Frontiers of Information Technology & Electronic Engineering
www.jzus.zju.edu.cn; engineering.cae.cn; www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

State space optimization of finite state machines from the viewpoint of control theory[*]

Jumei YUE[†1], Yongyi YAN[†‡2], Zengqiang CHEN[3], He DENG[2]
[1]*College of Agricultural Equipment Engineering, Henan University of Science and Technology, Luoyang 471003, China*
[2]*College of Information Engineering, Henan University of Science and Technology, Luoyang 471003, China*
[3]*College of Artificial Intelligence, Nankai University, Tianjin 300071, China*
[†]E-mail: yjm@mail.nankai.edu.cn; yyyan@mail.nankai.edu.cn

Received Nov. 6, 2020; Revision accepted Jan. 3, 2021; Crosschecked July 19, 2021; Published online Oct. 23, 2021

Abstract: Motivated by the inconvenience or even inability to explain the mathematics of the state space optimization of finite state machines (FSMs) in most existing results, we consider the problem by viewing FSMs as logical dynamic systems. Borrowing ideas from the concept of equilibrium points of dynamic systems in control theory, the concepts of *t*-equivalent states and *t*-source equivalent states are introduced. Based on the state transition dynamic equations of FSMs proposed in recent years, several mathematical formulations of *t*-equivalent states and *t*-source equivalent states are proposed. These can be analogized to the necessary and sufficient conditions of equilibrium points of dynamic systems in control theory and thus give a mathematical explanation of the optimization problem. Using these mathematical formulations, two methods are designed to find all the *t*-equivalent states and *t*-source equivalent states of FSMs. Further, two ways of reducing the state space of FSMs are found. These can be implemented without computers but with only pen and paper in a mathematical manner. In addition, an open question is raised which can further improve these methods into unattended ones. Finally, the correctness and effectiveness of the proposed methods are verified by a practical language model.

Key words: Finite state machines; Finite-valued systems; Logical systems; Logical networks; Semi-tensor product of matrices; Space optimization
https://doi.org/10.1631/FITEE.2000608 **CLC number:** TP13

1 Introduction

A finite state machine (FSM) is a mathematical model of computation and is widely used in some areas such as artificial intelligence (Voeten and van Zaanen, 2018), robotics (He et al., 2019), linguistics (Chu and Spinney, 2018), and video coding (Kamble et al., 2018). In addition, FSM theory plays an important role in many areas of modern computer science (Yan et al., 2019a).

State space optimization of FSMs is an important issue in FSM theory, and is of great significance in practical applications in engineering because the memory space needed for hardware realization grows exponentially with the increase of the number of states of FSMs. The problem of state space optimization of FSMs refers to the minimization of the number of the states of an FSM with the resulting FSM retaining the same functions as the original FSM.

The optimization of FSMs has been extensively studied from various aspects, including different methods used for the same type of FSM, similar approaches used for different types of FSM, and some with different emphases on some particular issues. For lattice-valued multiset FSMs, Li and Pedrycz (2007) and Wang and Li (2018) established

[‡] Corresponding author
[*] Project supported by the National Natural Science Foundation of China (Nos. U1804150, 62073124, and 61973175)
ORCID: Jumei YUE, https://orcid.org/0000-0003-1474-0088; Yongyi YAN, https://orcid.org/0000-0002-7181-5894

equivalences of nondeterministic lattice multiset FSMs and deterministic ones. Based on the equivalences, they further proposed effective algorithms to produce a minimal substitution of a given lattice multiset FSM. For Mealy and Moore FSMs, Solov'ev (2011, 2017) used two methods to study the state space optimization of Mealy FSMs by merging internal states and using output variables of state assignment based on structural models of FSMs. It was shown that the methods can reduce the area of FSM implementation for all families of field programmable gate arrays (FPGAs) of various manufacturers. The minimization of Moore FSMs was investigated by gluing states and representing an FSM in the form of a transition list (Solov'ev, 2010). With regard to the state space optimization of incompletely specified FSMs, the way of merging internal states used originally for Mealy FSM minimization is applied to this type of FSM by Klimowicz and Solov'ev (2013). The proposed minimization methods can considerably reduce the number of internal states and the number of FSM transitions. There are other studies on state space optimization with a specific emphasis, such as state reduction (Gören and Ferguson, 2007), exact minimization (Gören and Ferguson, 2002), and dimensionality minimization (Perkowski et al., 2001). On the state space optimization of deterministic FSMs (DFSMs), there are some excellent methods, for example, using backward depth information (Liu et al., 2016), inspired by Brzozowski's algorithm (García et al., 2014), the double reversal minimization algorithm (de Parga et al., 2013), and the complex method (Solov'ev, 2014). Also, there are some techniques with different focuses, for example, edge minimization (Melnikov, 2010), power consumption (Grzes and Solov'ev, 2015), and transition minimization (Dahmoune et al., 2014).

Most studies, like those mentioned above, are computer-based in nature. This can be either inconvenient or even unable to reveal the mathematical meanings of the optimization of FSMs. This motivates further consideration of the state space optimization problem from the standpoint of control science, where FSMs are viewed as logical dynamic systems, aiming to reveal the mathematical aspects of state space optimization. The research is based on the state transition dynamic equation of FSMs established under the framework in control theory in recent years (Xu and Hong, 2012; Yan et al., 2015). In Xu and Hong (2012) and Yan et al. (2015), the dynamics of an FSM is formulated as Eqs. (1) and (2), respectively:

$$\begin{cases} \boldsymbol{x}(t+1)=\boldsymbol{F}\boldsymbol{u}(t)\boldsymbol{x}(t), \\ \boldsymbol{y}(t)=\boldsymbol{H}\boldsymbol{u}(t)\boldsymbol{x}(t), \end{cases} \tag{1}$$

and

$$\begin{cases} \boldsymbol{x}(t+1)=\boldsymbol{T}^t\boldsymbol{x}_0\boldsymbol{u}(t), \\ \boldsymbol{y}(t)=\boldsymbol{K}^t\boldsymbol{x}_0\boldsymbol{u}(t), \end{cases} \tag{2}$$

where $\boldsymbol{x}(t+1)$ is the state at time step $t+1$, $\boldsymbol{x}_0$ is an initial state, $\boldsymbol{u}(t)$ is an input string of length t, $\boldsymbol{y}(t)$ is the output at time step t, and $\boldsymbol{T}^t$ is a matrix called a dynamical matrix of an FSM. Eqs. (1) and (2) share similarities in results while having differences conceptually. In Eq. (1) the state of an FSM is defined as a vector $\boldsymbol{x}(t)=(x_1(t), x_2(t), \ldots, x_n(t))^{\mathrm{T}}$, where $x_i(t)$ (i=1, 2, …, n) is the number of different paths by which the FSM can reach state x_i with an input string of length $t-1$, while in Eq. (2) state x_i is simply defined as a vector $\boldsymbol{\delta}_n^i$ ($\boldsymbol{\delta}_n^i$ is the column vector corresponding to the ith column of unit matrix $\boldsymbol{I}_n$). Eq. (1) is suitable for both DFSMs and nondeterministic FSMs (NFSMs), and describes the 1-step dynamics, while Eq. (2) is suitable only for DFSMs, and formulates the t-step dynamics. Thus, Eq. (1) excels in modeling FSMs, and Eq. (2) has advantages in expressing the dynamics. Based on Eq. (2), one can analyze and synthesize FSMs like studying dynamic systems using control theory. The ideas such as feedback (Han and Chen, 2018; Zhang ZP et al., 2018b), the concepts such as controllability, observability, reachability, and stabilization (Yan et al., 2016; Zhang KZ and Zhang, 2016; Han et al., 2018; Zhang ZP et al., 2018a), and the methods such as state feedback control, corrective control, and fault tolerant control (Gao et al., 2017; Lu et al., 2017; Zhang ZP et al., 2017) have been transplanted to FSMs.

