计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.200200073

关于同步部分规约的有限自动机的优化问题的近似难度

朱 凯[1,2] 毋国庆[1] 袁梦霆[1]
1 武汉大学计算机学院 武汉 430072
2 华南农业大学数学与信息学院 广州 510642
(zhukaics@aliyun.com)

摘 要 自动机是可同步的是指它具有满足以下性质的同步字:不论自动机当前所处的状态,以同步字为输入执行后它一定会到达某个特定状态。同步自动机问题的核心是计算最短同步字。聚焦于这一核心问题,文中就一类称为部分规约的确定的有限自动机的最短同步字问题,研究了近似计算这类自动机的最短同步字的复杂性,即近似计算它的难度,该工作有助于其近似算法的分析与设计。通过建立由两个优化问题(MAX SAT 问题以及 MAX FA-INT 问题)到最短同步字长度计算这一问题(即 Shortest-Syn)的归约,利用与概率可检验证明(Probabilistically Checkable Proofs,PCP)定理和概率可检验辩论(Probabilistically Checkable Debate,PCD)定理有关的若干结果证明了文中的主要结论:对于部分规约的确定的有限自动机,在某个近似因子内 Shortest-Syn 的近似难度是 NP-难的和 PSPACE-难的,除非 NP 和 PSPACE 分别坍塌到 P。

关键词:有限自动机;同步字;计算复杂性;近似难度;QSAT 问题
中图法分类号 TP301.5

On Hardness of Approximation for Optimized Problem of Synchronizing Partially Specified Deterministic Finite Automata

ZHU Kai[1,2], WU Guo-qing[1] and YUAN Meng-ting[1]
1 School of Computer Science, Wuhan University, Wuhan 430072, China
2 College of Mathematics and Informatics, South China Agricultural University, Guangzhou 510642, China

Abstract An automaton is synchronizing means it is with a word that has the following property: no matter which state the automaton is in, it will always reach a certain state after executing with a synchronizing word as input. For the problem of synchronization of automata, the core is to compute the shortest synchronizing word. Focusing on this core problem, this paper studies the complexity of approximate computing, i.e. the hardness of approximating, the shortest synchronizing word for a class of automata called partially specified deterministic finite automata, which is helpful for the analysis and design of its approximate algorithm. By two reductions from MAX SAT and MAX FA-INT, which are two optimizing problems, to the problem of computing the length of the shortest synchronizing word (i.e., Shortest-Syn), respectively, and by some existing results related to Probabilistically Checkable Proofs theorem and Probabilistically Checkable Debate theorem, the main results are proved. The results are as follows: for the partially specified deterministic finite automata, approximate computing the problem of Shortest-Syn is NP-hard and PSPACE-hard within some certain approximate ratios, unless NP and PSPACE collapse to P, respectively.

Keywords Finite automata, Synchronizing word, Computational complexity, Hardness of approximation, QSAT problem

1 前言

系统的可同步特性(synchronizability)可以通过执行某一输入序列,引导系统进入某个已知的确定的状态来获得。同步的要求常见于通信系统,在其他场合也渗透着同步的思想,比如,音乐会临近尾声时,往往由著名音乐家独奏华章(压轴),他将完全即兴演奏这个部分,指挥对他不会有任何干扰,管弦乐队需要等待指挥的信号重新开始。这时同步会按照音乐会的惯例而自动获得,这种隐含的惯例就可以看成同步字的使用[1]。对于计算系统,1964 年第一次出现“同步自动机”的概念,其由斯洛伐克计算机科学家Černý[2]提出,从此以后有关自动机同步的问题受到关注,尤其是以他的名字命名的迄今尚未解决的“Černý 猜想”。这个猜想本质上是计算最短同步字的最小上界。

早期有关同步自动机和Černý 猜想的研究可以参考文献

到稿日期:2020-01-15 返修日期:2020-03-15 本文已加入开放科学计划(OSID),请扫描上方二维码获取补充信息。
基金项目:国家自然科学基金(61640221,61872272)
This work was supported by the National Natural Science Foundation of China (61640221,61872272).
通信作者:袁梦霆(ymt@whu.edu.cn)

[3]。从文献[4-9]中可以看出，有关同步自动机的研究和应用跨越了多个学科，如软件工程（测试）、生物信息计算、控制论（机器人）、编码理论等。同步的重要性不言而喻，近年来对它的研究，一方面利用了非经典的计算复杂性理论知识，如参数复杂性理论、指数时间猜想（Exponential Time Hypothesis，ETH）和强指数时间猜想（Strong ETH，SETH）、概率可检验证明（PCP）定理及概率可检验辩论（PCD）定理等，较为深入地讨论了如参数复杂性及算法时间下界[10]、近似算法和不可近似性等理论问题[11-16]；另一方面将同步的自动机模型扩展到其他更复杂的自动机模型，如寄存器自动机[17]、马尔可夫决策过程[18]和时间自动机[19]等，研究它们的可计算性和计算复杂性。

对自动机的同步问题的理论研究主要集中在一类称为完全的确定的有限自动机上，这是一类较为特殊的自动机，在实际应用中并不常见。在自动机的实际应用场景中，部分规约的确定的有限自动机和非确定的有限自动机应用得更多。本文将关注部分规约的确定的有限自动机，由于判定这类自动机是否可同步以及它的最短同步字长是否为 $l(l\in\mathbb{N})$ 都是 PSPACE-完全的[20]，因此有理由相信这两个问题都无法在多项式时间内找到解，后一个问题还是搜索最小值的优化问题。如果进一步分析它的求解算法，大致上有两个选择[21]：1）考虑近似算法，将算法必须产生精确最优解的要求放宽，在考虑多项式时间算法的同时产生的解不必最优，但要求解的质量在最坏情况下满足一定的约束条件；2）利用参数复杂性理论，以更细的粒度分析算法的执行时间，此时不再以关于问题实例大小的函数来衡量运行时间，而是通过引入一个或多个参数的定义，来研究参数对算法执行时间的影响。

本文工作集中在第一种选择，即围绕部分规约的确定的有限自动机的同步问题及其最优问题，采用 PCP 定理的泛化——PCD 定理，证明了近似计算这类自动机的最短同步字的难度（不可近似性）的结论。本文得到的结论主要为：

1）对于部分规约的确定的有限自动机，存在常数 $\varepsilon>0$，使得在 n^{ε} 的比率内近似计算文中定义的 Shortest-Syn 问题是 PSPACE-难的。

2）对于部分规约的确定的有限自动机，对于每个常数 $\varepsilon>0$，使得在 $n^{1-\varepsilon}$ 的比率内近似计算文中定义的 Shortest-Syn 问题是 NP-难的。

本文第 2 节给出了证明上述结论需要用到的基础知识；第 3 节形式化定义了部分规约的有限制动机的同步问题和对应的优化问题，并给出了从多个有限自动机的交问题到部分规约的有限自动机的同步问题的多项式归约；第 4 节证明计算部分规约的有限自动机的最短同步字的不可近似性；第 5 节介绍相关工作；最后总结全文并展望下一步工作。

