From Artificial Evolution to Artificial Life

Timothy John Taylor

PhD
University of Edinburgh
1999

To Pete

Abstract

This work addresses the question: What are the basic design considerations for creating a synthetic model of the evolution of living systems (i.e. an 'artificial life' system)? It can also be viewed as an attempt to elucidate the logical structure (in a very general sense) of biological evolution. However, with no adequate definition of life, the experimental portion of the work concentrates on more specific issues, and primarily on the issue of open-ended evolution. An artificial evolutionary system called Cosmos, which provides a virtual operating system capable of simulating the parallel processing and evolution of a population of several thousand self-reproducing computer programs, is introduced. Cosmos is related to Ray's established Tierra system [Ray 91], but there are a number of significant differences. A wide variety of experiments with Cosmos, which were designed to investigate its evolutionary dynamics, are reported. An analysis of the results is presented, with particular attention given to the role of contingency in determining the outcome of the runs. The results of this work, and consideration of the existing literature on artificial evolutionary systems, leads to the conclusion that artificial life models such as this are lacking on a number of theoretical and methodological grounds. It is emphasised that explicit theoretical considerations should guide the design of such models, if they are to be of scientific value. An analysis of various issues relating to self-reproduction, especially in the context of evolution, is presented, including some extensions to von Neumann's analysis of self-reproduction [von Neumann 66]. This suggests ways in which the evolutionary potential of such models might be improved. In particular, a shift of focus is recommended towards a more careful consideration of the phenotypic capabilities of the reproducing individuals. Phenotypic capabilities fundamentally involve interactions with the environment (both abiotic and biotic), and it is further argued that the theoretical grounding upon which these models should be based must include consideration of the kind of environments and the kind of interactions required for open-ended evolution. A number of useful future research directions are identified. Finally, the relevance of such work to the original goal of modelling the evolution of living systems (as opposed to the more general goal of modelling open-ended evolution) is discussed. It is suggested that the study of open-ended evolution can lead us to a better understanding of the essential properties of life, but only if the questions being asked in these studies are phrased appropriately.

Acknowledgements

Many people have contributed in a variety of ways to the production of this thesis over the last three and a half years.

First of all, I would like to thank John Hallam, my supervisor. John took me on as a student, and has always been ready to offer guidance, discussion and constructive criticism of my work. He has also allowed me the freedom to follow my own academic interests in the course of my work, which I greatly appreciate. Thanks John!

The interdisciplinary nature of my work has required me to seek advice and discussion from a wide variety of people. Their generosity in giving up their valuable time to answer my questions and read drafts of some of the following chapters has consistently surprised me (especially those whom I know only by email exchanges).

Chief among my list of 'extra supervisors' is Mark Bedau from Reed College in the USA. Since first meeting Mark in early 1997 at the Philosophy of Artificial Life conference in Oxford, he has provided vital encouragement, suggestions and discussion of my work. From a practical point of view, he also developed a range of analysis and visualisation techniques for evolutionary systems, which I have used extensively. I would also like to thank Emile Snyder, who implemented the analysis software and has helped me to use it. Thanks, Emile, you saved me a lot of time!

Tom Ray read through an early draft of the Cosmos design document, and provided useful comments and suggestions. The work reported in Section 6.1 was originally published in the proceedings of the Sixth International Conference on Artificial Life in 1998, and I would like to thank Chris Adami and the four anonymous reviewers who offered comments on the original paper. Towards the end of my research I have had some invaluable discussions, and comments on drafts of various chapters, from a number of other people. I especially thank Nick Barton, John Holland, Howard Pattee and Moshe Sipper for reading and providing useful comments on Chapter 7 (and Chapter 2 as well in Nick Barton's case). Thanks also to Barry McMullin for providing clarification on some issues, and to Daniel Mange for some interesting discussion.

I have had a great time at a number of conferences over the last few years, and these have been made even more enjoyable by a number of people I've met at them. In particular, thanks to Jason Noble and Alastair Channon—see you at ECAL'99?

During my research I have also benefited from some much broader discussions about artificial life, artificial intelligence, life, the universe and everything. These discussions have often been lubricated with plenty of beer, and fed with copious amounts of chicken korma and egg foo yung. I am indebted to the other members of the Mobile Robots Group in the (former) Department of Artificial Intelligence, for making it such a fun place to work. In particular (at the risk of offending the others), I'd like to say a special thank you to Simon Perkins, Richard Reeve, Arturo Espinosa-Romero, John Demiris, Nuno Chagas, Sandra Gadanho and Heba Al-Lakany. Thanks also to Judith Good for the therapeutic moaning sessions about our theses, and to Bernard Goffin for putting up with our constant droning about them during dinners at Teviot.

There are many other people at the department who have made my life a lot easier. The librarians, Olga Franks and Janice Gailani, have always been more than helpful. To name a few more, Ken Dawson, Karen Konstroffer, Margaret Rennex, David Wyse and

Dougie Howie have also been very helpful in a number of ways. I'd also like to thank the organisers of the EUCS Perl course, for what must be the most useful half-day course I've ever attended.

I would not have been able to conduct this research without the financial assistance of a grant from the Engineering and Physical Sciences Research Council (grant number 95306471), for which I am very grateful.

I don't have enough room to adequately thank my parents for everything they have done, in so many ways, to enable me to be in the position of submitting this thesis. In retrospect, it was a Christmas gift from them some years ago of a paperback book—on the face of it insignificant in comparison to everything else they have done for me—that planted the seed for my growing interest in evolution and artificial life and ultimately led me to write this thesis. The book was *The Blind Watchmaker* by Richard Dawkins. Thanks, Mum and Dad, and thanks also to Anne, Katie, Rich and Tamsin.

Of course, it hasn't all been work over these last three and a half years, and I'd like to thank those who have kept me away from artificial life and helped to make my real life so enjoyable. Apart from various friends already mentioned, thanks to all the (ex-)Strathfillan Road gang—Claudio, Trish, Gordon, Jen, Matt and Wakako. A big thank you to Simon and Ögmundur for all the squash matches and pints in the bar afterwards—I hope we keep in touch. Thanks also to Stu for the regular and very welcome coffee breaks, and to Chris and Alan for really taking my mind off work!

But most of all, there is one person who has been with me through every hour of this time, and without whom I can't imagine having done this. Thank you Pete, for your constant love, affection, friendship and support.

Declaration

I hereby declare that I composed this thesis entirely myself and that it describes my own research.

Tim Taylor
Edinburgh
May 29, 1999

Contents

Abstract **v**

Acknowledgements **vii**

Declaration **ix**

List of Figures **xvii**

List of Tables **xxi**

1 Introduction **1**

1.1 The Big Picture 1

1.2 Organisation of the Thesis 3

1.3 Major Contributions 7

1.4 Typographical Conventions 8

2 Evolution and Life **9**

2.1 Two Views of Life 9

2.1.1 The Evolutionary View 10

2.1.2 The Ecological View 17

2.1.3 Hybrid Definitions 19

2.2 The Origin of Life 22

2.3 The Pattern of Life 23

2.3.1 Evolutionary Progress 24

2.3.2 Different Types of Complexity 26

2.3.3 Driven and Passive Trends 27

2.3.4 Major Evolutionary Transitions 29

2.3.5 Contingency in Evolution 31
2.4 Playing the Game 32
2.5 Definition of Terms 33
2.6 Summary 35

3 Artificial Life **37**
3.1 What is Artificial Life? 37
3.1.1 Weak Artificial Life versus Strong Artificial Life 38
3.1.2 Is Artificial Life Possible? 40
3.1.3 Relation to Theoretical Biology 43
3.2 Previous Work 46
3.2.1 Self-Reproduction 46
3.2.2 Open-Ended Evolution 57
3.2.3 Self-Organisation and Origin of Life Models 64
3.3 Methodology and Design Issues 66
3.3.1 Methodology 66
3.3.2 Self-Reproduction 67
3.3.3 Representation 68

4 Design Details of the Cosmos Platform **71**
4.1 Cosmos Design Philosophy 72
4.2 Preliminary Issues 73
4.2.1 Representation of Information 73
4.2.2 Spatial Structure 74
4.2.3 Time Slicing and the Top-Level Algorithm 74
4.2.4 Naming of Organisms 75
4.3 The Structure of an Individual Cell 75
4.3.1 Overview 75
4.3.2 The Genome 75
4.3.3 Regulators: Promoters and Repressors 76
4.3.4 The Translator 77
4.3.5 The Energy Token Store 78

4.3.6 Cell Division and Reproduction 79
4.3.7 Inter-Organism Communication Structures 80
4.3.8 Other Structures 83
4.3.9 Parallel Programs (Multicellular Organisms) 84
4.4 The Programming Language and Representation 87
4.5 The Environment 88
4.5.1 The Grid 88
4.5.2 Distribution of Energy Tokens 88
4.5.3 Collection of Energy Tokens 91
4.5.4 Moving around the Grid 91
4.5.5 Inter-Organism Communications 93
4.5.6 Environmental Information 94
4.5.7 Mutations and Flaws 94
4.6 Actions and Interactions 95
4.6.1 Implications of Intercellular and Inter-Organism Communications 96
4.7 The Top-Level Algorithm 97
4.8 Global Parameters 99
4.9 Input and Output Files 99
4.10 Major Differences between Cosmos and Tierra 99
4.11 Summary 104

5 Cosmos Experiments 1 **105**

5.1 Analysis and Visualisation Techniques 105
5.1.1 Individual-based Measures 106
5.1.2 Visualisation of Spatial Distributions 109
5.1.3 Population-based Measures 109
5.1.4 A Neutral Shadow 111
5.2 Detailed Analysis of a Standard Run 112
5.2.1 Program Length 114
5.2.2 Replication Period 117
5.2.3 Age at Death 118
5.2.4 Population Size and Diversity 118

5.2.5 Flaw Period and Proportion of Unfaithful Replications 121
5.2.6 Activity Measures 122
5.2.7 Analysis of Significant Genotypes 127
5.2.8 Spatial Distribution 130
5.2.9 Summary of Results 132

6 Cosmos Experiments 2 **133**

6.1 The Role of Chance 133
6.1.1 Method 134
6.1.2 Results 135
6.1.3 Activity Wave Diagrams 140
6.1.4 Discussion 143
6.2 Re-running the Standard Model 145
6.2.1 Results 145
6.3 Mutations and Flaws 151
6.3.1 High Mutation and Flaw Rates 151
6.3.2 Low Mutation and Flaw Rates 153
6.4 The CPU-time Distribution Scheme 156
6.4.1 Results 156
6.5 Energy 157
6.5.1 High Energy Levels 158
6.5.2 Private Energy Collection 158
6.5.3 Energy Gradient 162
6.5.4 Random Distribution 172
6.6 Reading Neighbouring Code 175
6.7 Inoculation with Sexual Ancestors 177
6.8 Summary and Discussion 181

7 Reappraisal of the Approach **187**

7.1 Problems with Tierra-like Models 187
7.1.1 Lack of an Explicit Theoretical Grounding 188
7.1.2 Predefined Organism Structure 190

7.1.3 Restricted Ecological Interactions . . . 191
7.1.4 No Competition for Matter or Energy . . . 192
7.1.5 Evolving a Self-Reproduction Algorithm . . . 195
7.2 Self-Reproduction and Evolution Revisited . . . 195
7.2.1 Some Definitions . . . 195
7.2.2 General Issues of Reproduction . . . 198
7.2.3 Self-Reproduction and Open-Ended Evolution . . . 202
7.3 Improving the Approach . . . 219
7.3.1 Beyond Digital Naturalism: The Need for Clear Goals . . . 220
7.3.2 A Full Specification for Open-Ended Evolution . . . 221
7.3.3 Evolution and Life Revisited . . . 229

8 Summary **233**

A Cosmos System Details **239**
A.1 Global Parameters . . . 239
A.1.1 Inoculation . . . 239
A.1.2 Start of Run . . . 240
A.1.3 Termination . . . 240
A.1.4 Environment . . . 240
A.1.5 Organism . . . 242
A.1.6 Cell . . . 242
A.1.7 Mutations and Flaws . . . 243
A.1.8 Input and Output . . . 244
A.2 The `REPLiCa` Instruction Set . . . 245
A.3 Predefined Ancestor Programs . . . 252
A.3.1 Ancestor A1 . . . 252
A.3.2 Ancestor A2 . . . 252
A.4 Running Cosmos . . . 254
A.5 Format of Input and Output Files . . . 254
A.5.1 Input Files . . . 254
A.5.2 Output Files . . . 257

A.6 Implementation Details 261

B Details of the Reported Cosmos Runs **263**

B.1 Default Values for Minor Parameters in Cosmos 263

B.2 The Standard Ancestor Program, 348AAAA 264

B.3 The Sexual Ancestor Program, 1314AAAA 265

B.4 Genetic Code 269

B.5 RNG Seeds for Runs in Chapter 6 270

B.5.1 The Role of Chance 270

B.5.2 Re-Running the Standard Model 271

B.5.3 Mutations and Flaws 271

B.5.4 The CPU-time Distribution Scheme 271

B.5.5 Energy 271

B.5.6 Reading Neighbouring Code 272

B.5.7 Inoculation with Sexual Ancestors 272

Glossary **273**

Bibliography **277**

List of Figures

2.1 Different Types of Complexity. 28

4.1 The Structure of a Cell in Cosmos. 76

4.2 Translation of the Genome. 78

4.3 Searching for Communications with the `rms_receive` Instruction. 82

4.4 An Example of Organism Fission. 86

4.5 Different Patterns of Energy Token Distribution. 90

4.6 The Top-Level Algorithm. 98

5.1 Schematic of the Standard Ancestor, 348AAAA. 112

5.2 Standard Run: Length. 115

5.3 Standard Run: Population Size. 115

5.4 Standard Run: Replication Period. 115

5.5 Standard Run: Diversity. 115

5.6 Standard Run: Flaw Period. 115

5.7 Standard Run: Age at Death. 115

5.8 Theoretical Relationship between Program Length and Replication Period. *Left*: Size of copy loop increases proportionally with size of program. *Right*: Size of copy loop is constant. 117

