Special course in Computer Science: Molecular Computing

Lecture 14: Molecular Computing Machineries - Models and Wet Implementations

Vladimir Rogojin

Department of CS, Abo Akademi

http://combio.abo.fi/teaching/special-course-in-computer-science-molecular-computing/

Fall 2015

Role of molecular machines

- In the early days it was understood that molecular machineries would replace silicon-based hardware with biological hardware through the development of molecular-based computers

- However, it is now recognized that molecular machineries should not substitute for existing computers but complement them

History

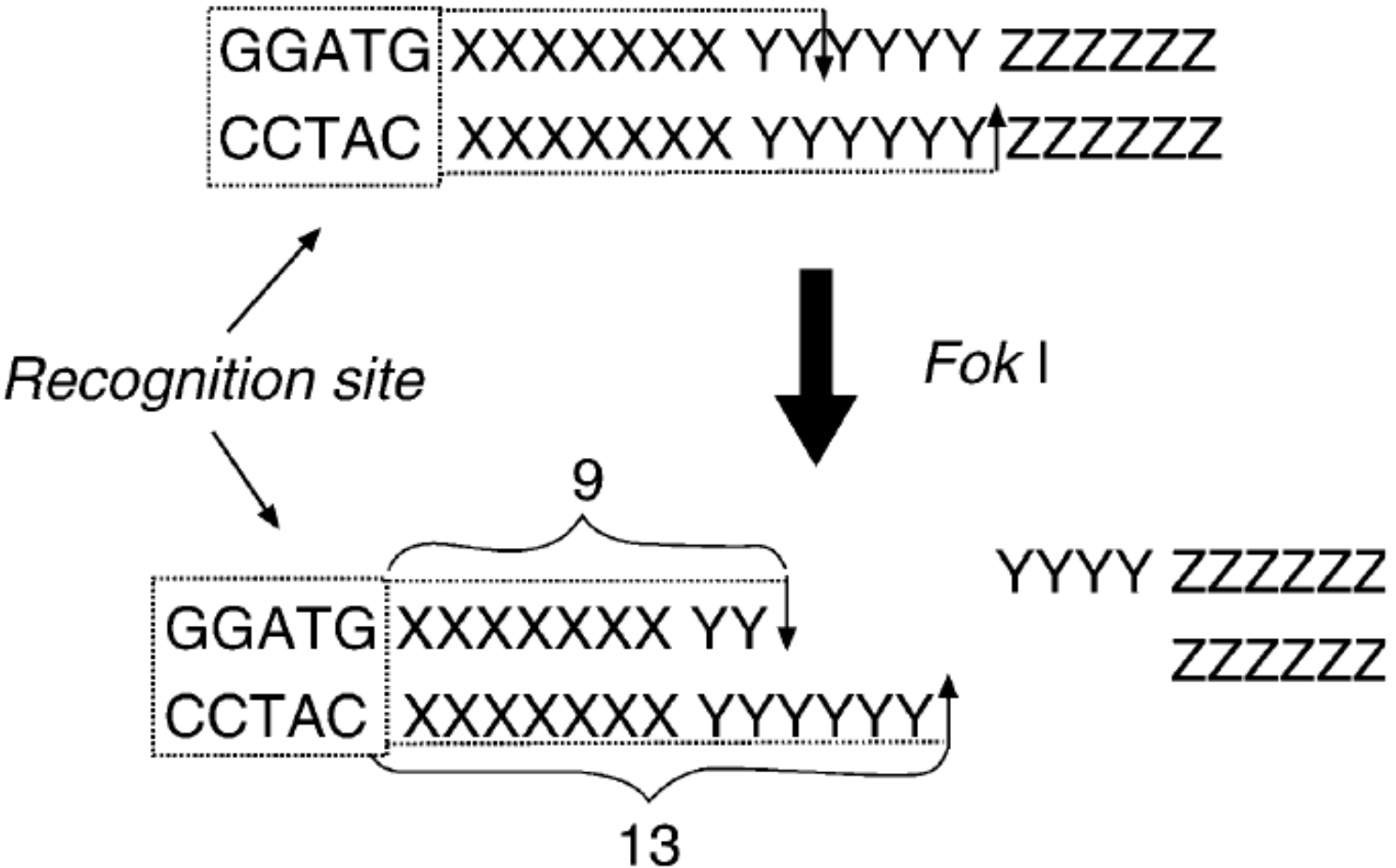
- 1973, Biofizika, Vaintsvaig and Liberman
  - Cell molecular computers discussed
- 1985, M. Conrad
  - Research on information-processing capabilities using macromolecules
- 1987, T. Head
  - Splicing systems and languages, first achievement of mathematical analysis using biochemical operations of DNA recombination
- 1994, Adleman
  - First experimental implementation of math problem solution with DNA molecules:
    - Solved an instance of HPP

Molecular Implementations and Applications

- We consider here several wet lab experimental works of molecular computing machineries based on structured DNA molecules

- Those molecular devices involve a variety of ideas in controlling molecular behaviors,
  - such as utilizing specific restriction enzymes, deoxyribozymes, sophisticatedly designed DNA tiles

Restriction enzyme : *Fok* I



Enzyme-Based DNA Automata

- An autonomous computing machine that comprises DNAs with DNA manipulating enzymes and behaves as a finite automaton

- The hardware of the automaton consists of a restriction nuclease and ligase, while the software and input are encoded by double-stranded DNAs

Encoding DFA onto DNA

- A symbol a
  - A sequence of nucleotides Sa
- A state qi while reading symbol a
  - A corresponding subsequence Sqi,a
- A rule (qi,a) → qj
  - A transition molecule mqi,a
- An input string S
  - A sequence SS encoding sequence of letters from S
- Computation
  - Disassembling the sequence SS till the terminal subsequence is produced

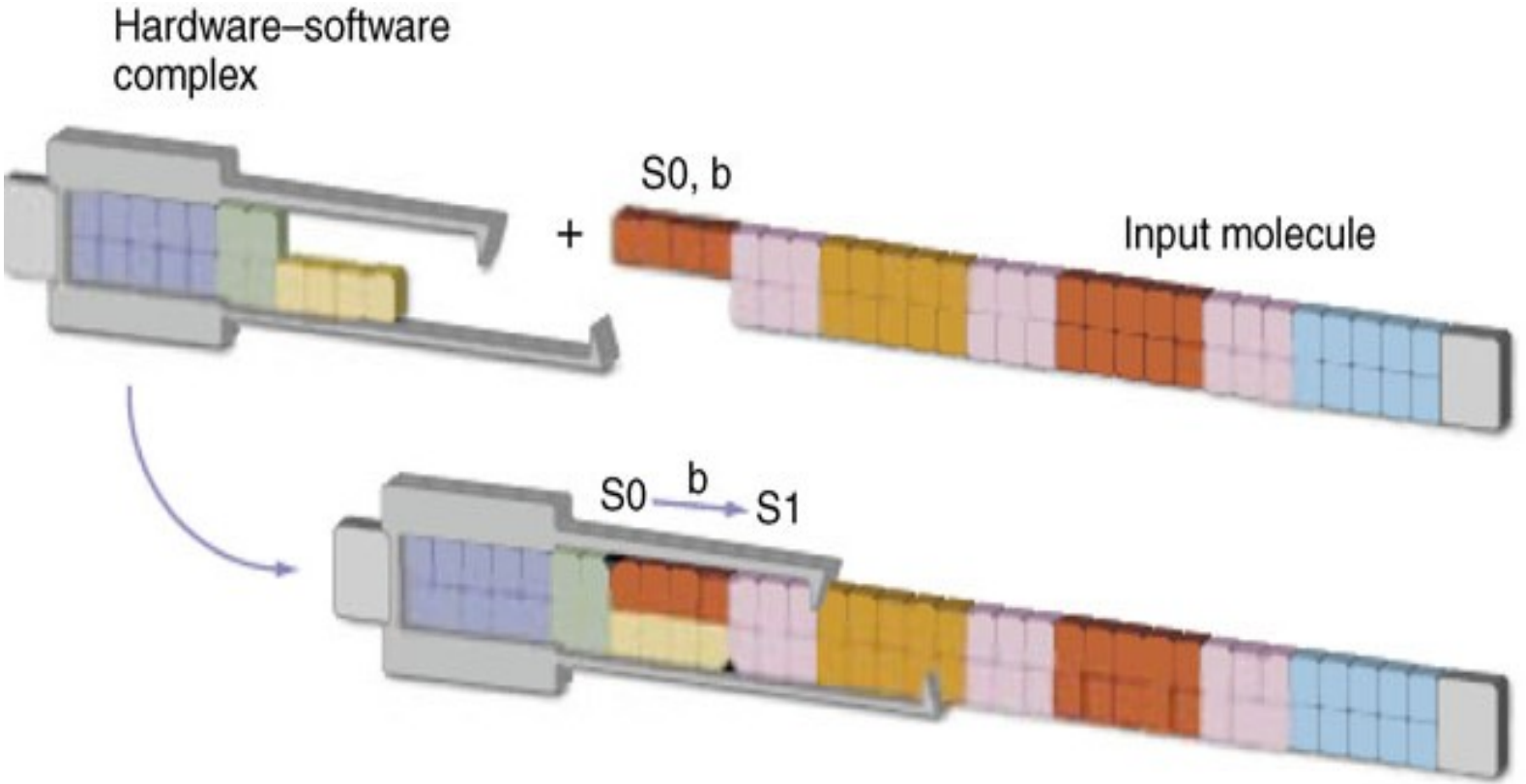
Finite-State Computations via Disassembly of DNA Nanostructures

- Reverse to the assembly-based computation techniques
- Input:
  - A linear double-stranded DNA nanostructure
- Computation:
  - Digesting part of the nanostructure each step
    - At each step: a sticky end at one end of the nanostructure encodes the current state and symbol
    - The finite transition is determined by hybridization of the current sticky end with a transition “rule” molecule
      - Which encodes the finite-state transition rule

