

基于 DNA 计算的分子下推自动机

张 征¹⁾ 刘 洁²⁾ 石晓龙¹⁾

¹⁾(华中科技大学控制科学与工程系 武汉 430074)

²⁾(华中科技大学材料科学科学与工程院 武汉 430074)

摘 要 DNA 分子计算的工作原理是对生物系统进行编码,以生物化学反应为基础,利用生物技术实现生物系统的状态转移来推进计算过程. 2001 年以色列的 Yaakov Benenson 等人在基于 DNA 计算的发卡模型实现了具有状态转移功能的分子有限状态自动机,国内则有利用 DNA 计算的方法构造可编程分子下推存储器的相关研究. 该存储器基于分子自动机的原理,能按一定逻辑进行自组装,是一种纳米尺度的生物存储机构. 文中首先通过在分子有限自动机上扩展一个分子下推存储器从而获得了一种简单的分子下推自动机,并基于该下推自动机提出了一类语言的分子自动机解法. 接着提出了两种改进的分子下推自动机的模型,通过增加模型复杂度,分别解决了基本型分子下推自动机存在输入字符串限制和输入分子形式不统一的问题. 计算理论表明,该种下推自动机的计算能力超过了已有的有限自动机.

关键词 分子下推自动机;DNA 计算

中图法分类号 TP301

Biomolecular Pushdown Automaton Based on the DNA Computing

ZHANG Zheng¹⁾ LIU Jie²⁾ SHI Xiao-Long¹⁾

¹⁾(Department of Control Science and Engineering, Huazhong University of Science & Technology, Wuhan 430074)

²⁾(Institute of Material Science and Engineering, Huazhong University of Science & Technology, Wuhan 430074)

Abstract DNA computing aims at using nucleic acids for computing. DNA solutions can act as billions of parallel nanoprocessors with few consume. A biomolecular finite automaton has been realized by Benenson in 2001, and the programmable biomolecular pushdown store based the DNA computing is available too. This pushdown store can self assemble with certain logical. In this paper, a simple biomolecular pushdown automaton is constructed with a finite automaton and a pushdown store firstly, and an algorithm is designed to solve a kind of languages. In addition, two improved pushdown automata are designed to solve some problems exiting in the original pushdown store. One of them can accept input string with infinite symbols in theory, and the other can uniform the input symbol. The pushdown automata described in this paper are more powerful than the existing finite automaton in computing theory.

Keywords biomolecular pushdown automaton; DNA computing

1 Introduction

Head gave the principle of molecular computing in 1987, and Adleman gave a successful solu-

tion of a seven-vertex instance of the NP-complete Hamiltonian directed path problem by a DNA algorithm which initiated the field of biomolecular computing in 1994^[1]. In recent years, both theory and

technology of molecular computing have improved very fast.

DNA computing is one of the biomolecular computing models, which includes sticker model^[2], splicing system model^[3-6], hairpin model^[7], and plasmid DNA model^[8], and every model can resolve some special problem.

The concept of Bio-Turning-machine was proposed by Bennet in 1972. This machine consists of DNA molecules and enzyme, and can compute as Turing machine. But the corresponding model cannot be realized at that time for lacking of some necessary biotechnologies^[9]. Recently, a realization of computing device operating autonomously on the molecule scale was available. This solution is based on the DNA hairpin formation which is presented in 2000 by Sakamoto. It's a programmable finite automaton comprising DNA and DNA-manipulating enzymes, which solves computational problems autonomously^[10]. The hardware of this automaton consists of restriction nuclease and ligase, the software and input are encoded by double-stranded DNA, and programming amounts to choosing appropriate software molecules. Upon mixing solutions containing these components, the automaton processes the input molecules via a cascade of restriction, hybridization and ligation cycles, producing detectable output molecules which encode the automata's final state, and thus the computational result. It acts as basic features and processes of finite automata with two internal states and an alphabet comprising two input symbols. Then this group extends this work in 2003 in medical area.

The extended system of this kind of automata was described in 2005, with three internal states and an alphabet comprising three input symbols. Another automaton was illustrated in 2006, which can be used as a programmable pushdown store. A new kind of restriction nuclease was used as hardware in this pushdown store^[11-13]. For the pushdown store has a programmable self-assembly computing device, it can be used to construct the initial solution space of the graph coloring problem.

