SCIENCE CHINA
Information Sciences

• LETTER •

February 2020, Vol. 63 129203:1–129203:3
https://doi.org/10.1007/s11432-018-9739-9

Three matrix conditions for the reduction of finite automata based on the theory of semi-tensor product of matrices

Jumei YUE[1], Yongyi YAN[2*] & Zengqiang CHEN[3]

[1]*College of Agricultural Engineering, Henan University of Science and Technology, Luoyang* 471023, *China;*
[2]*College of Information Engineering, Henan University of Science and Technology, Luoyang* 471023, *China;*
[3]*College of Computer and Control Engineering, Nankai University, Tianjin* 300071, *China*

Received 12 September 2018/Revised 20 November 2018/Accepted 11 January 2019/Published online 9 August 2019

Citation Yue J M, Yan Y Y, Chen Z Q. Three matrix conditions for the reduction of finite automata based on the theory of semi-tensor product of matrices. Sci China Inf Sci, 2020, 63(2): 129203, https://doi.org/10.1007/s11432-018-9739-9

Dear editor,

Reduction of finite automata (FA) is of great importance because of its practical applications in engineering; for example the memory space of hardware realization grows exponentially with the number of states of FSMs. Existing results for reducing FA can roughly be classified into four categories: merging of states [1], refining of the state set [2,3], trimming of FA [4,5], and dynamic reduction [6,7]. Most of these results are computer-based algorithms; they are efficient when applied to engineering problems but fail to explain the mathematical essence of the reduction in science. This study focuses on the explanation of the inner logics of the reduction mathematically.

Reduction problem. An FA is a six-tuple $M = (S, A, O, A \times S, f_o, f_s)$, where S, A, and O are the material bases denoting state, input and output sets, respectively; $A \times S$ is the Cartesian product of A and S; f_o and f_s are the dynamic expressions; f_o is the output function from $A \times S$ to O; f_s is the state transition function from $A \times S$ to 2^S; and 2^S is the power set of S. Figure 1(a) shows an example of an FA.

Two FAs are said to be equivalent to each other if they satisfy the condition that for an input sequence well-defined for an FA, there is at least one state in the other FA such that the input sequence is well-defined and the two FAs produce the same outputs.

Two states, such as s_i and s_j, are said to be k-different if there is an input sequence of length k that makes M produce different outputs when it receives the sequence at s_i and s_j, respectively. States s_i and s_j are called a compatible pair of states if they are not k-different for any $k \geqslant 0$; otherwise they are called an incompatible pair of states. Given an FA M with S as its state set, its incompatible graph is a graph $G = (S, E)$, where the node set S is the state set of M, $(s_i, s_j) \in E$ iff s_i and s_j are an incompatible pair of states. The internally stable set of G is called the compatible set (CS) of M. A set of CSes containing all the states of M is referred to as a compatible enclosure of the state set (CESS). A CESS that is a closed accumulation set is called a delegate set of the state set (DSSS). Physically, a DSSS serves as the state set for a potential reduced FA. A DSSS with the least number of states is called a least delegate set of state set (LDSSS), which is the state set of the resulting reduced FA.

It is known that the reduction of an FA is exploring different ways to determine the LDSSS. This study considers the problem under the condition that the CS of the considered FA is known. The methods used to obtain all the CS are reported in

* Corresponding author (email: yyyan@mail.nankai.edu.cn)

a previous study [8]. Let M be the concerned FA with state set $S=\{s_1,s_2,\ldots,s_n\}$; denote the CS of M by $\mathcal{C}=\{C_1,C_2,\ldots,C_r\}$.

Necessary and sufficient conditions. Semi-tensor product of matrices (STP) proposed by Cheng et al. [9] is used as the mathematical tool herein; Appendix A.1 details the definition and some special properties. The following are some of the commonly used notations in the framework of STP. (1) $A\ltimes B$ denotes the STP of matrices A and B. (2) δ_m^i denotes the i-th column of the identity matrix I_m; in particular, δ_m^0 is defined as the zero vector of dimension $m\times 1$. (3) $\Delta_m:=\{\delta_m^1,\delta_m^2,\ldots,\delta_m^m\}$, $\Delta:=\Delta_2$. (4) $\delta_m[i_1,i_2,\ldots,i_n]$ denotes the compact form of matrix $[\delta_m^{i_1},\delta_m^{i_2},\ldots,\delta_m^{i_n}]$. (5) $\mathrm{Col}_i(A)$ denotes the i-th column of matrix A. (6) $\mathbf{1}_m$ denotes the row vector of dimension m with all elements being 1.

