, \epsilon) = v\text{-close}(q_0)$. We also know that $q_D = v\text{-close}(q_0)$, because that is how the start state of D is defined. Finally, for a DFA, we know that $\hat{\delta}(p, \epsilon) = p$ for any state p, so in particular, $\hat{\delta_D}(q_D, \epsilon) = q_D = v\text{-close}(q_0)$. We have thus proved that $\hat{\delta_D}(q_D, \epsilon) = \hat{\delta}_{(s,t)}(q_0, \epsilon)$.

Induction. Suppose $w = xa$, where a is the final symbol of w, and assume that the statement holds for x. That is, $\hat{\delta}_{(s,t)}(q_0, x) = \hat{\delta_D}(q_D, x)$. Let both these sets of states be $\{p_1, p_2, \cdots, p_k\}$.

By the definition of $\hat{\delta}_{(s,t)}$ for SDVFA, we compute $\hat{\delta}_{(s,t)}(q_0, w)$ by:

1. Let $\{r_1, r_2, \cdots, r_m\}$ be $\bigcup_{i=1}^{k} \delta_{(s,t)}(p_i, a)$.

2. Then $\hat{\delta}_{(s,t)}(q_0, w) = \bigcup_{j=1}^{m} v\text{−close}(r_j)$.

If we examine the construction of DFA D described before Theorem 2.1, we see that $\delta_D(\{p_1, p_2, \cdots, p_k\}, a)$ is constructed by the same two steps (1) and (2) above. Thus, $\hat{\delta_D}(q_D, w)$ which is $\delta_D(\{p_1, p_2, \cdots, p_k\}, a)$ is the same set as $\hat{\delta}_{(s,t)}(q_0, w)$. We have now proved that $\hat{\delta}_{(s,t)}(q_0, w) = \hat{\delta_D}(q_D, w)$ and completed the inductive part. □

3. Conversion of ϵ-NFA to SDVFA of order $(1, n)$ and vice-versa

Let $E = (Q_E, \Sigma \cup \{\epsilon\}, \delta_E, q_0, F_E)$ be an ϵ-NFA [2,6]. We convert this ϵ-NFA to an SDVFA of order $(1, n)$. We construct an SDVFA of order $(1, n)$, say $V = (Q_V, \Sigma \cup \{\epsilon\}, \delta_{(1,n)}, \{q_0\}, F_V)$ such that $L(V) = L(E)$. Notice that the input alphabets of the two automaton are the same and the start state of V in the set contain only the start state of E. The other components of V are constructed as follows:

1. Q_V is the set of subsets of Q_E i.e. Q_V is the power of set of Q_E. Note that if Q_E has n states, then Q_V will have 2^n states. Often, not all theses states are accessible from the start state of Q_V. Inaccessible states can be "thrown away", so effectively, so that the number of states of V may be much smaller that 2^n.

2. F_V is the set of subset S of Q_E such that $S \cap F_E \neq \emptyset$. That is, F_V is all sets of E's states that include atleast one accepting state of E.

3. (i) For each set $S \subseteq Q_V$ and for each input symbol a in Σ

$$\delta_{(1,n)}(S, a) = \bigcup_{p \ in \ S} \delta_E(p, a).$$

(ii) For each set $S \subseteq Q_V$ and for virtual input symbol ϵ

$$\delta_{(1,n)}(S, \epsilon) = \bigcup_{p \ in \ S} \delta_E(p, \epsilon).$$

That is, to compute $\delta_{(1,n)}(S, a)$ (*or* $\delta_{(1,n)}(S, \epsilon)$), we look at all the states p in S see what states N goes to from p on input a (or ϵ), and take the union of all those states.

We now give an example to illustrate the conversion of ϵ-NFA to an SDVFA of order $(1, n)$.

Example 3.1. Consider the ϵ-NFA of Fig. 3.1. Here, $\Sigma = \{a, b, c\}$. The transitions table for the newly constructed VDFA using the above procedure is given in Table 3.1.