2　基础知识

计算复杂性理论中随机算法和近似复杂性的基础知识请参考文献[22-23]。

2.1　部分规约的自动机和同步字

定义 1（有限自动机）　（确定的）有限自动机（Deterministic Finite Automata，DFA）可定义为五元组 $A=(Q,\Sigma,\delta,q_0,F)$，其中 Q 是有限的非空状态集，Σ 是有限的非空字母表，变迁函数 $\delta:Q\times\Sigma\to Q$，$q_0\in Q$ 是初始状态，$F\subseteq Q$ 是最终状态集。如果 δ 是完全函数，即 $\forall q\in Q,\forall a\in\Sigma,|\delta(q,a)|=1$，那么称这个自动机是完全的有限自动机（Complete DFA），否则，由于存在变迁函数 $|\delta(q,a)|=0$，因此说明有变迁在状态 q 和字母 a 上没有定义，称它是部分规约的有限自动机（Partially Specified DFA）。实际上可以将后者看作前者的泛化。

记 $\delta(S,a):=\bigcup_{q\in S}\delta(q,a)$，其中子集 $S\subseteq Q$，字母 $a\in\Sigma$。可将 δ 扩展为 $\delta^*:Q\times\Sigma^*\to Q$，具体定义为：

$$\delta^*(q,w):=\begin{cases}q, & \text{如果 } w=\varepsilon\\ \delta^*(\delta(q,a),w'), & \text{如果 } w=a\cdot w'\end{cases}$$

其中，$w\in\Sigma^*$，ε 是空字，$a\in\Sigma$。对应的 $\delta^*(S,w):=\bigcup_{q\in S}\delta^*(q,w)$。为保持文献中符号的使用习惯，以下用 δ 代表 δ^* 和 δ，这在一定程度上滥用了符号 δ。

定义 2（同步字）　对于自动机 $A=(Q,\Sigma,\delta)$，如果存在 $w\in\Sigma^*$ 使得 $\forall q,q'\in Q,\delta(q,w)=\delta(q',w)$，其中 $\delta(q,w)\neq\perp$，即不能无定义，那么称 w 是自动机 A 的同步字（Synchronizing Word），有的文献也称之为重置字（Reset Word）。此时，称自动机 A 是可同步的（synchronizing or synchronizable）。注意，定义 2 中的自动机不同于定义 1，原因在于初始状态 q_0 和最终状态集 F 均与可同步性的定义无关。

图 1 给出了一个部分规约的自动机，不难验证 $w=bbababb$ 是它的最短同步字，同步后均进入状态 q_3。

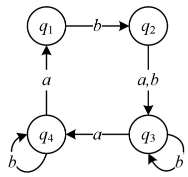

图 1　同步的部分规约的有限自动机示例

Fig. 1　Example of synchronizing PSDFA

2.2　近似算法基础

最优问题 Π 通常有如下结构：对于每个输入实例 x，问题 Π 会有多个可行解 y。Π 还有定义在可行解上的目标函数 $v_{\Pi}(y)$。最优问题往往是要搜索使 $v_{\Pi}(y)$ 达到最大或最小的可行解，称之为最优解。

设 $0<r<1$ 是实数，对于最小值问题 Π，可定义它的近似版本的问题 r-APPROX-Π 如下：给定输入 x，搜索可行解 y，使得 $v_{\Pi}(y)\cdot r\leqslant v_{\Pi}^*(x)$，其中 $v_{\Pi}^*(x)=\min\{v_{\Pi}(z)\mid z$ 是对输入 x 的可行解$\}$，即 Π 的最优解。一般来说，近似比 r 可以是常数，也可以是输入长度 n 的函数 $r(n)$，这个指标可用来约束可行解的质量。类似地，可定义最大值问题 Π 的近似版问题。

如果 NP 中的每个问题都可以多项式时间归约到函数 f，则称 f 是 NP-难的。如果 g 在比率 $\varepsilon(n)$ 之内近似 f，那么 g 是 NP-难的，则称在比率 $\varepsilon(n)$ 之内近似 f 是 NP-难的。类似地，可定义函数 f 是 PSPACE-难的和在比率 $\varepsilon(n)$ 内近似 f 是 PSPACE-难的。在比率 $\varepsilon(n)$ 之内近似 f 是 NP-难的或 PSPACE-难的，意思说，在 $\varepsilon(n)$ 的近似比率内不存在多项式时间的算法可以近似计算 f，除非 NP＝P 或 PSPACE＝P，即

除非 NP 或 PSPACE 坍塌到 P。

2.3 量化的布尔可满足问题及其变种

量化的布尔可满足问题 QSAT 是经典的 SAT 问题的泛化。设 Φ 是 QSAT 实例，不失一般性，假设它的量词是严格交错的，即形式满足：$\exists x_1 \forall x_2 \cdots \exists x_n \phi(x_1, x_2, \cdots, x_n)$。$\Phi$ 的赋值树 A 是深度为 n 的完全二叉树，其中从每个内部节点出发的一条边被标记为"0"，另一条边被标记为"1"。树上从根到叶子节点的每条路径 path 对应这些变量的一个赋值，如果这个赋值满足 ϕ，则称 path 满足 ϕ。在奇数层中的边称为 1-边，即对应存在量词量化的变量；在偶数层中的边称为 0-边，即对应全称量词量化的变量。$\exists$-赋值子树 A_1 是 A 的子树，它的每个位于偶数层的节点上有两条 0-边，每个位于奇数层的节点上只有一条 1-边。一棵 $\exists$-赋值子树是最优的，如果它最大化了满足公式 Φ 的路径条数。如果 Φ 为真，那么最优的 $\exists$-赋值子树的所有路径都满足 ϕ。对应地，$\forall$-赋值子树可类似说明。

Q3SAT 问题将 QSAT 中的子句的文字数量正好限定为 3 个。这时子句中最多有 3 个变量，是满足子句中变量个数不超过 $O(\log n)$ 的约束的，其中 n 是公式中布尔变量的个数。MAX Q3SAT 函数是 Q3SAT 的优化问题，其将量化的 3CNF 公式 Φ 映射到同时可满足的子句的最大个数。文献[24-25]指出，在确定的比率内近似 MAX Q3SAT 问题是 NP-难的。

2.4 概率可检验证明与概率可检验辩论

前言中提到，设计近似算法是求解难解问题的一种可行思路。对于研究某个优化问题在一定的近似比率内是否存在多项式时间的近似算法，目前的主要工具是概率可验证明 PCP，其有关资料请参考文献[22,26]。