5.9 Population Size Limit Imposed on Ancestor Programs by the Default Rate of Introduction of Energy into Environment. 120

5.10 Standard Run, Neutral Shadow: Diversity. 121

5.11 Standard Run: Unfaithful Replications. 122

5.12 Standard Run: Unfaithful Replications (omitting points with zero infidelity). 122

5.13 Standard Run: Activity Waves Plot. 124

5.14 Standard Run, Neutral Shadow: Activity Waves Plot. 124

5.15 Standard Run, Neutral Shadow: Activity Waves Plot (magnified). . . . 124

5.16 Standard Run: Cumulative Activity. 125

5.17 Standard Run, Neutral Shadow: Cumulative Activity. 125

5.18 Standard Run: Mean Cumulative Activity. 125

5.19 Standard Run, Neutral Shadow: Mean Cumulative Activity. 125

5.20 Standard Run: New Activity. 127

5.21 Standard Run, Neutral Shadow: New Activity. 127

5.22 Standard Run: Mean New Activity. 127

5.23 Standard Run, Neutral Shadow: Mean New Activity. 127

5.24 Standard Run: Phylogeny of Significant Genotypes. 128

5.25 Standard Run: Cell Energy Distribution, Time Slice 5,000. 131

5.26 Standard Run: Cell Energy Distribution, Time Slice 910,000. 131

5.27 Standard Run: Cell Age Distribution, Time Slice 900,000. 131

6.1 Contingency Experiments. Graphs of Cumulative Activity, Mean Cumulative Activity, Diversity, Program Length and Replication Period for Runs 10 (*left*) and 17 (*right*). 136

6.2 Procedure for Significance Testing in Contingency Experiments. 139

6.3 Cumulative Activity (concentration): Pairwise comparisons (p values) between runs. Raw Sample Data (*left*). Differenced Sample Data (*right*). 141

6.4 Mean Cumulative Activity (concentration): Pairwise comparisons between runs. 141

6.5 Diversity: Pairwise comparisons between runs. 142

6.6 Program Length: Pairwise comparisons between runs. 142

6.7 Replication Period: Pairwise comparisons between runs. 142

6.8 Activity Wave Diagram, Runs 10 (*left*) and 17 (*right*). 143

6.9 Class 2 versus Class 1 Dynamics. 147

6.10 Standard Model Re-Runs: Spatial distribution of programs in Run 1 during Class 1 phase. 149

6.11 Standard Model Re-Runs: Age at Death of programs in Run 1. 150

6.12 Standard Model Re-Runs: Number of faithful offspring of programs in Run 1. 151

6.13 Population size graphs for ancestors with different numbers of `et_collect` instructions within copy loop. 153

6.14 Low Mutation: Population size graphs for Run 3 (*left*) and Run 9 (*right*). 154

6.15 Low Mutation: Takeover of population by 'cancerous' mutant (darker squares) in Run 3. 155

6.16 CPU-time Distribution: Graphs of Program Length for 4 of the 9 Runs. 157

6.17 Private Energy Collection: Class 1-like behaviour. 160

6.18 Private Energy Collection: Population Size and Diversity for Run 9. . . 161

6.19 Large Energy Gradient, Runs 1 and 2: Change in constitution of population over time. 164

6.20 Large Energy Gradient, Runs 3, 5, 7 and 8: Change in constitution of population over time. 165

6.21 Large Energy Gradient, Run 3: Spatial Distribution of Programs at Time Slice 900,000. 167

6.22 Large Energy Gradient, Run 8: Spatial Distribution of Programs at Time Slice 900,000. 167

6.23 Large Energy Gradient: Equilibrium distribution of environmental energy. 167

6.24 Large Energy Gradient, Run 5: Spatial Distribution of Programs of Different Sizes at Time Slice 900,000. 169

6.25 Large Energy Gradient, Run 7: Spatial Distribution of Programs of Different Sizes at Time Slice 900,000. 169

6.26 Large Energy Gradient, Runs 4, 6 and 9: Change in constitution of population over time. 171

6.27 Large Energy Gradient, Run 4: Spatial Distribution of Programs of Different Sizes at Time Slice 900,000. 173

6.28 Large Energy Gradient, Run 6: Spatial Distribution of Programs of Different Sizes at Time Slice 900,000. 173

6.29 Large Energy Gradient, Run 9: Spatial Distribution of Programs of Different Sizes at Time Slice 900,000. 173

6.30 Schematic of the Sexual Ancestor, 1314AAAA. 178

7.1 Categorisation of Reproducers. 200

7.2 DNA Replication. 210

7.3 Embeddedness of Individuals and Richness of Interactions in Various Artificial Evolutionary Platforms. 217

A.1 Format of the `genetic_code.ini` file. 255

A.2 Format of the `params.ini` file. 255

A.3 Format of the `ancestor.ini` file. 256

List of Tables

1.1 Typographical Conventions. 8

4.1 Definitions of Biologically-Related Terms Used for Describing Cosmos. . 72

5.1 Default Values for Major Parameters in Cosmos. 113

5.2 Location of Differences Between Significant Genotypes and Their Ancestors. 129

6.1 Non-default Parameter Values for Contingency Experiments. 135

6.2 Mean Number of Runs that Each Run is Significantly Different from at the 0.05 Level ($\bar{n}_{0.05}$) and 0.001 Level ($\bar{n}_{0.001}$), and the Criterion Differential ($C.D.$). See text for details. 141

6.3 Non-default Parameter Values for High Mutation Experiments. 151

6.4 Non-default Parameter Values for Low Mutation Experiments. 154

6.5 Non-default Parameter Values for CPU-time Distribution Experiments. 156

6.6 Non-default Parameter Values for High Energy Experiments. 158

6.7 Non-default Parameter Value for Private Energy Experiments. 159

6.8 Non-default Parameter Value for Small Energy Gradient Experiments. . 163

6.9 Non-default Parameter Value for Large Energy Gradient Experiments. . 163

6.10 Non-default Parameter Values for Random Energy Experiments. 174

6.11 Non-default Parameter Value for Readable Neighbouring Code Experiments. 175

6.12 Non-default Parameter Values for Sexual Reproduction Experiments. . . 179

B.1 Default Values for Minor Parameters in Cosmos. 263

B.4 Genetic Code for Runs in Chapters 5 and 6. 270

Chapter 1

Introduction

> "*There is grandeur in this view of life, with its several powers, having been originally breathed into a few forms or into one; and that, whilst this planet has gone cycling on according to the fixed law of gravity, from so simple a beginning endless forms most beautiful and most wonderful have been, and are being, evolved.*"
>
> Charles Darwin, The Origin of Species ([Darwin 59] pp.459–460)

1.1 The Big Picture

This thesis is about life: what it is, how we recognise it, and whether it is possible to recreate it on a computer. The world-view introduced by Darwin asserts that the existence of living organisms can fundamentally be explained in terms of a small number of basic processes [Darwin 59]. Darwin argued that the most important of these are the processes of reproduction, variation and natural selection. Neo-Darwinism, the modern version of Darwin's position, asserts that these are in fact the *only* relevant processes for explaining the adaptations of organisms.

On the face of it, these arguments suggest that it might be possible to build an artificial world that exhibits these simple processes on a computer, such that entities evolve within it which could be regarded as living organisms. Most people would agree that the chances of success of such an endeavour are slim. However, there is less agreement over the reasons why the enterprise might fail. Some would argue that life is a

phenomenon which is fundamentally associated with the material world; for example, they might argue that some of the processes associated with living organisms, such as metabolism, could be simulated on a computer, but that a computer program could never be regarded as *really* metabolising. On the other hand, others would argue that life is fundamentally a process (or set of processes) and is quite independent of its particular medium of implementation; they would be quite happy to accept that artificial life could evolve on a computer. However, even some of these people would predict the failure of the endeavour, but for more practical reasons such as the lack of processing speed, memory, etc., of today's computers.

An attempt to create artificial life might therefore seem ill-fated from the start, but I believe it is still worth pursuing for a number of reasons. First, the approach of building a synthetic evolutionary system is very different to the traditional approach taken in theoretical biology of analysing evolution by tracking changes in population-level measures (e.g. gene frequencies) using models based upon differential equations. Different approaches involve looking at systems in different ways; one approach might suggest answers (and, indeed, questions) whose significance is not apparent from another approach. In this way, the synthetic approach can complement the more traditional approaches of theoretical biology, and lead us to ask different sorts of questions about evolution and life. This being the case, the endeavour might be worthwhile pursuing even if the attempt ultimately fails in achieving its grand goal. Secondly, we do not know how much of a barrier the practical limitations imposed by today's computers, mentioned above, really are. These are issues which can be resolved empirically.

The potential benefits of this work therefore include a contribution to our scientific knowledge of biological evolution. Even if we are ultimately unsuccessful in reaching the grand goal of synthesising artificial life, the nature of the ways in which the attempt fails will be instructive. In addition, any success in the endeavour would have tremendous implications in the longer-term for many kinds of computer applications. The ability to evolve an unlimited variety of complex adaptations is of obvious significance to areas such as machine learning systems, evolutionary design, computer games and genetic art, to name but a few.

1.2 Organisation of the Thesis

I will now give an overview of the thesis. I will briefly describe the content of each chapter, and emphasise the flow of thought connecting the chapters together. Where relevant, I highlight the key problems addressed in the chapters, and what I regard as the key contributions of the work.

Relevant issues from the biological literature are discussed in Chapter 2. The distinction between evolutionary and ecological views of life is emphasised, and attempts to produce definitions of life from each of these views, as well as hybrid definitions, are discussed. Similar distinctions in origin-of-life models are described. Current debates concerning various preconceptions about life and evolution are then discussed. These include debates over the notion of progress in evolution, and in particular the common assumption that evolutionary processes necessarily lead to an increase in the complexity of organisms. This leads to a discussion of the term 'complexity', and of the various issues it encompasses. The nature of the major evolutionary transitions that have occurred in the history of life on Earth is described. Current debates in the biological literature over the relative importance of general principles versus contingency ('historical accidents') in determining the course of evolution are also discussed. At the end of the chapter, definitions are provided for the term 'open-ended evolution' and for various sorts of complexity, but an adequately precise and satisfactory definition of life is still lacking. In this light, a decision is made to frame the research questions addressed in the rest of the thesis in terms of more specific concepts. In particular, the experimental work will concentrate on the issue of open-ended evolution.

A review of the relevant artificial life literature is presented in Chapter 3, together with an introduction to the more important issues and debates of the subject. The distinction between weak and strong artificial life is discussed, as are some of the theoretical and practical hurdles that the subject faces. Various views of the relationship between artificial life and theoretical biology are also presented. The majority of the chapter is devoted to a description of previous work with artificial life models. Particular attention is given to studies addressing the issues of self-reproduction and open-ended evolution, including work by John von Neumann, Nils Barricelli, John Holland and Tom Ray. More briefly, a number of models addressing issues of self-organisation and the origin

of life are also discussed. Having devoted some time to describing the literature, I end the chapter by talking about the various methodology and design issues for artificial life platforms that have been highlighted by the preceding review.

In Chapter 4 the platform used for the experimental portion of the thesis is introduced. This platform, called Cosmos, is based upon Ray's Tierra system [Ray 91]. Reasons for experimenting with such a system are discussed at the start of the chapter. As the approach pioneered by Ray with Tierra is fairly widely used in the artificial life community, and its validity is fairly widely accepted, I claim that it is a valuable endeavour to investigate such models more thoroughly. Although Cosmos shares the general Tierra approach of modelling evolving individuals as self-reproducing computer programs, there are a number of important differences in the design of the two systems. These include: a closer analogy between individual programs (or more accurately, processes) in Cosmos and biological cells, including features such as regulation of the genome; the introduction of the notion of energy (potential CPU-time) as a commodity that cells must collect, store and use to pay for the execution of instructions; the introduction of a two-dimensional discrete spatial environment in which the programs can move and interact; and finally, the potential for parallel programs (which are considered analogous to multicellular organisms) to evolve. (Tierra has also been modified to allow for parallel programs, e.g. [Thearling & Ray 96], but the implementation details are different.) These and other differences between Cosmos and Tierra are discussed throughout the chapter, and summarised in the final section.

The results of a wide variety of Cosmos runs are presented, analysed and discussed in Chapters 5 and 6. These two chapters together constitute possibly the fullest and most systematic investigation of a Tierra-like system to have been reported.

Chapter 5 is devoted to the detailed analysis of a single Cosmos run. The various measures and visualisation techniques used are first described, including a collection of measures suggested by Mark Bedau and colleagues (e.g. [Bedau *et al.* 98]). The results of the run were not spectacular; programs generally increased in length due to the acquisition of more instructions to collect energy from the environment (which thereby increased their chances of survival). No parallel programs emerged, neither were any parasites, or similar ecological phenomena, observed. Reasons for this behaviour are discussed. The lack of emergence of ecological phenomena reinforces the suspicion

that their appearance in runs of Tierra is due to some fairly specific details of the system's design rather than to any more general principles. The phylogenetic tree of the significant types of program to have emerged during the run is reconstructed, and suggests that chance events may have played a significant role in determining the outcome of the run. This issue is investigated in more detail in the following chapter.