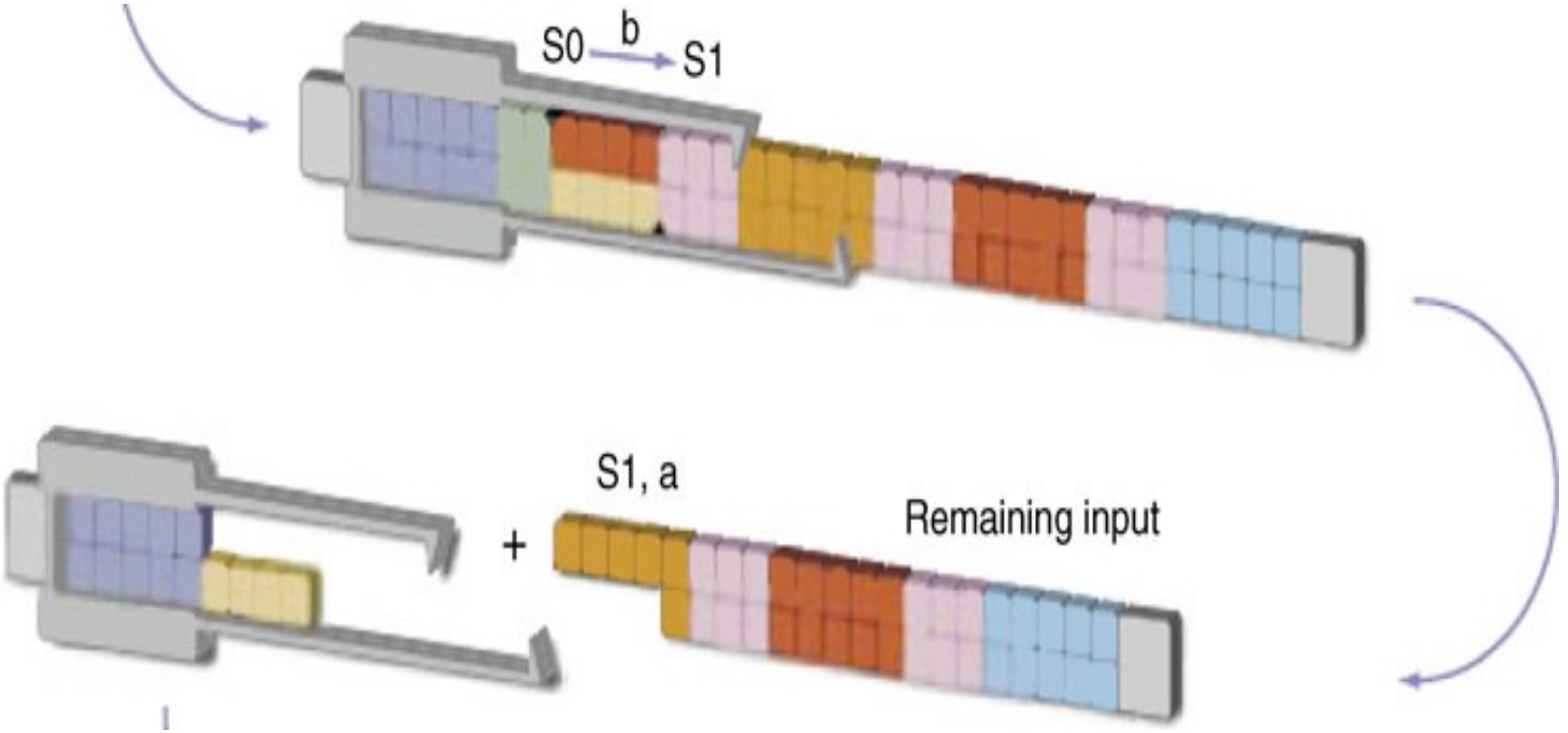
Finite-State Computations via Disassembly of DNA Nanostructures

- Digesting part of the nanostructure each step
  - At each step: a sticky end at one end of the nanostructure encodes the current state and symbol
  - The finite transition is determined by hybridization of the current sticky end with a transition “rule” molecule
    - Which encodes the finite-state transition rule
  - A restriction enzyme
    - recognizes the sequence encoding the current input as well as the current state,
    - cuts the appended end of the linear DNA and
    - exposes a new sticky end encoding the next state and symbol

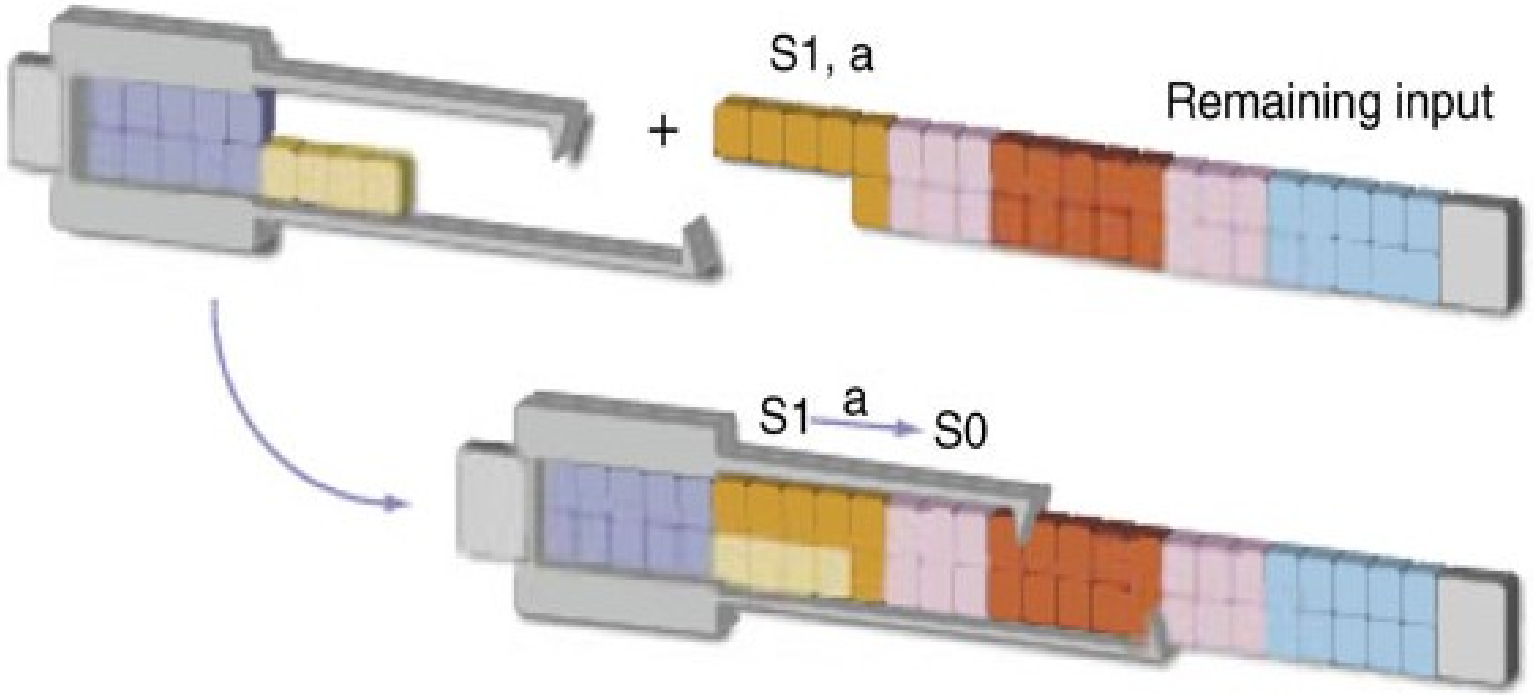
Finite-State Computations via Disassembly of DNA Nanostructures



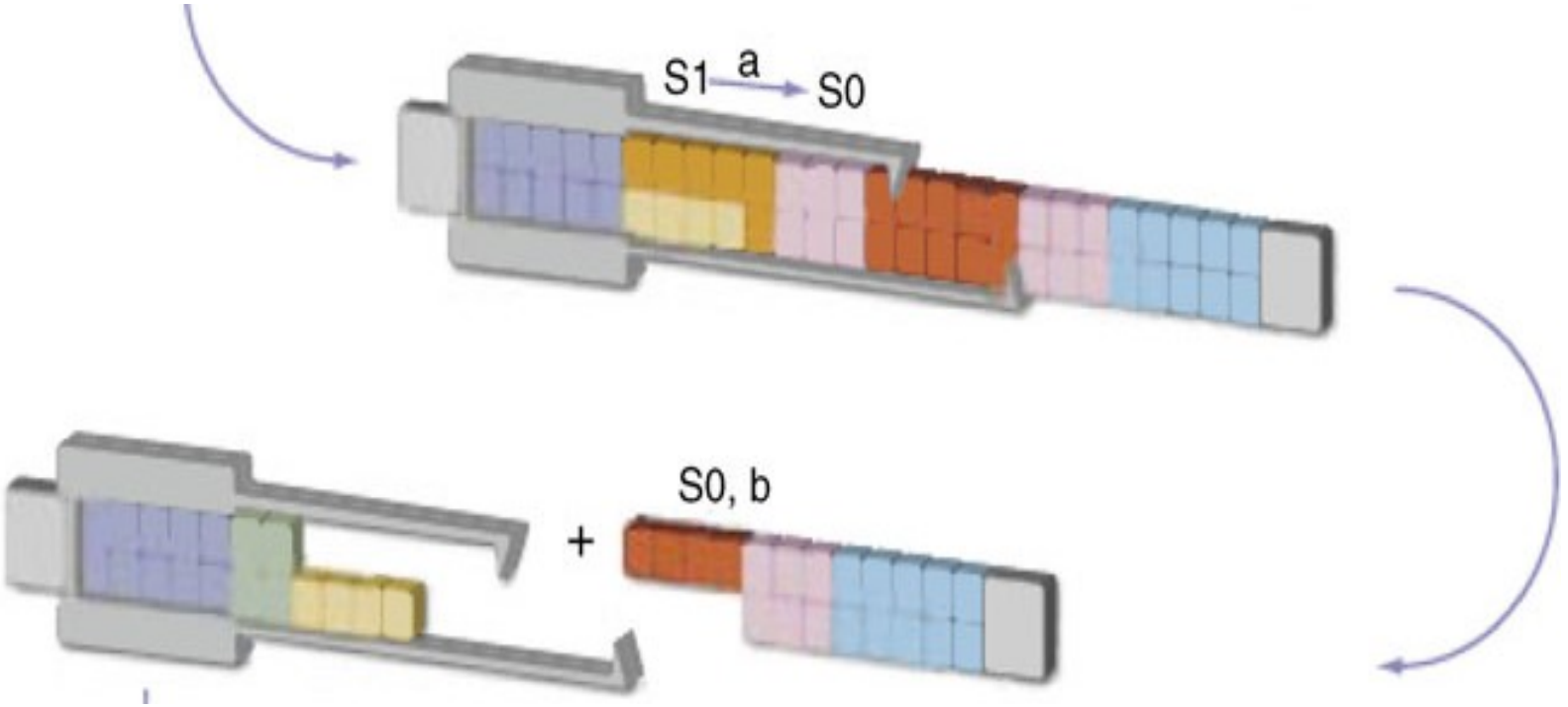
Finite-State Computations via Disassembly of DNA Nanostructures