The dynamic model of FSMs (2) is adopted in this study to investigate the state space optimization of FSMs because those considered here are limited to DFSMs since DFSMs are simple and easy to implement in engineering, and NFSMs are equivalent to DFSMs (Gören and Ferguson, 2007; Li and Pedrycz, 2007). Here "equivalent" means that for any NFSM there exists a DFSM with the same function as the nondeterministic one.

The main contributions of this paper are as follows. Taking the concept of equilibrium points in control theory, the concepts of *t*-equivalent states and *t*-source equivalent states of FSMs are proposed. Based on the state transition dynamic equation (2), several necessary and sufficient conditions are proposed for *t*-equivalent states and *t*-source equivalent states. These can be analogized to the necessary and sufficient conditions of equilibrium points of dynamic systems in control theory and thus explain mathematically the optimization problem. Two methods are designed to find all the *t*-equivalent states and *t*-source equivalent states of FSMs. These two methods are further applied to develop two methods of reducing FSMs. Unlike the computer-based algorithms proposed by researchers in computer science, the proposed methods are independent of computers and can be implemented mathematically with pen and paper; of course, they can also be easily conducted on computers.

2 Preliminaries and problem description

In this section we first briefly review the research tool, the semi-tensor product (STP) of matrices, and the research object, FSMs, and then give a problem description.

2.1 Semi-tensor product of matrices

Definition 1 (STP of matrices) (Cheng et al., 2012) For any two matrices $\boldsymbol{M}_{m\times n}$ and $\boldsymbol{N}_{p\times q}$, their STP, denoted by $\boldsymbol{M}\ltimes\boldsymbol{N}$, is defined as

$$\boldsymbol{M}\ltimes\boldsymbol{N}:=(\boldsymbol{M}\otimes\boldsymbol{I}_{s/n})(\boldsymbol{N}\otimes\boldsymbol{I}_{s/p}),$$

where s is the least common multiple of n and p, and $\otimes$ is the Kronecker product of matrices.

Remark 1 In practice, a special form of the STP where n and p are of multiple relations is often used, which is given as follows:

1. Let $\boldsymbol{X}$ be a row vector of dimension np, and $\boldsymbol{Y}$ be a column vector with dimension p. Split $\boldsymbol{X}$ into p blocks $\boldsymbol{X}_1, \boldsymbol{X}_2, \ldots, \boldsymbol{X}_p$ of equal size, which are $1\times n$ row vectors. The STP, denoted by $\ltimes$, is defined as

$$\begin{cases}\boldsymbol{X}\ltimes\boldsymbol{Y}=\sum_{i=1}^{p}\boldsymbol{X}_i y_i\in\mathbb{R}^n,\\ \boldsymbol{Y}^{\mathrm{T}}\ltimes\boldsymbol{X}^{\mathrm{T}}=\sum_{i=1}^{p}y_i(\boldsymbol{X}_i)^{\mathrm{T}}\in\mathbb{R}^n,\end{cases}$$

where $y_i\in\mathbb{R}$ is the ith component of $\boldsymbol{Y}$.

2. Let $\boldsymbol{A}$ and $\boldsymbol{B}$ be $m\times n$ and $p\times q$ matrices, respectively. If either n is a factor of p or p is a factor of n, then the STP of $\boldsymbol{A}$ and $\boldsymbol{B}$, denoted by $\boldsymbol{C}=\{\boldsymbol{C}_{ij}\}=\boldsymbol{A}\ltimes\boldsymbol{B}$, is defined as follows: $\boldsymbol{C}$ consists of $m\times q$ blocks and each block is defined as

$$\boldsymbol{C}_{ij}=\boldsymbol{A}^i\ltimes\boldsymbol{B}_j,\ i=1,2,\cdots,m,\ j=1,2,\cdots,q,$$

where $\boldsymbol{A}^i$ is the ith row of $\boldsymbol{A}$ and $\boldsymbol{B}_j$ is the jth column of $\boldsymbol{B}$.

Remark 2 (1) STP generalizes the conventional multiplication of matrices. It reduces to the latter when n=p. In this study the multiplication of matrices defaults to the STP of matrices, and the symbol "$\ltimes$" is omitted except for special emphases. (2) The STP of matrices not only retains almost all the major properties of the conventional matrix multiplication (for instance, the associative law is that for $\boldsymbol{A}_{m\times n}$, $\boldsymbol{B}_{p\times q}$, and $\boldsymbol{C}_{r\times s}$, we have $(\boldsymbol{A}\ltimes\boldsymbol{B})\ltimes\boldsymbol{C}=\boldsymbol{A}\ltimes(\boldsymbol{B}\ltimes\boldsymbol{C})$), but also overcomes some inherent defects of the latter. Let us look at an interesting phenomenon of the conventional matrix multiplication: "legal operations" leads to "illegal outcomes." Let $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{W}\in\mathbb{R}^n$ be column vectors. Since $\boldsymbol{Y}^{\mathrm{T}}\boldsymbol{Z}$ is a real number, we have that $(\boldsymbol{X}\boldsymbol{Y}^{\mathrm{T}})(\boldsymbol{Z}\boldsymbol{W}^{\mathrm{T}})=\boldsymbol{X}(\boldsymbol{Y}^{\mathrm{T}}\boldsymbol{Z})\boldsymbol{W}^{\mathrm{T}}=(\boldsymbol{Y}^{\mathrm{T}}\boldsymbol{Z})(\boldsymbol{X}\boldsymbol{W}^{\mathrm{T}})$ is an $m\times n$ matrix. Continuing application of the associativity leads to $(\boldsymbol{Y}^{\mathrm{T}}\boldsymbol{Z})(\boldsymbol{X}\boldsymbol{W}^{\mathrm{T}})=\boldsymbol{Y}^{\mathrm{T}}(\boldsymbol{Z}\boldsymbol{X})\boldsymbol{W}^{\mathrm{T}}$. Now there is a question "what is the term $\boldsymbol{Z}\boldsymbol{X}$?" Clearly, it is an illegal term. This shows that the conventional multiplication of matrices is not completely "compatible." However, under the framework of the STP of matrices, such an abnormal phenomenon turns normal. (3) The STP of matrices has been developed well and applied successfully in many areas, such as Boolean networks (Yan et al., 2019c; Yue et al., 2019a; Zhang QL et al., 2020), applied graph theory (Yan et al., 2019b), finite state machines (Yue et al., 2019b, 2020a, 2020b), and matrix theory (Yue and Yan, 2019).

2.2 Finite state machines

Definition 2 (FSMs) (Yan et al., 2019a) An FSM is a five-tuple M=(X, E, f, x_0, X_F), where X and E denote two finite sets of states and input symbols, respectively, f is a state transition function from X×E to X, x_0 is an initial state at which M begins to run, and X_F is a set of final states indicating that M accepts an input string if M reads the input string and stops at one state of X_F. Fig. 1a is an example, where X={x_1, x_2, x_3, x_4, x_5, x_6}, E={0, 1}, x_0=x_1, X_F={x_6}.