Chapter 6 describes the results of a wide variety of Cosmos runs, designed to explore the system's parameter space. This issue of contingency is investigated in detail, in a group of 19 experiments in which the system is run under identical conditions except for the number used as a seed for the random number generator. It is found that each run performed significantly differently, in a number of measures, to at least a third of the other runs. This result is used in a calculation which suggests that the system should be run at least nine times in future experiments to be sure of observing a variety of results due purely to contingency rather than any other factors. A range of other experiments are also reported. Some of the more interesting results from these include: the runs can generally by categorised into one of two classes proposed by Bedau and colleagues [Bedau *et al.* 98], where one class represents active evolutionary behaviour and the other represents a degenerative state where the system's evolutionary potential is effectively lost; and, in one set of experiments (reported in Section 6.5.3) 'speciation' is observed, i.e. the coexistence in the population of programs of different lengths, in different areas of the environment. A gradient in the energy distribution to the environment is required to achieve this, although no physical boundaries are necessary. Some experiments with a sexually reproducing program are also described, although asexual varieties quickly invaded the population and displaced the sexual programs in all of the reported runs. A summary and discussion of all of the results is presented at the end of the chapter.

In Chapter 7, with the benefit of the practical experience gained by designing, implementing and using Cosmos, I step back and reconsider the scientific value of the approach. A number of areas are identified in which Cosmos and similar Tierra-like systems are lacking, from both theoretical and methodological points of view. The major problem with using these systems for scientific purposes is the inadequacy of the theoretical grounding upon which their design is based. Various other problems are discussed, including the fact that these systems still do not seem to have the ability

to evolve *fundamentally new* types of innovations. These discussions lead on to a detailed analysis of various issues involved in the concept of self-reproduction, particularly within an evolutionary context. The reproduction process employed by the programs in these systems is analysed in terms of von Neumann's genetic self-reproduction architecture [von Neumann 66], and compared to the replication processes employed by von Neumann's self-reproducing automata, by DNA, and by a hypothetical 'proto-DNA' structure which has properties making it suitable for acting as a seed for an open-ended evolutionary process. It is argued that the explicit encoding of the self-reproduction process on programs in Tierra-like systems is a liability in terms of open-ended evolution, and that these systems would have greater evolutionary potential if the copying process was, initially at least, implicitly encoded in the operating system. In this situation, the focus of the design of such systems then shifts to the consideration of the sorts of phenotypic properties the reproducers should possess. The domain of interaction of the phenotypes should be within the evolving system itself, to allow for intrinsic rather than extrinsic selection (e.g. [Packard 88]), but beyond this there would seem to be few restrictions. Some ways in which the study of open-ended evolution in artificial life models may be improved are discussed in Section 7.3, where a number of open questions are also listed. A paradigm for studying biological evolution, introduced 30 years ago by C.H. Waddington [Waddington 69], is described, and it is argued that this represents a useful starting place for an improved synthetic approach. As an example of a future research direction, I show how the important question of how organisms evolve fundamentally new measuring instruments can be addressed within Waddington's framework (see Section 7.3.2). In the final section of Chapter 7 I return to the issue of studying life as opposed to open-ended evolution. I suggest that artificial life models of open-ended evolution and of other processes associated with the biosphere can usefully contribute to a broader understanding of the nature of life, but only if the questions being addressed are expressed in terms of precise concepts.

A summary of the thesis message is presented in Chapter 8.

Further details of the design and implementation of Cosmos are given in Appendix A, and further details of the runs reported in Chapters 5 and 6 can be found in Appendix B. A glossary of the biological terms used in the thesis is included after these appendices.

1.3 Major Contributions

In the following list I summarise what I regard as the major contributions of this work.

1. The design and implementation of Cosmos, a Tierra-like artificial life platform.

2. The description and analysis of a wide variety of experiments with Cosmos, constituting the fullest and most systematic investigation of the behaviour of such a system to have been reported. An important part of this investigation concerns the role of contingency in determining the course of evolution.

3. The identification of a number of problems, some theoretical and some methodological, that limit the utility of Tierra-like platforms as tools for the scientific investigation of open-ended evolution in general, and of the evolution of life in particular.

4. An analysis of the logic of reproduction in terms of more specific issues, with particular emphasis on processes of self-reproduction in the context of evolution. This includes topics such as: the superiority of genetic reproduction over reproduction by self-inspection; discussion of self-reproducing programs in Tierra-like platforms in terms of von Neumann's analysis of the logic of self-reproduction; issues relating to the explicit versus implicit encoding of the various components of von Neumann's genetic architecture; an emphasis on the role of phenotypes; the desirability of allowing unrestricted interactions between individuals to promote open-ended evolution; and discussion of the evolution of symbolic information and of fundamentally new measuring instruments, and how these may be achieved.

5. Suggestions for how the approach to modelling open-ended evolution, and the evolution of life, may be improved. In particular, it is emphasised that careful consideration must be given not only to modelling individuals, but also to modelling the environments in which they exist, and the sorts of interactions allowed. Waddington's paradigm for an evolutionary process [Waddington 69] is suggested as a possible unifying framework within which to develop a better approach to the synthetic modelling of evolution and life.

1.4 Typographical Conventions

Various typographical conventions have been used throughout the thesis, as listed in Table 1.1.

Example	Signification	Described in
`rng_seed`	Name of a Cosmos parameter, parameter value, instruction or file.	Apdx. A
`ExportData`	Name of a routine in Cosmos top-level algorithm.	Fig. 4.6
find_end	Label for a section of code on a predefined Cosmos ancestor program.	Figs. 5.1, 6.30
Translator	Structure within a Cosmos cell, or a class of object in the underlying implementation.	Sct. 4.3
A	Component of von Neumann's genetic architecture.	Sct. 3.2.1
$\mathcal{A}$	Object in Löfgren's definitions of production, reproduction and self-reproduction, or in Waddington's work.	Scts. 7.2.1, 7.3.2
ax	Register in a Cosmos cell.	Sct. 4.3.8

Table 1.1: Typographical Conventions.

Chapter 2

Evolution and Life

"*I may look happy, healthy and clean ...*
but behind the smile there is a Zerox machine"

Adam Ant, 'Zerox'[1]

In this chapter I present a review of some of the current viewpoints and debates on evolution, life and their interrelationship. This is by no means an exhaustive review, but serves to place the work that follows in some context. I also use it as a starting point from which to define more carefully the sense in which I will use words such as 'life' and 'open-ended evolution' throughout the rest of this thesis.

2.1 Two Views of Life

As emphasised by the evolutionary biologist John Maynard Smith, there are two distinct ways in which living organisms may be viewed. "One is of a population of entities which, because they possess a hereditary mechanism, will evolve adaptations for survival. The other is of a complex structure which is maintained by the energy flowing through it" [Maynard Smith 86] (p.7). In the following two subsections I will give a quick overview of these two pictures.

[1] From the album *Dirk Wears White Sox* by Adam and the Ants ©1979 Do-It Records.

2.1.1 The Evolutionary View

The first view sees organisms as players in an evolutionary process. This view originated from Charles Darwin's ideas [Darwin 59], and has been gradually refined by advances in knowledge to the current 'neo-Darwinist' position.[2]

Neo-Darwinism

Darwin's theory originated with the observation that organisms are exquisitely adapted to their environments. He further pointed out that organisms have the properties of multiplication, variation and heredity. Essentially, he showed that organisms with these three properties necessarily become adapted to their environment, and that there is therefore "a necessary connection between one set of observations, concerning reproduction, and another set, concerning adaptation" [Maynard Smith 86] (p.6).

Darwin claimed that all living things on Earth are descended, with modification, from one or a small number of simple original forms.[3] He further claimed that the major cause of evolutionary change was natural selection, i.e. those modifications which help the evolving entities to survive and reproduce will be maintained, and those which do not help will not. While Darwin himself was willing to accept that evolutionary changes may also be brought about by other causes (such as the effects of use and disuse), the neo-Darwinist position sees natural selection as the *only* relevant mechanism by which evolution leads to adaptation. The position does not rule out other causes of evolutionary change, but it does contend that any cause other than natural selection will result in essentially random changes.

An extreme neo-Darwinist definition of life might therefore be "any entity with the properties of multiplication, variation and heredity". There are a number of complications to this picture, which, while not contradicting the neo-Darwinist position, at least suggest other things to be taken into account in our efforts to build an artificial evolutionary system. Some of these (self-organisation and development, sex, and

[2] The major extension that produced the neo-Darwinist position was the synthesis of Darwin's original theory with Mendelian genetics.

[3] Darwin did not claim his theory accounted for the *origin* of life. However, more recently, a number of theories have been proposed which put the original transition from chemistry to biology in an evolutionary context (see, for example, [Eigen 71], [Maynard Smith & Szathmáry 95]).

symbiogenesis) are briefly described below, and will be returned to in later chapters.

Self-Organisation and Development

Although some people might accept this extreme neo-Darwinist definition, few would claim that it can give a full explanation of life on Earth. After all, life is embedded in the physical world, and the rules of physics and chemistry must surely impose important constraints on the evolutionary process.

In recent years there have been a growing number of attempts to integrate theories of self-organisation with evolutionary theory. Stuart Kauffman, professor at the Santa Fe Institute, argues that the laws of physics generate general principles of self-organisation which must be considered when studying the evolution of life (although he believes that science does not yet have an adequate conceptual or mathematical framework to deal with such issues) [Kauffman 93]. He says on the subject:

> "The building blocks of life at a variety of levels from molecules to cells to tissues to organisms are precisely the robust, self-organized, and emergent properties of the way the world works. If selection merely molds further the stable properties of its building blocks, the emergent lawful order exhibited by such systems will persist in organisms. The spontaneous order will shine through, whatever selection's further siftings.
>
> Can selection have reached beyond the spontaneous order of its building blocks? Perhaps. But we do not know how far. The more rare and improbable the forms that selection seeks, the less typical and robust they are and the stronger will be the pressure of mutations to revert to what is typical and robust. That natural order, we may suspect, will indeed shine through." [Kauffman 95] (p.189).

Many have argued that the developmental processes of organisms also impose significant constraints upon organismal design (e.g. [Wolpert 93], [Goodwin 94], [Raff 97]). Note that the nature of these constraints is subtly different to the general principles of self-organisation proposed by Kauffman and others; developmental constraints are somewhat arbitrary with respect to underlying physical laws, but they are nevertheless

real constraints for organisms employing any kind of developmental process. Maynard Smith says on the issue:

> "For 50 years to my knowledge, and maybe for much longer than that, people have been saying that our ideas about evolution will be transformed when we have an adequate theory of development." [Maynard Smith 96].

The distinguished geneticist C.H. Waddington remarked that:

> "The hereditary differences which arise in animals are not quite random, like the differences between two heaps of bricks. They result from changes in orderly systems of development, and each new variety has an order of its own, maybe less, but sometimes more, complex than that of the original form from which it was developed." [Waddington 57] (p.7).

Developmental biologist Lewis Wolpert argues that the self-organisational and feedback mechanisms operating during the development of an embryo to an adult organism radically restrict the space of morphologies that evolution is free to explore. In his book on animal development, *The Triumph of the Embryo*, Wolpert says:

> "Developmental mechanisms, together with their genetic control, put a severe constraint on the evolution of animal form. It is not selective pressures that have kept the basic pattern of the vertebrate arm the same, but the fact that altering the basic pattern is almost impossible. Therefore, not all imaginable animals are possible. Any theory of evolution must incorporate an appreciation of developmental mechanisms." [Wolpert 93] (p.195).

There are many arguments over the relative importance of the spontaneous order inherent in the physical world, of the constraints imposed by developmental processes, and of natural selection in determining morphology (e.g. [Maynard Smith 86] p.43), but all these factors clearly have some part to play in the process. These considerations indicate that we cannot understand life purely through studying the genes, but we must also think about their *interactions* with the physical medium in which they are embedded. Richard Belew and Melanie Mitchell warn that "common metaphors for the genes (e.g., programs, blueprints, etc.) as the *information*-carrying component in evolution all rest

on an inappropriate 'preformationist' view of information, as if information '... exists before its utilization or expression.' " [Belew & Mitchell 96] (p.14, original emphasis).[4]

Another important point about evolution taking place in a medium with some inherent self-organisational properties is that, although the possible forms achievable by evolution may be restricted, the 'instructions' for making those forms that *are* achievable can be compactly described on the organism's genome. In other words, the genome does not have to encode information about every aspect of the adult organism's design, because some features will just fall into place 'for free' as the developmental process unfolds, due to the self-organisational properties of the constituent matter.

Sex and Species[5]

Common conceptions of life are rooted firmly in our experience of organisms being grouped into species, with sexual reproduction within a species, and crosses between species being either infertile or impossible.[6] With the highly dissected flow of genetic information that results from the existence of different species, the possibility emerges for the specialisation of life forms into plants, animals, and, within these kingdoms, countless further subdivisions. Without species, evolution would be a continuous divergence—evolutionary novelty appearing in the genome of one individual would be passed on to its descendants, but would never spread to other lineages if there is no sex (in the most general sense of the mixing of genetic material from two or more sources in a single individual).