Given $\mathcal{G}\subseteq\mathcal{C}$ and $\mathcal{G}=\{C_{i_1},C_{i_2},\ldots,C_{i_s}\}$ $(1\leqslant s\leqslant r)$, for $C_i\in\mathcal{C}$ $(i=1,2,\ldots,r)$ define a vector $x_i^{\mathcal{G}}$ such that $x_i^{\mathcal{G}}=\delta_2^2$ if and only if $C_i\in\mathcal{G}$; otherwise $x_i^{\mathcal{G}}=\delta_2^1$. Here, $x_i^{\mathcal{G}}$ is referred to as the logical vector of C_i associated with $\mathcal{G}$.

Definition 1. Given $\mathcal{G}\subseteq\mathcal{C}$ and suppose that $s_i\in S$ $(1\leqslant i\leqslant n)$ is a member of $\mathcal{C}_{s_i}=\{C_{j_1},C_{j_2},\ldots,C_{j_{t_i}}\}$ $(1\leqslant t_i\leqslant r)$, according to Appendix A.5, there is a unique matrix L_{s_i} such that $x_{j_1}^{\mathcal{G}}\ltimes x_{j_2}^{\mathcal{G}}\ltimes\cdots\ltimes x_{j_{t_i}}^{\mathcal{G}}=L_{s_i}\ltimes x^{\mathcal{G}}$, where $x^{\mathcal{G}}=\ltimes_{i=1}^r x_i^{\mathcal{G}}=x_1^{\mathcal{G}}\ltimes\cdots\ltimes x_r^{\mathcal{G}}$. The enclosure vector of s_i is defined as

$$K_i^{\mathcal{G}}:=JL_{s_i},\quad J=[1\ \underbrace{0\cdots0}_{2^{t_i}-1}].$$

Theorem 1 (Compatible enclosure condition)**.** Given $\mathcal{G}=\{C_k|1\leqslant k\leqslant r,x_k^{\mathcal{G}}=\delta_2^2\}$ with $x^{\mathcal{G}}=\delta_{2^r}^l$ and $x^{\mathcal{G}}=\ltimes_{i=1}^r x_i^{\mathcal{G}}$. $\mathcal{G}$ is a CESS if and only if $\mathrm{col}_l(K)=0$, where $K=\sum_{i=1}^n K_i^{\mathcal{G}}$, $K_i^{\mathcal{G}}$ is the enclosure vector of s_i. The proof is presented in Appendix B.1.

Remark 1. (1) The compatible enclosure condition describes the mathematical meanings of a set of CSes enclosuring the state set of an FA. The mathematical description implies a mathematical algorithm for finding all the CESS of FA (refer to the compatible enclosure algorithm given in the sequel part). (2) The vector K embodies all the impossibilities that a set of CSes enclosing the state set of an FA; K is referred to as the enclosure vector of $\mathcal{G}$. (3) Theorem 1 suggests a mathematical algorithm for finding all the CESS of an FA. Given an FA M with state set $S=\{s_1,s_2,\ldots,s_n\}$ and compatible set $\mathcal{C}=\{C_1,C_2,\ldots,C_r\}$; assign each $C_i\in\mathcal{C}$ a vector $x_i\in\Delta_2$.

Algorithm 1 (Compatible enclosure algorithm)**.** Assume that s_i $(1\leqslant i\leqslant n)$ belongs to the compatible sets $\mathcal{C}_{s_i}=\{C_{j_1},C_{j_2},\ldots,C_{j_{t_i}}\}$. Every CESS can be obtained via the following procedure.

Step 1. Compute $K_i^{\mathcal{G}}$ $(i=1,2,\ldots,n)$ and K.

Step 2. Check whether there is an $l\in\{1,2,\ldots,2^r\}$ such that $\mathrm{col}_l(K)=0$. If not, there is no CESS. Otherwise, set $L=\{l\,|\,\mathrm{col}_l(K)=0\}$.

Step 3. For each $l\in L$, compute x_j by $x_j=S_j^r\ltimes\delta_{2^r}^l$, $S_j^r=(E_d)^{r-1}W_{[2^j,2^{r-j}]}$ $(j=1,2,\ldots,r)$. Then $\mathcal{F}_l=\{C_k\,|\,1\leqslant k\leqslant r,x_k=\delta_2^2\}$ is a CESS.

Step 4. Repeat Step 3 until every element in L is evaluated and all the CESS are obtained: $\mathcal{F}=\{\mathcal{F}_l\,|\,l\in L,\mathcal{F}_l$ is obtained using step 3$\}$.

Treat C_i as a "state package" and construct a set C_{i_a} as $C_{i_a}=\{s'\,|\,s'\in S,$ exist $s\in C_i$ such that $f_s(a,s)=s'\}$. The set $\mathcal{E}_{i_a}=\{C_k\,|\,1\leqslant k\leqslant r,C_{i_a}\subseteq C_k\}$ is called the state transfer package of C_i about an input $a\in A$.

