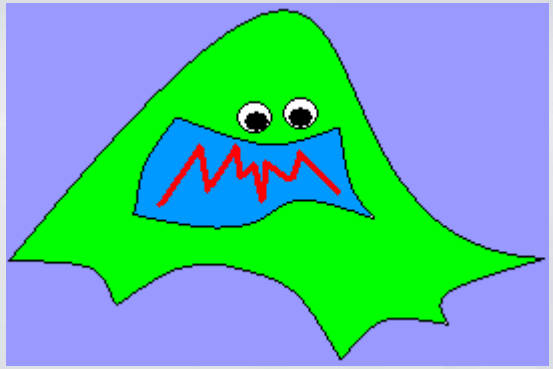


Cut & Paste machine was born

Andrew Kuznetsov

Institute of Biology III, Freiburg University, Germany

{Brains Phenotypes and Insects' International Nerdweek}
Puebla, Pue. Mex. 6-12 March 2005

Introduction

Computation by

- digital computers
- Fredkin gates
- billiard-ball collisions
- cellular automata
- neural networks
- enzymes operating on a polymer chain

Hypercomputation

DNA-as-string

- Bennet
- Shapiro

Splicing-as-combinatorial-optimization

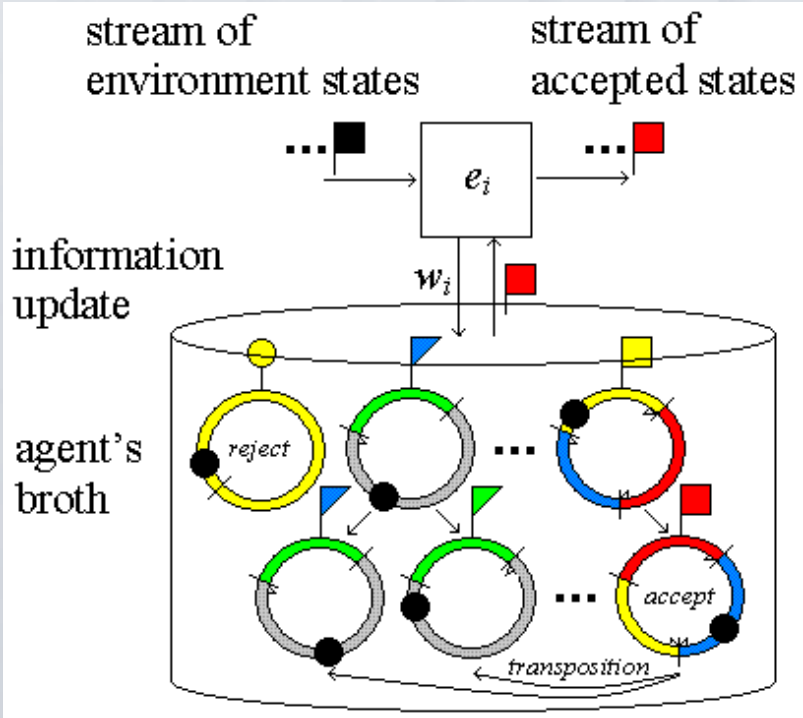
- Head
- Adleman
- Landweber and Kari

‘Molecular tape head’

An abstraction

Description

The *Cut&Paste machine* a finite set of non-deterministic cut-paste agents, which act in parallel on their own finite tapes, communicate by the transpositions of the tape, interact with the environment to compare the output state, and accept, reject, or run in a loop to fit to the environment



Agent algorithm

A* = “On word *w*:

1. Scan the tape to be sure that it contains at least two matches. If not, reject.
2. Cut at the matching sites and arbitrarily paste the tape’s fragments.
3. Take the output state according the new tape.
4. Check it with the state of environment. If satisfy, accept; otherwise loop.”

Definition

A Cut & Paste machine is the construct $\boldsymbol{M} = (\Sigma, O, E, A, \zeta)$, where

1. Σ is a finite alphabet, I and T are nonempty sets, such that I is the 'oracle' language over Σ and T is the set of finite strings (tapes) over Σ;
2. $O = \{ct, pt, sl, tn\}$ is a finite set of operators on T, here ct, pt, sl, tn denotes cut, paste, select and transpose (copy) respectively;
3. E is an infinite set of environment states;
4. A is a finite set of agents, such as $\boldsymbol{A} = (T, S, \delta)$, where

- S is a large finite set of output states, including sets: B, F, R, where B is the subset of initial states, F is the subset of final states, R is the subset of reject states;
- δ is the transition function, which according to the algorithm A* for each word $w_i \in I$ at the input of agent $a_j \in A$ after operations $\{ct, pt\}$ on the tape $t_j \in T$ assigns an output state $s_j \in S$ by the mapping $f: T \rightarrow S$. The state s_j that is compared with the current environment state $e_i \in E$. Formally this can be expressed as $\delta(w, t) = (ct, pt, s, e)$;

5. ζ is the super-transition function, which generates the transposition $\{tn\}$ of the tape $t_j \in T$ from the agent $a_j \in A$ in the state $s_j \in F$ to other agents $a^+ \in A$ after the selection event $\{sl\}$, assigns the new initial states $n^+ \in B$, and finally delivers the new word w_{i+1}, or formally $\zeta(e, s) = (sl, tn, n^+, w)$.