概率可检验辩论系统(Probabilistically Checkable Debate Systems, PCDS)[27-28]是概率可检验系统的泛化。语言 L 的 PCDS 由一个概率多项式时间验证者 V 和两名玩家 0 和 1 之间的辩论组成，玩家 1 宣称输入 $x \in L$，而玩家 0 宣称 $x \notin L$。Condon 证明了 PCD 定理，指出语言 L 存在一个 PCDS，其中验证者 V 只需利用 $O(\log n)$ 的随机位查询辩论上的 $O(1)$ 位，当且仅当 $L \in$ PSPACE 即 $\mathrm{PCDS}_{1,\varepsilon}(O(\log n), O(1)) =$ PSPACE，称 ε 是可靠性参数或出错概率，其中 $0 < \varepsilon \leqslant 1/2$。这个定理说明了 PSPACE 复杂性类中的每个问题都具有上述的概率可检验的辩论系统，即这个定理重新表征了 PSPACE 类，也就指明了对于一些 PSPACE-难的函数，在一定的近似比率内近似计算与准确计算是一样难的。

MAX FIX-QSAT 函数是优化问题 MAX Q3SAT 的变种，它的域是量化的公式集，其中子句是 $O(\log n)$ 个子子句的合取，每个子子句是 3 个文字的析取，其中 n 是公式集的布尔变量的个数。此外，给定这个域上的某个实例，该函数输出子句集的最大个数 k，使得不论玩家 0 对全称量化的变量如何选择，玩家 1 都可以保证集合中这 k 个子句能同时得到满足。因此，对于 MAX FIX-QSAT 问题，k 个可满足的子句集必须预先固定，也就是说这个集合对于玩家 0 的所有赋值必须是相同的。Condon 等[28]指出近似 MAX FIX-QSAT 问题是 PSPACE-难的。

3 有关自动机同步的问题定义

为清楚地讨论，首先定义以下几个判定问题，其中问题 1 和问题 2 是基础，与其相关的若干结论可以作为证明的起点；问题 3 是计算同步字的优化问题。本文的目的是研究近似计算问题 3 的复杂性。

问题 1 多个有限自动机的交问题 FA-INT

输入：k 个有限自动机 $A_1=(Q_1,\Sigma,\delta_1,s_1,F_1),\cdots,A_k=(Q_k,\Sigma,\delta_k,s_k,F_k)$，其中 $k \geqslant 2$。

询问：是否存在被所有自动机接受的字 $w \in \Sigma^*$，也就是说，是否存在 w 使得对每个 $i=1,\cdots,k$ 均有 $\delta_i(s_i,w) \in F_i$？

问题 2 限界的多个有限自动机的交问题 bounded-FAINT

输入：k 个有限自动机 $A_1=(Q_1,\Sigma,\delta_1,s_1,F_1),\cdots,A_k=(Q_k,\Sigma,\delta_k,s_k,F_k)$和参数 $l \in \mathbb{N}$，其中 $k \geqslant 2$。

询问：是否存在被所有自动机接受的字 $w \in \Sigma^l$，也就是说，是否存在限长的 w 使得对每个 $i=1,\cdots,k$ 均有 $\delta_i(s_i,w) \in F_i$？

在问题 1 和问题 2 的基础上，可以定义这两个问题对应的优化问题 MAX FA-INT 函数，它的域是 k 个有限自动机 $A_1,\cdots,A_k$，将它们映射到满足存在一个字符串 w 被其中 k' 个自动机接受的最大值 $k'_{\max}$。

问题 3 最短同步字长判定问题 Shortest-Syn

输入：可同步的部分规约的有限自动机 $A=(Q,\Sigma,\delta)$ 和 l，其中 $\Sigma=\{0,1,*,\#\}$，$l \in \mathbb{N}$。

询问：A 的最短同步字的长度是否为 l？

问题 2 是问题 1 的限界版，有的文献也称其为参数化版本，用于证明问题 1 的参数化复杂性和问题 3 的复杂性，如文献[20]，本文不做过多讨论。对于问题 3，证明的基础之一是从多个有限自动机的交问题到部分规约的确定的有限自动机的同步问题的归约。下面只针对问题 3 来讨论近似计算最短同步字的复杂性，结论的证明过程较复杂，为加强可读性，将其分成若干个步骤对一些关键点加以说明，整个过程尽量减少过度的形式化描述。

引理 1[20] 存在从 FA-INT 到部分规约的确定的有限自动机的同步问题的多项式时间归约。

为了方便主要结论的证明，这里仅给出引理 1 证明中需要用到的归约，与文献[20]中对应的归约相比，做了少量修改。

给定 FA-INT 的一个实例：$A_1=(Q_1,\Sigma',\delta_1,q_1,F_1),\cdots,A_k=(Q_k,\Sigma',\delta_k,q_k,F_k)$，其中 $k \geqslant 2$，可以构造一个部分规约的自动机 $B=(Q,\Sigma,\delta)$，它具有同步字当且仅当自动机 $A_1,...,A_k$ 均接受某个共同的字。设 $c \in \Sigma'$，$q \in Q_i$，$i \in \{1,\cdots,k\}$，B 的构造方案为：

$$Q:=\bigcup_{i=1}^{k} Q_i \cup \{q_{\text{init}}, q_{\text{end}}\};$$

$$\Sigma:=\Sigma' \cup \{*,\#\};$$

$$\delta(q,c):=\delta_i(q,c),\ \delta(q,*):=q_i$$

$$\delta(q,\#):=\begin{cases} q_{\text{end}}, & \text{如果 } q \in F_i \\ q_i, & \text{如果 } q \in Q_i \setminus F_i \end{cases}$$

$$\delta(q_{\text{end}},*):=q_{\text{end}}$$

$\delta(q_{\text{end}},\#):=q_{\text{end}}$

$\delta(q_{\text{end}},c):=q_{\text{end}}$

$\delta(q_{\text{init}},*):=q_1$

$\delta(q_{\text{init}},\#):=\perp$

$\delta(q_{\text{init}},c):=\perp$

其中，符号⊥表示此处无定义。

自动机 B 的大小是自动机 $A_1,\cdots,A_k$ 大小之和的线性关系。B 中 q_{end} 状态称为吸收状态，进入吸收状态后就无法再跳出这个状态。这个构造方案的示意图如图 2 所示，其中红色的状态是各个自动机 A_i 的最终状态。

以上初步建立了问题 1 与问题 3 的联系。如果以前者的复杂性结论为起点，在满足一定条件的归约(不一定是多项式归约)的情况下，可得到后者的复杂性下界，由于问题 1 在经典的复杂性理论中是 PSPACE-难的[29]，因此问题 3 也至少是 PSPACE-难的。例如，模型检测 CTL 和 CTL* 规约的公式，其计算复杂性是 PSPACE-难的，这个结论的证明是从问题 1 出发进行了归约[30]。