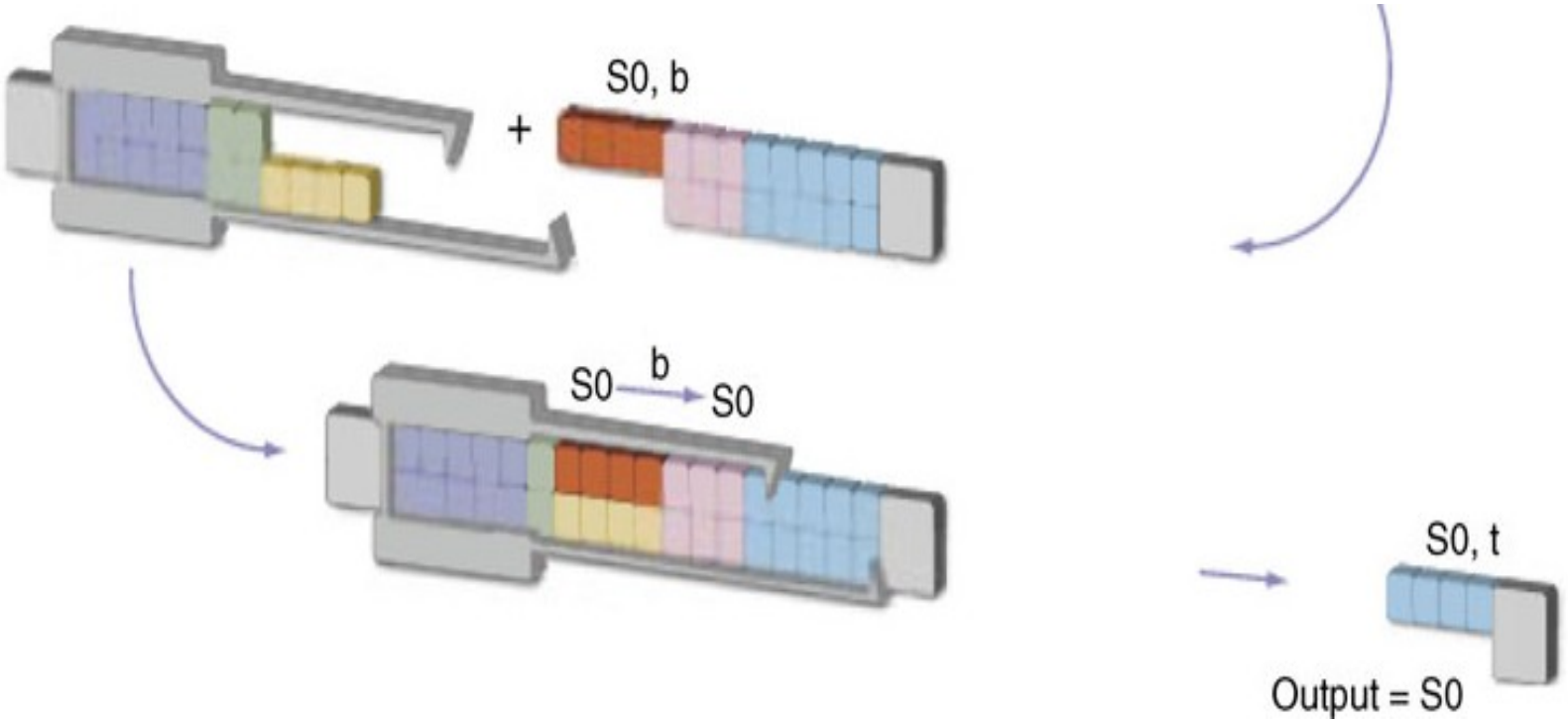
Finite-State Computations via Disassembly of DNA Nanostructures



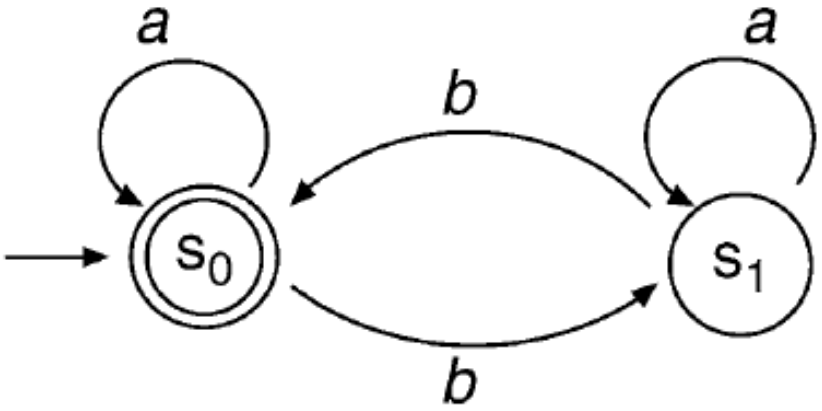
Finite-State Computations via Disassembly of DNA Nanostructures



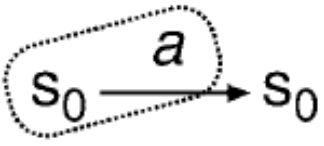
Finite-State Computations via Disassembly of DNA Nanostructures



Deterministic Finite Automata Example and its encoding

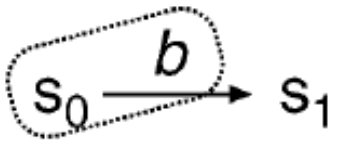


a	*b*	*t* (Terminator)
(s_1, a) CTGGCT (s_0, a)	(s_1, b) CGCAGC (s_0, b)	(s_1, t) TGTCGC (s_0, t)



GGATGTAC
C CTACATGCCGA

Center



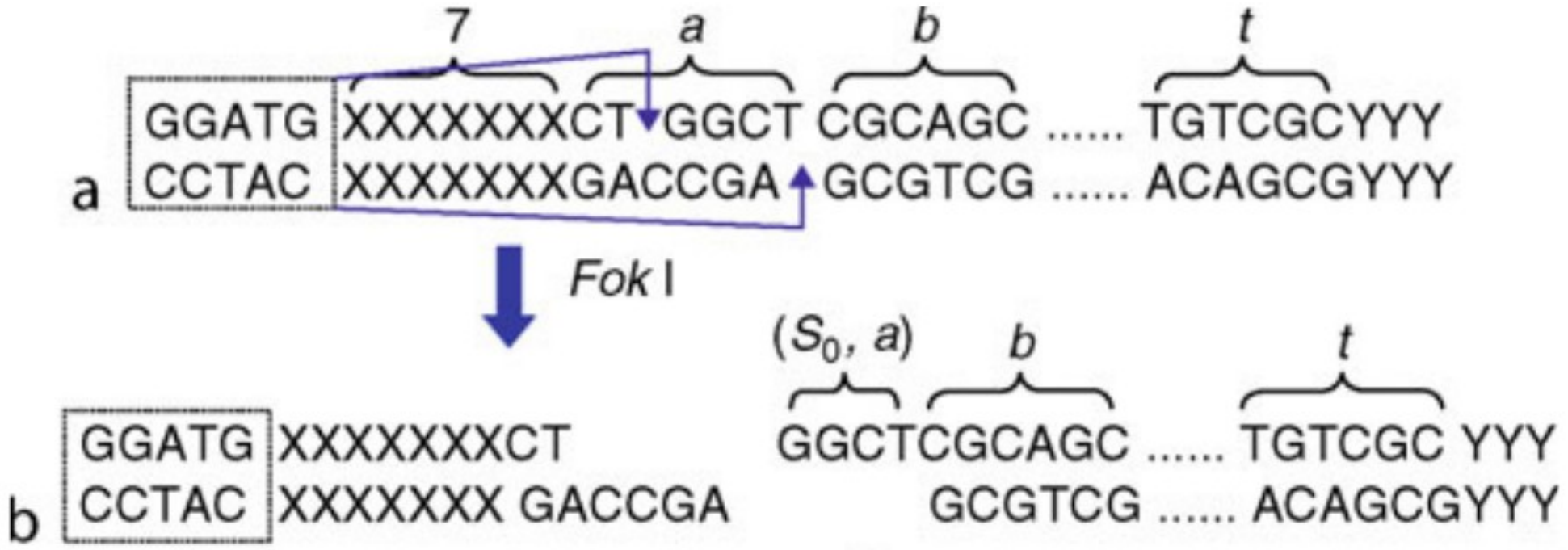
GGATGACGAC
C CTACTGCTGGTCG

Center

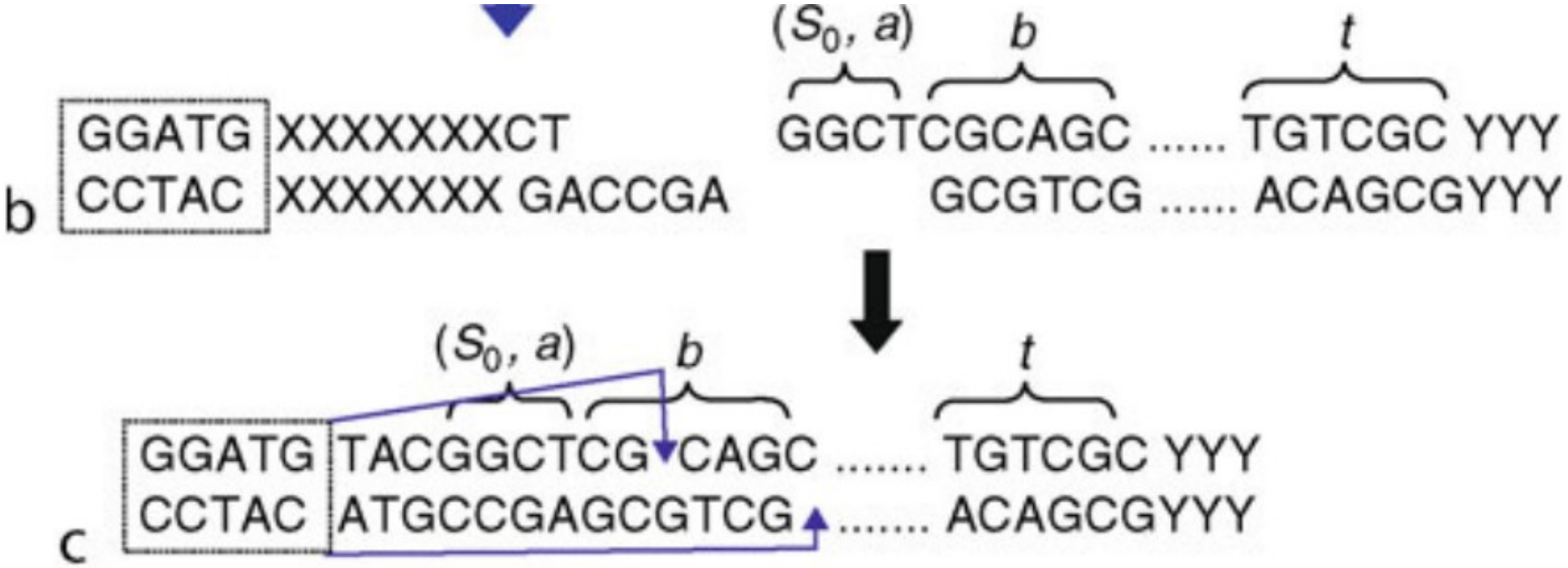
XXXXX
XXXXXAGCG

(Acceptance detection molecule)