The computation

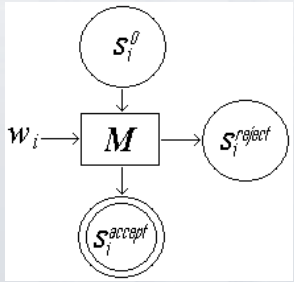
- Computation associated with *Cut&Paste machine* is the shuffling of tapes from the initial set T_0 until an accept state is reached – an *adaptation*

- In general, the computation never ends, because the environment changes permanently; if it happens, the case, called as a *catastrophe*, leads to a transposition, generates a super-transition from the accept-state to the set of new initial-states, and brings a new generative word

- A progression of adaptations and catastrophes – an *evolution*

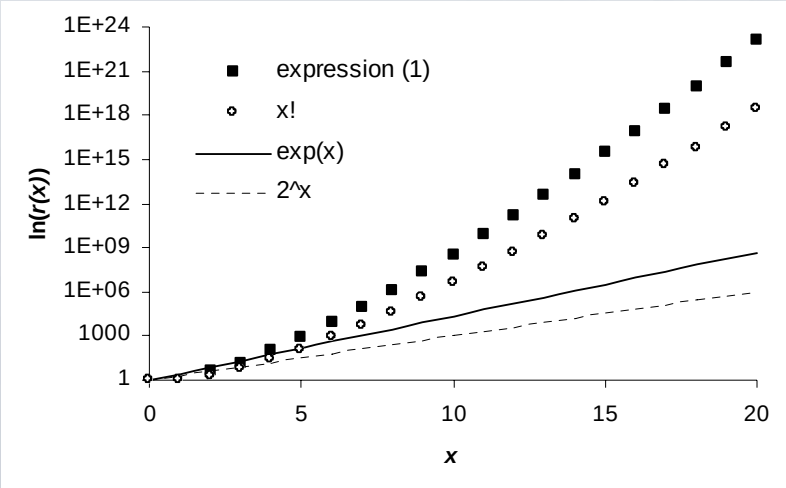
An analysis

Adaptation

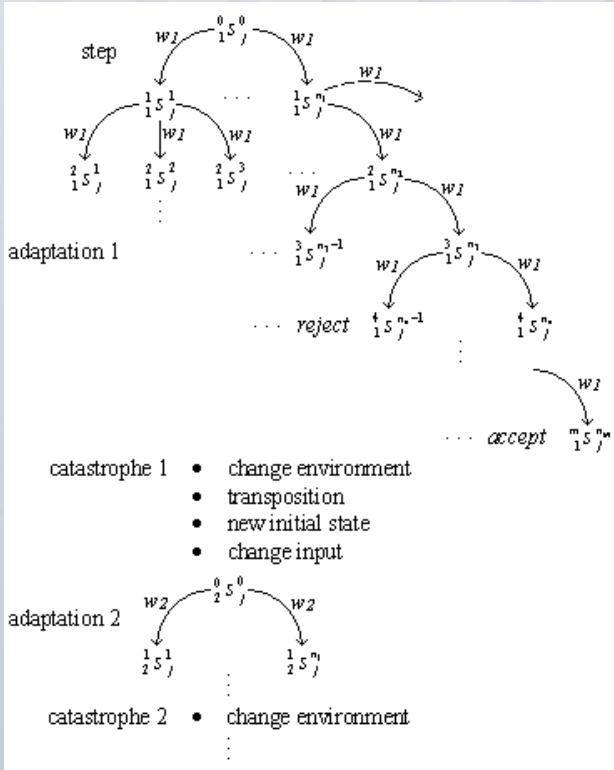


Combinatorial formula (1)

$$\begin{cases} r_0 = r_1 = 0, \\ r_x = 2^x * (x-1)!, \; x \geq 2 \end{cases} \quad (1)$$



Combinatorial power of expression (1)



Nondeterministic computation

Example

CPM computation in winning branch

Language notations:

~,<,(– strings, cut before open brackets;

- boundary symbol

Example 1. Adaptation without transposition:

environment '<~~>', word '<'

1.	<~~>	environment
2.	<	word
3.	#~<~<~<~#	tape_tick_1
4.	#~<~~><~#	tape_tick_2
5.	<~~>	accept

Example 2. Two adaptations with one transposition:

environment_1 '<~(~>', word_1 '<',

environment_2 '<~~~>', word_2 '('