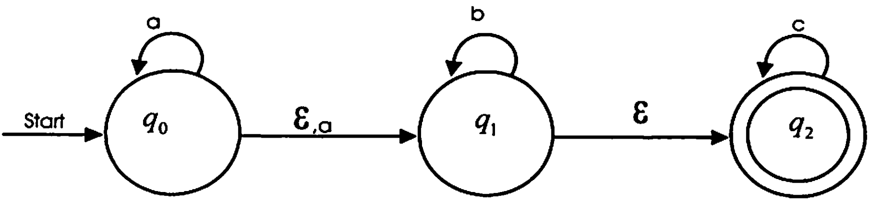

Fig. 3.1: ϵ–NFA accepting the language $\{a^u b^v c^w \mid u, v, w \geq 0\}$

State	a	b	c	ϵ
$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$
$\rightarrow \{q_0\}$	$\{q_0, q_1\}$	$\emptyset$	$\emptyset$	$\{q_0, q_1\}$
$\{q_1\}$	$\emptyset$	$\{q_1\}$	$\emptyset$	$\{q_1, q_2\}$
$* \ \{q_2\}$	$\emptyset$	$\emptyset$	$\{q_2\}$	$\{q_2\}$
$\{q_0, q_1\}$	$\{q_0, q_1\}$	$\{q_1\}$	$\emptyset$	$\{q_0, q_1, q_2\}$
$\{q_0, q_2\}$	$\{q_0, q_1\}$	$\emptyset$	$\{q_2\}$	$\{q_0, q_1, q_2\}$
$* \ \{q_1, q_2\}$	$\emptyset$	$\{q_1\}$	$\{q_2\}$	$\{q_1, q_2\}$
$* \ \{q_0, q_1, q_2\}$	$\{q_0, q_1\}$	$\{q_1\}$	$\{q_2\}$	$\{q_0, q_1, q_2\}$

Table 3.1: Transition table for the SDVFA of order $(1, n)$ constructed from the ϵ-NFA of Fig. 3.1.

Note that the state $\{q_0, q_2\}$ is not accessible in the newly constructed SDVFA and is, therefore, thrown away and not shown in the transition diagram of the newly constructed SDVFA. The three accepting states of the newly constructed SDVFA are $\{q_2\}, \{q_1, q_2\}$ and $\{q_0, q_1, q_2\}$.

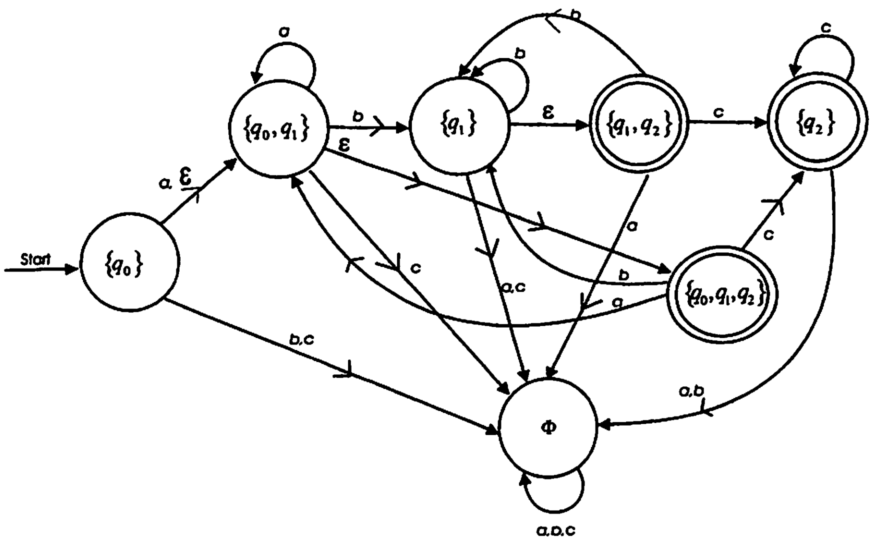

Fig. 3.2: SDVFA of order $(1, n)$ constructed from the ϵ-NFA of Fig. 3.1.

We now prove a theorem.

Theorem 3.1. *A language L is accepted by some ϵ-NFA if and only if L is accepted by some SDVFA.*

Proof. (if) Since every SDVFA of order (n, n) is ϵ-NFA where n is the total number of the states in the automaton, so this part follows trivially.

(only if) Let $V = (Q_V, \Sigma \cup \{\epsilon\}, \delta_{(1,n)}, \{q_0\}, F_V\}$ be an SDVFA of order $(1, n)$ constructed from ϵ-NFA $E = (Q_E, \Sigma \cup \{\epsilon\}, \delta_E, \{q_{0_1}\}, F_E\}$ by subset construction, then we show that $L(V) = L(E)$. What we actually prove first, by induction on $|w|$ is that $\hat{\delta}_{(1,n)}(\{q_0\}, w) = \hat{\delta}_E(q_0, w)$.

Notice that each of the $\hat{\delta}$ function returns a set of states from Q_E, but $\hat{\delta}_{(1,n)}$ interprets this set as one of the states of Q_V which is the power set of Q_E, while $\hat{\delta}_E$ interprets this set as a subset of Q_E.

Basis. Let $|w| = 0$, i.e. $w = \epsilon$. By the definition of $\hat{\delta}_{(s,t)}$ for SDVFA and ϵ-NFA, we have

$$\hat{\delta}_{(1,n)}(\{q_0\}, \epsilon) = \hat{\delta}_E(q_0, \epsilon) = v\text{-close}(q_0) = E\text{-close}(q_0).[2, 6]$$

Induction. Let w be of length $n+1$, and assume the statement for length n. Break w as $w = xa$ where a is the final symbol of w. By the induction hypothesis $\hat{\delta}_{(1,n)}(\{q_0\}, x) = \hat{\delta}_E(q_0, x)$. By the induction hypothesis $\hat{\delta}_{(1,n)}(\{q_0\}, x) = \hat{\delta}_E(q_0, x)$. Let both these sets of $E's$ states be $\{p_1, p_2, \cdots, p_k\}$.