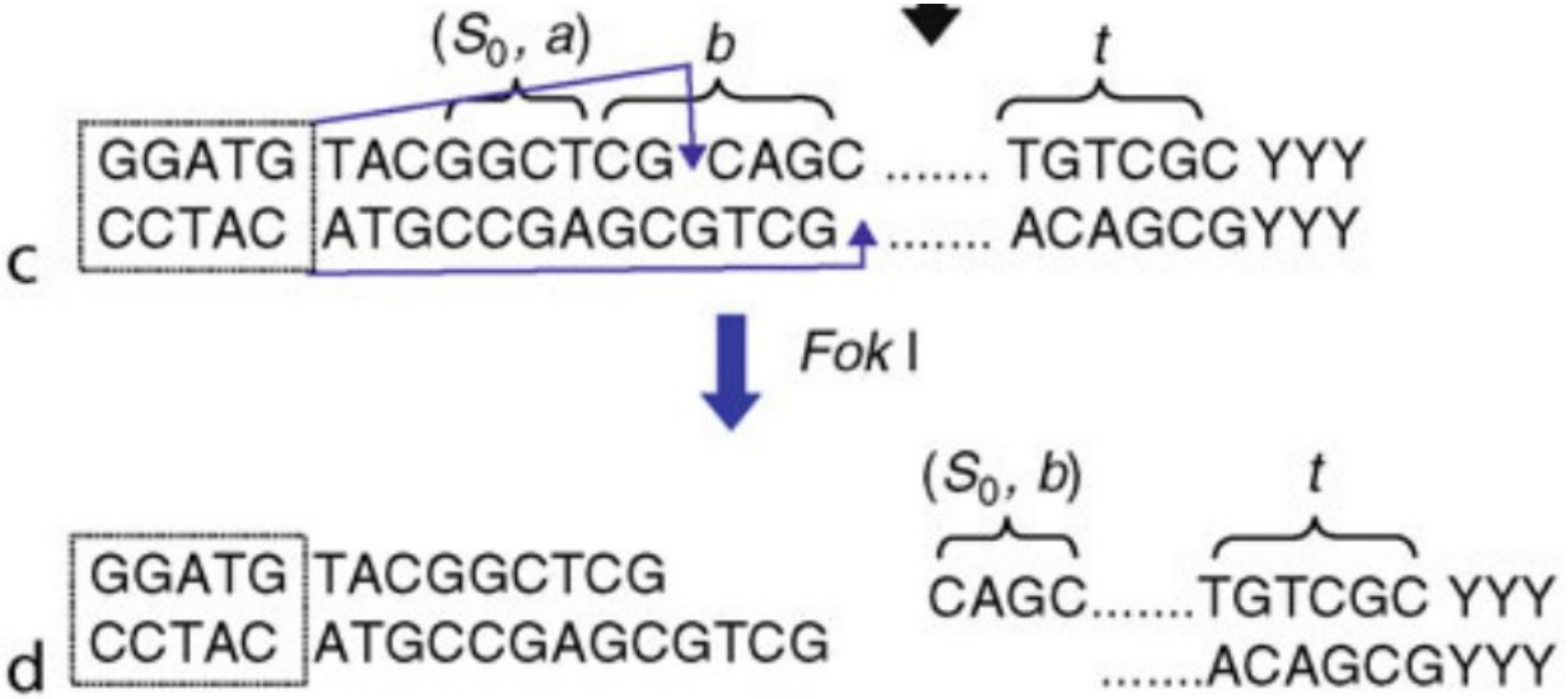
Implementing DFA Computing process of input w=ab(t)



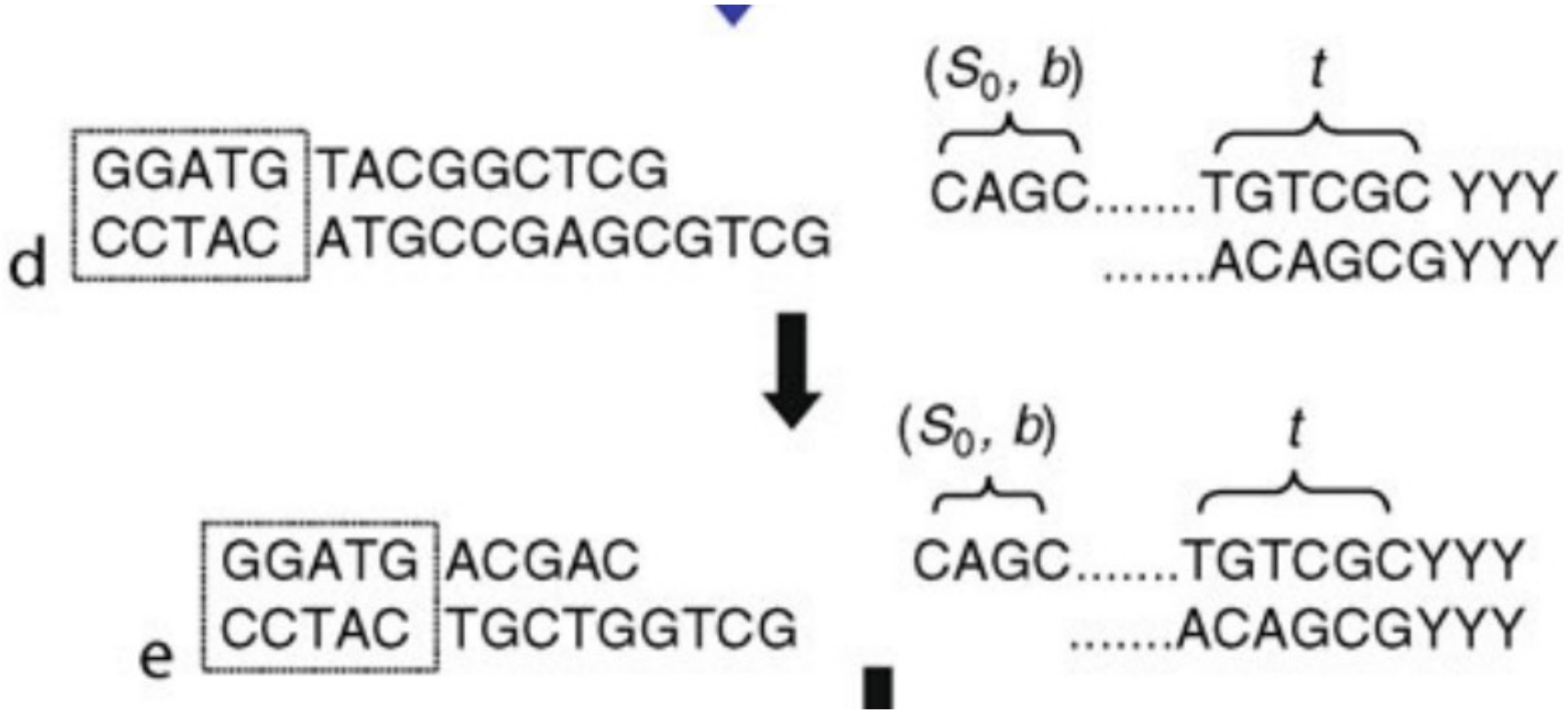
Implementing DFA Computing process of input w=ab(t)



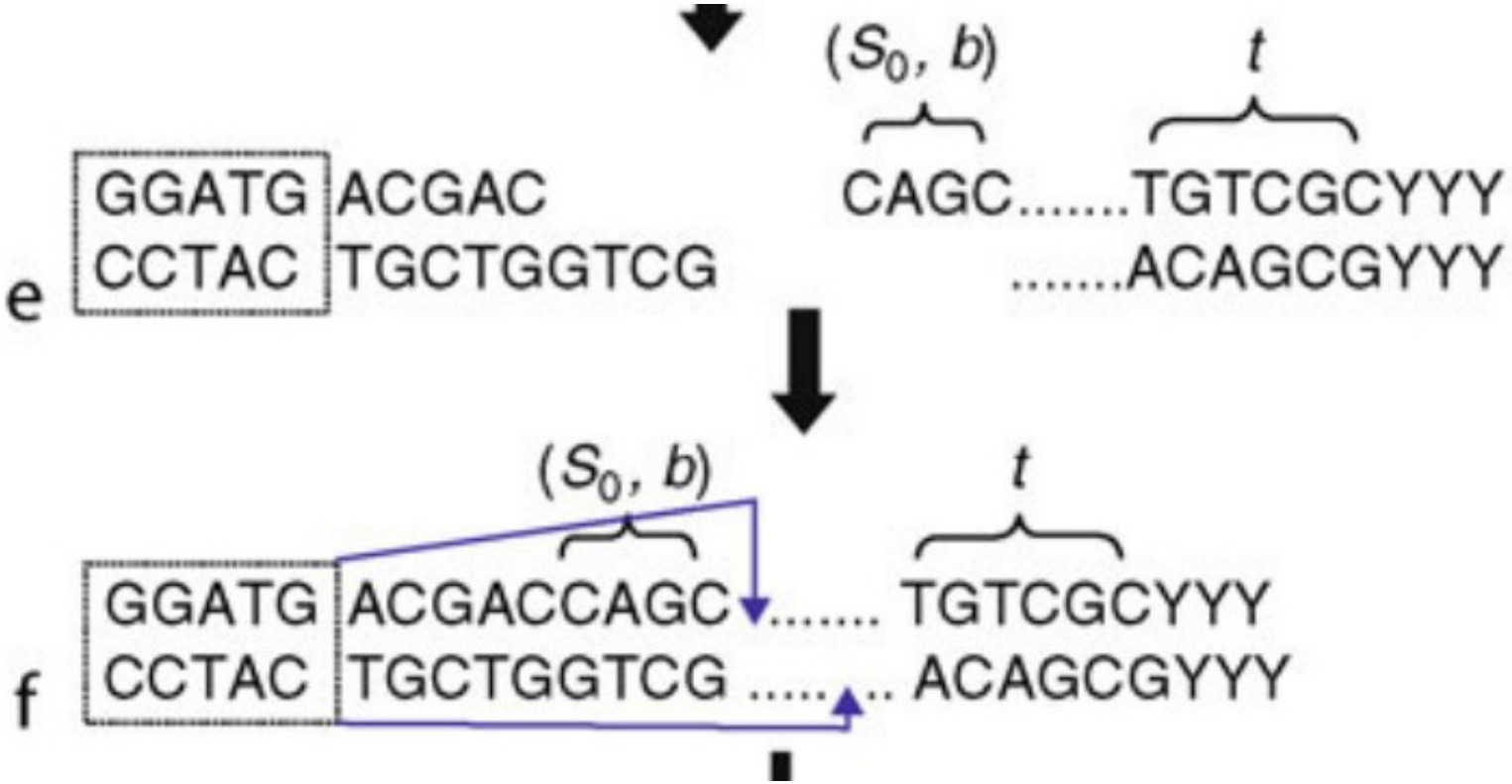
Implementing DFA Computing process of input w=ab(t)