However, the existence of species, and of sexual reproduction, is not a universal feature of life. In their book *Origins of Sex*, Lynn Margulis and Dorion Sagan argue that "The fact that we are descendants of that minority of organisms that is sexually reproducing has led us to a very unbalanced and unrealistic view of biological sexuality" [Margulis & Sagan 86] (p.15). Indeed, the very existence of sexual reproduction in nature is a great puzzle from a neo-Darwinian point of view, as organisms which

[4] The enclosed quotation is from [Oyama 85].

[5] The ideas in this section are partially based upon a talk given by Richard Dawkins at the 'Digital Biota 2' conference in Cambridge, England on 10th September 1998.

[6] In fact, it is still not entirely clear whether the existence of species is a consequence of sexual reproduction ([Maynard Smith & Szathmáry 95], Chapter 9), but this does not affect the argument being made in this section.

engage in it (and therefore only pass on half of their genes to their offspring) would appear to be at an evolutionary disadvantage to those which reproduce asexually. The question of how sexual reproduction evolved, and the related, but separate, question of how, once it appeared, it has been maintained in the face of apparent selection pressure to remove it, constitute major research topics of modern evolutionary biology (see, for example, [Maynard Smith 86], [Margulis & Sagan 86]).

The reason for mentioning this subject here is that our common notions of what it is to be alive, through our experience of biological life, may include some aspects which are by no means universal laws of biology, but rather contingent features of the particular way in which life has evolved on our planet. This raises the question of whether we would be able to recognise artificial life if we ever succeeded in creating it, and to what extent we might want to try to replicate the course of *biological* evolution in our artificial systems. These issues will be discussed in Chapter 3.

Symbiogenesis

Another complication to the neo-Darwinist picture is the fact that many species enter into close ecological relationships with other species. The general term for this is 'symbiosis'. Unfortunately this term is used in various different ways in the biological literature,[7] but I use it as an umbrella term to cover three distinct types of inter-species relationship: *parasitism*, where one organism benefits from the relationship to the detriment of the other organism(s) involved; *commensalism*, where one organism benefits with no significant detriment or benefit to the other organism(s); and *mutualism*, where both (all) species benefit from the relationship.

The intimacy of such symbiotic relationships varies considerably. In 'ectosymbionts' the associations between organisms are purely external, whereas in 'endosymbionts' one of the organisms is actually incorporated internally into the other (the 'host').

In the extreme, it is possible that the relationship between host and (endo)symbiont becomes so strong that the previously independent organisms are totally dependent on each other for their survival and reproduction, and effectively become a single organism. This process, the evolutionary origin of new morphologies and physiologies by symbiosis,

[7] For discussion of this, see, for example, [Margulis 91], [Daida *et al.* 96].

is called 'symbiogenesis', a term coined by the Russian biologist Merezhkovsky and subsequently developed by others such as Kozo-Poliansky (see [Khakhina 79]).

It is now widely accepted that the process of symbiogenesis played a key role in the transition from prokaryotic[8] cells to eukaryotic[9] cells (e.g. [Margulis 81], [de Duve 96], [Maynard Smith & Szathmáry 95]). This has led a growing number of people to question whether symbiosis also played a key role in other major events during the evolutionary history of life [Maynard Smith 91].

The significance of symbiogenesis with respect to neo-Darwinism is that Darwin's arguments concerning descent with modification seem to imply that evolutionary novelty comes about 'vertically', by mutations in the genes passed on from parents to offspring. In contrast, symbiogenesis is a mechanism for 'horizontal' gene transfer, where the genetic material from essentially unrelated organisms can be brought together in a single descendant. Whether this process is considered to be consistent with neo-Darwinism is a matter of debate. If one thinks about Darwinian evolution occurring at the fundamental level of the genes, it can certainly be argued that it is consistent (e.g. [Dawkins 76], [Maynard Smith 91]). For example, Maynard Smith says:

> "Whether [symbiosis constitutes a challenge to neo-Darwinism] depends, of course, on what one understands by 'neo-Darwinism'. I interpret it to mean the hypothesis that mutation (change in the genetic material) is not, except occasionally and by accident, adaptive to its causative agent, and therefore, insofar as organisms are adapted to their ways of life, that adaptation must have arisen by natural selection acting on originally nonadaptive genetic variation. In this sense I am a neo-Darwinist. However, the ways in which genetic material from different ancestors is united in a single descendant has profound effects." [Maynard Smith 91] (p.26).

Even if we accept that symbiogenesis is consistent with neo-Darwinism, we should recognise that this particular mechanism for producing evolutionary novelty has apparently played a key role at certain stages in the history of life on Earth. In terms of

[8] Present-day bacteria are examples of prokaryotic cells.

[9] All present-day animal, plant and fungi cells are eukaryotic. They are far more complex than prokaryotic cells, housing a wide array of specialised structures.

building artificial evolutionary systems, then, we should certainly give consideration to how symbioses and symbiogenesis might be achieved.

Life as Supple Adaptation

Mark Bedau has recently proposed an extreme evolutionary definition of life (e.g. [Bedau 96], [Bedau 98b]). Nearly all other definitions, whether evolutionary or ecological, see life primarily as a property of individuals (be they organisms, cells, or genes). In contrast, Bedau claims that a system capable of open-ended evolution (or 'supple adaptation' to use his phrase), *in its entirety*, should be considered as the primary form of life. A system exhibits supple adaptation by the "ongoing and indefinitely creative production of significantly new kinds of adaptive responses to significantly new kinds of adaptive challenges and opportunities" [Bedau 98b] (p.127). Bedau continues: "a supplely adapting system is an evolving population of organisms, or a whole evolving ecosystem of many populations, or, in the final analysis, a whole evolving biosphere with many interacting ecosystems" (*ibid.* p.131). In this view, particular components within the supplely adapting system, such as individual organisms, qualify as (secondary forms of) life by virtue of their specific relationship with the system. Exactly what relationships are required between a component and the system for the component to qualify as being alive is an open question. In suggesting this definition, Bedau emphasises that "natural selection is not sufficient for supple adaptation ... natural selection produces supple adaptation only when it is continually creative" (*ibid.* p.127).

At first glance, Bedau's suggestion might appear similar to James Lovelock's 'Gaia' hypothesis, that the whole Earth is equivalent to a single living organism [Lovelock 79]. However, the similarity is only superficial. Lovelock sees the Earth as a homeostatic, self-regulating entity; in other words, something which has the properties that we commonly associate with more standard biological organisms. Bedau, on the other hand, is certainly not claiming that the biosphere has these properties; what he is claiming is that the world must have a certain set of properties which enable it to act as a supplely adapting system if it is to be able to support the evolution of living (in the common sense of the word) organisms.

Bedau himself acknowledges that many obstacles must be overcome before a satisfactory definition can be produced in these terms, but claims that the approach has promise

for providing good explanations for a number of fundamental puzzles about life. The primary interest of his proposal in the present context is that it defines (secondary forms of) life as being the product of the *right kind* of evolutionary process (i.e. one capable of supple adaptation), rather than of evolutionary processes in general. The precise characterisation of the kind of evolutionary process which might possess this capacity is an endeavour requiring much further work. The intriguing implication is that many of the specific phenomena related to biological life which are not usually associated with purely evolutionary definitions (e.g. self-maintenance, food webs, etc.: see Section 2.1.2) might actually be necessary features of any system capable of supple adaptation. In this way, such features might ultimately be incorporated into an evolutionary framework of the type suggested by Bedau. Of course, this is just a theory, like any other, but it is worth serious consideration. We will return to this topic in Chapter 7. In the following section we return to more traditional individual-based approaches, but those that have concentrated on ecological rather than evolutionary considerations.

2.1.2 The Ecological View

Distinct from the evolutionary approach to defining life, a separate, and older, tradition has sought to define life in more direct, ecological terms. This approach does not seek to answer the question of what an organism is in terms of the history of its predecessors, but rather in terms of its characteristic organisation; in plain terms, what it *is* and what it *does*.

Some have even argued that it makes no sense to define life in terms of evolution, as evolution is purely a mechanism for *change*. Francisco Varela points out that "reproduction requires a unity to be reproduced; this is why reproduction is operationally secondary to the establishment of the unity, and it cannot enter as a defining feature of the organisation of living systems" [Varela 79] (p.33). Similarly, in discussing the distinction between genealogical (evolutionary) and ecological hierarchies of nature, Stanley Salthe questions whether organisms (or similar entities) have a place in the genealogical hierarchy. "After all, much of population genetics deals directly with gene frequencies in demes and does not bother with genotypes *per se*" [Salthe 85] (p.235). An organism "has no significant extension in time; it cannot evolve. The organism as we usually think of it (ourselves) is clearly an ecological entity" (*ibid.*).

In looking at an individual organism, it is remarkable that there is very little that stays constant over its lifetime. Energy is continuously expended as it goes about its activities, and this lost energy is replaced by the regular ingestion of food or by the direct harnessing of sunlight. Even the very molecules from which an organism is made are generally in a state of flux.[10] In contrast to most other objects in the world, organisms also have a remarkable capacity for self-maintenance and repair in the face of environmental perturbations. As Waddington puts it, the characteristic form of living organisms is "not only, in many cases, a complex one, but the entity by which it is expressed is more nearly comparable to a river than to a mass of solid rock" [Waddington 57] (p.2).

The image of organisms as self-maintaining structures that are open to both energy and matter has led many to consider them as dissipative systems (e.g. [Haskell 40], [O'Neill *et al.* 86]). The vortex produced when the plug is pulled from a basin, a flame, and, to take a larger example, the Earth's atmosphere, are all examples of physical dissipative structures. They all require a continuous input of energy to maintain their structure, and dissipate this energy as a result. However, organisms differ from physical dissipative structures in their ability to exert *control* on the flux of energy and matter through them. We are drawn to the idea of organisms as *self*-producing and *self*-maintaining wholes.

This view of life dates back to Immanuel Kant in the late eighteenth century, if not earlier. Kant's view of an organism was one in which "each part existed both for and by means of the whole, while the whole existed for and by means of the parts" (from [Kauffman 95] (p.274)). The Chilean biologists Humberto Maturana and Francisco Varela have attempted to formalise this notion of self-production and self-maintenance in their definition of 'autopoietic systems'[11] (e.g. [Maturana & Varela 80], [Varela 79]). The formal definition of autopoiesis is:[12]

> "An autopoietic system is organized (defined as a unity) as a network of processes of production (transformation and destruction) of compon-

[10] Although the extent to which this is true varies greatly between different organisms, and between different parts within a single organism [Maynard Smith 86] (p.2).

[11] They invented the word 'autopoiesis' from the Greek $\alpha\upsilon\tau\acute{o}\varsigma$ = self, and $\pi o\iota\epsilon\iota\nu$ = to produce.

[12] Unfortunately the definition, at least in its English translation, is somewhat opaque.

> ents that produces the components that: (1) through their interactions and transformations continuously regenerate and realise the network of processes (relations) that produced them; and (2) constitute it (the machine) as a concrete unity in the space in which they exist by specifying the topological domain of its realisation as such a network." [Varela 79] (p.13).

In other words, an autopoietic organisation is one which actively produces and maintains its own structure. Maturana and Varela also suggest an informal definition:

> "an autopoietic machine is a homeostatic (or rather a relations-static) system that has its own organization (defining network of relations) as the fundamental invariant." (*ibid.*)

Kauffman claims that the intellectual lineage starting from August Weismann's view of the "germ line" has led to the loss of the earlier image of cells and organisms as self-creating wholes [Kauffman 95]. Varela agrees, saying that "the great developments of molecular biology have led to an overemphasis on isolated components, and to a disregard of questions pertaining to what makes the living system a whole, autonomous unity that is alive regardless of whether it reproduces or not" [Varela 79] (p.5).

Some computer models of autopoiesis are mentioned in Section 3.2.3, and I will discuss the relationship of concepts such as autopoiesis and self-maintenance to evolutionary processes in Sections 7.1.4, 7.3.2 and 7.3.3.

2.1.3 Hybrid Definitions

For non-biologists, much of what has been said above might appear to be fairly far removed from the popular concept of what it means to be alive.

High school biology textbooks usually present a list of properties which collectively constitute a definition of life. A typical list, from [Day 85] (p.34), is: Movement, Feeding (obtaining energy and raw materials), Respiration, Excretion, Growth (increase in size and complexity), Reproduction, and Sensitivity.

Many similar lists have also been proposed, each combining some aspects of organisms from the evolutionary perspective, and some from the ecological perspective. One of

the major problems with check-list definitions, however, is that it is invariably possible to find an example of a real organism which fails to meet all of the requirements. A common problem case is the mule, which is certainly alive, although incapable of reproduction.

Considering the difficulties of such an endeavour, it is reasonable to ask whether we could *ever* arrive at a concise definition of life. Chris Langton, regarded by many as the founder of the scientific discipline of 'artificial life' (at least in its present incarnation), is skeptical that we will ever reach a consensus on this issue. From his experience of creating computer programs which exhibit many of the properties associated with living organisms, he remarks "every time we succeed in synthetically satisfying the definition of life, the definition is lengthened or changed."[13]

Attempts to fuse the ecological and evolutionary pictures in an illuminating and precise manner are complicated by the intricate, non-linear interactions involved. One of the major problems in developing a coherent picture of living systems is the description of the two-way interactions between processes occurring at vastly different time scales. Waddington says:

> "Perhaps the main respect in which the biological picture is more complex than the physical one, is the way in which time is involved in it ... [T]o provide anything like an adequate picture of a living thing, one has to consider it as affected by at least three different types of temporal change, all going on simultaneously and continuously. These three time-elements in the biological picture differ in scale. On the largest scale is evolution ... On the medium scale ... life history ... Finally, on the shortest time-scale ... rapid turnover of energy or chemical change." [Waddington 57] (pp.5–6).