2 Molecular Pushdown Automaton

Given a language $L(M) = \{a^n b^n | n \geq 1\}$, it cannot be accepted by any finite automata. For finite automata can only memorize the count of the read symbol, they can accept the above language. But the finite automata can only store the information with its inner states. For any positive number " k ", the finite automaton must have an inner state corresponding to the state: read k 'a'. This means

the finite automata must have k inner state. In brief the automata must have infinite states. It is conflict with the basic principle of the finite automata. A finite automaton with a push down store can accept the language above, and this automaton is more powerful than the finite automata. The biomolecular pushdown automata can be designed to increase the computing ability.

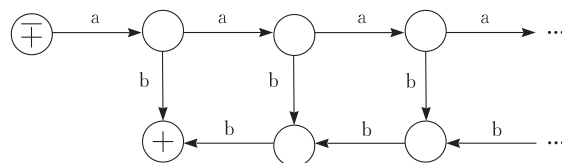


Fig. 1 A infinite automaton which can accept the string $a^n b^n$

2.1 A Simple Pushdown Automaton

Consider the language $L(M) = \{w | w \in (a, b)^*, \text{count}(a) = \text{count}(b)\}$, a simple biomolecular pushdown automaton can be constructed with a biomolecular finite automaton and a biomolecular pushdown store. The store molecule will lengthen when a symbol "a" is read, and shorten when a symbol "b" is read. The string will be accepted by the automaton only if the length of the store molecule equals the original length.

The constructed coding scheme and mechanism of a simple pushdown automaton is shown in the Fig. 2. The symbol biomolecule are shown in Fig. 2(a). Symbol "a" contains the recognition site of restrict enzyme BslI, and the symbol "b" contains the recognition site of restrict enzyme BstXI. The coding scheme of initial store biomolecule is shown in Fig. 2(b). Its sticky end NCC can be matched with the sticky end of symbol molecules (NGG).

The mechanism of the pushdown automaton is shown in the Fig. 2(c). There are store molecules, restrict enzyme BslI and BstXI in this solution. After reading the string 'aabb', gained store molecule is the same as the original store molecule, and the other store molecules in the intermediate step are not the same one during the process. So the string a, aa, aab is not a valid string of language: $L(M) = \{w | w \in (a, b)^*, \text{count}(a) = \text{count}(b)\}$, but the string "aabb" is a valid string of it.

In this solution, a Cut-stick reaction will occur when the automaton reads a symbol. At first, enzyme T_4 ligates the symbol molecules and the store molecules; Then a restrict enzyme cuts the ligated if it reads a symbol "a", and BslI will operate and the double-stranded DNA will be lengthened. When a symbol "b" is read, the store molecule will be shortened by the enzyme BstXI. The pushdown automaton will be terminated at the