引理 2[20]　对于上述归约中构造的自动机 B，它的最短同步字具有 $u=*w\#$ 的形式，其中 w 属于多个自动机的交。

该引理给出了当 k 个自动机有共同的接受字 w 时，B 的最短同步字是 $*w\#$。但文献[20]忽略了以下情况：当 k 个自动机没有共同的接受字时，如果 B 仍是可同步的(假设 $A_1,\cdots,A_k$ 均为完全的确定的有限自动机)，那么不难看出 $*w_1\#*w_2\#\cdots*w_k\#$ 是 B 的同步字，其中 w_i 是自动机 A_i 的接受字。因此，本文将主要讨论当 k 个自动机没有共同的接受字，但其中最多有 k' 个自动机有共同的接受字时，它与前两种情况相比，同步字之间明显存在长度上的间隙，即问题的解的间隙。

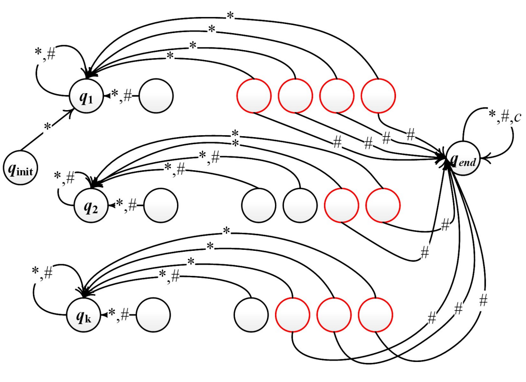

图 2　从 FA-INT 到部分规约的确定的有限自动机的同步问题的归约(电子版为彩色)

Fig. 2　Reduction from FA-INT to problem of synchronizing PSDFA

4　近似难度的讨论

本节讨论近似计算问题 3 的复杂性，即近似难度。

4.1　基于 PCD 定理的结论及证明

首先直接给出本文的主要结论。

定理 1　对于每个常数 $\varepsilon>0$，使得在 $2-\varepsilon$ 的比率内近似计算 Shortest-Syn 问题是 PSPACE－难的。。

定理 2　存在常数 $\varepsilon>0$，使得在 n^{ε} 的比率内近似计算 Shortest-Syn 问题是 PSPACE-难的。

定理 2 中的近似比是关于 n 个函数的，强于定理 1 中的常数近似比。

接着给出证明思路：选择合适问题作为归约起点，构造出具体最优解和近似解间隙的归约，讨论构造的性质，特别是与最优解和近似解的目标函数有关的不相等的数量关系，由此推理出结论。这两个定理的证明直接从 PCD 定理出发，较为复杂，因此利用文献[27-28]中已证明的有关 MAX FIX QSAT 和 MAX FA-INT 的不可近似性来证明。分析归约时，除了要保证是多项式的归约，还要注意归约前后的出错概率(或间隙)之间的数量关系。

4.1.1　从 MAX FIX-QSAT 归约到 MAX FA-INT

引理 3[27-28]　存在常数 $\varepsilon>0$，使得在 n^{ε} 的比率内近似计算 MAX FA-INT 是 PSPACE-难的。

这个定理的证明思路是：建立从 MAX FIX-QSAT 到 MAX FA-INF 的归约，使得玩家 1 可以确保 MAX FIX-QSAT 的实例的子句集中有 k 个子句可以同时得到满足，当且仅当 MAX FA-INT 的实例有 k 个自动机接受共同的字。

这里不进行引理 3 的证明，仅完整地给出从 MAX FIX-QSAT 到 MAX FA-INF 的归约，证明细节可参考文献[28]。设 $\Phi=\exists x_1\forall x_2\cdots\exists x_n\phi(x_1,x_2,\cdots,x_n)$ 是 MAX FIX-QSAT 的实例，其中 ϕ 有 m 个子句，每个子句是 $O(\log n)$ 个子子句的合取，每个子子句是 3 个文字的析取。不失一般性，假设不存在子句中未出现的变量，并假设 n 是偶数。

首先给出含 w 字的集合 $Valid$。该集合中的每个字(对应 QSAT 公式的每个解，即玩家 1 在辩论中的获胜策略)描述玩家 1 对存在量词量化的变量赋值的可能选择，应对玩家 0 对全称量词量化的变量赋值的所有可能选择。

为每个子句构造一组自动机，m 个子句对应 m 组自动机，使得 m 中有 k 组，每组中的所有自动机构成的新自动机都接受某个字 $w\in Valid$，当且仅当量化的公式 Φ 有 k 个同时可满足的子句。

每个 $w\in Valid$ 均有 $\$w_1\$w_2\$\cdots\$w_N\$$ 的形式(这里加入符号 \$ 作为子串之间的分隔符是为了方便说明，不是必需的)，其中 $N=2^u$，u 是 Φ 中全称量词量化的变量的个数，每个子串 $w_i=w_i[1]w_i[2]\cdots w_i[n]$ 是长为 n 的 01 串，与赋给 ϕ 的 n 个变量的一个真赋值对应(将 $w_i[j]$ 看作对 x_j 的赋值)。$w\in Valid$，当且仅当所有的 01 串 w_i 对应一棵∃-赋值子树的路径集，它们描述玩家 1 的获胜策略，应对玩家 0 的所有可能赋值。这棵赋值子树有 N 个叶子，可以从上到下、从左到右的顺序枚举这些路径。

编码 w 在语法形式上须满足以下 3 个性质：

1)如果变量 x_j 是由全称量词量化的，那么 $w_1[j]=0$ 且 $w_N[j]=1$。如果 w_i 使得对于所有 j，若 $w_i[j]=1$，则 x_j 是全称量化的，那么 $i=N$。

2)如果变量 x_j 是由存在量词量化的且 $i>1$，那么 $w_i[j]=w_{i-1}[j]$。如果对所有的下标 $j'>j$，使得若变量 $x_{j'}$ 是由全称量词量化的，则 $x_{i-1}[j']=1$，那么 $w_i[j]$可不受约束。

3)如果变量 x_j 是由全称量词量化的且 $i>1$，如果对所有 $j'>j$，使得若变量 $x_{j'}$ 是由全称量词量化的，则 $w_{i-1}[j']=1$，那么 $w_i[j]=\bar{w}_{i-1}[j]$(符号 $\bar{x}$ 表示对 x 的值取反，即 0 变为 1，

1 变为 0),否则 $w_i[j]=w_{i-1}[j]$。

如果在下标 j 处不满足性质 2)或性质 3),那么称 w 在下标 j 处不合法。

以上 3 个条件是 $w\in Valid$ 的充分必要条件。例如,图 3 是 QBF 的实例 $\exists x_1 \forall x_2 \exists x_3 \forall x_4(((x_1 \vee x_2 \vee x_3) \wedge (\neg x_2 \vee x_3 \vee x_4)) \wedge (x_1 \vee x_3 \vee \neg x_4))$的一棵 ∃-赋值子树,假设 $C_1=(x_1 \vee x_2 \vee x_3) \wedge (\neg x_2 \vee x_3 \vee x_4)$和 $C_2=x_1 \vee x_3 \vee \neg x_4$ 是公式的两个子句。这棵赋值子树的所有路径均使得公式为真,编码 $w=\$0010\$0011\$0110\$0111\$$ 对应树的 4 条路径。不难验证,$w=\$0010\$0011\$0110\$0111\$$ 满足这 3 个性质。