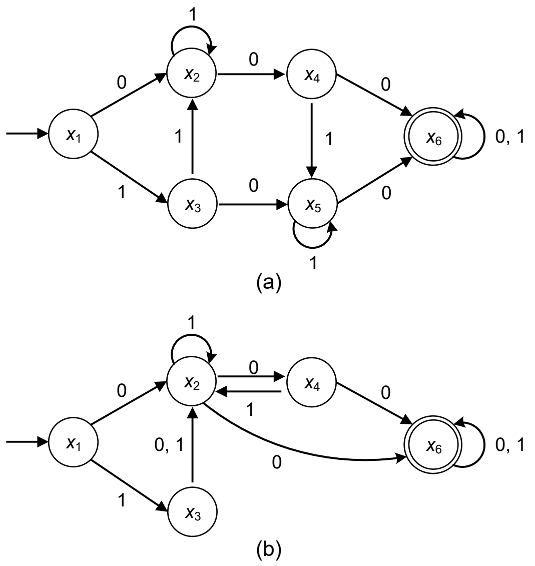

Fig. 1 The original finite state machine (a) and merged finite state machine (b)

FSMs are of two kinds, deterministic and nondeterministic. An FSM M is said to be deterministic if $|f(x, e)|\leq 1$ holds for each $x\in X$ and $e\in E$, where $|f(x, e)|$ denotes the number of states to which M is moved by input e from state x; otherwise, M is called an NFSM. Since NFSMs are equivalent to DFSMs, and DFSMs are used widely in practice, the FSMs considered in this study are limited to DFSMs.

The dynamical evolution that FSM M reads an input string $e=e_1e_2\ldots e_t\in E^*$ at state $x\in X$ is defined as

$$f(e,x)=f(f(\cdots f(f(x,e_1),e_2)\cdots),e_t), \tag{3}$$

where E^* denotes the set of finite strings on E={e_1, e_2, …, e_m}. The meaning of Eq. (3) is that M starts in state x and makes a transition to state $f(x, e_1)\in X$ if event e_1 occurs at state x. Next, if an event e_2 occurs at the new state $f(x, e_1)$, M transfers to state $f(f(x, e_1), e_2)$, and so on.

2.3 Problem description

An FSM is originally a concept in the field of computer science, where the state evolution is expressed by diagrams, tables, or discrete functions. These kinds of representations are simple and intuitive, and provide the basis for designing algorithms to accomplish various kinds of particular functionalities. However, they are not easy for studying the state evolution of FSMs mathematically, i.e., not like investigating the dynamics of linear and nonlinear systems by manipulating their dynamic equations.

In recent years the state transition dynamics of an FSM is formulated as a dynamic equation (4) (Yan et al., 2015), called the state transition dynamic equation, with which one can analyze the dynamics of FSMs like using dynamic equations to investigate the behaviors of dynamic systems in the area of control theory:

$$\boldsymbol{x}(t+1)=\boldsymbol{T}^t\boldsymbol{x}_0\boldsymbol{u}(t). \tag{4}$$

In Eq. (4) both the states $\boldsymbol{x}(t)$ and $\boldsymbol{x}_0$ and input string $\boldsymbol{u}(t)$ are vectors. Specifically, for FSM M=(X, E, f, x_0, X_F), where X={x_1, x_2, ..., x_n} and E={e_1, e_2, …, e_m}, state x_i is identified with $\boldsymbol{\delta}_n^i$, and input symbol e_j with $\boldsymbol{\delta}_m^j$; $\boldsymbol{\delta}_n^i$ and $\boldsymbol{\delta}_m^j$ are called the vector forms of state x_i and input symbol e_j, respectively, where $\boldsymbol{\delta}_p^q$ is the column vector corresponding to the q^{th} column of the p×p unit matrix $\boldsymbol{I}_p$. For an input string $\boldsymbol{u}(t)$=$e_1e_2\ldots e_t$, its vector form is $\ltimes_{j=1}^t\boldsymbol{\delta}_m^j$. This representation is referred to as the vectorization of M. This is intended to introduce the STP of matrices to model the dynamics of M as a dynamic equation. In this study the vector forms of state x_i and input symbol e_j are used interchangeably with x_i and e_j for the convenience of use.

The aim of this paper is to find, based on the dynamic equation (4), the simplest substitutions of FSMs by following the style of control theory. A simplest substitution of an FSM is an FSM that achieves the same function with the minimum number of states, for example, generation of the same language. For this purpose, we propose the concepts of t-string set between states, t-equivalent states and t-source equivalent states; we explore some mathematics of t-equivalent states, and further design several algebraic methods to find the simplest substitutions of FSMs. In addition, an open question is posed, by which the proposed optimization method can be further optimized to an unattended one.

The following are the notations used in this paper: $\boldsymbol{\delta}_n^i$, the column vector corresponding to the ith column of the unit matrix $\boldsymbol{I}_n$; Col$_i(\boldsymbol{A})$, the ith column of matrix $\boldsymbol{A}$; $|w|$, the length of string w, i.e., the number of symbols contained in w; α_β, the connection of strings α and β; S_1-S_2, the subtraction of sets S_1 and S_2, i.e., $S_1-S_2=\{x|x\in S_1, x\notin S_2\}$; $\varnothing$, the empty set.

3 Main results

In this section we present the main results of this paper in two parts. The first part includes the theoretical results containing the concepts and mathematical formulations of t-equivalent states and t-source equivalent states, which are the basis of designing the methods of reducing FSMs presented in the second part.

3.1 Theories

Definition 3 (t-string set between states) Let $M=(X, E, f, x_0, X_F)$ be an FSM, where $X=\{x_1, x_2, ..., x_n\}$ and $E=\{e_1, e_2, …, e_m\}$. The set of input strings of length t moving M to x_j from x_i is called the t-string set from x_i to x_j, denoted by $S_t(x_i\rightarrow x_j)$, i.e.,

$$S_t(x_i \rightarrow x_j)=\{\omega|\omega\in E^*,\ |\omega|=t, f(x_i,\omega)=x_j\}.$$

Definition 4 (t-equivalent states and t-different states) States x_{i_1} and x_{i_2} are called t-equivalent states if the following Eq. (5) holds for any $\omega\in S_t(x_{i_1}\rightarrow x_{j_1})\cap S_t(x_{i_2}\rightarrow x_{j_2})$:

$$f(x_{i_1},\omega)=f(x_{i_2},\omega). \tag{5}$$

Otherwise, they are called t-different states; that is, there is an $\omega\in S_t(x_{i_1}\rightarrow x_{j_1})\cap S_t(x_{i_2}\rightarrow x_{j_2})$ such that

$$f(x_{i_1},\omega)\neq f(x_{i_2},\omega).$$

Remark 3 (1) Since $f(x_{i_1},\omega)=x_{j_1}$ and $f(x_{i_2},\omega)=x_{j_2}$, the t-equivalence of states x_{i_1} and x_{i_2} implies that $x_{j_1}=x_{j_2}$ in Definition 4. (2) Clearly, as a special case, state x_i is t-equivalent to itself. (3) The t-equivalence of states satisfies the following three properties: reflexivity, symmetry, and transitivity; therefore, it is an equivalence relation on the set of states X. (4) If states $x_{i_1}, x_{i_2}, \cdots, x_{i_s}$ are t-equivalent, they have exactly the same functions. In other words, they exhibit exactly the same behaviors in the dynamic evolutions of FSMs. (5) To reduce the state space of FSMs, it is sufficient to find the equivalent states and merge them to one.

For the convenience of description in the sequel, we introduce a partition of matrices.

r-column equipartition of matrices: let $\boldsymbol{M}$ be a matrix of size $p\times qr$. The r-column equipartition of $\boldsymbol{M}$ refers to the following partition that divides $\boldsymbol{M}$ into r equally sized blocks:

$$\boldsymbol{M}=[\mathrm{Blk}_1(\boldsymbol{M}),\cdots,\mathrm{Blk}_i(\boldsymbol{M}),\cdots,\mathrm{Blk}_r(\boldsymbol{M})],$$

where

$$\mathrm{Blk}_i(\boldsymbol{M})=[\mathrm{Col}_{(i-1)q+1}(\boldsymbol{M}),\mathrm{Col}_{(i-1)q+2}(\boldsymbol{M}),\\ \cdots,\mathrm{Col}_{iq}(\boldsymbol{M})],\ i=1,2,...,r.$$

By the definition of the STP of matrices and direct calculation, it is easy to obtain the following proposition, which will be used extensively in the proofs of the theorems of this paper:

Proposition 1 Let $\boldsymbol{M}$ be a matrix of size $p\times qr$. Then

$$\boldsymbol{M}\boldsymbol{\delta}_r^i=\mathrm{Blk}_i(\boldsymbol{M}), \tag{6}$$

where Blk$_i(\boldsymbol{M})$ (i=1, 2, …, r) is the ith block of r-column equipartition of $\boldsymbol{M}$.