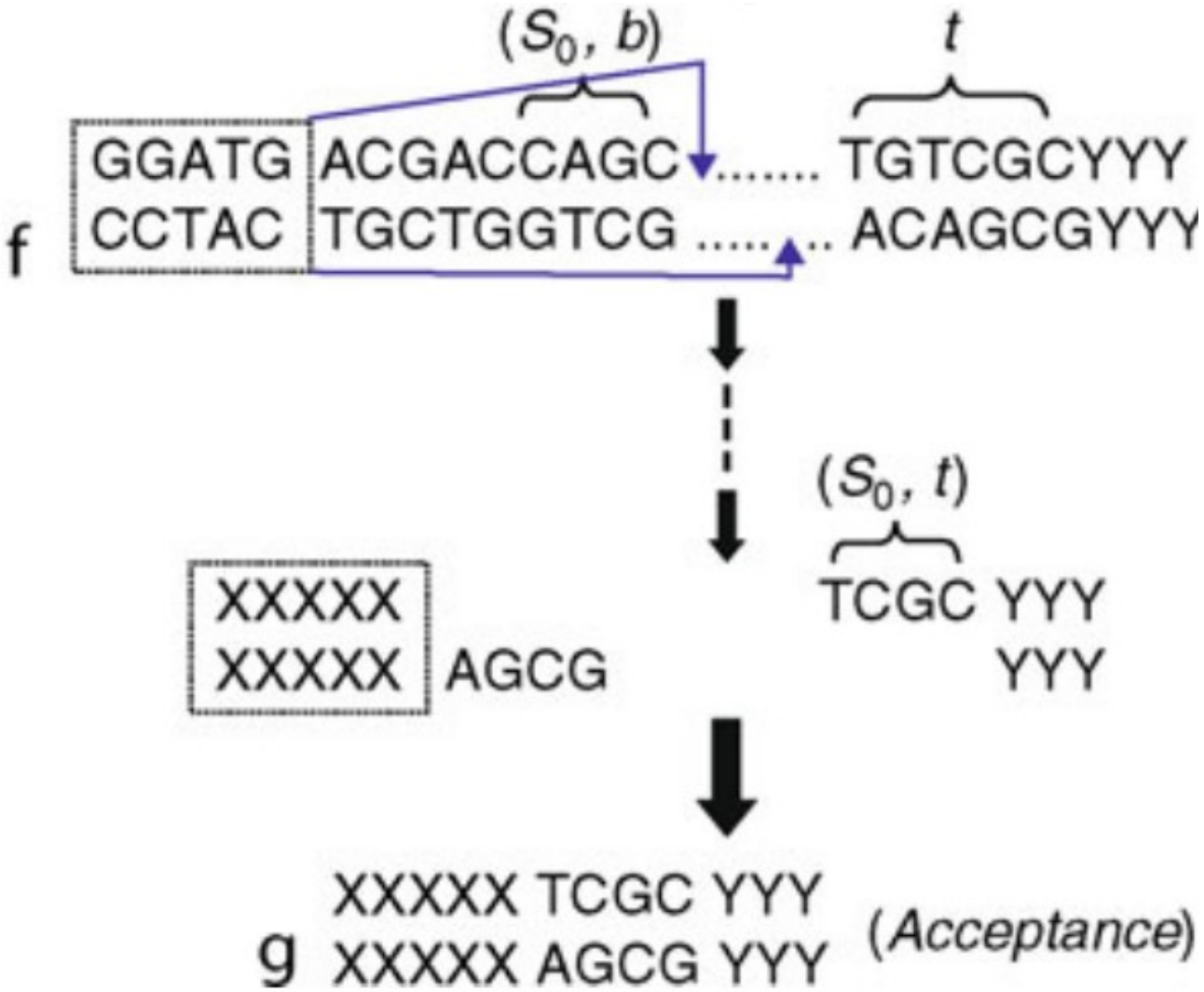


Implementing DFA Computing process of input w=ab(t)



Implementing DFA Computing process of input w=ab(t)




(S_0, b)
t
GGATG ACGACCAGC..... TGTCGCYYY
f CCTAC TGCTGGTCG ..... ACAGCGYYY
(S_0, t)
XXXXX TCGC YYY
XXXXX AGCG YYY
XXXXX TCGC YYY
g XXXXX AGCG YYY
(Acceptance)

Application to Drug Delivery

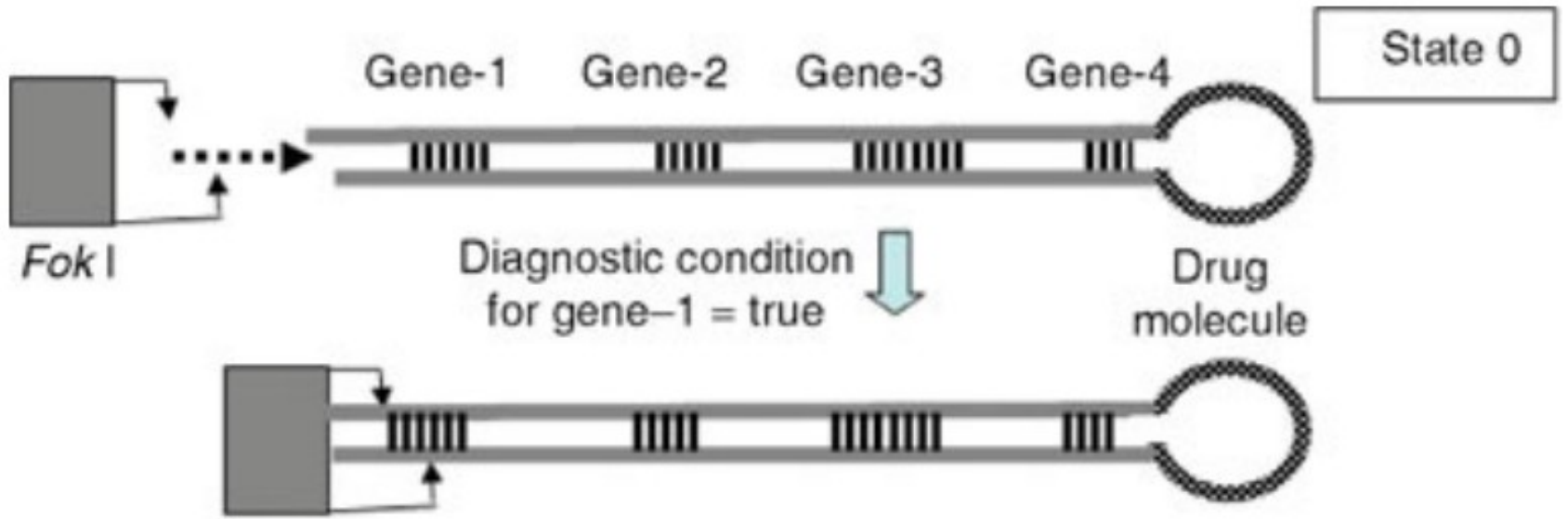
- Carrying out a diagnosis in vivo
- Demonstrated experimentaly the feasibility of logical control of gene expression for drug delivery
- In order to carry out a medical diagnosis, it is necessary to achieve in vivo checking of a medical statement such as “if a certain series of diagnostic conditions are true, then the drug is administered.”
- Such an *if–then* mechanism is critically essential in computing models and in diagnostic process as well
- It is assumed that each condition is described in the form:
  - a certain type of gene expression is regulated by either high or low level of concentrations of particular indicator mRNA for the gene

Diagnostic molecular automaton

- Suppose that a certain disease involves four conditions of gene expression to be checked before administering a drug for the disease,
- Each gene-i (i=1, …, 4) as its label is encoded into a structured molecule with a hairpin in which the drug is enclosed
- The diagnostic rule:
  - if the conditions for gene-1 through gene-4 are all true, then the drug is released

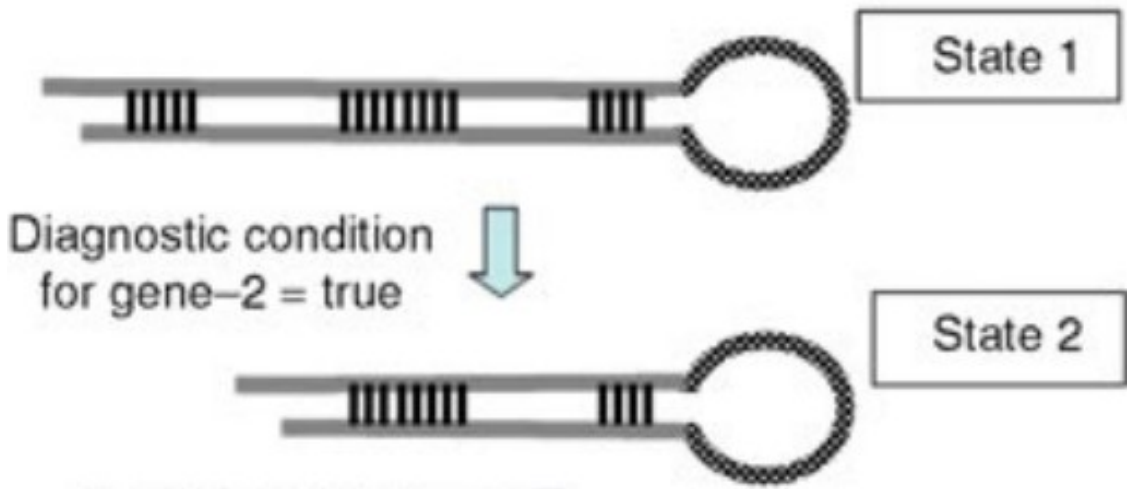
DNA automaton for logical control of drug delivery

At the initial state (State 0), a hairpin molecule (computation molecule) is designed so that a hairpin may maintain the drug molecule that is guarded by four specific sequences gene-i (i = 1,. . .,4) each of which corresponds to one of four conjunctive logical statements in a diagnostic rule for the disease.