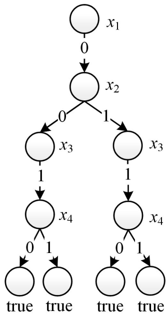

图 3 ∃-赋值子树的示例

Fig. 3 Example of ∃-assignment subtree

为完成这个归约需要两类自动机,分别是"串句法检查"自动机和"子句可满足性检查"自动机。本文将使用这些自动来机构造 MAX FA-INT 实例中的 k 个自动机。

串句法检查自动机有 $n+1$ 个。自动机 S_{n+1} 检查 w 是否可以被分隔符 \$ 分割成 N 个长度为 n 的子串 w_i 以及是否满足性质 1)。对于每个 $j(1\leqslant j\leqslant n)$,自动机 S_j 检查每个子串 w_i 的第 j 位的合法性,即检查性质 2)或性质 3)是否满足,这由对应的变量 x_j 是全称量词量化的还是存在量词量化的决定。明显地,串 w 被这 $n+1$ 个自动机接受,当且仅当它属于 $Valid$。这些自动机的状态个数是 $poly(n)$,以下约定 $poly(n)$是关于 n 的多项式函数,其中 n 是布尔变量的个数。

有 m 个子句可满足性检查自动机 $Cl_1,\cdots,Cl_m$,每个自动机有 $poly(n)$个状态,使得 $w\in Valid$ 被自动机 Cl_j 接受,当且仅当在对应的 ∃-赋值子树的所有的路径上,第 j 个子句得到满足。子句可满足性检查自动机的构造将在 4.1.2 节单独讨论。

现在构造 m 个自动机,第 i 个自动机 A_i 在它的输入串 w 上进行串句法检查和子句可满足性检查,具体过程如下。

1)采用自动机 S_{n+1} 执行检查。这个检查可以通过含有 $poly(n)$个状态的自动机完成。

2)如果 x_j 或 $\neg x_j$ 是子句 C_i 中的文字,那么执行自动机 S_j 检查(在这种情况下,称 A_i 检查变量 x_j 对应的位)。对于每个 i,自动机存储 $w_{i-1}[j]$和 $w_i[j]$这两个位,其中 x_j 对应的位被 S_j 检查。此外,保存 w_i 中的具有 0 值的最右边的全称量化的变量的下标位置。利用这个信息,在字符串 w_i 上执行 S_j 的检查。

3)通过自动机 Cl_i 执行检查。

由于自动机 A_i 是由完成以上 1)-3)步的检查所用到的各自动机的组合而成,这些自动机的大小均为 $poly(n)$,执行检查 1)有 1 个自动机,执行检查 2)最多有 n 个自动机,执行检查 3)最多有 $O(\log(n))$个自动机,每个自动机 Cl_i 的大小是 $poly(n)$的,因此 A_i 的大小仍是 $poly(n)$。实际上,在对 m 个 A_i 的检查中,对 S_j 和 S_{n+1}的检查存在多次,只需检查一次即可。

由以上分析得到:如果玩家 1 能够确保对玩家 0 的所有赋值,ϕ 的 k 个子句构成的固定子句集合是满足的,具体表现为枚举所有可能构成的 01 串,这个串中的子串都是 ϕ 中固定的 k 个子句的解,那么存在对应的 01 串 w 被 k 个自动机所接受。

可以假设 $n+1$ 个串的句法检查自动机始终接受 w,下面讨论每个子句可满足性检查自动机是否接受 w。

4.1.2 子句可满足性检查自动机的构造和性质

每个子句可满足性检查自动机 Cl_j 的构造,将利用 Gawrychowski 等[16]给出的从 3SAT 问题到完全的确定的有限自动机的同步问题的归约中的构造思想,以及第 3 节中多个有限自动机的交到同步字问题的归约。

对于 $\Phi=\exists x_1 \forall x_2 \cdots \exists x_n \phi(x_1,x_2,\cdots,x_n)$中的公式 ϕ,它有 m 个子句,每个子句由 $O(\log n)$个子子句的合取构成,每个子子句又由 3 个文字的析取构成。可以令 $\phi=C_1 \wedge C_2 \wedge \cdots \wedge C_m$,其中每个 C_i 有 $\bigwedge_{1\leqslant t\leqslant O(\log n)} C_i^t$ 的形式。

将为每个子子句 C_i^t构造一个自动机 $Cl_i^t=(\boldsymbol{Q},\{0,1,*,\#\},\delta)$,使得自动机 Cl_i 由不超过 $O(\log n)$个自动机 Cl_i^t 的交构成。由文献[16]的构造容易得到:自动机 Cl_i^t 的关键部分可以看成一棵有 8 个叶子的树,其中每个叶子对应子子句 C_i^t中出现的 3 个变量的一种赋值。对 n 个变量取所有可能的赋值有 2^n 种情况,这构成了高度为 n 的完全二叉树。树的每条从父亲节点到孩子的边用 0 或 1 标记,每种赋值对应树的一个叶子。本文在这里采用文献[16]中提出的通过合并同构子树得到大小为 n 的线性函数的自动机的方法。具体为:用 $L_0,L_1,\cdots,L_n$ 分别表示节点所在的第 0 层、第 1 层、…、第 n 层,于是 $L_0=\{r\}$,这里 r 是根,L_n 是叶子的集合。假设变量 x_k 没有出现在子子句 C_i^t 中。取任意顶点 $v\in L_{k-1}$,用 T_0 和 T_1分别表示以 v 的两个孩子为根的子树。容易看出,若 T_0 和 T_1 是同构的,则可合并,于是得到两条分别标记为 0 和 1 的从 v 出发的边,它们都到达顶点 v',也就是 T_0 的根 。继续合并直到没有这样的顶点。

该构造过程可形式化为:在第 j 层$(0\leqslant j\leqslant n)$的状态集 $L_j=\{q_j^w:w\in\{0,1\}^d\}$中,$d$ 是出现在子子句 C_i^t 中的变量 x_i 的下标满足 $i\leqslant t$ 的变量的个数,显然它不会超过 3。L_0 为 $\{q_0^\varepsilon\}$。给定 q_j^w,应该将 $w\in\{0,1\}^d$ 看成对出现在子子句 C_i^j 中的变量 $x_{j_1},x_{j_2},\cdots,x_{j_d}$ 的某个布尔赋值,其中 $j_1\leqslant j_2\leqslant\cdots\leqslant j_d\leqslant j$。

变迁函数的定义为:任取变量 x_k 和顶点 $q_{k-1}^w\in L_{k-1}$,那么:

1)如果 x_k 在子子句 C_i^t 中出现,那么 q_{k-1}^w 有两个孩子 $\delta(q_{k-1}^w,0)=q_k^{w0}$ 和 $\delta(q_{k-1}^w,1)=q_k^{w1}$;