In the 40 years since Waddington wrote this, little progress has been made in developing models which satisfactorily capture the interactions of processes happening at these different time-scales. Recent attempts to achieve such a synthesis are discussed by John Holland in his book *Hidden Order* [Holland 95] (esp. Chapter 5). Further efforts from the theoretical biology community at modelling the interface between ecology and evolution are discussed in [Travis & Mueller 89], [Stanley 89] and [Feldman 89].

[13] Quoted from [Kelly 94] (p.447).

In Section 2.1.2 some objections were raised to defining life in terms of evolution, as evolution is just a mechanism for change. Margulis and Sagan accept this, but claim that both autopoiesis *and* reproduction are distinguishing processes of living matter [Margulis & Sagan 86]. They argue that, while autopoiesis defines the organisation of an individual living entity, we still need to retain the idea that organisms reproduce in order to appreciate the larger picture of life:

> "Autopoiesis occurs, then, to maintain an organism during its own life, but by itself autopoiesis does not guarantee that an organism will show genetic continuity or that the characteristics of any given organism will persist faithfully through time. The process that ensures genetic continuity is reproduction. But autopoiesis remains the primary process. On the one hand, without it the organism would not survive to reach the stage at which reproduction becomes feasible. On the other hand, autopoiesis does not depend on reproduction, at least within a single generation. An infertile but healthy person with muscle, circulatory, excretory, and other organ systems in excellent running order is autopoietic even though he is unable to reproduce. In an evolutionary sense, however, such an individual has forfeited genetic continuity; he is already dead." [Margulis & Sagan 86] (p.13).

Any approach that redefines life using terms such as reproduction and autopoiesis has the great advantage that such terms can be precisely defined. Nils Barricelli, possibly the first person to implement and experiment with an artificial evolutionary system ([Barricelli 57], [Barricelli 62], [Barricelli 63]) clearly realised the problems with using words such as 'life'. In his work, Barricelli gave a precise definition of what he called a 'symbioorganism' (the work was based upon the concept of symbiogenesis, discussed earlier). Rather than asking whether the organisations that evolved in his system were alive, he turned the question upon its head:

> "As a matter of fact [the question of whether the evolved organizations are alive] has no meaning as long as there is no agreement on a definition of 'living being'. However, the reciprocal question 'whether the objects we are used to call [*sic*] living beings are a particular class of symbioorganisms'

> has a meaning. This is the question we have been trying to answer in this paper ..." [Barricelli 63] (p.99).

This tactic of reversing the question, and using terms which can be precisely defined in preference to terms such as 'life', will be the one taken throughout most of this work. Clarification of some of the terms to be used is given in Section 2.5. We will return to the problem of providing an adequate characterisation of the concept of life in Section 7.3.3.

2.2 The Origin of Life

In the previous section, the distinction was made between definitions of life which emphasise the evolutionary perspective and those which emphasise the metabolic and/or ecological perspective. A similar distinction exists in models put forward to explain the *origin* of life on Earth.[14] The two approaches can be called the *replicator-first* approach and the *metabolism-first* approach.

The replicator-first approach assumes that the original seed for life was the existence of some sort of material that could exist in a large or infinite variety of forms, could reproduce more or less faithfully without the assistance of complicated machinery, and had some mechanism whereby specific other reactions or processes could become associated with specific forms of the material. This view was first explicitly expounded by Muller in the 1920s (see [Muller 66]). Candidates for the original self-replicating material of this type include RNA (for references see, for example, [Nuño *et al.* 95] and [Lazcano 95]), and clay minerals ([Cairns-Smith 71], [Cairns-Smith 85]). The presence of a simple self-reproducer of this nature is enough to begin a process of evolution. The idea is that some forms of the material may be such that they have processes associated with them (e.g. they may act as a catalyst for some reaction) that act to stabilise the material. Such forms will be favoured by natural selection (precisely because they are more stable), and evolution proceeds by selecting reproducers that catalyse more and more reactions that are beneficial to the stability of the replicator. At some point the reactions will effectively give the replicator complete control over the composition of its local environment, at which stage the network of reactions will probably fulfil our cri-

[14] Here I am only talking about models that assume a terrestrial origin of life.

teria for being a living organisation. Many prominent evolutionary biologists and chemists favour this approach, e.g. John Maynard Smith [Maynard Smith & Szathmáry 95], Richard Dawkins [Dawkins 76] and Graham Cairns-Smith [Cairns-Smith 85].

In contrast, the metabolism-first approach assumes that self-maintaining organisations were the seed of life. These models assume that the world is such that self-maintaining (collectively autocatalytic) organisations of chemical reactions occur spontaneously with reasonable probability. Being self-maintaining, they persist for reasonable durations. Another consequence of being self-maintaining is that they produce all of the components from which they are composed, so it is easy to imagine scenarios by which some organisations of this type might reproduce (e.g. by splitting in two). Such self-reproducing organisations will become more abundant, and will replace non-reproducers if there is competition for resources. With self-reproduction comes the possibility of evolution, so any variations of these self-reproducing and self-maintaining organisations that make them more stable will be selected for. By this process, the idea is that a genetic representation emerged by natural selection (in the right conditions)[15] to give the organisation a high degree of stability. With a genetic representation comes an unlimited range of hereditary variation, enabling the organisations to participate in prolonged evolutionary processes (see [Maynard Smith & Szathmáry 95] pp.67–72). This approach to the origin of life is favoured by Maturana and Varela (see, for example, Chapter 5 of [Varela 79]), and variations have also been suggested by, among others, Freeman Dyson [Dyson 85] and Stuart Kauffman [Kauffman 86].

2.3 The Pattern of Life

At this stage it is worth stepping back from the debate over definitions of life from the point of view of individual organisms, and considering the way in which life has evolved on Earth on a larger scale. We are interested in the fundamental question of which features of evolution we might expect to be universal (and therefore be apparent in any evolutionary system, natural or artificial), and which features are due to mere contingencies in the particular history of life on our planet (cf. the discussion of sex and species in Section 2.1.1). These considerations will affect the expectations we should

[15] One necessary condition is that the self-maintaining organisations were enclosed in some form of compartment, to enable natural selection to act *between* the compartments [Szathmáry & Demeter 87].

have for how an evolutionary process might unfold in an artificial system.

2.3.1 Evolutionary Progress

We might ask whether there is any underlying direction in evolution, such as a tendency for organisms to become more complicated. Although this might at first sight appear to be obviously so, the concepts of evolutionary progress and of the evolutionary increase in complexity are in fact the subject of great debate. (Note that an increase in complexity would be just one possible example of evolutionary progress; in other words, the concept of evolutionary progress is broader than that of complexity increase, and could be used in relation to many other measures.)

Countless putative examples of evolutionary progress exist in the literature, such as an increase in the maximum sizes of organisms [Bonner 88], an increased resistance of taxonomic groups to extinction [Raup & Sepkoski 82], and net progress in the expansion of life (e.g. increase in the number of species, the number of individuals, and in biomass and energy flow) [Ayala 88]. Daniel McShea criticises the evolutionary progress literature for concentrating on theorising rather than empirical inquiry, but he also warns that "marshaling cases does not document a pervasive trend either. The many increases could well be offset by an equal (or greater) number of decreases" [McShea 96] (p.477). Referring to the question of whether there has been an increase in the complexity of multicellular animals (metazoans) over the Phanerozoic Era, McShea concludes that "the evidence so far supports only agnosticism, indeed it supports an emphatic agnosticism" (*ibid.* p.489).

Almost as numerous as the number of examples of claimed evolutionary progress are the number of measures that have been used to track this progress. These measures are often intended to indicate the complexity of an organism, and include the length of coding DNA in an organism's genome [Cavalier-Smith 85], the number of different cell types in an organism [Bonner 88], and the number of species in a community (Bonner, *ibid.*).

Among the mechanisms which have been proposed to drive evolutionary progress, a common theme is the selective pressure and ecological niche opportunities that arise from other species in the ecosystem. Coevolution and symbiosis are general principles

here, and have been applied to explain evolutionary progress in a number of specific ways. Examples include Van Valen's Red Queen hypothesis, in which it is postulated that even in a constant physical environment there will be indefinite evolutionary change as each species evolves to meet changes in others in the ecosystem [Van Valen 73]. A similar scenario was suggested by Waddington ([Waddington 69] pp.115–6). On a larger scale, consideration of population sizes and spatial heterogeneity are also common factors in explanations of differentiation between populations, such as allopatric models of speciation,[16] and Wright's "shifting balance" theory.[17]

Although many different mechanisms have been proposed to explain apparent evolutionary progress, McShea notices that most are based upon a small number of underlying principles [McShea 91]. These include some kind of ratcheting process, where a progressive step, once it occurs, is not easily lost (perhaps because the new development has become very tightly integrated with the rest of the system) and is thereby able to act as a stable base for further progression. Another common principle is the repetition-and-differentiation-of-parts theme (as noted by Darwin ([McShea 91] p.308) and Maynard Smith [Maynard Smith 86] (pp.45–47) among others).

However, many would claim that there is no consistent progress in evolution. D'Arcy Thompson, in his seminal book *On Growth and Form*, remarked "That things not only alter but improve is an article of faith, and the boldest of evolutionary conceptions ... I for one imagine that a pterodactyl flew no less well than does an albatross, and that Old Red Sandstone fishes swam as well and easily as the fishes of our own seas" [Thompson 17].[18] George Williams doubted that there was any consistent notion of complexity which can be said to have increased from the Paleozoic to the present day [Williams 66] (pp.42–43), and Stephen Jay Gould certainly agrees with these sentiments (e.g. [Gould 89]).

That Darwin's theory does not in itself predict evolutionary progress is emphasised by

[16] That is, speciation occuring as the result of the restriction (or total absence) of gene flow between two populations caused by some sort of physical barrier (e.g. a sea or mountain range). See, for example, [Maynard Smith 89].

[17] In the terminology of evolutionary algorithms, Wright's theory is a mechanism whereby an evolving population can avoid getting stuck in local (sub-optimal) fitness peaks. It involves the population being divided into a large number of small, partially isolated demes. Individuals in one deme may move to a higher fitness peak by chance (i.e. genetic drift), and subsequently spread throughout the whole population. See, for example, [Wright 31] and [Wright 82]; and also [Coyne *et al.* 97].

[18] Quoted from [Nitecki 88] (p.14).

Maynard Smith: "our theory of evolution does not predict an increase in anything. At first sight, Fisher's [Fisher 30] 'fundamental theorem of natural selection' might seem to predict an increase in 'mean fitness', but it would be a mistake to think that there is any quantity that necessarily increases" [Maynard Smith 88] (p.220). Whether Darwin himself believed in the idea of evolutionary progress is itself a matter of debate (for an overview, see [Nitecki 88]). This is not surprising when some people quote passages such as: "as natural selection works solely by and for the good of each being, all corporeal and mental endowments will tend to progress towards perfection" [Darwin 59] (Chapter 14), and others quote: "After long reflection, I cannot avoid the conviction that no innate tendency to progressive development exists" [Darwin 72].[19]

At the root of much of this controversy lie the facts that the notion of progress is poorly defined, it can be used in a variety of different ways, and, above all, it is a value-laden notion. In an analysis of the concept, Francisco Ayala notes that progress "contains two elements: one *descriptive*, that directional change has occurred; the other *axiological* (= evaluative), that the change represents betterment or improvement" [Ayala 88] (p.78). It is the latter aspect of the concept that is the source of much of the disagreement. Many argue that we, as human beings, are not impartial observers, and that our value-judgements are easily influenced (even subconsciously) by cultural, political and religious considerations (e.g. [Ayala 88], [Hull 88], [Ruse 88], [Nitecki 88], [Gould 88], [McShea 91]). In stark terms, Gould says "progress is a noxious, culturally embedded, untestable, nonoperational, intractable idea that must be replaced if we wish to understand the patterns of history" [Gould 88] (p.319). The use of the term complexity is also problematic. McShea says "Complexity is more specific, more concrete, and therefore more tractable, but on account of its historical and still-commonplace association with progress, it carries the axiological taint" [McShea 91] (p.320).

2.3.2 Different Types of Complexity

McShea identifies a number of methodological problems with many previous studies of complexity increase, and remarks that "specifying an operational metric remains a difficult problem" [McShea 91] (p.318). In a later paper, he provides a useful analysis of the concept of complexity, and suggests that it can be broken down into a number

[19] Quoted from [Gould 89] (p.257).

of independent measures [McShea 96]. McShea's hope is that these measures are both operational (they can unambiguously be measured in real systems) and universal (they can be applied to all systems).

The definitions that McShea proposes are based upon two dichotomies: object versus process, and hierarchical versus nonhierarchical structure. These dichotomies yield four different types of complexity: (1) Nonhierarchical object complexity; (2) nonhierarchical process complexity; (3) hierarchical object complexity; and (4) hierarchical process complexity.[20] These different types are illustrated in Figure 2.1. In the context of his work on metazoan complexity, object complexity refers to morphological complexity, and process complexity to developmental complexity. McShea discusses various measures and proxies for obtaining these data from fossils. By taking this narrower, more specific view of complexity, a much needed degree of objectivity can be introduced. As a final remark on the notion of overall complexity, McShea says "Is a human more complex than a trilobite *overall*? The question seems unanswerable in principle because the types of complexity are conceptually independent ... Thus, it is hard to imagine how a useful notion of overall complexity could be devised" [McShea 96] (p.480).