Theorem 1 (Equivalent condition of states) Let Eq. (7) be the state transition dynamic equation of FSM $M=(X, E, f, x_0, X_F)$, where $X=\{x_1, x_2, ..., x_n\}$ and $E=\{e_1, e_2, …, e_m\}$:

$$\boldsymbol{x}(t+1)=\boldsymbol{T}^t\boldsymbol{x}_i\boldsymbol{u}(t). \tag{7}$$

Then x_{i_1} and x_{i_2} are t-equivalent if and only if

$$\mathrm{Blk}_{i_1}(\boldsymbol{T}^t)=\mathrm{Blk}_{i_2}(\boldsymbol{T}^t).$$

Proof (Necessity) Since states x_{i_1} and x_{i_2} are t-equivalent, it follows from Definition 4 that the following Eq. (8) holds for any $\omega \in S_t(x_{i_1} \to x_{j_1}) \cap S_t(x_{i_2} \to x_{j_2})$:

$$f(x_{i_1}, \omega) = f(x_{i_2}, \omega). \tag{8}$$

By Eq. (7), Eq. (8) can be rewritten as

$$\boldsymbol{T}^t x_{i_1} \omega = \boldsymbol{T}^t x_{i_2} \omega, \tag{9}$$

where ω is the input string $\boldsymbol{u}(t)$ of Eq. (7).

Note that the vector forms of x_{i_1} and x_{i_2} are $\boldsymbol{\delta}_n^{i_1}$ and $\boldsymbol{\delta}_n^{i_2}$, respectively. Substituting the vector forms into Eq. (9) yields

$$\boldsymbol{T}^t \boldsymbol{\delta}_n^{i_1} \omega = \boldsymbol{T}^t \boldsymbol{\delta}_n^{i_2} \omega. \tag{10}$$

Combining Eq. (6) with Eq. (10), we have that $\mathrm{Blk}_{i_1}(\boldsymbol{T}^t)\omega = \mathrm{Blk}_{i_2}(\boldsymbol{T}^t)\omega$. The arbitrariness of ω guarantees that $\mathrm{Blk}_{i_1}(\boldsymbol{T}^t) = \mathrm{Blk}_{i_2}(\boldsymbol{T}^t)$. The necessity is proved.

(Sufficiency) The known condition is $\mathrm{Blk}_{i_1}(\boldsymbol{T}^t) = \mathrm{Blk}_{i_2}(\boldsymbol{T}^t)$. We prove the sufficiency by contradiction. Assume x_{i_1} and x_{i_2} are not t-equivalent. By Definition 4, there exists an $\omega' \in S_t(x_{i_1} \to x_{j_1}) \cap S_t(x_{i_2} \to x_{j_2})$ such that

$$f(x_{i_1}, \omega') \neq f(x_{i_2}, \omega'). \tag{11}$$

Using Eq. (7), we know that the left side of inequality (11) is

$$\boldsymbol{T}^t x_{i_1} \omega', \tag{12}$$

and the right side of inequality (11) is

$$\boldsymbol{T}^t x_{i_2} \omega'. \tag{13}$$

Substituting the vector forms of x_{i_1} and x_{i_2}, $\boldsymbol{\delta}_n^{i_1}$ and $\boldsymbol{\delta}_n^{i_2}$, into expressions (12) and (13), respectively, we have

$$\boldsymbol{T}^t \boldsymbol{\delta}_n^{i_1} \omega', \tag{14}$$

and

$$\boldsymbol{T}^t \boldsymbol{\delta}_n^{i_2} \omega'. \tag{15}$$

According to Proposition 1, expressions (14) and (15) are $\mathrm{Blk}_{i_1}(\boldsymbol{T}^t)\omega'$ and $\mathrm{Blk}_{i_2}(\boldsymbol{T}^t)\omega'$, respectively.

Thus, inequality (11) can be rephrased as $\mathrm{Blk}_{i_1}(\boldsymbol{T}^t)\omega' \neq \mathrm{Blk}_{i_2}(\boldsymbol{T}^t)\omega'$.

On the other hand, let the vector form of ω' be $\ltimes_{i=1}^{t} \delta_m^i = \delta_{m^t}^l$. Then Proposition 1 shows that the l^{th} block of $\mathrm{Blk}_{i_1}(\boldsymbol{T}^t)$ is different from the l^{th} block of $\mathrm{Blk}_{i_2}(\boldsymbol{T}^t)$, i.e.,

$$\mathrm{Blk}_{i_1}(\boldsymbol{T}^t) \neq \mathrm{Blk}_{i_2}(\boldsymbol{T}^t).$$

This contradicts the known condition $\mathrm{Blk}_{i_1}(\boldsymbol{T}^t) = \mathrm{Blk}_{i_2}(\boldsymbol{T}^t)$. The proof of sufficiency is completed.

Corollary 1 For the FSM described in Theorem 1, if states x_{i_1} and x_{i_2} are t-equivalent, they are (t+k)-equivalent (k≥1).

Proof Since states x_{i_1} and x_{i_2} are t-equivalent, it follows from Theorem 1 and Definition 4 that the following Eq. (16) holds for any $\omega \in S_t(x_{i_1} \to x_{j_1}) \cap S_t(x_{i_2} \to x_{j_2})$:

$$\mathrm{Blk}_{i_1}(\boldsymbol{T}^t) = \mathrm{Blk}_{i_2}(\boldsymbol{T}^t). \tag{16}$$

By Proposition 1, we know that Eq. (16) is equivalent to

$$\boldsymbol{T}^t x_{i_1} = \boldsymbol{T}^t x_{i_2}. \tag{17}$$

For any input symbol $e \in E$, we consider the string $\omega\_e \in S_{t+1}(x_{i_1} \to x_{j_1}) \cap S_{t+1}(x_{i_2} \to x_{j_2})$.

Multiplying both sides of Eq. (17) by T on the left and by ω_e on the right (for convenience, in the following we use ω_e to denote the vector form of the string ω_e), we obtain

$$\boldsymbol{T}\boldsymbol{T}^t x_{i_1}\omega\_e=\boldsymbol{T}\boldsymbol{T}^t x_{i_2}\omega\_e, \tag{18}$$

i.e.,

$$\boldsymbol{T}^{t+1} x_{i_1}\omega\_e=\boldsymbol{T}^{t+1} x_{i_2}\omega\_e. \tag{19}$$

According to Eq. (7), Eq. (19) can be rewritten as

$$f(x_{i_1},\omega\_e)=f(x_{i_2},\omega\_e).$$

Since ω_e is arbitrary, it follows from Definition 4 that x_{i_1} and x_{i_2} are (t+1)-equivalent. Setting t=t+1, we have that x_{i_1} and x_{i_2} are (t+2)-equivalent. Following this procedure, the proof can be completed.

Definition 5 (t-source equivalent states) Let x_{i_1} and x_{i_2} be two states of an FSM. x_{i_1} and x_{i_2} are t-source equivalent states if x_{i_1} and x_{i_2} are t-equivalent, but not (t−1)-equivalent (t≥2). If x_{i_1} and x_{i_2} are 1-equivalent, they are called 1-source equivalent states.