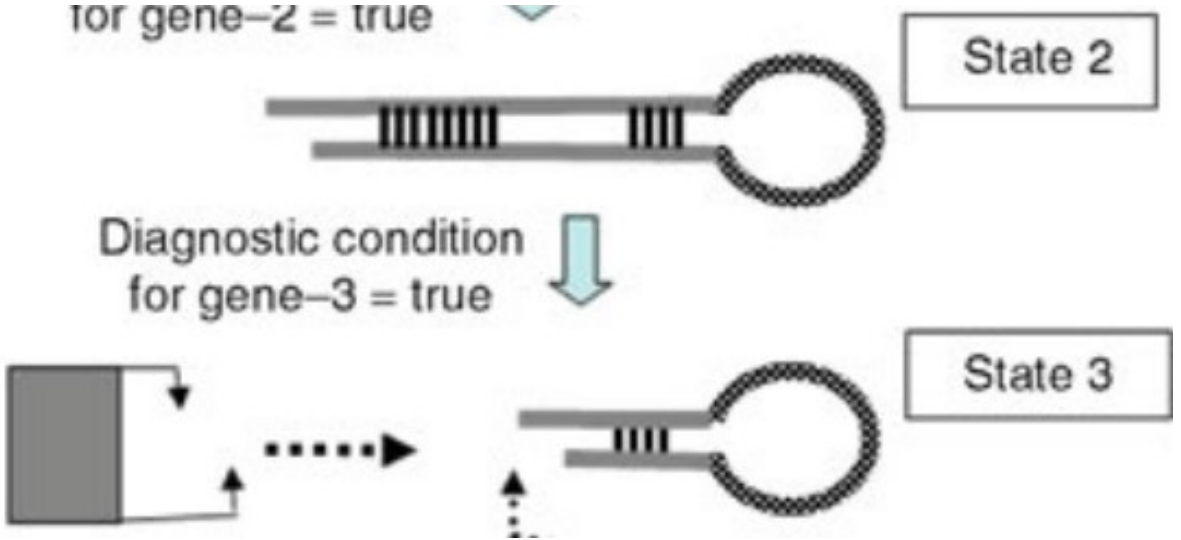


DNA automaton for logical control of drug delivery

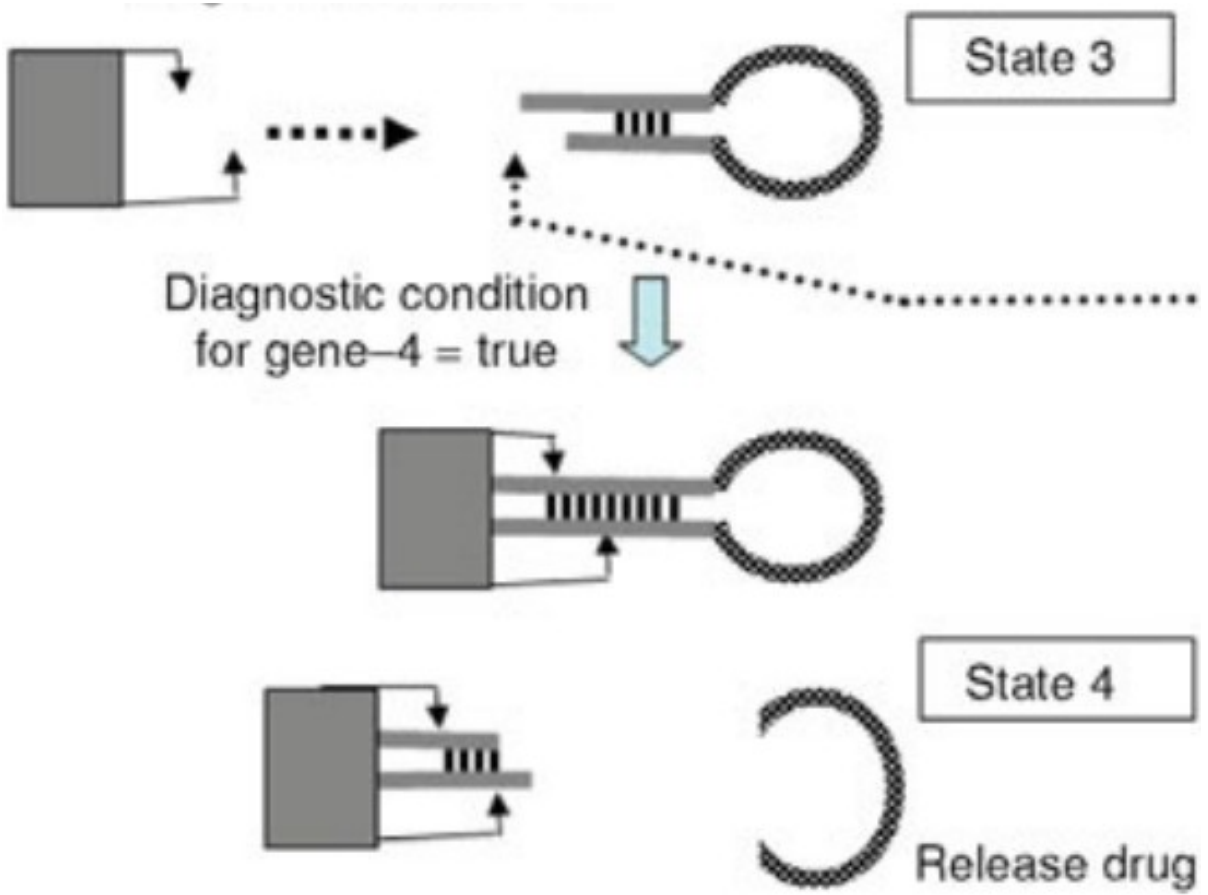
Each State (i-1) is associated with a collection of molecules, and in the presence of the indicator molecule for gene-i, the collection can produce a specific molecule (bridge molecule) which is necessary for cleaving the guarded gene-i, resulting in a successful transition to State i.



DNA automaton for logical control of drug delivery



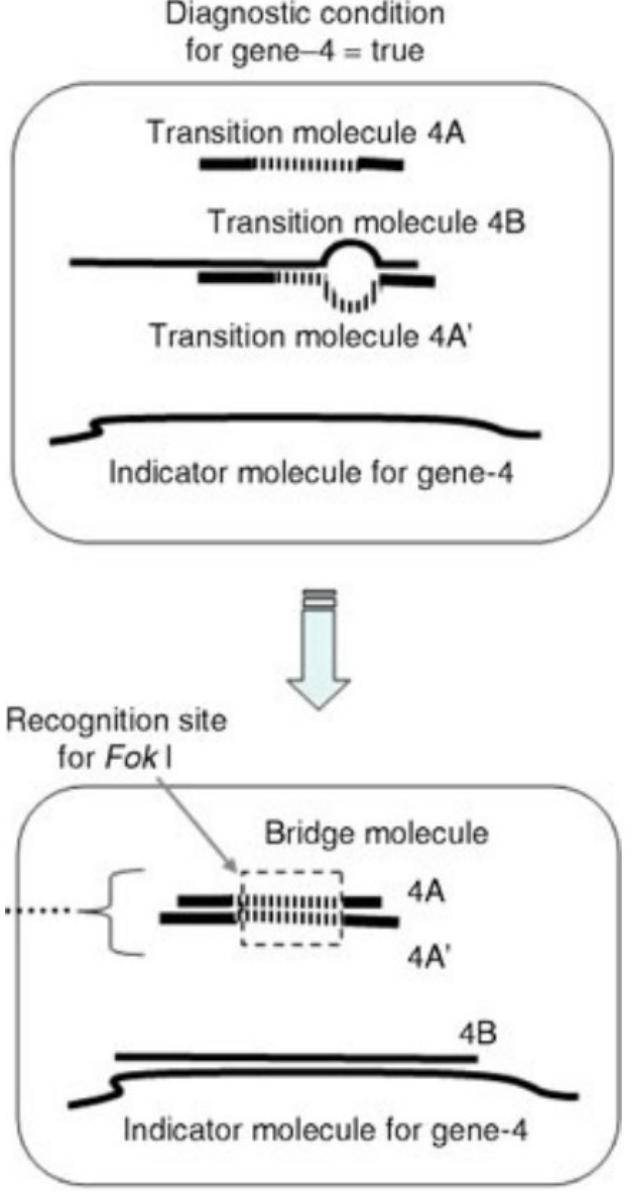
DNA automaton for logical control of drug delivery



Generating bridge molecule

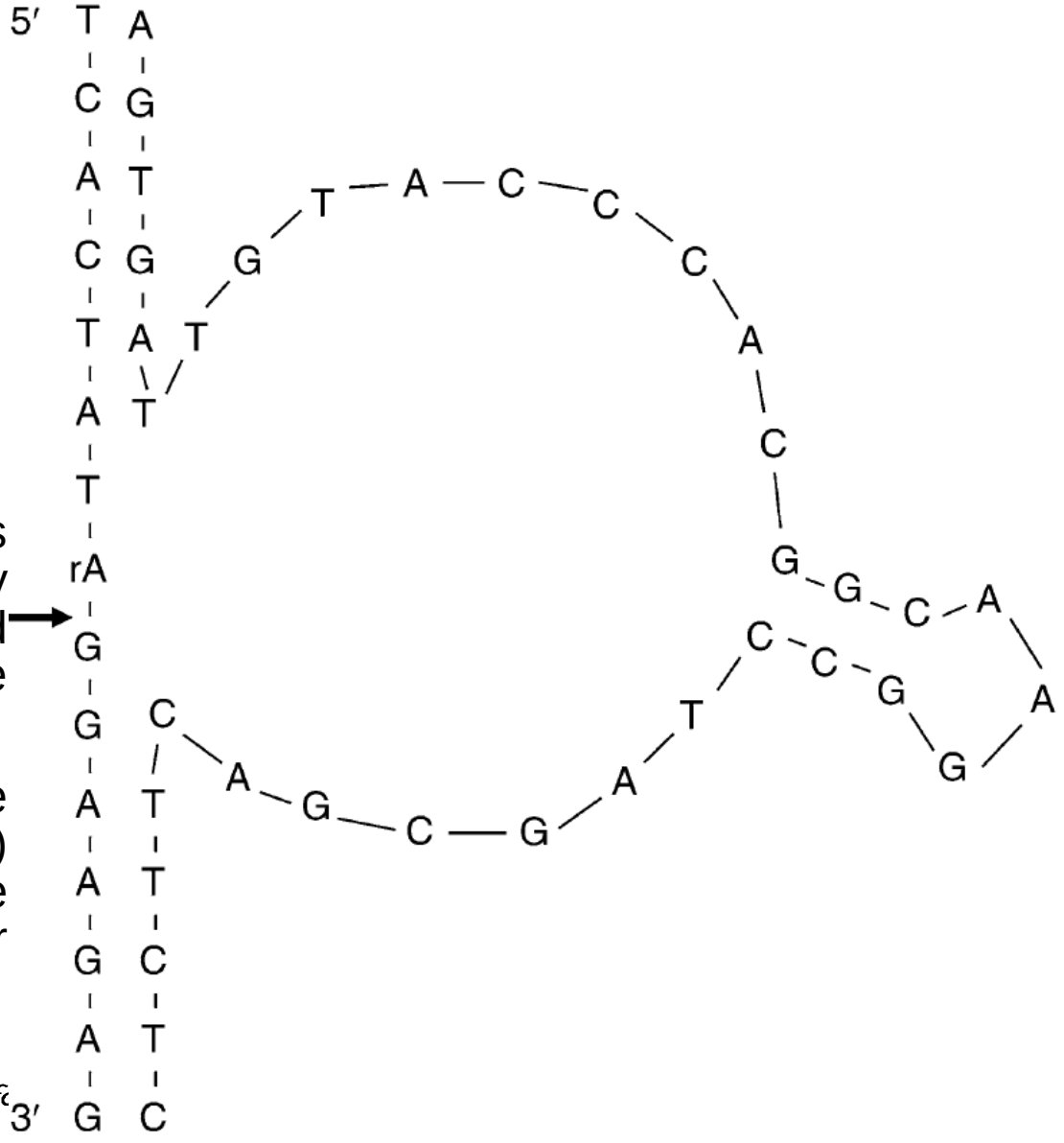
- A team of molecules needed for a transition from State 3 to State 4 comprises transition molecules 4A, 4B, and 4A0.

- In the presence of the indicator molecule for gene-4, members in the team can be relocated into the new formation in which a bridge molecule (involving the recognition site of FokI) is created to contribute to a transition to State 4.



DNAzymes

- Enzyme made of DNA
  - a DNA molecule that has some catalytic function
- This example
  - The strand forming a loop structure is the DNAzyme that partially hybridizes with the target strand and cleaves it at the site shown by the arrow
  - The base denoted as rA is a single ribonucleotide (RNA nucleotide) inserted in the target strand. The enzyme cuts the phosphodiester bond at this site