2.3.3 Driven and Passive Trends

In his work, McShea also considers the possible causes that have been proposed to explain evolutionary trends, and highlights a broad distinction between *driven* and *passive* accounts (e.g. [McShea 94], [McShea 98]). A driven trend can occur when a directed, pervasive force exists in a system, causing a particular measure to move in one direction rather than another. However, McShea points out that trends can occur in measures even in the absence of such driving forces (i.e. passive trends). These may occur when some sort of boundary exists in the system, representing a minimum (or maximum) below (above) which the measure is unable to move. The distinguishing feature of such boundaries, compared to the biasing forces that cause driven trends, is that they only operate on a small subset of the state space. As a simple (if somewhat implausible) example, imagine a multicellular organism (consisting of a very small

[20] McShea also mentions a third possible dichotomy: differentiation versus configuration. The four types of complexity mentioned above are differentiational. Configurational complexity, on the other hand, is "irregularity of arrangement of parts and interactions, independent of their differentiation" [McShea 96] (p.480). McShea claims that configurational complexity has received little attention in the biological literature, so he does not consider it further in his work.

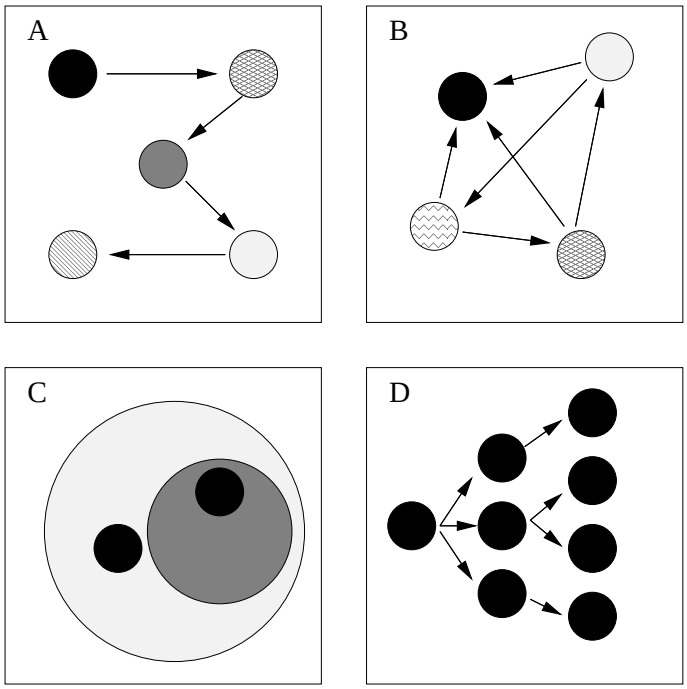


Figure 2.1: Different Types of Complexity. (A, B) Nonhierarchical complexity: A has greater nonhierarchical *object* complexity than B (it has more different parts); B has greater nonhierarchical *process* complexity than A. (C, D) Hierarchical complexity: C has a degree of hierarchical *object* complexity, D has a degree of hierarchical *process* complexity. Adapted from [McShea 96].

number of cells) which is free to evolve both in the direction of more cells and in the direction of fewer cells, with no selection pressure favouring evolution one way or the other. Both larger and smaller organisms arise by chance as evolution proceeds, so the diversity of observed sizes increases over time. Now, imagine that there is no particular upper limit on the size of the organisms which may evolve (at least, perhaps, until they consist of *millions* of cells). However, there must be a lower boundary on organism size—each organism must comprise *at least one* cell. In this situation, the mean observed organism size will increase, due to the existence of a boundary but in the absence of any directed force.

McShea suggests various ways by which these two causes may be distinguished, but remarks that "most explanations that have been proposed for complexity trends implicitly invoke biases and thus are driven ... In contrast, little has been said about possible causes of boundaries, a subject which is ripe for deeper theoretical investigation" [McShea 96] (p.486). In our work on artificial evolutionary systems it is therefore important to remember that some of the patterns of life observed in the biological world

may not be due to pervasive directed forces, but rather to the existence of boundaries.

2.3.4 Major Evolutionary Transitions

Much of the debate over evolutionary progress, based upon evidence from the fossil record, is concerned with the pattern of evolution during the Phanerozoic Era (i.e. from the first occurrence of abundant fossils of multicellular organisms to the present day). However, multicellular organisms evolved from unicellular organisms, and those, in turn, must have evolved from simpler origins [Maynard Smith & Szathmáry 95]. The one measure of progress upon which there seems to be some agreement is that of life having passed through a succession of major hierarchical transitions. In his book *The Evolution of Individuality*, Leo Buss remarks that:

> "Life, beginning as self-replicating molecules, did not persist in this initial state. Rather, it subsequently elaborated 'vehicles' in which the original heritable units became increasingly distanced from direct interaction with the external environment" [Buss 87] (p.183).

One succession that has been proposed begins with self-replicating molecules as the first stage of the evolution of life (but see Section 2.2). These molecules later grouped together into larger collections of self-replicators, which became enclosed within a membrane. From these 'protocells', prokaryotic cells evolved, and collections of prokaryotes subsequently became associated with each other to form eukaryotes. From simple unicellular eukaryotes, multicellular organisms evolved, and single multicellular organisms then grouped together in various social patterns. Such a progression has been proposed by Maynard Smith and Szathmáry ([Szathmáry & Maynard Smith 95], [Maynard Smith & Szathmáry 95]). There is still some concern over the consistency of the criteria used to define each of the transitions (e.g. [McShea 96]), but the majority involve an increase in hierarchical object complexity at least.

In Buss' treatment of the subject, he says "At Darwin's strong urging, the notion of evolution by natural selection and the notion of progress have been divorced from the outset ... [However] in the limited sense that life evolves hierarchically ... progress is inherent in the evolutionary process" [Buss 87] (p.186).

If we restrict ourselves to considering only those transitions which *do* involve an increase in hierarchical object complexity, we see that each entails the absorption of a number of previously free-living individuals into a new, hierarchically higher, unit of selection. When a transition of this nature occurs, two major factors are involved [Buss 87]. The first is a synergism between the interests of the individuals at the lower level and the new 'individual' at the higher level. This synergism "will act to create novel organizations allowing exploitation of the external environment in ways that the lower unit alone could not accomplish" (*ibid.* p.185). However, the individuals at the lower level might evolve in ways that disrupt the higher level organisation. Thus, the second factor is a conflict between the two levels of selection. "The rate and magnitude of such conflicts must be limited, or the higher unit will perish. If variants arise in the lower unit whose effect is to limit the occurrence or magnitude of subsequent variation, then the higher unit will eventually become resistant to further perturbation" (*ibid.* p.185).

Once a transition has occurred, then, the new unit of selection must become stabilised if it is not to break down into its component subparts again. (Maynard Smith and Szathmáry also suggest that the genetic relatedness of individuals in the lower level may also play an important role in conferring immediate selective advantage at the time of transition [Maynard Smith & Szathmáry 95] (p.8).) If the new organisation *does* become stabilised, then it, in turn, is able to participate in the same sort of process, leading to yet higher organisational levels. Bronowski's term "stratified stability" captures this picture of progression well [Bronowski 73].[21]

Buss summarises: "the organization of any unit will come to reflect those synergisms between selection at the higher and the lower levels which permit the new unit to exploit new environments and those mechanisms which act to limit subsequent conflicts between two units" [Buss 87] (p.viii).

From this point of view, a new perspective on the pattern of evolution emerges. It is a picture of major hierarchical transitions, with long periods in between during which it is debatable whether any consistent notion of progress applies.[22] Buss claims that

[21] Although Bronowski used the term in a somewhat broader context.

[22] Note that I am *not* talking about the distinction between gradualism and punctuated equilibria here; this view of evolution *does* allow for evolution in the periods between major transitions. However, the view holds that evolution during these periods will not necessarily be in the direction of increased complexity.

"the major features of evolution were shaped during periods of transition between units of selection" [Buss 87] (p.188). In the period after a transition, an adaptive radiation of forms is possible to fill the novel evolutionary niches afforded by the new kind of organisation. In the context of organic evolution, Salthe suggests that this expansion continues until a saturation point of number of kinds is reached, "when the emphasis of the process ... shifts toward coevolutionary elaborations of pairs, guilds, and even more complex symbioses" [Salthe 85] (p.253). The saturation point can be released if, for example, a subset of the organisms become geographically isolated and able to participate in a new adaptive radiation, but none of these processes will necessarily lead to any general evolutionary progress.

This view of evolution is also consonant with Schwemmler's notion of a constant oscillation between divergent and convergent phases [Schwemmler 89], and, to some extent, with Gould's picture of diversification and decimation [Gould 89].[23]

2.3.5 Contingency in Evolution

Proponents of the general picture of evolution discussed in the previous section tend to emphasise the role of contingency ('historical accidents') in determining the course of evolution. In *The Evolution of Individuality*, Buss says:

> "the fact that life is hierarchical redirects attention to the effect of history on biological phenomena. The major features of genomic organization, of cell architecture, and of organismal ontogeny arose as the products of history-dependent variation at the time of the transition from one unit of selection to another. Units which persist today do so because variants which restrained further interaction between two units of selection arose and fixed the organization of the unit in question at a given state" [Buss 87] (p.187).

Similarly, Maynard Smith says "My own view is of a series of historical accidents, subject to engineering constraints on the one hand, and to the conservatism of development on the other" [Maynard Smith 86] (p.45). The 'engineering constraints' arise when there may be only a small number of ways of solving a particular engineering

[23] Although Gould's view is also connected to the concept of punctuated equilibria, which is not specifically related to transitions between units of selection.

problem faced by a species. By 'conservatism of development', Maynard Smith is referring to the fact that "it seems to be a general feature of evolution that new functions are performed by organs which arise, not *de novo*, but as modifications of pre-existing organs" (*ibid.* p.46).

Finally, Gould's view on the subject is as follows:

> "Invariant laws of nature impact the general forms and functions of organisms; they set the channels in which organic design must evolve. But the channels are so broad relative to the details that fascinate us! The physical channels do not specify arthropods, annelids, mollusks, and vertebrates, but, at most, bilaterally symmetrical organisms based upon repeated parts ... When we set our focus upon the level of detail that regulates most common questions about the history of life, contingency dominates and the predictability of general form recedes to an irrelevant background ... Charles Darwin recognized this central distinction between 'laws in the background' and 'contingency in the details' ..." [Gould 89] (pp.289–290).

2.4 Playing the Game

Having discussed some of the issues that arise when considering how evolution has unfolded on Earth, it is time to ask how this affects our attempts at producing a definition of life. The distinction has already been made between evolutionary and ecological views of life (Sections 2.1.1 and 2.1.2 respectively), and in Section 2.1.3 it was noted that an adequate picture of living things would require a combination of these perspectives. However, it was also noted that it is hard to imagine a succinct definition which could capture this.

The discussion in the previous section only serves to emphasise the entanglement between the evolutionary and ecological pictures. Living organisms are taking part in the 'game' of life. Fundamental aspects of this game include ecological interactions with other organisms and with the abiotic environment, the exchange and transformation of resources, and the coevolution of all organisms, together with aspects of their abiotic environment. If we define life in terms of an individual organism, it is easy to underestimate the importance of that organism's interactions with the rest of the

world. These interactions are inseparably linked to the concept we refer to as 'life'. Organisms exist both as products and as players in this game. It is only as both of these things that an individual organism can be said to be 'alive'.

2.5 Definition of Terms

Before continuing it is worth clarifying some of the terminology relating to evolutionary systems that will be used in subsequent chapters. It has been noted by various people that some confusion exists in the use of terms such as *genes*, *replicators* and *lineages* (e.g. [Hull 80], [Salthe 85], [McMullin 95]). The confusion arises because terms such as gene and replicator are sometimes used to refer to an *individual* instance of a specific form, e.g. a specific portion of DNA within one particular organism, and sometimes to refer to a *class* of forms, members of which can be considered identical from the point of view of their acting in an evolutionary process (e.g. it is in this sense that we refer to a gene being *passed on* (*replicated*) from one generation to the next).[24]

In an attempt to remove this ambiguity between referring to tokens (individual instances) and classes, McMullin introduces two prefixes, 'A-' (standing for *Actor*) and 'L-' (standing for *Lineage*) [McMullin 95]. The former signifies an individual token, and the latter signifies a class (read the 'L-' prefix as 'a lineage of ...'). For example, an A-replicator is an individual (molecule, strand of DNA, etc.), and an L-replicator is a lineage of A-replicators. McMullin goes on to single out lineages of replicators in which all members are identical to the founder of the lineage with respect to a particular selectively relevant character. These lineages, which he calls similarity lineages, or S-lineages, are, he argues, the fundamental units of selection in a Darwinian evolutionary process.[25] Phrases such as "genes are potentially immortal" refer to L-genes, and, in particular, to S-lineages. Although McMullin regrets the introduction of yet

[24] Salthe, in considering the hierarchical nature of life, argues that there is a clear distinction between the ecological hierarchy and the genealogical (evolutionary) hierarchy [Salthe 85]. The confusion in terminology under discussion here can also be seen as a confusion between entities in these two distinct (but related) hierarchies.