Definition 2. Given $\mathcal{G}\subseteq\mathcal{C}$, let $x_i^{\mathcal{G}}$ be the logical vector of C_i associated with $\mathcal{G}$ and $\mathcal{E}_{i_a}=\{C_{j_1},C_{j_2},\ldots,C_{j_{t_i}}\}$ be the state transfer package of C_i about an input $a\in A$. According to Appendix A.7, there is a unique matrix P_i such that $\bar{x}_i^{\mathcal{G}}\ltimes x_{j_1}^{\mathcal{G}}\ltimes x_{j_2}^{\mathcal{G}}\ltimes\cdots\ltimes x_{j_{t_i}}^{\mathcal{G}}=P_i\ltimes x^{\mathcal{G}}$, where $\bar{x}_i^{\mathcal{G}}$ is the logical negation of $x_i^{\mathcal{G}}$, $x^{\mathcal{G}}=x_1^{\mathcal{G}}\ltimes x_2^{\mathcal{G}}\ltimes\cdots\ltimes x_r^{\mathcal{G}}$. The transfer vector of C_i about an input $a\in A$ is defined as

$$T_{i_a}^{\mathcal{G}}=\begin{cases}J\ltimes P_i,\ J=[1\ \underbrace{0\cdots0}_{2^{t_i+1}-1}], & \text{if } |C_{i_a}|>1;\\ [0\cdots0]_{1\times 2^r}, & \text{otherwise.}\end{cases}$$

Theorem 2 (Delegate set condition)**.** Given $\mathcal{G}=\{C_k\,|\,1\leqslant k\leqslant r,x_k^{\mathcal{G}}=\delta_2^2\}$ with $x^{\mathcal{G}}=\delta_{2^r}^l$ and $x^{\mathcal{G}}=\ltimes_{i=1}^r x_i^{\mathcal{G}}$. $\mathcal{G}$ is a DSSS if and only if $\mathrm{col}_l(H)=0$, where

$$\begin{cases}H=T+K,\\ T=\displaystyle\sum_{i=1}^r\sum_{a\in A}T_{i_a}^{\mathcal{G}},\end{cases}$$

in which $T_{i_a}^{\mathcal{G}}$ is the transfer vector of C_i about an input $a\in A$ and K is the enclosure vector of $\mathcal{G}$. See Appendix B.2 for the proof.

Remark 2. (1) The delegate set condition mathematically explains how a set of CSes can be a DSSS of an FA. The mathematical explanation provides a way to find all the DSSS of FA, shown in the delegate set algorithm in the sequel part. (2) The vector H contains all the conditions under which a set of CSes become a DSSS of an FA; we call H the delegate vector of $\mathcal{G}$. (3) Theorem 2 offers a way to develop an algorithm to find all the DSSS.

Algorithm 2 (Delegate set algorithm)**.** Let $\mathcal{E}_{i_a}$ be the state transfer package of C_i about an input $a\in A$, $1\leqslant i\leqslant r$. The following procedure obtains every DSSS.

Step 1. Compute K following the compatible enclosure condition.

Step 2. Compute T and H.

Step 3. Check whether there is an $l \in \{1, 2, \ldots, 2^r\}$ such that $\mathrm{col}_l(H) = 0$. If not, there is no DSSS. Otherwise, set $L = \{l \,|\, \mathrm{col}_l(H) = 0\}$.

Step 4. For each $l \in L$, compute x_j by $x_j = S_j^r \ltimes \delta_{2^r}^l$, $S_j^r = (E_d)^{r-1} W_{[2^j, 2^{r-j}]}$ $(j = 1, 2, \ldots, r)$. A DSSS $\mathcal{R}_l$ is then obtained $\mathcal{R}_l = \{C_k \,|\, 1 \leqslant k \leqslant r, x_k = \delta_2^2\}$.

Step 5. Repeat Step 4 till every element in L is evaluated and all the DSSS are obtained $\mathcal{R} = \{\mathcal{R}_l \,|\, l \in L, \mathcal{R}_l$ is obtained by Step 4$\}$.

Definition 3. Let $x \in \Delta_k$ be k-valued logical variables; the vector $Q = JE$ is called the sum vector of x, where $J = [1\ 0]$ and E is defined as in Appendix A.8.

Theorem 3 (Least delegate set condition)**.** Given $\mathcal{G} = \{C_k \,|\, 1 \leqslant k \leqslant r, x_k^{\mathcal{G}} = \delta_2^2\}$ with $x^{\mathcal{G}} = \delta_{2^r}^l$, and $x^{\mathcal{G}} = \ltimes_{i=1}^r x_i^{\mathcal{G}}$. $\mathcal{G}$ is an LDSSS if and only if $\mathrm{col}_l(N) = \min(\mathrm{col}(N))$, where $N = 2rH - Q$, in which H is the delegate vector of $\mathcal{G}$ and Q is the sum vector of $x_i^{\mathcal{G}}$ $(i = 1, \ldots, r)$. See Appendix B.3 for the details of the proof.