1.	<~(~>	environment_1
2.	<	word_1
3.	#~(<~<~)<~#	tape_tick_1.1
4.	#~(<~(~><~#	tape_tick_1.2
5.	<~(~>	accept_1
6.	<~~~>	environment_2
7.	#~(<~<~)<~##~(<~(~><~#	transposition
8.	(	word_2
9.	#~(<~<~)<~~(<~(~><~#	tape_tick_2.1
10.	#~(<~<~)<~~~>)(~><~#	tape_tick_2.2
11.	<~~~>	accept_2

Requirements to CPM

- definition, description, and refinement of CPM
- investigation of CPM behavior: a sample run of CPM on input in the environment
- variants of CPM: isomorphism, robustness
- comparison of CPM with TM and others machines: decidability, halting problem
  - proof of equivalence in power
  - simulate one by the other

Two machines are equivalent if they recognize the same language

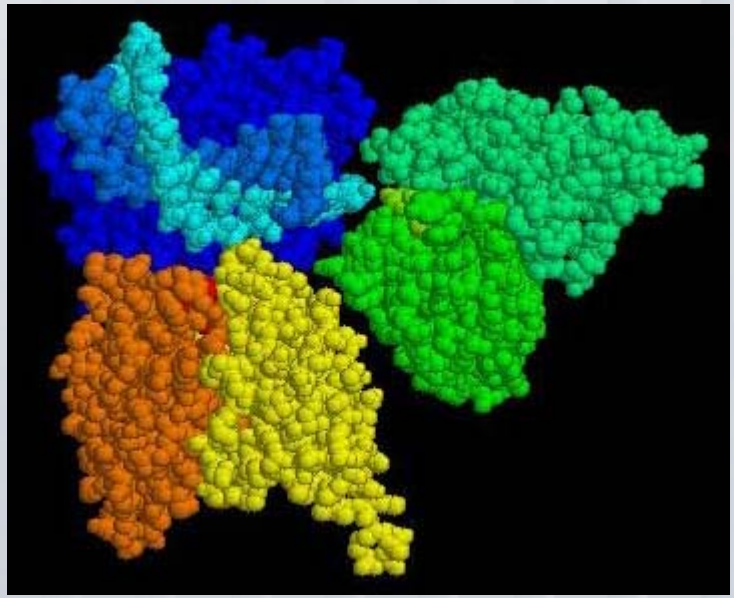
- implementation on
  - conventional computer (special case)
  - bio-molecules
  - living cells

Modeling

- **twin-shuffle language** to correctly represent DNA strands
- **π-calculus** to describe the distributed concurrent computation by bio-molecules
- **BioSPI-simulator** to design the bio-computer by reaction rules in the real chemistry

A real world

- IGNAF comprises NucA non-specific endonuclease from cyanobacterium *Anabaena sp*. and 4-4-20 scFv mouse single-chain antibody to fluorescein
- To increase the robustness of the enzyme we develop two versions of IGNAF (α) by optimisation NucA-domain via epPCR, shuffling, PHD, and (β) NucA split domain activated by selfassembling
- Input: oligonucleotides or PNA labelled by fluorescein
- IGNAF binds to flourescein and cuts DNA at the target DNA



Implementation

in vitro:

- *catalytical approach* - nuclease as a catalytic with substrate turnover above T_m
- *robust approach* - carrying out repeated hybridizations and cleavage reactions

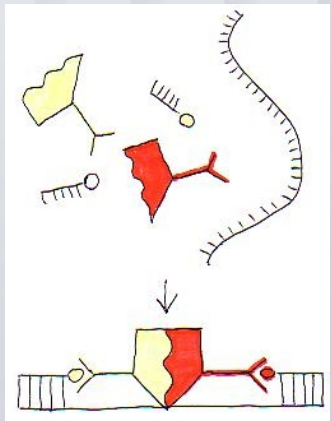
in vivo:

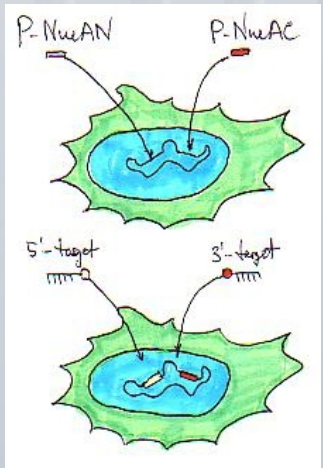
1. preinstallation of *ignaf*-transgene into living cells
2. introduction of gene markers (input)
3. activation of IGNAF nuclease at the target site

Computation in loop

1. introduction of input molecules into the system
2. target cleavage
3. arbitrary ligation

Bio-computation and Nanotechnology





Questions and conclusion

- No any computational definition of life; no minimal set of conditions needed for life to exist
- We try to find a minimal set to define life and to create a small tool to drive life
- The questions to minimal life :
  - How much the alphabet?
  - How long should words and tapes be?
  - How many tapes does it require?
  - What about rules for the input words to effective search in the environment space and to creative design?
  - What about super computing power and halting problem?
- Is Cut&Paste computer beyond Turing machine?
- If the Turing machine is a stupid clerk, the Cut&Paste computer is like a chaotic hacker's community