[25] That is to say, S-lineages are the fundamental *types of things* which act as units of selection. McMullin is *not* arguing here about whether it is more appropriate to look at evolution from the point of view of the gene or the organism, for example, but claims that "*whichever* of these viewpoints may be adopted, there will be a crucial distinction between actors and S-lineages, with only the latter being properly regarded as units of selection, or entities for whose benefit (Darwinian) adaptations may be said to exist" [McMullin 95] (p.166, original emphasis).

more terminology (as do I), he sees no other way to disambiguate confusing terms such as 'replicator'. I shall therefore use these prefixes throughout this thesis in situations where potential ambiguities may otherwise arise.

We are now in a position to be somewhat more precise about the meaning of a number of concepts that were used in Chapter 1.

Open-Ended Evolution. This term refers to a system in which components continue to evolve new forms continuously, rather than grinding to a halt when some sort of 'optimal' or stable position is reached. In other words, using the terminology just introduced, new, adaptively successful, S-lineages are continuously appearing and displacing existing S-lineages. Note that open-ended evolution does not necessarily imply any sort of evolutionary *progress*. Also note that this is still not a perfect definition; there are some more subtle issues involved in determining what counts as 'new' in this context. Also, by using the term 'open-ended' I wish to imply that an indefinite variety of phenotypes are attainable through the evolutionary process, rather than continuous change being achieved by, for example, cycling through a finite set of possible forms. I will return to some of these issues in Chapter 7 (for example, in Sections 7.1.2 and 7.3.2).

Complexity. The distinctions proposed by McShea, described in Section 2.3.2, will be adopted here, so we will not use the term 'complexity' unqualified. As the major evolutionary transitions described in Section 2.3.4 were critical points in the evolution of life on Earth, we will be especially interested in investigating hierarchical complexity (primarily hierarchical object complexity) in artificial evolutionary systems.

Life. Unfortunately we seem to be no closer to arriving at a satisfactory definition of life. Maturana and Varela's concept of autopoiesis may be adequate for the classification of individual organisms. However, as we are ultimately interested in *creating* artificial life, or at least in understanding why life exists on Earth, we need to look at a somewhat broader picture which considers the generative processes by which life has emerged. Both evolutionary and ecological considerations will be important for this purpose. We will keep these ideas in mind when developing our artificial models, but specific research

questions will be phrased in terms of more specific concepts such as 'hierarchical object complexity' and 'open-ended evolution'. In Chapter 7 we will revisit the problem of providing an adequate characterisation of the concept of life, in the light of experience gained from building and experimenting with an 'artificial life' system (described in Chapters 4–6).

2.6 Summary

In this chapter we have discussed the two distinct ways of looking at life: the evolutionary and the ecological pictures. Although distinct from a conceptual point of view, both processes have contributed in an intricately interrelated manner to the pattern of life we see on Earth. Both views are therefore necessary to gain a satisfactory picture of life.

The concept of evolutionary progress was also discussed. Despite common preconceptions, evidence for evolutionary progress is only available in a relatively small number of fairly specific cases. Whether evolutionary progress, and in particular an evolutionary increase in complexity, has been a general feature of biological evolution is a subject of much debate. Despite the lack of evidence, many theories have been put forward to explain how progress may come about. Common mechanisms that have been utilised in these theories include some kind of ratcheting, and the idea of repetition-and-variation-of-parts. For explaining speciation and other macroevolutionary patterns, consideration of population sizes and spatial heterogeneity is also often employed.

One sense of progress on which there does seem to be broad agreement is that life has evolved in a number of hierarchical steps, starting with individual replicating molecules, and progressing through stages including unicellular organisms and then multicellular organisms. Although the details of each transition may be different, and depend on fairly particular, contingent factors, it is possible that some common features exist. The level of explanation at which these commonalities will be clearest is likely to be in the consideration of synergisms and conflicts between levels of selection. It has also been suggested that many of the major features of life are forged at the time of such transitions.

In the next chapter, a review is presented of the many ways in which people have

applied the ideas discussed here to computer simulations of evolution and life, and to the creation of 'artificial life'.

Chapter 3

Artificial Life

> "*You know, the universe is the only thing big enough to run the ultimate game of life. The only problem with the universe as a platform, though, is that it is currently running someone else's program.*"
>
> Ken Karakotsios[1]

3.1 What is Artificial Life?

Chris Langton, whom many regard as the founder of the research discipline known as Artificial Life, defines the subject as the study

> "of man-made systems that exhibit behaviors characteristic of natural living systems. It complements the traditional biological sciences concerned with the *analysis* of living organisms by attempting to *synthesize* life-like behaviors within computers and other artificial media. By extending the empirical foundation upon which biology is based beyond the cabon-chain [*sic*] life that has evolved on Earth, Artificial Life can contribute to theoretical biology by locating *life-as-we-know-it* within the larger picture of *life-as-it-could-be*." [Langton 88] (p.1, original emphasis).

As a brief aside, one might argue that the implication in this quotation that traditional theoretical biologists are concerned only with terrestrial biology is not entirely accurate.

[1] Quoted from [Kelly 94] (p.451).

Although the approach may be different, the goal of attempting to develop a truly universal theory of biology (i.e. a general theory which might, for example, predict the forms life might take on other planets, in contrast to our rather provincial view of life on Earth) is one that is shared by many biologists—for example, John Maynard Smith (e.g. [Maynard Smith 86] pp.21–23) and Richard Dawkins (e.g. [Dawkins 83]), to name but two.

Langton's publications in the mid-1980s in which he brought the term 'artificial life' into common usage (e.g. [Langton 86]),[2] together with his organisation of a workshop on the subject in Los Alamos in September 1987 (which has subsequently grown into the biennial International Conference on Artificial Life), were certainly the principal factors contributing to the current popularity of the subject (and to its crystallisation as a coherent subject[3]). However, research which would today certainly be classified as 'artificial life' began almost as soon as the modern digital computer was developed, with Nils Barricelli's work at the U.S. Institute for Advanced Study commencing in 1953, described in Section 3.2.2. The theoretical background for the subject was being developed even earlier, by von Neumann [von Neumann 66], for example, and by proponents of the cybernetics field (e.g. [Weiner 48]). A good history of much (but certainly not all) of this earlier work is provided by Langton himself in [Langton 88].

3.1.1 Weak Artificial Life versus Strong Artificial Life

The word 'artificial' is commonly used in two distinct ways. One carries the connotation that the thing being referred to is a fake—an imitation of something else (e.g. 'artificial flowers'), but the other merely suggests that the process by which the referent was produced is man-made or unnatural in some sense, but the thing itself is just as real as the original (e.g. 'artificial light').

These two uses of the term correspond to the contrasting philosophical positions of *Weak* and *Strong* Artificial Life respectively. Proponents of Weak Artificial Life view their work as an attempt to learn more about biological life (or to create artefacts

[2] Although others had used the term in earlier publications, such as [Overton 82] (in which it appears in the title without definition in the text, and in fact, by comparing the contents of the main text with that of the editor's note at the beginning, it appears that the title (including the term 'artificial life') was chosen by the editor, Edmund C. Berkeley, rather than by Overton himself).

[3] Although some would argue over the extent to which it *is* a coherent subject. See Sections 3.1.1–3.1.3.

such as robots) by creating synthetic models of various processes associated with living organisms (e.g. evolution) on computers and other artificial media, but do not claim that any parts of their models are actually living themselves. On the other hand, proponents of Strong Artificial Life claim that by instantiating such processes in artificial media, the end product will be just as deserving of the term 'living' as are biological organisms. In other words, the former group see their computer programs as *simulations* of life, while the latter group see them (potentially) as *realisations* of life [Pattee 88].

This distinction is related in some ways to the argument between those who claim that life depends crucially upon the physical medium in which it is implemented, and those who claim that life is fundamentally a *process* (or a collection of processes) and as such can be implemented in any physical medium which has the logical structure to support such processes. Arguments for the former view have been put forward by, for example, Elliott Sober [Sober 91], Claus Emmeche [Emmeche 92], Eric Olson [Olson 97], and Margaret Boden [Boden 99] (although she claims that this depends on how we define metabolism). Proponents of the latter point of view, that life is independent of matter, include Langton himself [Langton 86] and Tom Ray [Ray 91], creator of the 'Tierra' system (described in Section 3.2.1). A version of the former view which offers more hope for the endeavour of Artificial Life, and the view with which I have most sympathy, is that life must be embedded within a 'physical' environment, but that the symbolic environment provided by a computer program (that is, an 'artificial physics') *might* be sufficient for this purpose (e.g. [Pattee 88], [Rasmussen 91], [Pattee 95a], [Bedau 98b]). However, this view comes with the proviso that the artificial physics of the computer model must resemble the important aspects of the physics of the real world if the model is to be treated as science rather than just a computer game (see, for example, [Moreno *et al.* 94], [Morán *et al.* 97] and [Ruiz-Mirazo *et al.* 98]). A major challenge for the science of artificial life will be to develop theories of exactly which aspects of the real world are important in this sense.

A glance at any of the recent proceedings of conferences on artificial life (for example, [Adami *et al.* 98], [Husbands & Harvey 97]) reveals that the field now encompasses a very wide variety of research, including synthetic evolutionary models, artificial chemistries, models of autopoiesis, self-organising systems, evolutionary robotics and evolutionary hardware. The methods of artificial life are now also being employed

for artistic purposes,[4] and work of this nature is also being integrated into these conferences. The majority of current work in the field (at least as published in scientific journals and conferences) does not postulate any more than the weak position. In other words, the premise is that by modelling particular natural processes such as evolution, we can improve our understanding of those processes and their relevance to the biological world, or, alternatively, we can profitably use such processes for various practical purposes (e.g. to develop robust control systems for robots).

However, a minority of published artificial life research explicitly takes the strong position. The protagonists, not surprisingly, include those who are most vociferous about life being independent of matter, such as Langton and Ray. For example, Ray says "I would consider a system to be living if it is self-replicating, and capable of open-ended evolution" (*ibid.* p.372).[5] Ray's requirement that living systems have the capacity of open-ended evolution is similar to Bedau's concept of life as supple adaptation, discussed in Section 2.1.1. However, the definition does not prescribe how such a capacity can be assured. In the following chapters, much time will be spent investigating the behaviour of an artificial life platform called 'Cosmos', based upon Tierra, in which populations of self-replicating computer programs evolve. I have developed this system largely to study its evolutionary behaviour, but I certainly do not equate self-replication with life. However, I think the relationship between open-ended evolution and life is more interesting. After reporting the lessons learned from running an extensive series of experiments with Cosmos (in Chapters 5–6), I will return to a further discussion of this relationship in Chapter 7.

3.1.2 Is Artificial Life Possible?

As mentioned in the previous chapter (Section 2.1.3), without a clear definition of the word 'life', questions such as "is artificial life possible?" really have no meaning. However, assuming that the question can be suitably rephrased in appropriately concrete

[4] For example, see Steven Rooke's online portfolio at `http://www.concentric.net/~Srooke/` and the list of genetic art-related sites at `http://gracco.irmkant.rm.cnr.it/luigi/alg_art.htm`.

[5] The concern with self-replication has also been a preoccupation for other artificial life researchers from von Neumann [von Neumann 66] to the present day, and can be related to the work of biologists such as Muller and Dawkins, discussed in Section 2.2.

terms,[6] there are still a number of philosophical and practical hurdles to be overcome.

A major practical barrier to creating artificial life is simply the vastness of life in the biological realm, both in terms of the numbers of molecules comprising the biosphere, and of the billions of years over which evolution has been proceeding. Even if artificial life were a theoretical possibility, why should we expect it to proceed any faster on a computer only capable of supporting a trivially small population?

If we are eventually hoping to produce *intelligent* artificial life, the chances seem even slimmer. Gould remarks that "*Homo sapiens* is an entity, not a tendency" [Gould 89] (p.320). Expanding upon this, he says:

> "Life arose at least 3.5 billion years ago, about as soon as the earth became cool enough for stability of the chief chemical components ... [However], a good deal more than half the history of life is a story of prokaryotic cells alone, and only the last one-sixth of life's time on earth has included multicellular animals ... But cosmologists tell us that the sun is just about at the halfway point of existence in its current state; and that some five billion years from now, it will explode ... Since human intelligence arose just a geological second ago, we face the stunning fact that the evolution of self-consciousness required about half of the earth's potential time. Run the tape again, and even if the same general pathways emerge, it might take twenty billion years to reach self-consciousness this time—except that the earth would be incinerated billions of years before." [Gould 89] (pp.309–311).

Another argument against the possibility of artificial life concerns the nature of the major evolutionary innovations in biology, which seem to have provided organic evolution with an unlimited supply of phenotypic novelty. It has been suggested by Howard Pattee that most of these innovations involve the invention of techniques for measuring new aspects of the environment, and of arbitrarily mapping these patterns to symbols or to specific actions [Pattee 88]. Pattee emphasises that one of the major hurdles for

[6] This, of course, is a big assumption. Those who maintain that life is of necessity a biochemical phenomenon would argue that concepts such as autopoiesis do *not* capture all of the relevant features of life. However, this eventually boils down to a linguistic debate, and scientific progress can only be made when we do adopt precise definitions, even if these are not universally agreed upon.

(strong) artificial life will be to develop a satisfactory theory of measurement which suggests how we might equip our programs with the potential to evolve new measurement devices (*ibid.*). The evolution of new forms of action (e.g. new effectors) is equally problematic.

