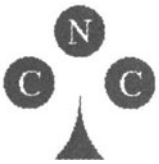
Natural Computing Series



Springer-Verlag Berlin Heidelberg GmbH

Andrzej Ehrenfeucht Tero Harju
Ion Petre David M. Prescott
Grzegorz Rozenberg

Computation in Living Cells

Gene Assembly in Ciliates

With 92 Figures and 2 Tables

Springer

Andrzej Ehrenfeucht

Department of Computer Science
University of Colorado
Boulder, CO 80309-0347, USA
email: andrzej@cs.colorado.edu

Tero Harju

Department of Mathematics
University of Turku
FIN-20014 Turku, Finland
email: harju@utu.fi

Ion Petre

Department of Computer Science
Åbo Akademi University
FIN-20520 Turku, Finland
email: ipetre@abo.fi

David M. Prescott

Department of Molecular, Cellular and
Developmental Biology
University of Colorado
Boulder, CO 80309-0347, USA
email: prescotd@colorado.edu

Grzegorz Rozenberg

Leiden Institute for Advanced Computer Science
Leiden University
Niels Bohrweg 1
2333 CA Leiden, The Netherlands
e-mail: rozenber@liacs.nl

Cataloging-in-Publication Data applied for

A catalog record for this book is available from the Library of Congress.

Bibliographic information published by Die Deutsche Bibliothek

Die Deutsche Bibliothek lists this publication in the Deutsche Nationalbibliografie; detailed bibliographic data is available in the Internet at http://dnb.ddb.de.

ACM Computing Classification (1998): F.1, G.2.3, J.3

ISBN 978-3-642-07401-1 ISBN 978-3-662-06371-2 (eBook)
DOI 10.1007/978-3-662-06371-2

http://www.springer.de

Originally published by Springer-Verlag Berlin Heidelberg New York in 2004
Softcover reprint of the hardcover 1st edition 2004

Cover design: KünkelLopka, Heidelberg
Typesetting: Digital Data supplied by authors

Printed on acid-free paper 45/3142XO - 5 4 3 2 1 0

This book is dedicated to Pat, Eija, Luigia, Gayle, and Maja

Preface

Natural Computing is a research area concerned with computing taking place in nature and with human-designed computing inspired by nature. It is a fast growing, genuinely interdisciplinary field involving, among others, biology and computer science.

The contribution of Natural Computing to computer science is quite significant, and it comes in the period when computer science is undergoing an important transformation that combines knowledge about human-design computing (as going on in computer science) with knowledge about computing observed in nature. Several areas of natural computing, such as evolutionary algorithms (see, e.g., Ghosh and Tsutsui [23]), neural networks (see, e.g., Haykin [27]), quantum computing (see, e.g., Hirvensalo [29]), and DNA computing (see, e.g., Păun et al. [41]; Jonoska and Seeman [30]), are flourishing in computer science. Characteristic for these areas is the use of paradigms underlying natural systems. Thus, e.g., evolutionary algorithms use the concepts of mutation, recombination, and natural selection from the theory of evolution, while neural networks are inspired by the highly interconnected neural structures in the brain and nervous system.

DNA computing is based on paradigms from molecular biology; researchers in DNA computing study the use of DNA (and other) molecules for the purposes of computing. Research in DNA computing can be roughly divided into two (not disjoint) streams: DNA computing in vitro and DNA computing in vivo. The former is concerned with the theoretical foundations and experimental work on building DNA-based computers in test tubes. The latter is concerned with constructing computational components in living cells (such as simple switching circuits, see [57]), and with studying computational processes taking place in living cells. In recent years, some of the life processes going on in ciliates have attracted the attention of researchers in the DNA computing community.

Ciliates (ciliated protozoa) are single-celled eukaryotic organisms (see, e.g., [47]). It is an ancient group of organisms that originated around two billion years ago, and it is a very diverse group – some 8,000 different species

are currently known. Two characteristics unify ciliates as a single group: the possession of hairlike cilia used for motility and food capture, and the presence of two kinds of functionally different nuclei in the same cell – a micronucleus and a macronucleus.

The macronucleus is the "household nucleus" that provides RNA transcripts for producing proteins, while the micronucleus is a dormant nucleus, where no production of RNA transcripts is attempted at all. The micronucleus is activated only in the process of sexual reproduction, where at some stage (the genome of) the micronucleus gets transformed into (the genome of) the macronucleus in the process called gene assembly – it is the most involved DNA processing known in living organisms. Gene assembly is so involved because the form of the micronuclear genome is drastically different from the form of the macronuclear genome (see, e.g., [45], [46], [48]).