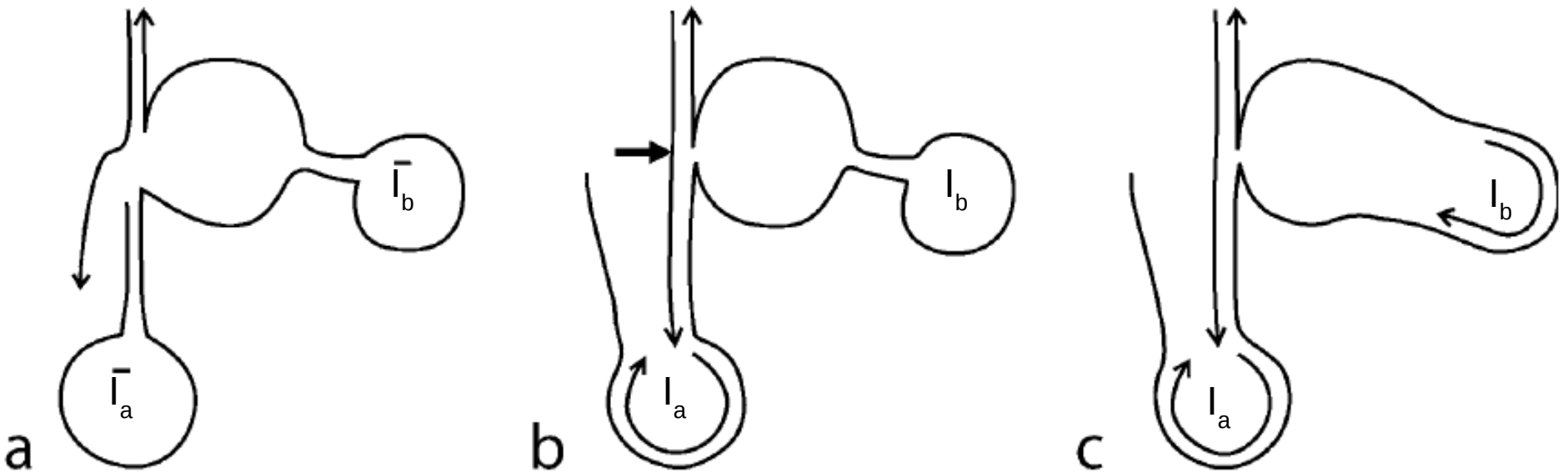


DNAzymes

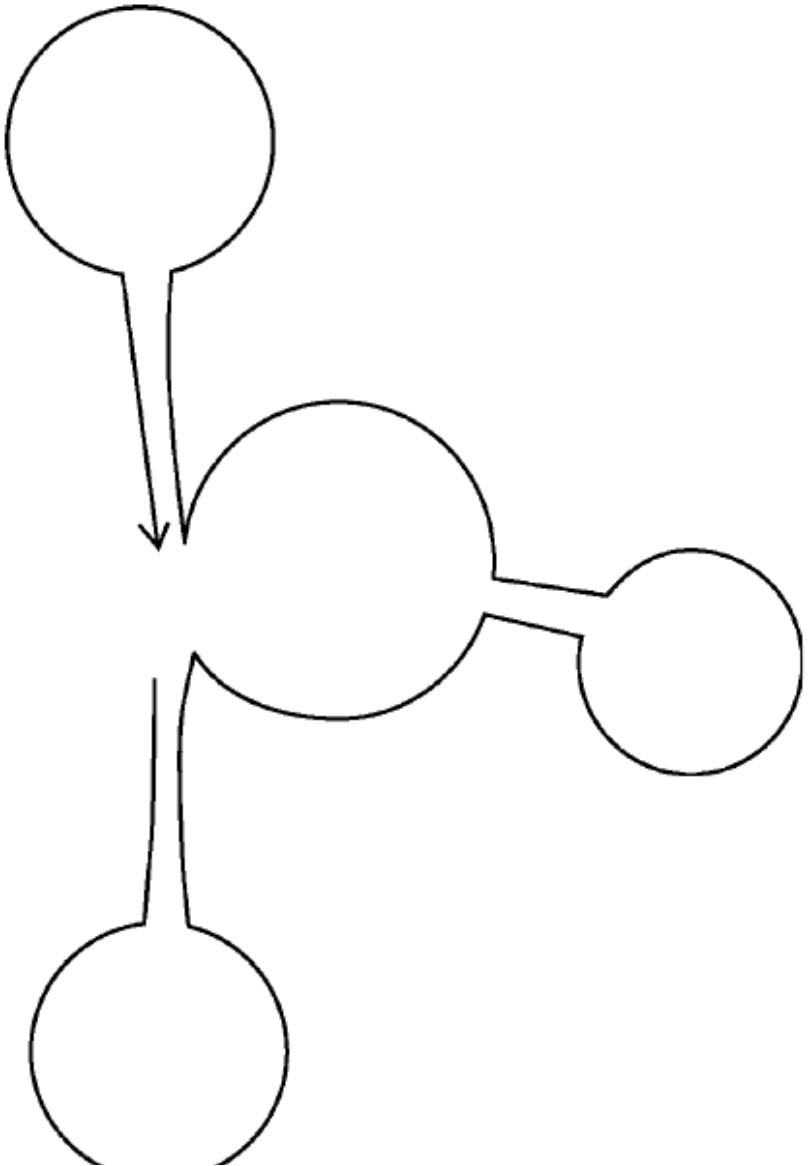
- After the target strand is cleaved, the partially double-stranded structure becomes unstable and the two pieces that arise from the target strand come apart.
- They can be regarded as outputs of the logic gate.
- If a fluorescent group and a quencher group are attached at the terminals of the target strand, an increase of fluorescence is observed when the strand is cleaved.

Implementing logic gates with DNAzymes

The schematic figure of the gate A∧¬B. The DNAzyme is functional only if A is present and B is not.



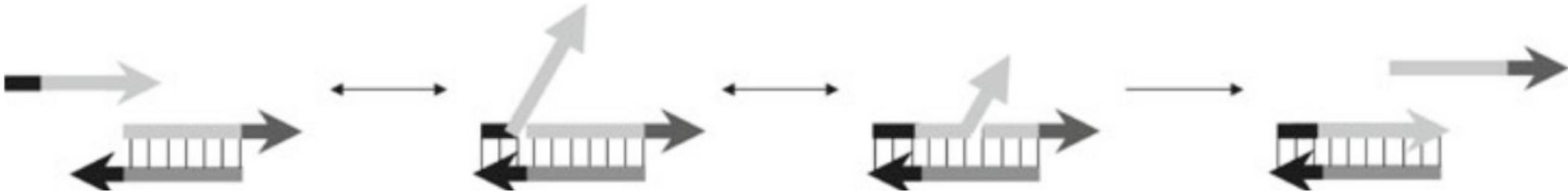
A schematic figure of a three-literal (



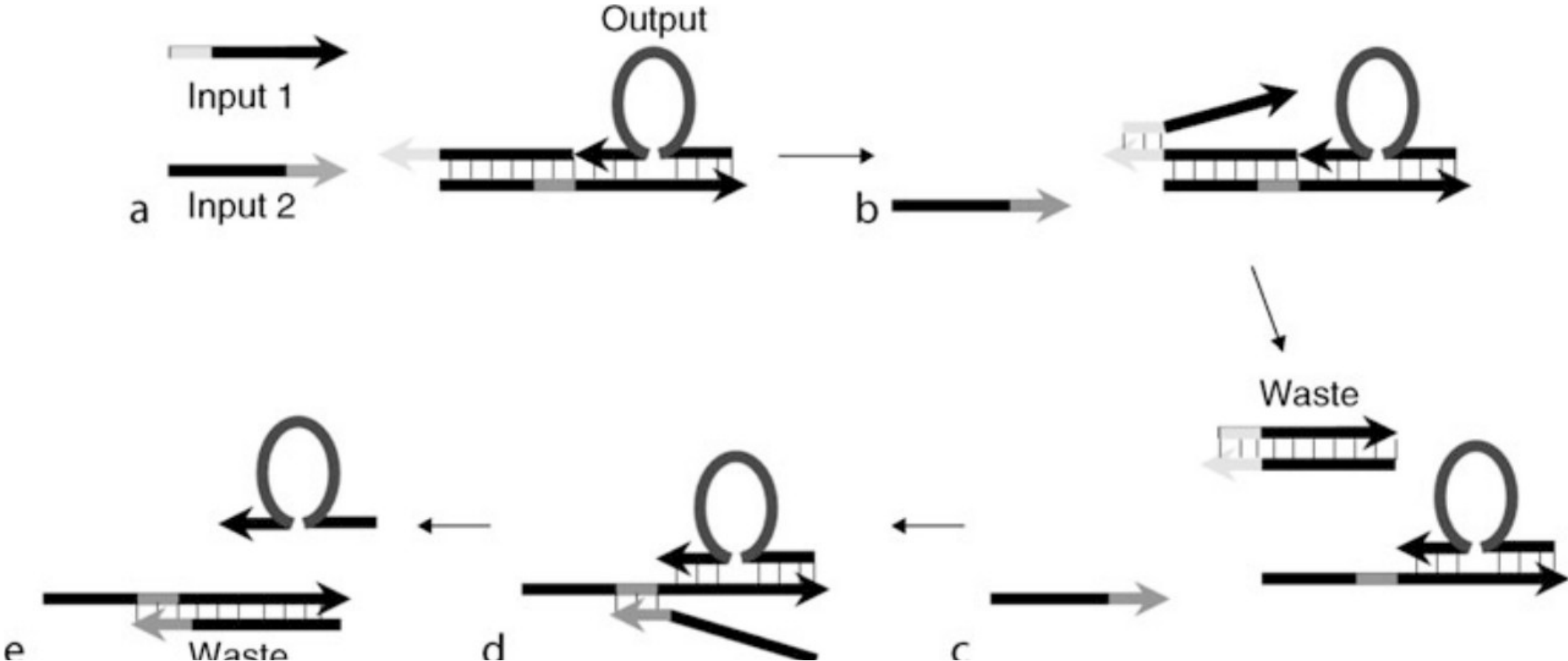
Strand Displacements

- Many bio-molecular machineries and logic gates utilize various restriction enzymes or DNAzymes to control their state transitions.
- Enzyme-based logic gates have a disadvantage in that it is difficult to cascade from one gate to another one because of design restrictions.
- We show here enzyme-free logic gates
- The driving force is a branch migration rather than enzymatic reaction

