Genome as a functional program

S.V. Kozyrev, Steklov Mathematical Institute

Approach *"genome as a program"* using functional programming — genome is a functional program (lambda–term).

Functional programming — parallelism, simple system of states allows easy modification of programs (error control).

Comparison to biology — parallelism of processes in cells and in evolution, random modifications of genetic program in evolution usually do not break the program immediately.

Darwinian evolution — generation of programs by data — machine learning.
Learning problems for functional programming.

Gibbs distribution and scaling: genomics, Zipf's law

Scaling in sizes of families of paralogous genes, scaling in metabolic networks and networks of interacting genes (scale free graphs).

E.V.Koonin: genome is a "*gas of interacting genes*", scaling should be related to Gibbs distribution for this model.

Yu.I.Manin: model of statistical mechanics with Hamiltonian equal to Kolmogorov complexity ("*Complexity as Energy*") — Gibbs distribution should give the Zipf's scaling law for distribution of words in texts.

S.K.: These two approaches can be unified if biological evolution is a model of temperature learning with regularization equal to estimate for Kolmogorov complexity.

E. V. Koonin, The Logic of Chance: The Nature and Origin of Biological Evolution, FT Press, 2012.

Y. I. Manin, Complexity vs energy: theory of computation and theoretical physics, Journal of Physics: Conference Series 532 (2014) 012018. arXiv:1302.6695

S.V. Kozyrev, Biology as a constructive physics, p-Adic Numbers, Ultrametric Analysis and Applications, 10:4 (2018), 305–311. arXiv:1804.10518

S.V. Kozyrev, Learning problem for functional programming and model of biological evolution, p-Adic Numbers, Ultrametric Analysis and Applications, 12:2 (2020), 112–122.

S.V. Kozyrev, Genome as a functional program, arXiv:2006.09980

Biology:

Molecules are linear polymers (proteins and nucleic acids) — strings of symbols, state of a system — set of strings with multiplicity (multistring).

Chemical reactions — transformations of multistrings local in substrings (gluing, cuttings, substitutions, duplications, other multistring editing operations). Physical transformations (transfer of molecules) — changes of multiplicities of strings in multistrings. Generalization of Chomsky context free grammars to multistrings.

Genome — set of genes, each gene defines a transformation of multistrings. Each gene is represented by a string, genome is a multistring.

Genes as transformations operate in parallel.

Biological evolution — transformation of genomes as multistrings by a set of genome editing operations.

Genome as a program

J.Backus systems of functional programming. Functions (not everywhere defined) act in the set of objects (map objects to objects). Functional forms (in particular composition of functions $\circ$) create new functions starting from existing ones.

J. Backus, Can Programming Be Liberated from the von Neumann Style? A Functional Style and its Algebra of Programs, Comm. ACM 21 (8), 613–641 (1978).

Genome is a functional program defined recursively by a list of functions $G=[g_1,\dots,g_n]$, each of functions g_k is a transformation of multistrings $S\to S$

$$\widetilde{G}=\widetilde{G}\circ G=[\widetilde{G}\circ g_1,\dots,\widetilde{G}\circ g_n]. \tag{1}$$

Here the space of objects is the space S of multistrings.
g_k – genes, G – genome as a list of genes, $\widetilde{G}$ – genome as a program.

Gene g_k as a function is not everywhere defined in S and the corresponding map is multivalued (application of function g_k to object v is represented by lambda–term where reduction can be made in multiple ways). In particular g_k may cut a string at the position of substring uv

$$u'uvv' \mapsto u'u + vv',$$

string may contain several such substrings and g_k can act to different strings in a multistring.

List $G = [g_1, \dots, g_n]$ of genes is a multivalued function $S \to S$: any function g_k in the list can be applied to an object $v \in S$.

The set S of multistrings is the space of sets of biological sequences (molecules), g_k is a gene which encodes protein performing transformation of biological sequences (chemical reaction). Also we have operations describing transfer of molecules which change multiplicities of some strings in a multistring (we consider these operations as some "genes" g_j in G).

The set $G = [g_1, \dots, g_n]$ is a genome (list of genes) which defines the program $\widetilde{G}$ (multivalued map $S \to S$) recursively (the program $\widetilde{G}$ is a fixed point of the genome G as a lambda–term).

Any gene g_k (which defines a map $S \to S$) is encoded by some biological sequence, i.e. gene g_k is a string and a genome G is a multistring in S.