The computational nature of gene assembly in ciliates was brought to the attention of the DNA computing community in a series of papers by Kari and Landweber (see, e.g., [37], [38]), where the authors note that the process of assembling a macronuclear gene from its micronuclear form resembles the structure of the solution of the so-called directed Hamiltonian path problem proposed by Adleman in his seminal paper [1] that invigorated DNA computing research. (See also [55] for an even earlier hint on the computational nature of gene assembly.) Since then research on the computational nature of gene assembly in ciliates has developed rapidly, and it has involved both biologists and computer scientists. One line of this research has followed the original view of Kari and Landweber, and it has focused on the computational power (in the sense of computability theory) of their intermolecular model. The other line of this research, carried out by the authors of this book and based on an intramolecular model, has focused on the gene assembly itself, including topics such as the possible forms of the genes generated during gene assembly and possible strategies for the gene assembly. This book centers on the phenomena of gene assembly.

DNA computing represents one side of the cooperation/interaction between computer scientists and biologists: molecular biology assisting computer scientists to achieve the really bold goal of providing an alternative to or a complement for silicon-based computers. On the other side of this cooperation/interaction, in bioinformatics (see, e.g., Lesk [39]) and in computational molecular biology (see, e.g., Pevzner [43]), computer scientists and mathematicians assist biologists in understanding the structure and function of biomolecules, such as DNA and proteins, in living cells. The research presented in this book lies at the intersection of all three areas: DNA computing, bioinformatics, and computational biology. But most naturally it belongs to natural computing because it is deeply concerned with the computational nature of complex biological phenomena.

This book is organized as follows.

Part I of the book gives the biological background, and it consists of three chapters. Chapter 1 provides an overview of the structures common to cells and the molecular principles on which cells operate. Chapter 2 describes the features of ciliates that make them uniquely useful for the study of natural computing. In Chapter 3 we postulate three molecular operations, *ld*, *hi*, and *dlad*, that accomplish gene assembly in ciliates.

Part II introduces formal models for studying gene assembly. Chapter 4 describes the process of model forming that leads to the formulation of three models: MDS descriptors, legal strings, and overlap graphs. In particular, it explains how abstracting from more details of gene structure leads to these three models (in this increasing level of abstraction). This chapter is an informal introduction to the formal framework of this book — it lays a foundation for biologists to acquire intuitive insights and understanding about the more formal chapters of Part II. Chapter 5 introduces basic mathematical notions and notations needed in this book. Formalization of gene assembly on the MDS descriptors level is presented in Chaps. 6 and 7, on the level of legal strings in Chaps. 8 and 9, and on the level of overlap graphs in Chaps. 10 and 11.

Part III gives three examples of research topics concerned with gene assembly. In Chapter 12 we consider properties of gene assembly that are independent of the choice of gene assembly strategy. Since at present we do not know which strategies are actually used by ciliates, studying properties that are common to all strategies is, of course, important. In Chap. 13 we analyze the influence of molecular operations on the form of the genes that they assemble. In particular, we give formal characterizations of the forms of genes that can be assembled by each subset of the set of the three molecular operations *ld*, *hi*, and *dlad*. In Chap. 14 we use graph theory for formulating yet another point of view on gene assembly. We view it here as a process of dynamically changing decomposition of a graph representing a gene. One can view this chapter as a structural graph-theoretic formulation of the novel paradigm "computing by folding and recombination" that underlies a big part of research on computational aspects of gene assembly.

Finally, Part IV is an epilogue for this book. Chapter 15 demonstrates how to formulate an intermolecular model of gene assembly using the "pointer approach" of this book. In this way we formulate one possible bridge to the original intermolecular model of Kari and Landweber. Chapter 16 provides a perspective on the research presented in this book, and in particular it outlines a number of possible future lines of research.

Acknowledgements

The research of T. Harju, I. Petre, and G. Rozenberg was supported by European Union project MolCoNet, IST-2001-32008. T. Harju gratefully acknowledges support by the Academy of Finland, project 39802. D. M. Prescott and G. Rozenberg gratefully acknowledge support under NSF grant 0121422. We are also grateful to Springer-Verlag, in particular Mrs. Ingeborg Mayer, for cooperation, excellent in every respect.