Theorem 2 (Necessary condition of (t+1)-equivalent states) Let M=(X, E, f, x_0, X_F) be an FSM, where X={x_1, x_2, ..., x_n} and E={e_1, e_2, …, e_m}. If there are two (t+1)-equivalent states x_{i_1} and x_{i_2}, there exist at least two t-equivalent states.

Proof Since x_{i_1} and x_{i_2} are (t+1)-equivalent, by Definition 4, the following Eq. (20) holds for any $e\_\omega\in S_{t+1}(x_{i_1}\to x_{j_1})\cap S_{t+1}(x_{i_2}\to x_{j_2})$, where |$\omega$|=$t$:

$$f(x_{i_1},e\_\omega)=f(x_{i_2},e\_\omega)\,. \tag{20}$$

The expansion of Eq. (20) is

$$f(f(x_{i_1},e),\omega)=f(f(x_{i_2},e),\omega). \tag{21}$$

Suppose that

$$f(x_{i_1},e)=x_{k_1}, \tag{22}$$

$$f(x_{i_2},e)=x_{k_2}. \tag{23}$$

Setting Eqs. (22) and (23) into Eq. (21), we obtain

$$f(x_{k_1},\omega)=f(x_{k_2},\omega).$$

Because ω (|ω|=t) is arbitrary, it follows from Definition 4 that x_{k_1} and x_{k_2} are t-equivalent. This proves the theorem.

The following corollaries are useful consequences of Theorem 2 and will be used in the state optimization of FSMs presented in the following subsection:

Corollary 2 Let M=(X, E, f, x_0, X_F) be an FSM, where X={x_1, x_2, ..., x_n}. If M contains no t-equivalent states, there are no (t+1)-equivalent states.

Proof (Deduction to absurdity) Assume that x_{i_1} and x_{i_2} are (t+1)-equivalent states. According to Theorem 2, we know that there exist at least two t-equivalent states. This contradicts the known condition that M contains no t-equivalent states. Thus, Corollary 2 holds.

Based on Corollary 2 we have the following result:

Corollary 3 If an FSM contains no 1-equivalent states, then it is the simplest substitution of itself.

Remark 4 Theorem 1 and Corollaries 1–3 show that the dynamical matrix $\boldsymbol{T}^t$ (Eq. (7)) completely depicts the properties of the states of FSM: states are either t-equivalent or t-different. Corollaries 2 and 3 provide a theoretical basis for the termination condition of designing methods to reduce FSMs.

3.2 Methods

In this subsection we present several algebraic methods of finding t-equivalent states and t-source equivalent states and of reducing FSMs.

The following Method 1 of finding all the t-equivalent states of FSMs is designed based on the sufficiency and necessity of Theorem 1:

Method 1 (Finding t-equivalent states) Let $\boldsymbol{x}(t+1)=\boldsymbol{T}^t\boldsymbol{x}_i\boldsymbol{u}(t)$ be the state transition dynamic equation of FSM M=(X, E, f, x_0, X_F), where X={x_1, x_2, ..., x_n}. Taking the following procedure, we can obtain all the t-equivalent states of M:

Step 1: divide matrix $\boldsymbol{T}^t$ using the n-column equipartition, and denote the ith block by Blk$_i$($\boldsymbol{T}^t$), i=1, 2, …, n.

Step 2: check if there are identical blocks in {Blk$_i$($\boldsymbol{T}^t$)|i=1, 2, …, n}. If not, M contains no t-equivalent states and the method terminates. Otherwise, construct the set K={(i_1, i_2, …, i_s)|2≤s≤n, Blk$_i$($\boldsymbol{T}^t$) are identical, j=1, 2, …, s}.

Step 3: the set of t-equivalent states of M is

$$X_t=\left\{(x_{i_1},x_{i_2},\cdots,x_{i_s})\,|\,(i_1,i_2,\cdots,i_s)\in K\right\}.$$

The following Method 2 is to find the t-source equivalent states of FSMs:

Method 2 (Finding t-source equivalent states) Let $\boldsymbol{x}(t+1)=\boldsymbol{T}^t\boldsymbol{x}_i\boldsymbol{u}(t)$ be the state transition dynamic equation of FSM $M=(X, E, f, x_0, X_F)$, where $X=\{x_1, x_2, ..., x_n\}$. The following steps yield the set of t-source equivalent states of M:

Step 1: use Method 1 to obtain the set of t-equivalent states of M, X_t.

Step 2: use Method 1 to obtain the set of $(t-1)$-equivalent states of M, X_{t-1}.

Step 3: obtain the set of t-source equivalent states of M, $X_{\mathrm{ot}}=X_t-X_{t-1}$.

Next we consider how to reduce the state space of FSM; we start with a preparation of regulating an operation of merging states into one state.

Merging of states of FSMs: let X be the state set of an FSM M, where $X=\{x_1, x_2, ..., x_n\}$. Let $X'=\{x_{i_1}, x_{i_2}, \cdots, x_{i_s}\}$ be a subset of X. The merging of states refers to the process of merging states in X' into one and moving the related "edges" accordingly in the state transition diagram of M. We use an example, as shown in Fig. 1, to illustrate it, where $X=\{x_1, x_2, x_3, x_4, x_5, x_6\}$, $X'=\{x_2, x_5\}$.

Method 3 (State minimization of FSMs) Let M be an FSM, where the state set is $X=\{x_1, x_2, ..., x_n\}$. Take the following steps and we can obtain the simplest substitution of M:

Step 1: set $t=1$ and Box=$\varnothing$.

Step 2: use Method 2 to obtain the set of t-source equivalent states of M, denoted by X_{ot}. If there are no t-source equivalent states, let $X_{\mathrm{ot}}=\varnothing$.

Step 3: set Box=Box$\cup X_{\mathrm{ot}}$.

Step 4: check whether X_{ot} is empty or not. If yes, go to step 5; otherwise, set $t=t+1$, and go to step 2.

Step 5: merge the states in each box$\in$Box. The resulting FSM is the simplest substitution of M.

Remark 5 The dimension of $\boldsymbol{T}^t$ is $n\times m^t n$, which shows that the size of $\boldsymbol{T}^t$ increases exponentially with m and t. Thus, $\boldsymbol{T}^t$ is very large when m or t is large. This causes a heavy burden on the memory of computers during the use of Method 3 to reduce FSMs (Note that $\boldsymbol{T}^t$ appears in Method 1, Method 1 is used in Method 2, and Method 2 is used in Method 3). To avoid this situation, Method 3 can be further improved.

Method 4 (Improvement of Method 3)

Step 1: use Method 1 to obtain the set of 1-equivalent states, X_1. If X_1 does not exist, M is a simplest FSM and the method terminates; otherwise, go to step 2.

Step 2: merge the states in X_1. Denote the resulting FSM by M_{new}.

Step 3: set $M=M_{\mathrm{new}}$ and go to step 1.

Remark 6 In Method 4, the iterative merging of the 1-equivalent states is used to avoid the appearance of $\boldsymbol{T}^t$ and prevent the heavy burden on computer memory.

3.3 Complexity analysis

In this subsection we analyze the computational complexity of the presented minimization method (Method 4) from the two aspects of time complexity and space complexity. Method 4 is an improved version of Method 3; Method 3 is conducted based on Methods 1 and 2. Therefore, we need only to discuss the complexities of Methods 1–3. Consider an FSM $M=(X, E, f, x_0, X_F)$, where $X=\{x_1, x_2, ..., x_n\}$ and $E=\{e_1, e_2, \ldots, e_m\}$.

Time complexity: Methods 1 and 2 have no circulation iteration, and thus the time complexity is $O(1)$. In Method 3, if M itself is the simplest FSM, the method comes to an end without iteration. Suppose M is not the simplest one. The number of t-source equivalent states is $n-1$ in the worst situation, and for this reason the case of $X_{\mathrm{ot}}=\varnothing$ occurs, thus causing Method 3 to iterate $n-1$ times. Therefore, the method iterates $n-1$ times at most. The time complexity is $O(n)$.