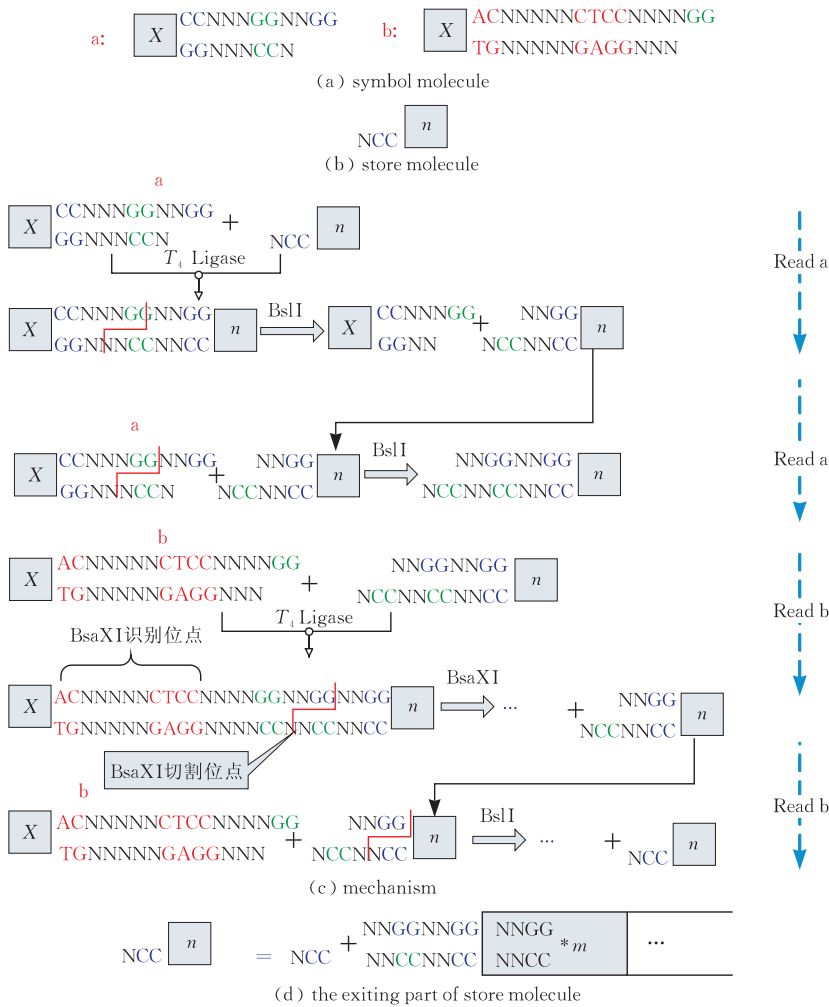


Fig. 2 The mechanism of a simple pushdown automaton

original state when the automaton reads the equal symbol “a” and “b”. Otherwise it will halt at the other state, and the store molecule will be different with the initial state. The mechanism of the pushdown store is detailed in literature^[13]. But this simple pushdown store has a bug when the number of the ‘b’ is much more than the number of the ‘a’ in a string. The entire store molecule will be cut and the automaton will halt in abnormal state (as Shown in Fig. 2(d)). For we can’t forecast the input string, the store molecule must be infinite to avoid the bug, or design a suitable length in a certain situation.

2.2 Improved Pushdown Automaton

Because the simple pushdown store has a disadvantage, the better pushdown automaton should be designed to avoid the problem.

Consider the following language:
 $L(M)=\{w \mid w \in (a,b)^*, count(a)=count(b)\}$,
then the following algorithm will construct strings which are accepted by pushdown automaton.

Initializing: Empty the store and scan the string $w \in (a,b)^*$ from left to right. If the store is

empty and the current symbol is ‘a’, then push the symbol ‘a’ to the stack; if the store is empty and the current symbol is b, then push the symbol ‘b’ to the stack. If the top symbol of the stack is different from the current symbol, then pop the top symbol from the stack; if the top symbol of the stack is same with the current symbol then push the current symbol to the stack. Then the store operating molecule will be designed as Fig. 3.

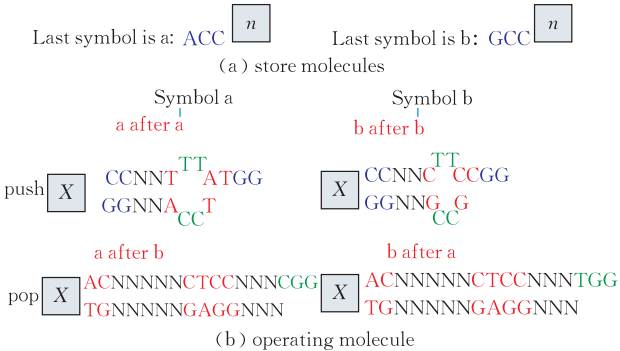


Fig. 3 Store operating molecules

As shown in Fig. 3, each input symbol has a

pop operating molecule and a push operating molecule. Two operating molecule will work separately. When the top element of the stack is the same symbol, the push molecule activated, otherwise the pop molecule activated. This solution solves the problem of the simple pushdown automaton. But this automaton needs two operating molecule of each input symbol. It will be puzzled in some situation.