August 2003
Boulder, Turku, Leiden

A. Ehrenfeucht
T. Harju
I. Petre
D.M. Prescott
G. Rozenberg

Contents

Notation

Sets

$\mathrm{card}(X)$	Number of elements in a set	57
$[k, n]$	Interval $\{k, k+1, \ldots, n\}$	57
$X \cup Y$, $X \cap Y$	Union and intersection of sets	57
$X \setminus Y$	Set difference	57
$\mathbb{N}$	Set of positive integers	57
$\emptyset$	Empty set	58
$\delta_{(p)}$	Interval of a pointer in δ	71
$O_u(a)$, $O_u^+(a)$, $O_u^-(a)$	Overlapping pointers of a legal string	84
Ld, Hi, Dlad	Sets of the operations ld, hi, dlad	76
Snr, Spr, Sdr	Sets of the operations snr, spr, sdr	92
Gnr, Gpr, Gdr	Sets of the operations gnr, gpr, gdr	110

Strings

Σ^*	Set of strings over Σ	58
$\overline{\Sigma}$	A signed copy of Σ	60
Σ^{Φ}	Set of signed strings over Σ	60
$\mathrm{dom}(u)$	Domain of the string u	60
$\|\|v\|\|$	Removes the bars from v	61
$[v]$	Circular string of v	61
$u_{(a)}$	a-interval of a legal string u	83
Λ	Empty string	58
$\overline{u}$	Inversion of the string u	60
u^R, u^C	Reversal and complement of u	60
Θ_κ	Alphabet of MDSs	68
Ω	Alphabet of IESs	122
Δ_κ	Alphabet $\{2, 3, \ldots, \kappa\}$	69
$\mathcal{M}$	Markers $\{b, e, \overline{b}, \overline{e}\}$	69
Π_κ	Alphabet of pointers $2, \overline{2}, \ldots, \kappa, \overline{\kappa}$	69
Γ_κ	Alphabet of MDS descriptors	70
w_δ	String associated with an MDS descriptor	91

Functions

ld_p, hi_p, $\mathsf{dlad}_{p,q}$	Assembly rules for MDS descriptors	76–77
$\underline{\mathsf{ld}}_p$, $\underline{\mathsf{hi}}_p$, $\underline{\mathsf{dlad}}_{p,q}$	Assembly rules for MDS–IES descriptors	123
snr_p, spr_p, $\mathsf{sdr}_{p,q}$	Reduction rules for legal strings	92
gnr_p, gpr_p, $\mathsf{gdr}_{p,q}$	Reduction rules for signed graphs	110
$\mathsf{sir}_p(w)$	Local reversal	113
ψ_κ	Translation of MDS arrangements to MDS descriptors	70
rem_κ	Removes parentheses and markers from MDS descriptors	84
ϱ_κ	Maps MDS arrangements to legal strings	85
$\mathsf{loc}_p(\gamma)$	Local complement of γ	109
$\mathrm{con}(\delta)$	Context of δ	124
$\mathrm{res}(\delta)$	Residual string of δ	125

Graphs

$E(V)$	$\{\{x,y\} \mid x,y \in V,\ x \neq y\}$	62
$\gamma = (V,E,\varepsilon)$	(Multi)graph	62
$\gamma = (V,E,\sigma)$	Signed graph	99
$\gamma = (V,E)$	Simple graph	64
$\gamma = (V,E,\varepsilon,f,h)$	MI-graph	151
$\sum_{i=1}^{m} \gamma_i$	Disjoint union of MI-graphs	154
$\gamma * p$	Folded MI-graph	157
$\gamma \diamond p$	Unfolded MI-graph	159
$\gamma \circledast P$	Folded and unfolded MI-graph	159
ε_γ	Endpoint mapping of γ	62
η_γ	Natural pairing function of γ	157
$N_\gamma(x)$	(Closed) neighborhood of x	62
$x \xrightarrow{i} y$	Oriented edge (x,y) with label i	64
A_w	Graph associated with a double occurrence string w	87
γ_v	Overlap graph of a legal string v	99
$E^+_\gamma(x)$, $E^-_\gamma(x)$	Incoming and outgoing edges	152
$\mathrm{val}_\gamma(x)$	Valency of a vertex	152
$\mathrm{val}_\gamma(x;c)$	Valency of color x of a vertex	152
$\mathcal{G} = (\gamma, P)$	Genome	165
$A(\mathcal{G},R)$	Assembled genome $(\gamma \circledast R, P \setminus R)$	165