For Method 4, since the number of states of an FMS is at least one, the number of iterations of the method is no larger than $n-1$. Therefore, the time complexity is of polynomial time $O(n)$.

Space complexity: in Method 1, matrix $\boldsymbol{T}^t$ of size $n\times m^t n$ and matrices $\mathrm{Blk}_i(\boldsymbol{T}^t)$, $i=1, 2, \ldots, n$, need to be stored in step 1. The sum of the sizes of these n matrices is $n\times m^t n$. Thus, step 1 requires a total of $2n^2m^t$ storage units. The set K needs to be stored in step 2; the number of elements of K is the combinatorial number $\mathrm{C}_n^2=n(n-1)$ in the worst case. The set X_t in step 3 has also $n(n-1)$ elements at most. Therefore, the space complexity of Method 1 is $O(n^2m^t)$. Method 2 uses Method 1 to find the t-source equivalent states of M, and no additional storage is required. The space complexity of Method 2 is also $O(n^2m^t)$. The storage needed by Method 3 lies in step 2, in which Method 2 is applied to obtain the t-source equivalent states of M. In the worst situation, step 2 is iterated n times, and thus the space complexity is $nO(n^2m^t)=O(n^3m^t)$.

For Method 4, only step 1 needs memory. Step 1 is to find the 1-source equivalent states of M using Method 1. Therefore, the space complexity of Method 3 can be obtained by setting t=1 in the space complexity of Method 1, $O(n^2m^t)$, and the result is of polynomial time $O(n^2m)$.

4 Discussions (an open question)

Existing results of FSM minimization can roughly be classified into four categories according to research methods (Martinek, 2018): merging of states, refinement of the state set, trim of FSMs, and dynamic minimization. Most of them are computer-based algorithms, which focus on the minimization method and stress the computation gain. They are efficient for engineering problems; however, they fail to describe the minimization in a mathematical manner.

Compared with most previous work, although our work is as simple and efficient as them, more importantly, they differ in concept, purpose, approach, and the focus on both the theoretical and algorithmic results. The existing work frequently develops new algorithms to gain computational improvement; of course, most are supported by theoretical results. However, they are not good at explaining the mathematical meaning of FSM minimization. We reconsider the problem from the standpoint of control theory, where FSMs are treated as logical dynamic systems, aiming to reveal the mathematical meaning of the minimization.

The advantage of this study is the mathematical formulation and mathematical methods for simplifying FSMs. These explain the mathematics of the simplification problem.

Method 4 can be further improved. Let $\boldsymbol{T}_{\text{new}}$ and $\boldsymbol{T}$ be the dynamical matrices of the new FSM M_{new} and the original FSM M in Method 4, respectively. Denote by $\boldsymbol{T}_{\text{new}}=g(\boldsymbol{T})$ the relation between $\boldsymbol{T}_{\text{new}}$ and $\boldsymbol{T}$. If $\boldsymbol{T}_{\text{new}}=g(\boldsymbol{T})$ is known, then $\boldsymbol{T}_{\text{new}}$ can be obtained directly from $\boldsymbol{T}$ without replacing M_{new} by M in Method 4. Consequently, Method 4 can be optimized to run offline.

Method 5 (Unattended version of Method 4)

Step 1: split matrix $\boldsymbol{T}$ using n-column equipartition and denote by $\text{Blk}_i(\boldsymbol{T})$ the ith block, i=1, 2, …, n.

Step 2: check whether there are identical blocks in $\{\text{Blk}_i(\boldsymbol{T})|i=1, 2, \dots, n\}$. If not, M is a simplest FSM and stops. Otherwise, set $X_1=\{(x_{i_1},x_{i_2},\cdots,x_{i_s})\mid 2\le s\le n,\ \text{Blk}_{i_j}(\boldsymbol{T})\text{ are identical}, j=1,2,\cdots,s\}$.

Step 3: merge the states in X_1 and denote by M_{new} the resulting FSM.

Step 4: set $\boldsymbol{T}=\boldsymbol{T}_{\text{new}}=g(\boldsymbol{T})$, and go to step 1.

An open question: does the relation $\boldsymbol{T}_{\text{new}}=g(\boldsymbol{T})$ between $\boldsymbol{T}_{\text{new}}$ and $\boldsymbol{T}$ exist? If yes, what is it? This problem is under our study.

5 Correctness and effectiveness analysis

In this section we use an example in Chen (2007) to verify the correctness and effectiveness of the methods presented in this paper. Consider the FSM $M=(X, E, f, x_0, X_F)$, as shown in Fig. 2, which recognizes the language over alphabet {0, 1} that contains no "00" as its substring, where $X=\{x_0, x_1, x_2, x_3, x_4, x_5\}$, $E=\{0, 1\}$, $x_0=x_1$, and $X_F=\{x_1, x_2, x_4, x_5\}$.

Note that the main results of this paper have the relations shown in Fig. 3, which suggests that verifying the effectiveness and correctness of Method 4 will suffice.

Step 1: run step 1 of Method 4. Use Method 1 to obtain the set of 1-equivalent states X_1. The state transition dynamic equation is $\boldsymbol{x}(t+1)=\boldsymbol{T}^t\boldsymbol{x}_1\boldsymbol{u}(t)$, where the state set $X=\{x_1, x_2, \dots, x_5\}$ and $\boldsymbol{T}=\delta_5[2, 5, 3, 5, 3, 3, 3, 5, 4, 5]$.

Step 1.1: run step 1 of Method 1. Divide $\boldsymbol{T}$ into 5-column equipartition:

$$\begin{aligned}&\text{Blk}_1(\boldsymbol{T})=\delta_5[2,5],\ \text{Blk}_2(\boldsymbol{T})=\delta_5[3,5],\\&\text{Blk}_3(\boldsymbol{T})=\delta_5[3,3],\ \text{Blk}_4(\boldsymbol{T})=\delta_5[3,5],\\&\text{Blk}_5(\boldsymbol{T})=\delta_5[4,5].\end{aligned}$$

Step 1.2: run step 2 of Method 1. In $\{\text{Blk}_i(\boldsymbol{T})|i=1, 2, \dots, 5\}$, $\text{Blk}_2(\boldsymbol{T})=\text{Blk}_4(\boldsymbol{T})$. We have $K=\{(2, 4)\}$.

Step 1.3: run step 3 of Method 1. The set of 1-equivalent states of M is $X_1=\{(x_2, x_4)\}$.

Step 2: run step 2 of Method 4. Merge the states x_2 and x_4. The resulting FSM M_{new} is shown in Fig. 4.

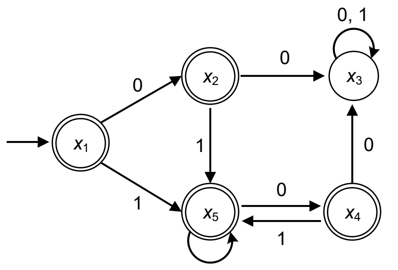

Fig. 2 Finite state machine to be reduced

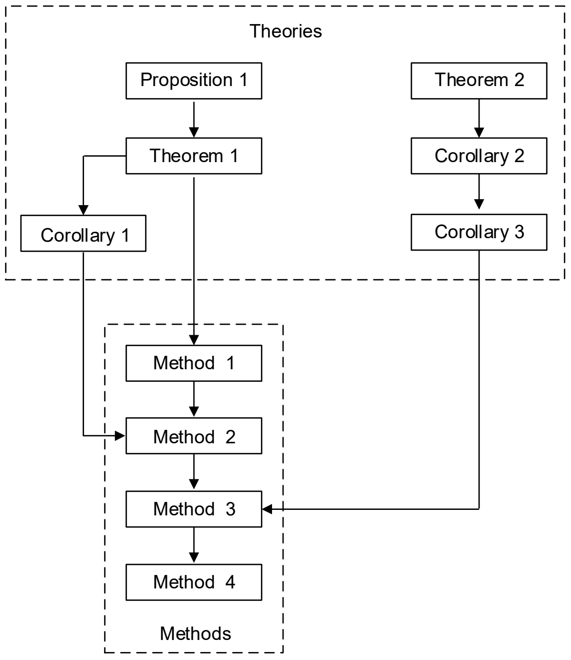

Fig. 3 Logical relations among the main results of this paper

Step 3: run step 3 of Method 4. Set M=M_{new} and go to step 1 of Method 4.