2)如果 x_k 没有出现在子子句 C_i^t 中,那么 q_{k-1}^w 有一个孩子 $\delta(q_{k-1}^w,0)=\delta(q_{k-1}^w,1)=q_k^w$。

这就定义了由第 0 层、第 1 层、…、第 $n-1$ 层中的状态出发的变迁。接着定义第 n 层的状态出发的变迁。设 q_n^w 是最后这一层上的状态,参考图 2 中的 $*$-变迁和 #-变迁的定义可得,如果对应 C_i^j的可满足的赋值存在,那么 $\delta(q_n^w,\#)=$

q_{end}；否则 $\delta(q_n^w, *)=q_0^\varepsilon$。此外，定义初始状态（树的根）$r=\{q_0^\varepsilon\}$，对于每个 $q\in Cl_i^t$，$\delta(q,*)=r$，定义吸收状态 q_{end}。

因此，子句 $\bigwedge_{1\leqslant t\leqslant O(\log n)} C_i^t$ 对应的自动机是这些子子句自动机的组合。子子句个数是 $O(\log n)$，子子句自动机的状态个数又是 n 的线性函数，因此 $\bigwedge_{1\leqslant t\leqslant O(\log n)} C_i^t$ 对应的自动机的状态个数为 $O(n\log n)$，这显然是 $poly(n)$ 的。于是，这个构造是可以在多项式时间内完成的。由此得到 4.1.1 节所述的整个构造是多项式时间的。

根据图 3 的例子，考虑 $C_2=x_1\vee x_3\vee\neg x_4$ 所构造的自动机 Cl_2 的关键部分如图 4 所示，红色的叶子节点表示使得公式为真的赋值。

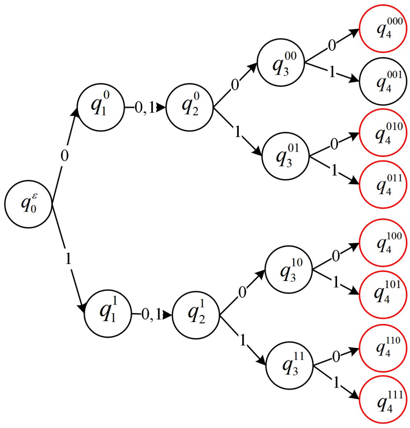

图 4　$C_2=x_1\vee x_3\vee\neg x_4$ 对应的自动机（电子版为彩色）

Fig. 4　Automaton corresponding to $C_2=x_1\vee x_3\vee\neg x_4$

下面讨论由这种构造得到的一些性质。

性质 1　每个子句 C_i 所对应的自动机 A_i 是由多个子自动机构成的，它们由自动机 S_{n+1}、不超过 n 个自动机 S_j 和不超过 $O(\log n)$ 个自动机 C_i^t 组成。一般地，如果每个 C_i 都是可满足的，存在长度为 $n\cdot 2^{n/2}$ 的 01 串 w 作为赋值策略，是子句 C_i 的一个解。

注意：若要保证长度是多项式的串 w，只要每个子句中出现的变量个数不超过 $O(\log n)$ 即可，而在 4.1.2 节的构造过程中每个子句的变量数为 3，这显然是满足的。

性质 2　对于公式 Φ，由于它是 m 个子句的合取，当 Φ 可满足时，存在长度为 $n\cdot 2^{n/2}$ 的 01 串 w 使得 w 属于这 m 个自动机的交，那么由引理 1 和引理 2 构造的自动机 B 是可同步的，具有最短的同步字长 $n\cdot 2^{n/2}+2$。

下文将讨论 Φ 不满足时，即子句可满足性检查自动机的交为空时，最多有 k 个自动机有共同的接受子 w 的情况。显然，存在串长度为 $n\cdot 2^{n/2}+2$ 的 01 串 w 可以将以上的 k 个自动机同步到吸收状态 q_{end}，其他的自动机执行 w 后，应该处于它们各自的初始状态。此时，若要同步这个自动机 B，就要同步这些剩余的自动机的初始状态，剩下的自动机有 $(1-r)m$ 个，其中 r 是出错概率，因为至少会剩一个自动机上的状态没有到达吸收状态 q_{end}，将这个自动机同步到吸收状态 q_{end}，还需要长度为 $n\cdot 2^{n/2}+1$ 的一个串。

4.1.3　定理 1 的证明

证明：由 4.1.1 节和 4.1.2 节的分析可以得到，当构造出的多个自动机具有共同的接受字时，同步自动机 B 需要长度为 $n\cdot 2^{n/2}+2$ 的最小同步字 $*w\#$，其中由每个长度为 $2^{n/2}$ 的 n 个子串所构成的 w，编码了对应的起始 QSAT 问题的一个解（即获胜策略）。当构造出来的多个自动机没有共同的接受字时，最多有 k 个自动机有共同的接受字，反过来至少有一个自动机 A' 与这 k 个自动机的交为空。此时，假设这 k 个有共同接受字的自动机共同的最短接受字是 w，那么串 $*w\#$ 可以将这 k 个自动机上的任何状态同步到吸收状态 q_{end}，同时在 A' 上执行 $*w\#$ 会令其回到初始状态，此时由它的初始状态进入吸收状态，需要执行一个与它对应的子句的可满足赋值策略对应的串 w'，$|w'|=n\cdot 2^{n/2}$，由于 B 是可同步的，因此它对应的子句是可满足的，于是 w' 存在。此时，$*w\#w'\#$ 串是最短的可以同步 B 的串。显然这个最短串的长度是 $2n\cdot 2^{n/2}+3$。这也就意味着，当构造出来的多个自动机没有共同的接受字时，同步自动机 B 的串长度至少是 $2n\cdot 2^{n/2}+3$。显然近似比满足 $\dfrac{2n\cdot 2^{n/2}+3}{n\cdot 2^{n/2}+2}=\dfrac{2(n\cdot 2^{n/2}+2)-1}{n\cdot 2^{n/2}+2}<2$。

于是得到：对于每个常数 $\varepsilon>0$，除非 PSPACE 坍塌到 P，在 $2-\varepsilon$ 的近似因子内计算 shortest-Syn 没有多项式时间算法。定理 1 得证。

4.1.4　定理 2 的证明

证明：定理 2 证明的关键在于说明验证的随机算法有亚常数的出错概率。Cohen 等[31] 给出以下结论：存在常数 $0<\varepsilon<1$，使得如果存在 RP(Randomized Polynomial time) 算法，它以 ε 的出错概率界限，采用 r 位随机位判定语言 L，那么存在一种 RP 算法，它以 2^{-k} 的出错概率界限，采用 $m=r+O(k)$ 的随机位判定语言 L。

这个定理说明，采用随机抽样，只需增加常数因子的随机位的数量，就可以将任何 RP 算法的出错概率减小到问题实例大小的指数小。Condon 等[28] 采用该方法证明了引理 3 的近似因子的阈值为 n^ε。