Metabolic network as a reduction graph

Let $v_0 \in S$ be multistring. Let us define a graph $\Gamma_{\widetilde{G}}(v_0)$ for the program $\widetilde{G}$ (which corresponds to reduction graph in "lazy" evaluation strategy):

Step 0) We start from vertex v_0.

Step 1) Let us apply G to v_0, any $g_k \in G$ can be applied to v_0 (g_k itself is multivalued). Let us include to the graph all vertices obtained from v_0 by multivalued map G (we identify vertices which coincide as multistrings), the obtained vertices are connected to v_0 by edges.

Step 2 etc.) By recursion let us apply to obtained at the previous step vertices multivalued map G. Let us include to the graph $\Gamma_{\widetilde{G}}(v_0)$ all vertices and edges obtained in this way (again, we identify vertices which coincide as multistrings). Iteration of the process gives graph $\Gamma_{\widetilde{G}}(v_0)$.

Some genes g_k in G correspond to "physical" transfer operations which change multiplicities of some strings in a multistring. These operations allow to close metabolic cycles in the graph. In particular this allows to unify some graphs $\Gamma_{\widetilde{G}}(v_0)$ with some different v_0 to a single graph. We denote this "large" graph by $\Gamma_{\widetilde{G}}$. This graph unifies graphs $\Gamma_{\widetilde{G}}(v_0)$ with different "reasonable" v_0 and corresponds to metabolic network for the genome G.

Let us put in correspondence to any gene g in the genome G the pair of non-negative numbers $r_+(g)$, $r_-(g)$ — transition rates of corresponding reaction. These rates define a Markovian random walk in the graph $\Gamma_{\widetilde{G}}$ with transition rates along and against edges equal to $r_+(g)$, $r_-(g)$ (where edges correspond to action of genes), and define the corresponding system of kinetic equations for distribution functions on vertices of the graph.

Let us consider some linear functional $A(f)$ of distribution $f(v)$ on vertices of the graph. In particular we consider the functional of current along the edge v_1v_2 with rates r_+ and r_- along and against the edge (from v_1 to v_2 and against) which equals to $r_+f(v_1)-r_-f(v_2)$. Different edges v_1v_2 corresponding to the same gene may be related to the same chemical reaction (in particular with different values of reagents). In this case to obtain the complete current one has to sum up values $r_+f(v_1)-r_-f(v_2)$ over all such edges v_1v_2.

Let us assume that for the system of kinetic equations for random walk under consideration there exists a unique stationary state $f_{\widetilde{G}}$, moreover the solution of the system tends to this stationary state. We will discuss functionals (in particular currents) $A(f_{\widetilde{G}})$ in this stationary state.

Remarks

The program $\widetilde{G}$ is highly parallel. Correctness of operation of metabolic networks is related to Church–Rosser property for lambda–calculus (in different order of application of genes one can obtain the desired result).

The program $\widetilde{G}$ for a genome loops — this describes cycles in the metabolic network $\Gamma_{\widetilde{G}}$.

The set of numbers $r_+(g_k)$, $r_-(g_k)$ — rates of transitions along edges of the graph $\Gamma_{\widetilde{G}}$ (reaction and transfer rates) and the corresponding stationary state $f_{\widetilde{G}}$ — correspond to a state for a genome as program $\widetilde{G}$.

Gene regulation — changing values $r_+(g)$, $r_-(g)$ we will change the stationary state $f_{\widetilde{G}}$ and contributions to the functional $A(f_{\widetilde{G}})$ from different metabolic pathways.

Physically gene regulation works by enhancers/repressors/promoters (in particular lac operon) which define priority in expression of genes.

Analog in functional programming — monads (computations with effects, modification of states, input/output operations — interaction with the environment).

Another mechanism of gene regulation: epigenetics — genome methylation, histone code, folding of chromatin.

Biological evolution — action of "evolutionary program" $\widetilde{E}$ with "evolution genes" $E=[e_1,\dots,e_m]$ (operations of editing of genomes) defined recursively

$$\widetilde{E}=\widetilde{E}\circ E=[\widetilde{E}\circ e_1,\dots,\widetilde{E}\circ e_m]. \tag{2}$$

Evolution transforms genomes to genomes (as multistrings), transforms rates $r_+(g)$, $r_-(g)$ for genes, the stationary state $f_{\widetilde{G}}$ and functional $A(f_{\widetilde{G}})$.

Darwinian evolution — machine learning (generation of a program by the data). Machine learning was proposed by A.Turing, he also mentioned analogy to Darwinian evolution.