Step 4: run step 1 of Method 4. Use Method 1 to obtain the set of 1-equivalent states of M, X_1. Now the state transition dynamic equation is $\boldsymbol{x}(t+1)=\boldsymbol{T}\ltimes\boldsymbol{x}_1\boldsymbol{u}(t)$, where $X=\{x_1, x_2, x_3, x_4\}$ and $\boldsymbol{T}=\delta_4[2, 4, 3, 4, 3, 3, 2, 4]$.

Step 4.1: run step 1 of Method 1. Divide $\boldsymbol{T}$ into 4-column equipartition:

$$\begin{aligned}&\mathrm{Blk}_1(\boldsymbol{T})=\delta_4[2,4],\ \mathrm{Blk}_2(\boldsymbol{T})=\delta_4[3,4],\\ &\mathrm{Blk}_3(\boldsymbol{T})=\delta_4[3,3],\ \mathrm{Blk}_4(\boldsymbol{T})=\delta_4[2,4].\end{aligned}$$

Step 4.2: run step 2 of Method 1. In $\{\mathrm{Blk}_i(\boldsymbol{T})|i=1, 2, 3, 4\}$, $\mathrm{Blk}_1(\boldsymbol{T})=\mathrm{Blk}_4(\boldsymbol{T})$. We have $K=\{(1, 4)\}$.

Step 4.3: run step 3 of Method 1. The set of 1-equivalent states of M is $X_1=\{(x_1, x_4)\}$.

Step 5: run step 2 of Method 4. Merge the states x_1 and x_4. The resulting FSM M_{new} is as shown in Fig. 5.

Step 6: run step 3 of Method 4. Set M=M_{new} and go to step 1 of Method 4.

Step 7: run step 1 of Method 4. Use Method 1 to obtain the set of 1-equivalent states of M, X_1. Now the state transition dynamic equation of M is $\boldsymbol{x}(t+1)=\boldsymbol{T}\ltimes\boldsymbol{x}_1\boldsymbol{u}(t)$, where $X=\{x_1, x_2, x_3\}$ and $\boldsymbol{T}=\delta_3[2, 1, 3, 1, 3, 3]$.

Step 7.1: run step 1 of Method 1. Divide $\boldsymbol{T}$ into 3-column equipartition:

$$\begin{aligned}&\mathrm{Blk}_1(\boldsymbol{T})=\delta_3[2,1],\ \mathrm{Blk}_2(\boldsymbol{T})=\delta_3[3,1],\\ &\mathrm{Blk}_3(\boldsymbol{T})=\delta_3[3,3].\end{aligned}$$

Step 7.2: run step 2 of Method 1. There are no identical blocks in $\{\mathrm{Blk}_i(\boldsymbol{T})|i=1, 2, 3\}$. Then M has no 1-equivalent states. Method 1 is terminated. X_1 does not exist, and M is a simplest FSM. Method 4 stops.

It is easy to verify that the reduced FSM (Fig. 5) has the same functions as the original FSM (Fig. 2). Both recognize the language on alphabet {0, 1} in which there is no substring "00." This shows the correctness and effectiveness of the results of this paper.

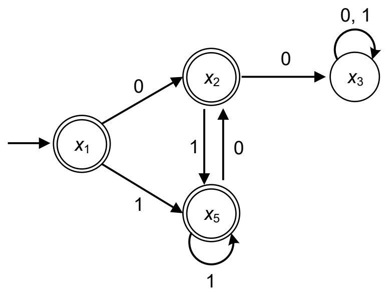

Fig. 4 Finite state machine produced by merging 1-equivalent states x_2 and x_4

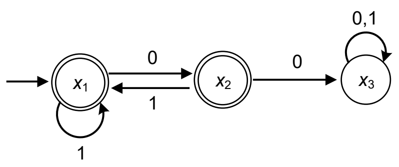

Fig. 5 Finite state machine produced by merging 2-equivalent states x_1 and x_4

6 Conclusions

FSMs can be viewed as logical systems. In this paper, we use the concepts, ideas, and methods of system science to reconsider the problem of state space optimization of FSMs. Two general kinds of equivalent states of FSMs, *t*-equivalent states and *t*-source equivalent states, are proposed by following the notion of the equilibrium points of dynamic systems in control theory. Further we explore some mathematics of these two kinds of equivalent states, including properties and a necessary and sufficient condition of *t*-equivalent states, and the relation between the *t*-equivalent and (*t*+1)-equivalent states. Based on these results, several algebraic methods are developed to find all the *t*-equivalent states and to reduce FSMs. The idea and methods proposed in this paper may provide a new angle and means to analyze and synthesize FSMs mathematically.

Contributors

Jumei YUE raised the research questions and the ideas to solve them. Yongyi YAN and Zengqiang CHEN guided Jumei YUE and He DENG to carry out the research in the whole process. Jumei YUE drafted the manuscript. He DENG helped organize the manuscript. Yongyi YAN and Zengqiang CHEN revised and finalized the paper.

Acknowledgements

The authors would like to express their sincere thanks for the advice given by Assistant Prof. Xiangru XU and Prof. Yiguang HONG.

Compliance with ethics guidelines

Jumei YUE, Yongyi YAN, Zengqiang CHEN, and He DENG declare that they have no conflict of interest.

References

Chen WY, 2007. The Theory of Finite Automata. University of Electronic Science and Technology of China Press, Chengdu, China (in Chinese).

Cheng DZ, Qi HS, Zhao Y, 2012. An Introduction to Semi-tensor Product of Matrices and Its Applications. World Scientific, Singapore.

Chu D, Spinney RE, 2018. A thermodynamically consistent model of finite-state machines. *Interf Focus*, 8(6): 20180037. https://doi.org/10.1098/rsfs.2018.0037

Dahmoune M, El Abdalaoui EH, Ziadi D, 2014. On the transition reduction problem for finite automata. *Fundam Inform*, 132(1):79-94. https://doi.org/10.3233/fi-2014-1033

de Parga MV, García P, López D, 2013. A polynomial double reversal minimization algorithm for deterministic finite automata. *Theor Comput Sci*, 487:17-22. https://doi.org/10.1016/j.tcs.2013.03.005

Gao N, Han XG, Chen ZQ, et al., 2017. A novel matrix approach to observability analysis of finite automata. *Int J Syst Sci*, 48(16):3558-3568.

García P, López D, de Parga MV, 2014. Efficient deterministic finite automata split-minimization derived from Brzozowski's algorithm. *Int J Found Comput Sci*, 25(6): 679-696. https://doi.org/10.1142/s0129054114500282

Gören S, Ferguson FJ, 2002. Chesmin: a heuristic for state reduction in incompletely specified finite state machines. Proc Design, Automation and Test in Europe Conf and Exhibition, p.248-254. https://doi.org/10.1109/DATE.2002.998280.