Remark 3. The least delegate set condition formulates the mathematical structure of an LDSSS of an FA, which provides methods for obtaining all the LDSSS.

Algorithm 3 (Least delegate set algorithm)**.** For the given FA described in Algorithm 2, all LDSSS can be obtained using the following procedure.

Step 1. Compute N.

Step 2. Set $L = \{l \,|\, \mathrm{col}_l(N) = \min(\mathrm{col}(N))\}$.

Step 3. For each $l \in L$, compute x_j by $x_j = S_j^r \ltimes \delta_{2^r}^l$, where $S_j^r = (E_d)^{r-1} W_{[2^j, 2^{r-j}]}$ $(j = 1, 2, \ldots, r)$. Thus, an LDSSS is obtained $\mathcal{R}_l = \{C_k \,|\, 1 \leqslant k \leqslant r, x_k = \delta_2^2\}$.

Step 4. Repeat step 3 until every element in L is evaluated and all the LDSSS are obtained $\mathcal{R} = \{\mathcal{R}_l \,|\, l \in L, \mathcal{R}_l$ is obtained by step 3$\}$.

Remark 4. Several studies have proposed new methods of solving the reduction problem of FA for various purposes. This study uniquely contributes by the mathematical formulation of the problem within the framework of the STP theory. The former is a mathematical description that focuses on how to simplify FA from the theoretical standpoint, aiming to reveal the mathematics of the reduction. The latter are computer-based algorithms that lay emphasis on reduction algorithm, stressing the computation gain.

Illustrating example. Using the mathematical algorithms, such as compatible enclosure algorithm, delegate set algorithm, and least delegate set algorithm, the FA with five states, shown in Figure 1(a), can be reduced into an FA with three states, 20% decrease in the number of states; Figure 1(b) shows three examples of such reduced FA.

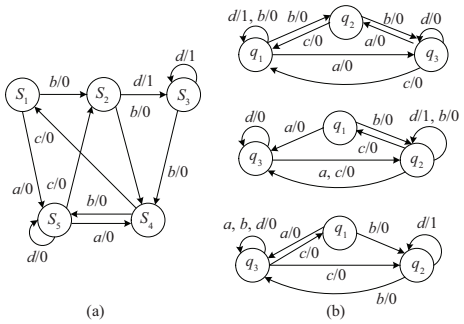

Figure 1 (a) An example of FA; (b) reduced counterparts of the FA.

Remark 5. The three reduced FAs are equivalent to the original FA and they are, according to [10], isomorphic in nature.

Acknowledgements This work was supported by National Natural Science Foundation of China (Grant Nos. U1804150, 61573199) and 2018 Henan Province Science and Technique Foundation (Grant No. 182102210045).

Supporting information Appendixes A and B. The supporting information is available online at info.scichina.com and link.springer.com. The supporting materials are published as submitted, without typesetting or editing. The responsibility for scientific accuracy and content remains entirely with the authors.

References

1 Almeida J, Zeitoun M. Description and analysis of a bottom-up DFA minimization algorithm. Inf Process Lett, 2008, 107: 52–59
2 David J. The average complexity of Moore's state minimization algorithm is $O(n\log\log n)$. Lect Notes Comput Sci, 2010, 6281: 318–329
3 Peeva K. Equivalence, reduction and minimization of finite automata over semirings. Theory Comput Sci, 1991, 88: 269–285
4 Lamperti G, Scandale M. Incremental determinization and minimization of finite acyclic automata. In: Proceedings of IEEE International Conference on Systems, Man, and Cybernetics, 2013. 2250–2257
5 Holzer M, Kutrib M. Descriptional and computational complexity of finite automata: a survey. Inf Comput, 2011, 209: 456–470
6 Carrasco R C, Forcada M L. Incremental construction and maintenance of minimal finite-state automata. Comput Linguist, 2002, 28: 207–216
7 Zhang ZP, Chen Z Q, Liu Z X. Modeling and reachability of probabilistic finite automata based on semitensor product of matrices. Sci China Inf Sci, 2018, 61: 129202
8 Yue J M, Chen Z Q, Yan Y Y, et al. Solvability of k-track assignment problem: a graph approach. Control Theory Appl, 2017, 34: 457–466
9 Cheng D Z, Qi H S, Zhao Y. An Introduction to Semi-Tensor Product of Matrices and Its Applications. Singapore: World Scientific Publishing, 2012
10 Aho A V, Sethi R, Ullman J D. Compilers, Principles, Techniques, and Tools. Boston: Addison-Wesley Publishing Corporation, 1986