The improved input symbol can be designed as shown in Fig. 4. Some tag molecules are added to

the new solution. The new input symbols are shown in Fig. 4(a): The sticky end of the symbol ‘a’ can be matched with the sticky end of the tag molecules for symbol ‘b’. And the sticky end of the symbol ‘b’ will be matched with the tag molecules for symbol ‘a’. The tags have the similar sequence with the operating molecules. In this solution, the input symbol controls the consistency of the tag, and then affects the consistency of the operating molecule. For example(as show in Fig. 5), when a symbol ‘a’ is read, symbol ‘a’ will ligate

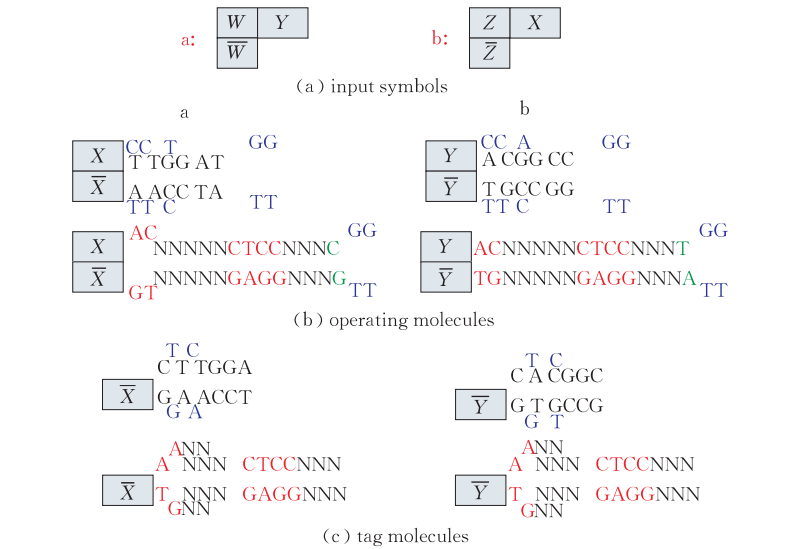


Fig. 4 Tag molecules

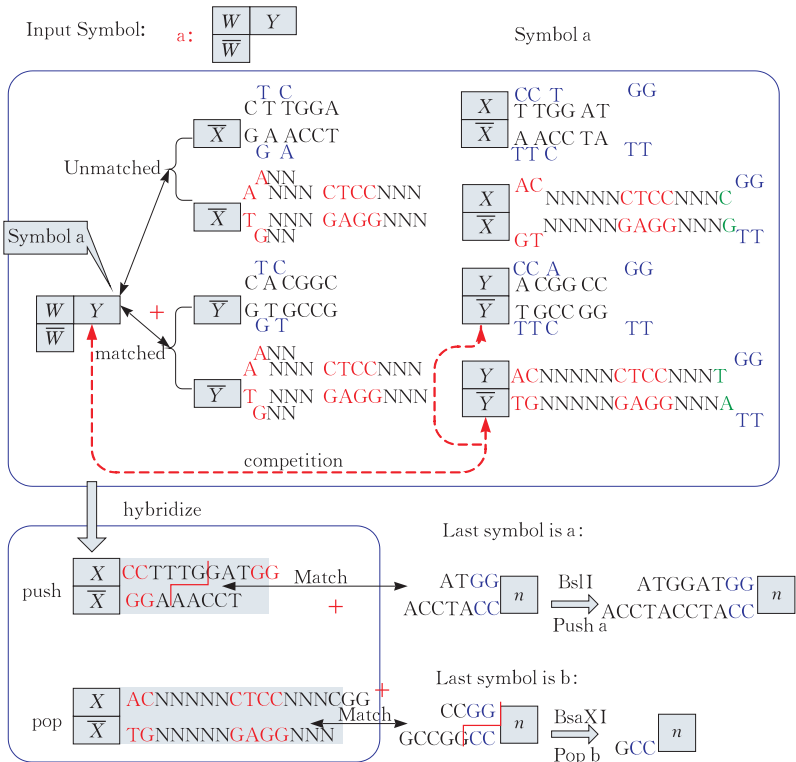


Fig. 5 The mechanism of tag molecule

with the tag molecule for the symbol 'b'. This tag molecule can't be matched with the initial operating molecules. Then the final operating molecules for symbol 'b' can't be built in solution. In the other hand, the building of the final operating molecules for symbol 'a' will not be affected by symbol 'a'. When the top element of the stack is symbol 'a', the push molecules for symbol 'a' will be activated, otherwise the pop molecules for symbol 'a' will work.