Gören S, Ferguson FJ, 2007. On state reduction of incompletely specified finite state machines. *Comput Electr Eng*, 33(1):58-69. https://doi.org/10.1016/j.compeleceng.2006.06.001

Grzes TN, Solov'ev VV, 2015. Minimization of power consumption of finite state machines by splitting their internal states. *J Comput Syst Sci Int*, 54(3):367-374. https://doi.org/10.1134/s1064230715030090

Han XG, Chen ZQ, 2018. A matrix-based approach to verifying stability and synthesizing optimal stabilizing controllers for finite-state automata. *J Franklin Inst*, 355(17):8642-8663. https://doi.org/10.1016/j.jfranklin.2018.09.009

Han XG, Chen ZQ, Liu ZX, et al., 2018. The detection and stabilisation of limit cycle for deterministic finite automata. *Int J Contr*, 91(4):874-886. https://doi.org/10.1080/00207179.2017.1295319

He YS, Yu ZH, Jian L, et al., 2019. Fault correction of algorithm implementation for intelligentized robotic multipass welding process based on finite state machines. *Robot Comput-Int Manuf*, 59:28-35. https://doi.org/10.1016/j.rcim.2019.03.002

Kamble SD, Thakur NV, Bajaj PR, 2018. Fractal coding based video compression using weighted finite automata. *Int J Amb Comput Intell*, 9(1):115-133. https://doi.org/10.4018/IJACI.2018010107

Klimowicz AS, Solov'ev VV, 2013. Minimization of incompletely specified Mealy finite-state machines by merging two internal states. *J Comput Syst Sci Int*, 52(3): 400-409. https://doi.org/10.1134/s106423071303009x

Li YM, Pedrycz W, 2007. Minimization of lattice finite automata and its application to the decomposition of lattice languages. *Fuzzy Sets Syst*, 158(13):1423-1436. https://doi.org/10.1016/j.fss.2007.03.003

Liu DS, Huang ZP, Zhang YM, et al., 2016. Efficient deterministic finite automata minimization based on backward depth information. *PLoS ONE*, 11(11): e0165864. https://doi.org/10.1371/journal.pone.0165864

Lu JQ, Li HT, Liu Y, et al., 2017. Survey on semi-tensor product method with its applications in logical networks and other finite-valued systems. *IET Contr Theory Appl*, 11(13):2040-2047.

Martinek P, 2018. Some notes to minimization of multiset finite automata. *AIP Conf Proc*, 1978(1):470019.

https://doi.org/10.1063/1.5044089

Melnikov B, 2010. Once more on the edge-minimization of nondeterministic finite automata and the connected problems. *Fundam Inform*, 104(3):267-283. https://doi.org/10.3233/fi-2010-349

Perkowski MA, Jóźwiak L, Zhao W, 2001. Symbolic two-dimensional minimization of strongly unspecified finite state machines. *J Syst Archit*, 47(1):15-28. https://doi.org/10.1016/s1383-7621(00)00038-2

Solov'ev VV, 2010. Minimization of Moore finite automata by internal state gluing. *J Commun Technol Electron*, 55(5):584-592. https://doi.org/10.1134/s1064226910050153

Solov'ev VV, 2011. Minimization of Mealy finite state machines via internal state merging. *J Commun Technol Electron*, 56(2):207-213. https://doi.org/10.1134/s1064226911020136

Solov'ev VV, 2014. Complex minimization method for finite state machines implemented on programmable logic devices. *J Comput Syst Sci Int*, 53(2):186-194. https://doi.org/10.1134/s1064230714020154

Solov'ev VV, 2017. Minimization of Mealy finite-state machines by using the values of the output variables for state assignment. *J Comput Syst Sci Int*, 56(1):96-104. https://doi.org/10.1134/s1064230717010129

Voeten C, van Zaanen M, 2018. The influence of context on the learning of metrical stress systems using finite-state machines. *Comput Ling*, 44(2):329-348. https://doi.org/10.1162/COLI_a_00317

Wang YB, Li YM, 2018. Minimization of lattice multiset finite automata. *J Intell Fuzzy Syst*, 35(1):627-637. https://doi.org/10.3233/jifs-161382

Xu XR, Hong YG, 2012. Matrix expression and reachability analysis of finite automata. *J Contr Theory Appl*, 10(2): 210-215. https://doi.org/10.1007/s11768-012-1178-4

Yan YY, Chen ZQ, Liu ZX, 2015. Semi-tensor product approach to controllability and stabilizability of finite automata. *J Syst Eng Electron*, 26(1):134-141. https://doi.org/10.1109/jsee.2015.00018

Yan YY, Chen ZQ, Yue JM, et al., 2016. STP approach to model controlled automata with application to reachability analysis of DEDS. *Asian J Contr*, 18(6): 2027-2036. https://doi.org/10.1002/asjc.1294

Yan YY, Yue JM, Fu ZM, et al., 2019a. Algebraic criteria for finite automata understanding of regular language. *Front Comput Sci*, 13(5):1148-1150. https://doi.org/10.1007/s11704-019-6525-x

Yan YY, Yue JM, Chen ZQ, et al., 2019b. Algebraic expression and construction of control sets of graphs using semi-tensor product of matrices. *IEEE Access*, 7:113440-113451. https://doi.org/10.1109/access.2019.2935321

Yan YY, Yue JM, Chen ZQ, 2019c. Algebraic method of simplifying Boolean networks using semi-tensor product of matrices. *Asian J Contr*, 21(6):2569-2577. https://doi.org/10.1002/asjc.2125

Yue JM, Yan YY, 2019. Exponentiation representation of Boolean matrices in the framework of semi-tensor product of matrices. *IEEE Access*, 7:153819-153828. https://doi.org/10.1109/ACCESS.2019.2948357

Yue JM, Yan YY, Chen ZQ, et al., 2019a. Identification of predictors of Boolean networks from observed attractor states. *Math Method Appl Sci*, 42(11):3848-3864. https://doi.org/10.1002/mma.5616

Yue JM, Yan YY, Chen ZQ, 2019b. Language acceptability of finite automata based on theory of semi-tensor product of matrices. *Asian J Contr*, 21(6):2634-2643. https://doi.org/10.1002/asjc.2190

Yue JM, Yan YY, Chen ZQ, 2020a. Matrix approach to simplification of finite state machines using semi-tensor product of matrices. *Asian J Contr*, 22(5):2061-2070. https://doi.org/10.1002/asjc.2123

Yue JM, Yan YY, Chen ZQ, 2020b. Three matrix conditions for the reduction of finite automata based on the theory of semi-tensor product of matrices. *Sci China Inform Sci*, 63(2):129203. https://doi.org/10.1007/s11432-018-9739-9

Zhang KZ, Zhang LJ, 2016. Observability of Boolean control networks: a unified approach based on finite automata. *IEEE Trans Autom Contr*, 61(9):2733-2738. https://doi.org/10.1109/tac.2015.2501365

Zhang QL, Feng JE, Yan YY, 2020. Finite-time pinning stabilization of Markovian jump Boolean networks. *J Franklin Inst*, 357(11):7020-7036. https://doi.org/10.1016/j.jfranklin.2020.05.010

Zhang ZP, Chen ZQ, Han XG, et al., 2017. Static output feedback stabilization of deterministic finite automata. Proc 36th Chinese Control Conf, p.2421-2425. https://doi.org/10.23919/ChiCC.2017.8027721

Zhang ZP, Chen ZQ, Liu ZX, 2018a. Modeling and reachability of probabilistic finite automata based on semi-tensor product of matrices. *Sci China Inform Sci*, 61(12):129202. https://doi.org/10.1007/s11432-018-9507-7

Zhang ZP, Chen ZQ, Han XG, et al., 2018b. On the static output feedback stabilization of deterministic finite automata based upon the approach of semi-tensor product of matrix. *Kybernetika*, 54(1):41-60. https://doi.org/10.14736/kyb-2018-1-0041