如果取 $k=\log n$，则出错概率的界限是无限接近 n^{-1} 的，采用的随机位为 $r+O(\log n)$。

PCD 定理中的检验算法只允许单面错误，属于 RP。采用 Cohen 等的上述结论可得到 PSPACE$\subseteq$PCDS$_{1,1/n}$$(O(\log n),O(\log n))$，这里的 n 是证明串的长度不再是 QSAT 中公式的布尔变量个数。结合定理 1 的证明过程，可以得到最短同步字的阈值上界是不会超过 $1/n^{-1}$ 的。

由此可得，存在常数 $\varepsilon>0$，使得在 n^ε 的比率内近似计算 Shortest-Syn 问题是 PSPACE-难的。

4.2　基于 PCP 定理的结论

引理 4　Shortest-Syn 是 NP-难的。

证明：它的 NP-难的下界由 bounded-FA-INT 问题是 NP-难的[32] 和第 3 节给出的引理 1 可得。

于是，计算最短同步字的长度是 NP-难的函数。可以采用 PCP 定理处理 NP-难的计算函数的近似难度，得到的结论和证明方法同文献[16]，在构造过程中只需稍微修改，归约到部分规约的有限自动机即可。由此可以直接得到以下结论。

定理 3　对于每个常数 $\varepsilon>0$，使得在 $2-\varepsilon$ 的比率内近似计算 Shortest-Syn 问题是 NP-难的。

定理 4　存在常数 $\varepsilon>0$，使得在 n^ε 的比率内近似计算 Shortest-Syn 问题是 NP-难的。

定理 5　对于每个常数 $\varepsilon>0$，使得在 $n^{1-\varepsilon}$ 的比率内近似

计算 *Shortest-Syn* 问题是 NP-难的。

注意:1)文献[16]给出的3个结论来自不同形式的 PCP 定理,近似因子与 PCP 定理中的出错概率的上界直接相关,其中后两个结论为了获得更强的出错概率,分别采用了膨胀图上的随机游走和自由位复杂性来放大间隙。

2)在证明对应部分规约的有限自动机也具有定理3—定理5时,可以采用从 MAX SAT 出发的归约,对每个子句 C_i 仍然可以构造一个自动机,对应到图2中的 A_i,其不再按文献[16]的方法构造,而是参考图2的方式建立多个自动机的交问题到部分规约的自动机的同步问题。

3)定理1和定理2对应的最短同步字长是状态个数的指数函数,而定理3—定理5对应的最短同步字长是状态个数的多项式函数,前两个定理和后3个定理分别从空间复杂性是 PSPACE-难的和时间复杂性是 NP-难的两个方面给出了它的难解性,对应地,本文采用了 PCD 定理和 PCP 定理的结论来处理近似的难度的证明过程。

5 相关工作

本节讨论文献[3]发表以后的工作,时间以2008年为界。近似难度(不可近似性)的研究工作主要集中在完全的有限自动机上,这其实是特殊的部分规约的有限自动机,是一类最简单的自动机。在证明了完全的确定的有限自动机的最短同步字问题是 DP-完全的之后,研究者最先想到的是寻找质量好的多项式时间的近似算法,后来逐渐意识到在一定的近似比内,近似计算最优解的多项式时间算法是不存在的。在这个认识过程中,计算复杂性理论从经典的 NP 完全理论发展到一些新的理论,特别是 PCP 定理及其应用,近似难度的研究工作对近似算法难度的分析起到了积极的作用。

最早的对近似算法的研究是 Berlinkov[11] 提到的 Gawrychowski 的求解近似比为2的同步字是 NP-难的。Berlinkov[11] 也得到了相同的结论,在其文章中将近似比定义为出错比。接着,Gerbush 等[12] 研究了自动机的最短同步序列问题,他们给出了一个近似因子为 $\lceil(n-1)/(k-1)\rceil$ 的贪心算法,它的时间算法度为 $O(kmn^k+n^4/k)$,其中 k 是算法的参数,代表在每一轮迭代中同步的状态个数,n 和 m 分别是状态数和字母表的大小。近年,Ananichev 等[13] 在文献[12]的基础上,研究了可同步的自动机的同步阈值(即最短同步字的最小长度)的近似计算问题,主要结论为:对一类形式化为 SYNCH-SUBSET(k)的贪心算法,当 $k<n/3$ 时,它的近似比率至少可达 $\frac{n}{6(k-1)}$。

对于近似难度的研究,Berlinkov[14] 指出,除非 NP 坍塌到 P,在常数因子内近似计算给定的可同步的自动机的最短同步字的长度不存在多项式时间的算法。Berlinkov[15] 进一步指出,对于字母表大小为2的 n 个状态的完全的确定的自动机,在 $c\ln n$ 的近似因子之内这个问题没有多项式时间算法,其中常数 $c>0.1134$。Gawrychowski 等[16] 给出了它的强不可近似结论,即对于每个 $\varepsilon>0$,在 $n^{1-\varepsilon}$ 的函数因子内近似计算给定的可同步的自动机的最短同步字的长度是 NP-难的。Fernau 等[10] 在研究完全的确定的有限自动机的参数复杂性时,得到不可近似结论:除非 NP$\subseteq$DTIME($n^{\log(\log n)}$),对于任意常数 $\varepsilon>0$,最短同步字问题不能在 $(1-\varepsilon)\log(|\Sigma|)$ 的比率内近似。至于利用 PCP 定理证明其他 NP-难的问题的不可近似性的研究工作有很多,不再列举。

对于同步自动机问题在部分规约的确定的有限自动机上的研究,Martyugin[20] 给出判定部分规约的有限自动机的可同步性和判定它的最短同步字的长度都是 PSPACE-完全的。此外,针对部分规约的有限自动机的研究工作还有两个:1)Bondt 等[33] 在对部分规约的有限自动机的最短同步字长的下界的研究中,构造了几种最短字长达到状态数的指数级的可同步自动机;2)Shabana 等[34-35] 以同步字长度和字母表大小为参数,将问题归约到 SAT(这个归约其实是固定参数可解的归约,而非他们阐述的多项式归约),多次调用 SAT 求解器来二分搜索求解最短同步字。

此外,我们关注到国内的崔振河等研究了限界的偏序自动机的同步问题[36]。

结束语 本文围绕部分规约的有限自动机的最短同步字计算,研究了其近似计算的复杂性,即近似难度,指出:对于部分规约的确定的有限自动机,如果要在某个近似比率内近似计算其最短重置字,那么它的近似难度是 NP-难的和 PSPACE-难的,除非 NP 和 PSPACE 分别坍塌到 P。

进一步研究的两个方向:1)考虑部分规约的有限自动机的同步字问题的参数复杂性和算法下界;2)考虑这个问题的近似计算的参数复杂性[37-38]。

参 考 文 献

[1] JÜRGENSEN H. Synchronization[J]. Information and Computation,2008,206(9/10):1033-1044.

[2] ČERNÝ J. Poznámka k-homogénnym eksperimentom s konečnými automatami[J]. Mathematicko-fyzikalny Časopis Slovenskej, Akadémie Vied,1964,14(3):208-216.