This pushdown automaton can uniform the code sequence of the input symbol, but the extra tag biomolecule must be added to this solution. And it will be more complex and unstable. So the suitable pushdown automata should be chosen to fit the certain situation.

3 Results and Discussion

The biomolecular automaton can act as a traditional automaton with very low consume and vast parallel. It can be applied to more areas and its result is more stable than other model of the DNA computing. But the biomolecular automaton has some disadvantage too. Firstly, the biomolecular automaton is limited by its hardware-restrict enzyme. Secondly, although the biomolecular automaton can be used as a normal automaton, it will be less efficient than the other DNA computing model for some reason.

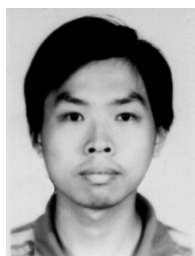
In this paper, an improved biomolecular automaton was designed. Three biomolecular pushdown automata were constructed with a finite automaton and a pushdown store. For the computing power of a Turning machine equals to a pushdown automaton with two pushdown store, an automaton made of biomolecule may be designed to simulate the Turning machine

For the pushdown automaton is not limited by the length of the DNA molecules, the automaton motioned above can run continuously in theory. A molecular pulsator may be designed as a probe to

check the atom molecule. This pulsator will work when its store molecules match with the target molecules, and the state transition will be consumed. The result can be gained by checked consistency of the transition molecules.

References

- [1] Adleman L M. Molecular computation of solutions to combinatorial problems. *Science*, 1994, 266: 1021-1023
- [2] Roweis S, Winfree E, Burgoyne R, Chelyapov, Goodman M, Rothmund P, Adleman L A. Sticker based model for DNA computation//*Proceedings of the 2nd DIMACS Workshop on DNA Based Computers*. Princeton, 1996, DIMACS Series, Vol. 44, AMS Press, 1999: 1-29
- [3] Kari L. DNA computing: Arrival of biological mathematics. *The Mathematical Intelligencer*, 1997, 19(2): 9-22
- [4] Pan Lin-Qiang, Xu Jin, Liu Ya-Chun. A surface-based DNA algorithm for the maximal clique problem. *Chinese Journal of Electronics*, 2002, 11(2): 469-471
- [5] Pan Lin-Qiang, Xu Jin, Liu Ya-Chun. A surface-based DNA algorithm for the minimal vertex problem. *Progress in Natural Science*, 2003, 13(1): 81-84
- [6] Praun G, Rozenberg G, Salomaa A. *DNA Computing—New Computing Paradigms*. Berlin: Springer, 1998
- [7] Sakamoto K, Gouzu H, Komiya K et al. Molecular computation by DNA hairpin formation. *Science*, 2000, 288: 1223-1226
- [8] Head T, Rozenberg G, Bladergroen R B, Breek C K D, Lommerse P H M, Spaink H P. Computing with DNA by operating on plasmids. *BioSystems*, 2000, 57: 87-93
- [9] Bennett C H. On constructing a molecular computer. *IBM Journal of Research and Development*, 1973, 17: 525-532
- [10] Benenson Y, Paz-Elizur T, Adar R et al. Programmable and autonomous computing machine made of biomolecules. *Nature*, 2001, 414: 430-434
- [11] Shi X L, Li X, Zhang Z et al. Improve capability of DNA automaton: DNA automaton with three internal states and tape head move in two directions//*Lecture Notes in Computer Science 3645*. Springer, 2005: 71-79
- [12] Shi Xiao-Long, Pan Lin-Qiang, Xu Jin. General DNA automaton model with R/W Tape//*Lecture Notes in Computer Science 4115*. Springer, 2006: 258-266
- [13] Zhang Zheng, Xu Jin, Liu Jie, Pan Lin-Qiang. Programmable pushdown store base on DNA computing//*Lecture Notes in Computer Science 4115*. Springer, 2006: 286-293



ZHANG Zheng, born in 1976, lecturer. His research interests include DNA computing and biomolecular automaton.

LIU Jie, born in 1972, Ph. D. candidate, lecturer. Her main directions: rapid prototyping & rapid tooling; rapid development & applications of drip emitter.

SHI Xiao-Long, born in 1975, associate professor. His research interests include DNA computing, neural networks, parallel and distributed processing, computer vision and image processing.