Now, let

$$\bigcup_{i=1}^{k} \delta_E(p_i, a) = \{r_1, r_2, \cdots, r_m\} \tag{3.1}$$

Then

$$\hat{\delta}_E(q_0, w) = \cup_{j=1}^{m} E\text{-close}(r_j) = \cup_{j=1}^{m} v\text{-close}(r_j) \tag{3.2}$$

The subset construction, on the other hand tell us that

$$\delta_{(1,n)}(\{p_1, p_2, \cdots . p_k\}, a) = \bigcup_{i=1}^{k} \delta_E(p_i, a). \tag{3.3}$$

Now, let us use (3.1) and the fact that $\hat{\delta}_{(1,n)}(\{q_0\}, x) = \{p_1, p_2, \cdots, p_k\}$ in the inductive part of the definition of $\hat{\delta}_{(1,n)}$ for SDVFA,

$$\hat{\delta}_{(1,n)}(\{q_0\}, w) = \delta_{(1,n)}(\{p_1, p_2, \cdots, p_k\}, a) = \cup_{j=1}^{m} v\text{-close}(r_j) = \hat{\delta}_E(q_0, w)$$

where r_j's are given by (3.1).

Thus $\hat{\delta}_{(1,n)}(\{q_0\}, w) = \hat{\delta}_E(q_0, w)$. When we observe that V and E both accept w if and only if $\hat{\delta}_{(1,n)}(\{q_0\}, w)$ or $\hat{\delta}_E(q_0, w)$ respectively contain a state in F_E, we have a complete proof that $L(V) = L(E)$.

Hence the theorem. □

We now prove a theorem for the equivalence of SDVFA, VDFA and NFA.

Theorem 3.2. *A language L is accepted by an SDVFA if an only if L is accepted by some NFA if and only if L is accepted by some VDFA.*

Proof. We have shown that,

A Language L is acccepted by a DFA

$\iff$ L is acccepted by a SDVFA (Theorem 2.1)

$\iff$ L is acccepted by some ϵ-NFA (Theorem 3.1). (3.4)

Also, we know that for a language L,

$$\begin{aligned} L \text{ is accepted by an } \epsilon\text{-NFA} &\iff L \text{ is accepted by some NFA} \\ &\iff L \text{ is accepted by some} \\ &\qquad \text{VDFA}[2,6,8]. \end{aligned} \tag{3.5}$$

Combining (3.4) and (3.5), we get the result. □

References

[1] A.V. Aho, J.E. Hopcroft and J.D. Ullman, The Design and Analysis of Computer Algorithms, Addison-Wesley, Reading, MA, 1974.

[2] A.V. Aho and J.D. Ullman, Foundations of Computer Science, Computer Science Press, New York, 1994.

[3] J. Anderson, Automata Theory with Modern Applications, Cambridge University Press, 2006.

[4] N. Chomsky, in review of Belevitch, V., *Language des machines et langage humain*, Language, 34 (1958), p.100.

[5] N. Chomsky, *Three models for the description of languages*, IRE Trans. on Information Theory, 2:3 (1956), pp.113-124.

[6] S. Eilenberg, *Automata, Languages and Machines*, Vol. A-B, Academic Press, New York, 1974.

[7] C.C. Elgot and G. Mezei, *On relations defined by generalized finite automata*, IBM J. of Res. and Dev., 9 (1965), 47-65.

[8] M. Ganesan, P. Bhattacharya and A. Jain, *A Notion of Virtual Deterministic Finite Automaton (VDFA) in Language Processing*, J. Comput. Math. Optim., 2 (2006), 181-206.

[9] A. Gill, Introduction to the Theory of Finite-State Machines, McGraw-Hill Book Company, New York, 1962.

[10] M.A. Harrison, Introduction to Formal Language Theory, Addison-Wesley Publishing Company, Reading Maas., 1978.

[11] F.C. Hennie, Finite-State Models for Logical Machines, John Wiley & Sons, New York, 1968.

[12] A. Jain, *Semi-deterministic Virtual Finite Automaton of order* (s,t), to appear in J. of Comp. math. Optmz.

[13] M.O. Rabin and D. Scott, *Finite automata and their decision problems*, IBM J. Research and Development 3:2 (1959), pp.115-125.

[14] M.P. Schutzenberger, *On the definition of a family of automata*, Information and Control, 4 (1961).

[15] C.E. Shannon and J. McCarthy, Automata Studies, Princeton Univ. Press, 1956.

32, Uttranchal
5, I.P. Extension
Delhi
India
E-mail: jainarihant@live.com