[3] VOLKOV M V. Synchronizing automata and the Černý conjecture[C]//International Conference on Language and Automata Theory and Applications. Springer,Berlin,Heidelberg,2008:11-27.

[4] POMERANZ I,REDDY S M. On achieving complete testability of synchronous sequential circuits with synchronizing sequences [C]//Proceedings.,International Test Conference. IEEE,1994:1007-1016.

[5] SANDBERG S. Homing and Synchronizing Sequences [M]// Model-based testing of reactive systems. Springer,Berlin,Heidelberg,2005:5-33.

[6] BENENSON Y,ADAR R,PAZ-ELIZUR T,et al. DNA molecule provides a computing machine with both data and fuel[J]. Proceedings of the National Academy of Sciences,2003,100(5):2191-2196.

[7] EPPSTEIN D. Reset sequences for monotonic automata [J]. SIAM Journal on Computing,1990,19(3):500-510.

[8] GOLDBERG K Y. Orienting polygonal parts without sensors [J]. Algorithmica,1993,10(2/3/4):201-225.

[9] BISKUP M T,Plandowski W. Shortest synchronizing strings for Huffman codes[J]. Theoretical computer science,2009,410(38/39/40):3925-3941.

[10] FERNAU H,HEGGERNES P,VILLANGER Y. A multi-pa-

rameter analysis of hard problems on deterministic finite automata[J]. Journal of Computer and System Sciences,2015,81(4):747-765.

[11] BERLINKOV M V. On calculating length of minimal synchronizing words[J]. arXiv:0909.3787v1.

[12] GERBUSH M,HEERINGA B. Approximating minimum reset sequences[C]// International Conference on Implementation and Application of Automata. Springer, Berlin, Heidelberg, 2010: 154-162.

[13] ANANICHEV D S,GUSEV V V. Approximation of reset thresholds with greedy algorithms[J]. Fundamenta Informaticae,2016,145(3):221-227.

[14] BERLINKOV M V. Approximating the minimum length of synchronizing words is hard[C]// International Computer Science Symposium in Russia. Springer,Berlin,Heidelberg,2010:37-47.

[15] BERLINKOV M V. On two algorithmic problems about synchronizing automata[C]// International Conference on Developments in Language Theory. Springer,Cham,2014:61-67.

[16] GAWRYCHOWSKI P,STRASZAK D. Strong inapproximability of the shortest reset word[C]// International Symposium on Mathematical Foundations of Computer Science. Springer, Berlin, Heidelberg,2015:243-255.

[17] QUAAS K,SHIRMOHAMMADI M. Synchronizing Data Words for Register Automata[J]. ACM Transactions on Computational Logic (TOCL),2019,20(2):1-27.

[18] Doyen L,Massart T,Shirmohammadi M. The complexity of synchronizing Markov decision processes[J]. Journal of computer and system sciences,2019,100:96-129.

[19] ZHU K,WU G Q,WU L H,et al. Computational complexity of several problems for resetting timed automata[J]. Ruan Jian Xue Bao/Journal of Software,2019,30(7):2033-2051.

[20] MARTYUGIN P. Computational complexity of certain problems related to carefully synchronizing words for partial automata and directing words for nondeterministic automata[J]. Theory of Computing Systems,2014,54(2):293-304.

[21] MARX D. Parameterized complexity and approximation algorithms[J]. The Computer Journal,2008,51(1):60-78.

[22] DU D Z,GE K Y,WANG J. Introduction to computational complexity[M]. Higher Eeucation Press,2002.

[23] ARORA S,BARAK B. Computational complexity:a modern approach[M]. Cambridge University Press,2009.

[24] ARORA S,SAFRA S. Probabilistic checking of proofs; A new characterization of NP[C]// 33rd Annual Symposium on Foundations of Computer Science,FOCS 1992. IEEE Computer Society,1992:2-13.

[25] ARORA S,LUND C,MOTWANI R,et al. Proof verification and hardness of approximation problems[C]// Proceedings., 33rd Annual Symposium on Foundations of Computer Science. IEEE,1992:14-23.

[26] XU D Y. PCP theorem and its applications to research on non-approximatable problems[J]. Journal of frontiers of computer science and technology,2008,2(1):20-31.

[27] CONDON A,FEIGENBAUM J,LUND C,et al. Probabilistically checkable debate systems and approximation algorithms for PSPACE-hard functions[C]// Proceedings of the twenty-fifth annual ACM symposium on Theory of computing. 1993:305-314.

[28] CONDON A,FEIGENBAUM J,LUND C,et al. Probabilistically Checkable Debate Systems and Nonapproximability Results for PSPACE-Hard Functions[J]. Chicago J. Theoretical Computer Science,1995(4):1-35.

[29] KOZEN D. Lower bounds for natural proof systems[C]// 18th Annual Symposium on Foundations of Computer Science (SFCS). IEEE,1977:254-266.

[30] KUPFERMAN O,VARDI M Y,WOLPER P. An automata-theoretic approach to branching-time model checking[J]. Journal of the ACM (JACM),2000,47(2):312-360.

[31] COHEN A,WIGDERSON A. Dispersers,deterministic amplification,and weak random sources[C]// 30th Annual Symposium on Foundations of Computer Science (SFCS). IEEE,1989:14-19.

[32] WAREHAM H T. The parameterized complexity of intersection and composition operations on sets of finite-state automata [C]// International Conference on Implementation and Application of Automata. Berlin:Springer,2000:302-310.

[33] DE BONDT M,DON H,ZANTEMA H. Lower bounds for synchronizing word lengths in partial automata[J]. International Journal of Foundations of Computer Science, 2019, 30(1): 29-60.

[34] SHABANA H,VOLKOV M V. Using Sat solvers for synchronization issues in partial deterministic automata[C]// International Conference on Mathematical Optimization Theory and Operations Research. Springer,Cham,2019:103-118.

[35] SHABANA H. Exact synchronization in partial deterministic automata[J]. Journal of Physics: Conference Series,2019,1352(1):012047.

[36] CUI Z H,HE Y,SUN S Y. Synchronizing bounded partially ordered automata[J]. Chinese Journal of Computers,2019(3):610-623.

[37] CHALERMSOOK P,CYGAN M,KORTSARZ G,et al. From gap-eth to fpt-inapproximability: Clique, dominating set, and more[C]// 2017 IEEE 58th Annual Symposium on Foundations of Computer Science (SFCS). IEEE,2017:743-754.

[38] LAEKHANUKIT B,MANURANGSI P. On the parameterized complexity of approximating dominating set[C]// Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing(STOC). ACM,2018:1283-1296.

ZHU Kai,born in 1979,doctorial student,lecturer. His main research interests include theoretical computer science and formal method.

YUAN Meng-ting, born in 1976, Ph.D, associate professor,is a member of China Computer Federation. His main research interests include programming language theory, compiler optimization and software engineering.