A. M. Turing, Can machines think? Computing Machinery and Intelligence. Mind 49: 433–460 (1950).

Temperature learning

Problem of machine learning — minimization over the space of parameters of the sum of the loss (or risk) functional and the regularization functional

$$H(s) = R(s) + Reg(s) \to \min .$$

Regularization is important to control overfitting (by reducing entropy of the space of parameters s, see VC–theory).

Temperature learning — instead of minimization we compute the statistical sum ($\beta > 0$ is the inverse temperature)

$$Z = \sum_{s} e^{-\beta H(s)}.$$

In the zero temperature limit $\beta \to \infty$ problem of computation of Z becomes the problem of minimization of H (temperature learning becomes standard learning).

Ideas of machine learning in evolution — regularization by estimate of Kolmogorov complexity to control overfitting.

Teleology — explanation of phenomena by their purpose. In evolution theory teleology can be formulated as solvability of the corresponding learning problem.

Mechanism — combination of function and low complexity.

Darwinian evolution as learning problem — minimization of the loss functional (i.e. function) with regularization by complexity — biological systems are mechanisms to perform biological function.

Darwinian evolution as temperature learning — Gibbs distribution. Temperature models in evolution — effective population size is the inverse temperature. Scaling in genomics — Gibbs distribution of interacting gas of genes (Koonin). Zipf's scaling law — complexity as energy (Manin).

Universal scaling in genomics can be explained by universal regularization by complexity in the corresponding learning problems.

Temperature learning for functional programs

Let us consider the evolution program $\widetilde{E}$ of the form (2) with reduction graph $\Gamma_{\widetilde{E}}(G_0)$ where G_0 is the ancestor genome (vertices are descendants).

Let us put in correspondence to action of evolution operation e_k weight (positive number) $K(e_k)$ and to oriented path p between vertices u and v in the graph $\Gamma_{\widetilde{E}}(G_0)$ (path from ancestor to descendant) we put in correspondence the action functional — the sum of weights of edges in the path

$$K_{\widetilde{E}}(p) = \sum_{k \in p: u \to v} K(e_{i_k}). \tag{3}$$

This functional can be considered as the cost of computation along the path p or weighted estimate for Kolmogorov complexity of generation v from u.

Let us define Darwinian evolution as the temperature learning problem with inverse "evolution temperature" β' with statistical sum

$$Z[\widetilde{E}, G_0] = \sum_{G \in \Gamma_{\widetilde{E}}(G_0)} A(f_{\widetilde{G}}) \sum_{p \in \mathrm{Path}\left(\Gamma_{\widetilde{E}}(G_0)\right): G_0 \to G} e^{-\beta' K_{\widetilde{E}}(p)}. \qquad (4)$$

The summation runs over paths p between the ancestor genome G_0 and the descendant genome G, then we sum over descendants G. This statistical sum is concentrated at genomes with large functional $A(f_{\widetilde{G}})$ (for example selection for large current functional).
Summation over paths describes parallelism in evolution (computation of typical functional $A(f_{\widetilde{G}})$ includes summation over paths which describes parallelism in metabolism). Gibbs factor of the action functional reduces the complexity of evolution operations which contribute to the statistical sum of evolutionary program. This corresponds to regularization by complexity as energy.

Nondeterministic algorithm is described by a Nondeterministic Turing Machine (NTM) which at some steps of computation can duplicate and perform several branches of computation (this gives brute-force search).

Programs (1), (2) which describe operation and evolution of genomes are programs for NTM since G and E are multivalued functions and recursive application of multivalued functions generate many branches of computation.

We propose to consider parallelism in biology (parallelism of processes in cells and in evolution) as a manifestation of nondeterministic algorithms. Biological processes correspond to nondeterministic computations and Darwinian evolution is a temperature learning problem for a functional nondeterministic algorithm.

Summary

Model of genome as a functional program (1) defined recursively by the set of genes is proposed, *"life is a fixed point of a genome"*.

Operation of this program is described by distribution $f_{\widetilde{G}}$ for *"interacting gas of genes"*.

"Darwinian evolution by selection is a temperature learning problem" for functional program (2) given by the statistical sum (4). This statistical sum contains a sum over paths of reduction of Gibbs factors of the action functional (3) of *"complexity as energy"* and describes scaling laws in genomics.

Parallelism of processes in cell and in evolution can be described by nondeterministic algorithms.