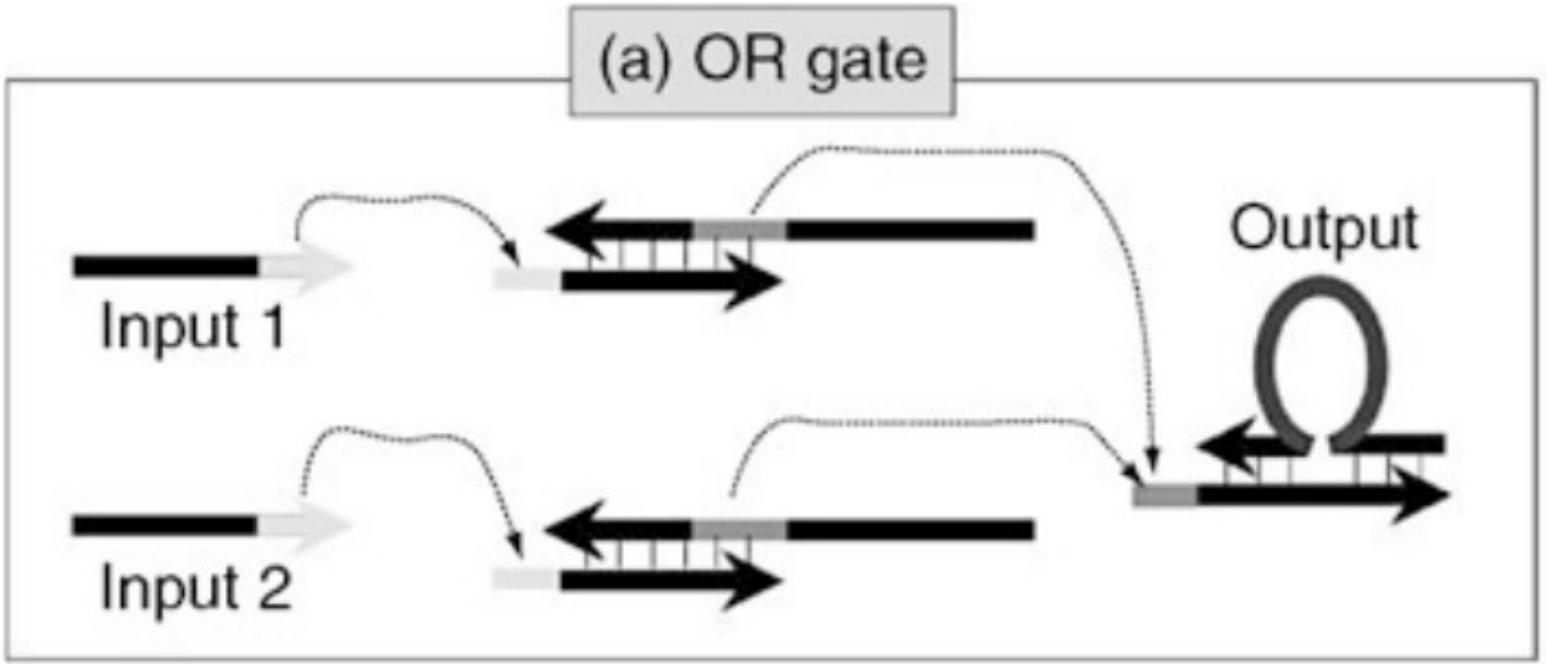
Branch migration



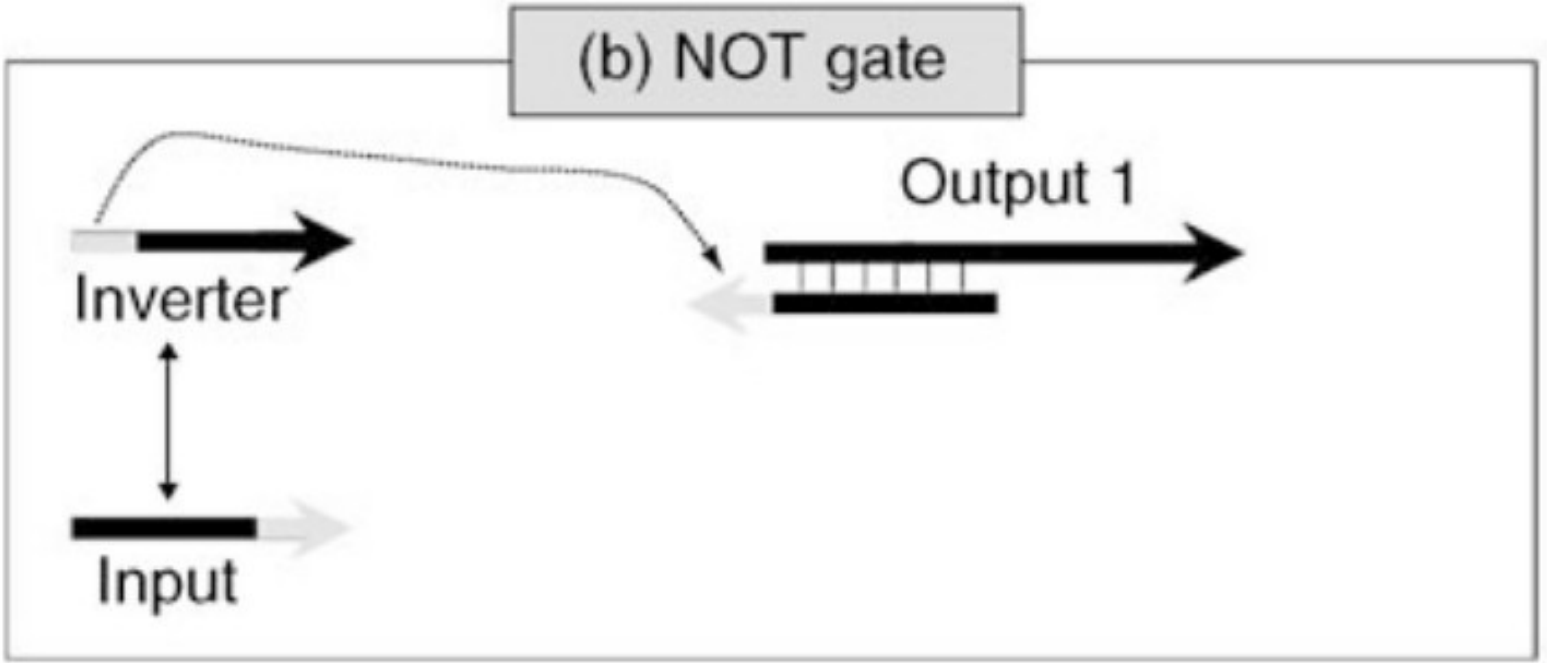
Branch migration – AND gate



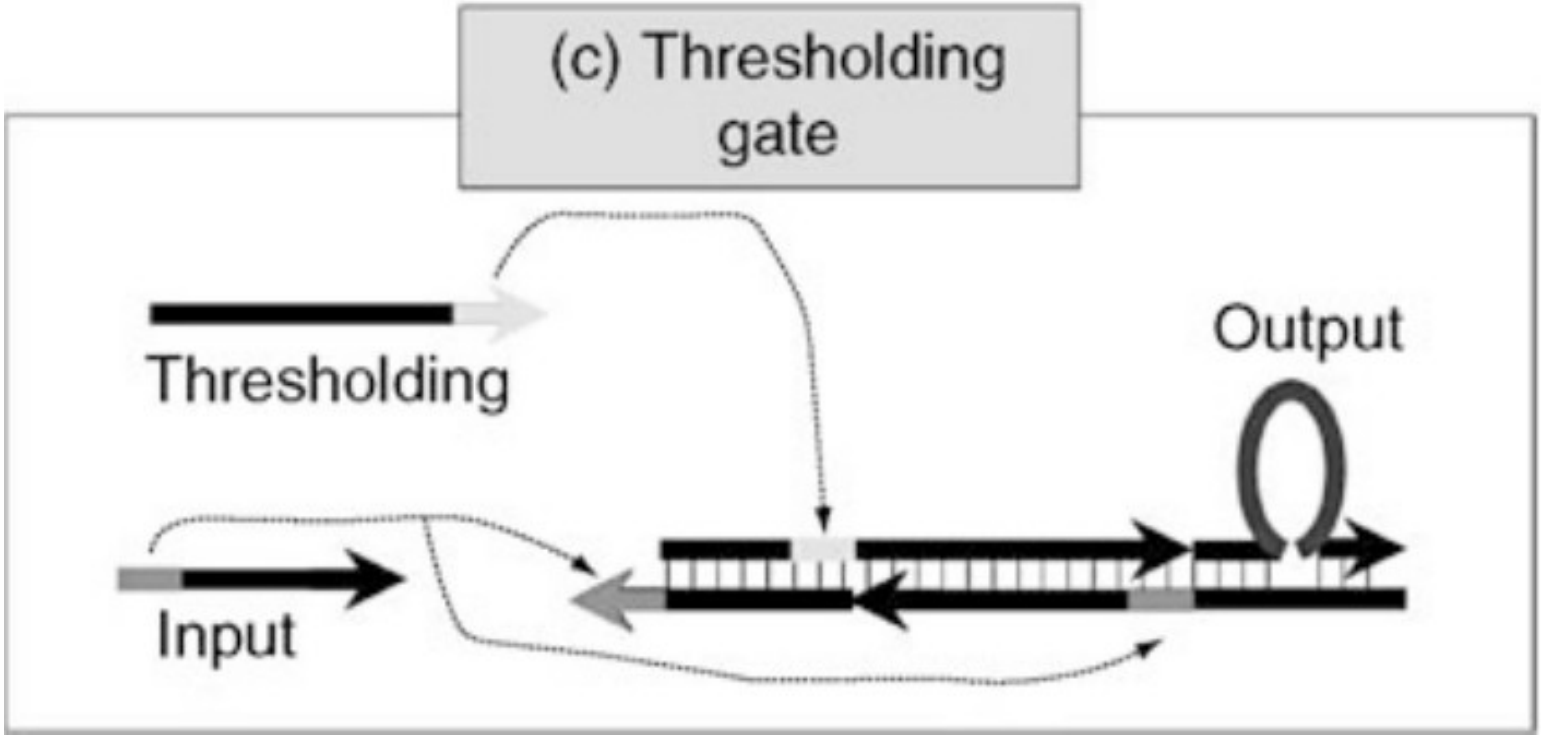
Branch migration – OR gate



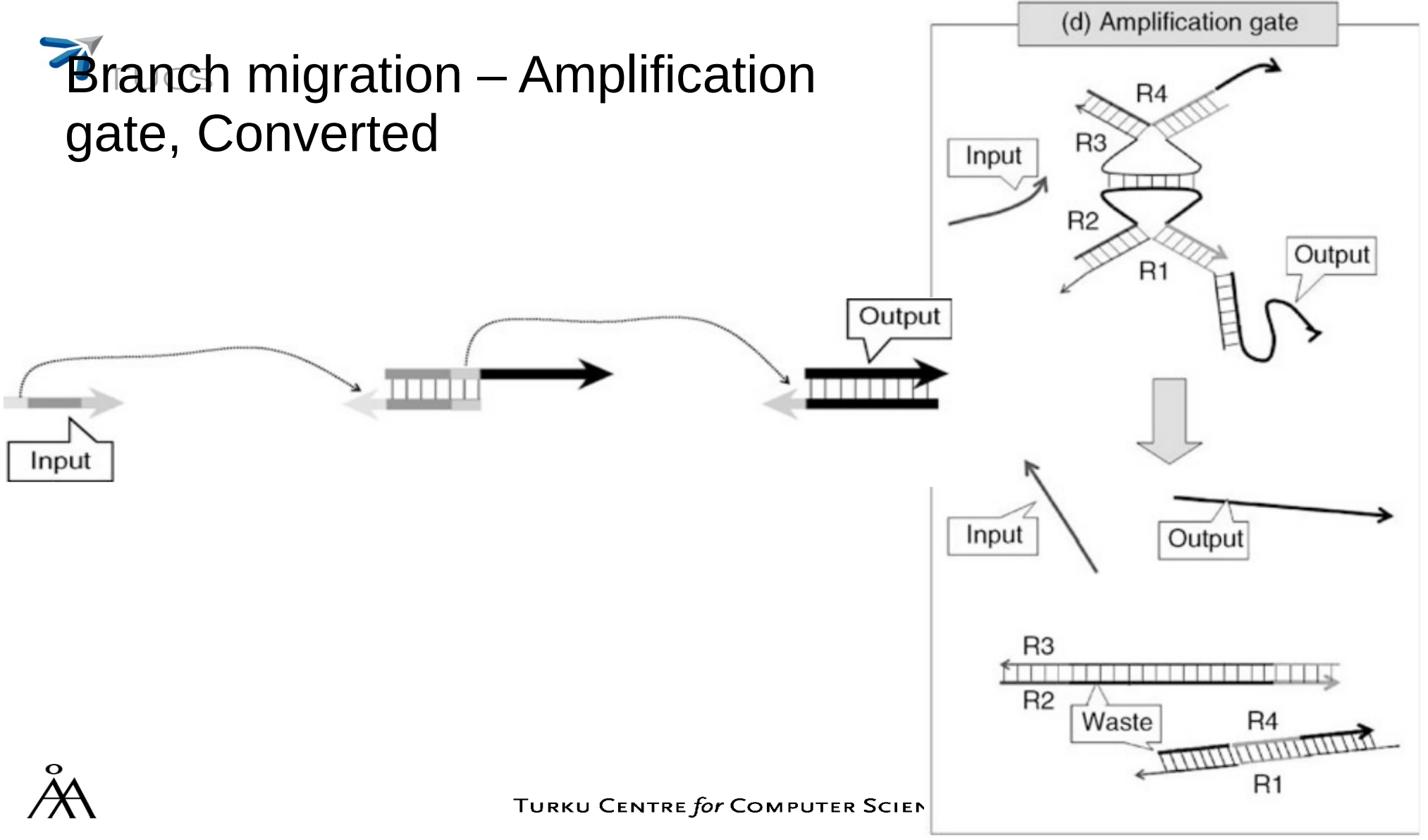
Branch migration – NOT gate



Branch migration – Threschold gate



Branch migration – Amplification gate, Converted



DNA Comparator