Furthermore, such innovations usually develop from existing phenotypic structures employed for unrelated functions, but which happen to perform (however crudely) the novel task as a side-effect. To the extent that this is true, this presents a problem for the designers of artificial life systems. A computer model, like any other scientific model, is an idealisation of the real world, designed to model (what the designer thinks are) the important components of a system, while ignoring irrelevant aspects. However, the present argument suggests that a primary source of evolutionary novelty might lie in precisely those aspects of the physical world which might ordinarily be deemed irrelevant for the purposes of constructing a model. We are again confronted by our lack of an adequate theory of measurement, as pointed out by Pattee. Even if we were to model a wide variety of properties for each component in our artificial life system, we might only expect complex forms to emerge if they existed in a very complex environment which afforded many possibilities for perception and action. Some steps which we might take to ensure a rich environment are discussed in Chapter 7, but the issue of evolving novel sensors and effectors in a computer model remains problematic.

These are formidable arguments, and there are a number of others which can also be made against the likelihood of success (see [Bonabeau & Theraulaz 94]). The number of points that computer models have in their favour seem to be few in comparison (speed of execution might be one example).[7] I take the pragmatic attitude that we simply do not know how successful the approach might be until we try it. Eric Bonabeau and Guy Theraulaz argue that "rather than true limitations, [issues such as these] constitute questions asked to AL [Artificial Life]. And AL is precisely a constructive way of checking whether these limitations are real obstacles" [Bonabeau & Theraulaz 94] (p.314). A detailed discussion of the philosophy of artificial life can be found in [Boden 96]. Even if the quest for strong artificial life ultimately ends in failure, the attempt can highlight questions and gaps in our knowledge which were not apparent, or did not seem particularly relevant, beforehand, and indeed it already has done.

[7] But note that artificial life does not restrict itself to using digital computers as the medium of implementation.

3.1.3 Relation to Theoretical Biology

Traditionally, most models used in ecology and evolutionary biology have been systems of differential equations. These equations track changes in macroscopic measures of the system being modelled, such as the frequency of an allele in a population's gene pool, or the amount of energy in a trophic level. This approach is appropriate for modelling a wide variety of systems, and has led to very successful models in numerous cases. It is, however, subject to a number of limitations. Charles Taylor (a biologist) and David Jefferson (a computer scientist) give some examples:

> "For example, in many models it is common to refer to the derivative of a variable with respect to population size N. This in turn implies the assumption of very large populations in order for such a derivative to make sense, which has the effect of washing out small population effects, such as genetic drift, or extinction. Another difficulty is that it would take tens to hundreds of lines of equations to express even a simple model of an organism's behavior as a function of the many genetic, memory, and environmental variables that affect its behavior, and there are simply no mathematical tools for dealing with equational systems of that complexity. Furthermore, equational models are generally poor at dealing with highly nonlinear effects such as thresholding or if-then-else conditionals, which arise frequently in the description of animal behavior." [Taylor & Jefferson 94].

Additionally, it is difficult to satisfactorily model spatial inhomogeneity and incomplete mixing with this approach, especially when considering continuous rather than discrete spatial structure (see [van Baalen & Rand 98], [van Baalen 98]). In discussing ecological modelling techniques, M.A.R. Koehl refers to models of macroscopic measures of a system as phenomenological models. He says of the shortcomings of such models:

> "The limitations of phenomenological models render them inappropriate for certain types of analysis. Whenever we make a prediction using a phenomenological model, we implicitly assume (1) that conditions do not change, and (2) that the phenomena that go into the model adequately sample the causal pattern of interest. Therefore, phenomenological models are best for making short-term predictions." [Koehl 89].

In other words, by their very nature differential equation models require that all components to be modelled are explicitly specified in the equations, so they cannot generally be used to model the emergence of *new* components.

In contrast, artificial life models dispense with equations, and represent individuals explicitly. That is, the artificial life approach is fundamentally synthetic rather than analytic (see Section 3.1). Artificial life models are similar to what Koehl refers to as mechanistic models in ecology [Koehl 89], where it is the essential processes governing components of the system that are modelled. As computers become faster and more powerful, it becomes possible to model larger populations of individuals, and to use a more sophisticated representation for each individual. Indeed, the growth of interest in artificial life in the late 1980s can be largely attributed to the fact that sufficiently powerful computers first became readily available at around that time. However, some interesting and worthwhile studies were conducted a long time before then, and will be reviewed in Section 3.2.

Artificial life models may be compared to microcosm experiments in ecology [Koehl 89], where a limited ecology of organisms with small size and short generation times are studied in the hope that they may reveal basic principles which are valid for larger systems as well. Such microcosm experiments are also easier to control and replicate than are larger-scale studies, and this capacity for controlled experimentation is even greater in computer-based artificial life models. In his discussion of microcosm models, Koehl warns that they "are certainly useful to test models, but are they so unrealistic that they tell us little about nature?" (*ibid.* p.43). One of his main concerns is the size of the microcosm compared to natural systems, and similar worries about the size of artificial life models were raised in Section 3.1.2. Koehl takes the pragmatic view that we should not forget about such issues, but use microcosm studies in order to "pursue sensible answers to these questions" (*ibid.* p.43). I have a similar view of artificial life studies, as discussed in Section 3.1.2.

In considering the scientific status of artificial life models, Holland argues that computer simulations (weak artificial life) can be seen as a bridge between experimentation and theory—a sort of half-way house on the road to a more rigorous mathematical treatment (which might not be possible until new mathematical tools are developed) [Holland 95]. Daniel Dennett suggests that they can also be used as a bridge between experimentation

and philosophy [Dennett 94]. Jason Noble, adopting the philosophical viewpoint known as 'unrepresentative realism', argues that artificial life models *can* be treated as scientific theories in their own right, but *only* if appropriately formulated (e.g. they must be based upon explicit axioms)[8] [Noble 97]. In particular, he points out that they are useful for testing the logical coherence of a given set of assumptions. Pattee also emphasises the need for simulations to be based upon solid theories, and additionally discusses the conditions under which we might consider an artificial life program as a *realisation* of life rather than a *simulation* of life [Pattee 88].

A number of people have warned of the dangers of taking the concept of 'life-as-it-could-be' too far, and argue that artificial life can only be treated as science if it sufficiently reflects the constraints and boundary conditions operating in the real world (e.g. [Bonabeau & Theraulaz 94], [Morán *et al.* 97], [Noble 97]). Geoffrey Miller has suggested that the science of artificial life should restrict itself to tackling established problems of theoretical biology [Miller 95], although others have argued (and I agree, at least if Miller's suggestion is taken too restrictively) that this goes too far, and that the approach opens up new areas for research which were not amenable to more traditional methods [Di Paolo 96].

The nature of artificial life models makes them suitable for studying emergent phenomena and open-ended evolution, not least because they have the potential for new types of individuals or components to develop within them [Miller 95]. Spatial structure can often be modelled more easily using this approach than by using more traditional theoretical biological methods (see, for example, [Boerlijst & Hogeweg 91], [Collins 92], [Nuño *et al.* 95]).

As should be apparent from the above, the artificial life approach, like more traditional theoretical biological approaches, has both strong and weak points. The synthetic and analytic approaches are complementary, and ideally they can reinforce each other. Further discussion of the relationship between artificial life and theoretical biology can be found in [Collins 92], [Taylor & Jefferson 94], [Bonabeau & Theraulaz 94], [Miller 95], [Roughgarden *et al.* 96] and [Toquenaga & Wade 96].

[8] David Marr's [Marr 82] analysis of complex systems on three levels (the computational theory level, the representation and algorithm level, and the hardware implementation level), and his insistence that the theoretical distinction between these levels should be recognised when devising a model—especially the distinction between *what* is being computed (level 1) and *how* (level 2)—is also relevant for the formulation of scientific artificial life models.

3.2 Previous Work

In this section, a review is presented of some previous artificial life studies. This is not a comprehensive review of the whole field, but focuses on work related to the central topics of this thesis, such as self-reproduction, open-ended evolution and self-maintenance. Coverage of some of the more important contributions to the field that are not discussed here can be found in [Fogel 98].

3.2.1 Self-Reproduction

Nearly all of the practical implementations of artificial worlds to be described are related in some way to the seminal work of John von Neumann. In the late 1940s and early 1950s, he devoted considerable time to the question of how complicated machines could evolve from simple machines.[9] Specifically, he wished to develop a formal description of a system that could support self-reproducing machines which were robust in the sense that they could withstand some types of mutation and pass these mutations on to their offspring. Such machines could therefore participate in a process of evolution.

Inspired by Alan Turing's earlier work on universal computing machines [Turing 36], von Neumann devised an architecture which could fulfil these requirements. The machine he envisaged was composed of three subcomponents [von Neumann 66]:

1. A general *constructive* machine, $\mathbf{A}$, which could read a description $\phi(\mathbf{X})$ of another machine, $\mathbf{X}$, and build a copy of $\mathbf{X}$ from this description:

$$\mathbf{A} + \phi(\mathbf{X}) \rightsquigarrow \mathbf{X} \tag{3.1}$$

   (where + indicates a single machine composed of the components to the left and right suitably arranged, and $\rightsquigarrow$ indicates a process of construction.)

2. A general *copying* machine, $\mathbf{B}$, which could copy the instruction tape:

$$\mathbf{B} + \phi(\mathbf{X}) \rightsquigarrow \phi(\mathbf{X}) \tag{3.2}$$

[9] Von Neumann had difficulties in defining precisely what the term 'complicated' meant. He said "I am not thinking about how involved the object is, but how involved its purposive operations are. In this sense, an object is of the highest degree of complexity if it can do very difficult and involved things." [von Neumann 66].

3. A *control* machine, **C**, which, when combined with **A** and **B**, would first activate **B**, then **A**, then link **X** to $\phi(\mathbf{X})$ and cut them loose from $(\mathbf{A}+\mathbf{B}+\mathbf{C})$:

$$\mathbf{A}+\mathbf{B}+\mathbf{C}+\phi(\mathbf{X}) \rightsquigarrow \mathbf{X}+\phi(\mathbf{X}) \tag{3.3}$$

Now, if we choose **X** to be $(\mathbf{A}+\mathbf{B}+\mathbf{C})$, then the end result is:

$$\mathbf{A}+\mathbf{B}+\mathbf{C}+\phi(\mathbf{A}+\mathbf{B}+\mathbf{C}) \rightsquigarrow \mathbf{A}+\mathbf{B}+\mathbf{C}+\phi(\mathbf{A}+\mathbf{B}+\mathbf{C}) \tag{3.4}$$

This complete machine plus tape, $[\mathbf{A}+\mathbf{B}+\mathbf{C}+\phi(\mathbf{A}+\mathbf{B}+\mathbf{C})]$, is therefore self-reproducing. From the point of view of the evolvability of this architecture, the crucial feature is that we can add the description of an arbitrary additional automaton **D** to the input tape. This gives us:

$$\mathbf{A}+\mathbf{B}+\mathbf{C}+\phi(\mathbf{A}+\mathbf{B}+\mathbf{C}+\mathbf{D}) \rightsquigarrow \mathbf{A}+\mathbf{B}+\mathbf{C}+\mathbf{D}+\phi(\mathbf{A}+\mathbf{B}+\mathbf{C}+\mathbf{D}) \tag{3.5}$$

Furthermore, notice that if the input tape $\phi(\mathbf{A}+\mathbf{B}+\mathbf{C}+\mathbf{D})$ is mutated in such a way that the description of automaton **D** is changed, but that of **A**, **B** and **C** are unaffected—that is, the mutated tape is $\phi(\mathbf{A}+\mathbf{B}+\mathbf{C}+\mathbf{D}')$—then the result of the construction will be:

$$\mathbf{A}+\mathbf{B}+\mathbf{C}+\mathbf{D}+\phi(\mathbf{A}+\mathbf{B}+\mathbf{C}+\mathbf{D}) \overset{\text{mutation}}{\rightsquigarrow} \mathbf{A}+\mathbf{B}+\mathbf{C}+\mathbf{D}'+\phi(\mathbf{A}+\mathbf{B}+\mathbf{C}+\mathbf{D}') \tag{3.6}$$

The reproductive capability of the architecture is therefore robust to some mutations (specifically, those mutations which only affect the description of **D**), so the machines are able to evolve. Von Neumann pointed out that it was the action of the general copying automaton, **B**, which gave his architecture the capacity for evolving machines of increased complexity, because **B** is able to copy the description of any machine, no matter how complicated [von Neumann 66] (p.121). This ability is clearly demonstrated in Equation 3.5 above.

The original implementation envisaged by von Neumann was a constructive system, which Burks has referred to both as the 'robot model' and as the 'kinematic model' [Aspray & Burks 87] (p.374). However, von Neumann decided that the system was too complicated to capture in a set of rules that were both simple and enlightening, so he turned his attention to developing the cellular automata (CA) framework with Stanislaw Ulam. Von Neumann described the detailed design of a self-reproducing machine

in a cellular automata space, according to the architecture described above.[10] Recently, a slightly modified and simplified version of this design was successfully implemented on a computer [Pesavento 95]. One of the major achievements of von Neumann's work was to clarify the logical relationship between *description* (the instruction tape, or genotype), and *construction* (the execution of the instructions to eventually build a new individual, or phenotype) in self-replicating systems. However, as already mentioned and as emphasised recently by Barry McMullin (e.g. [McMullin 92a]), his work was always within the context of self-replicating systems which would also possess great *evolutionary* potential.

Von Neumann concentrated on the *logic* required for a self-replicating machine to be able to evolve increased complication. He therefore did not specifically deal with various biological concerns, most notably concerns of energy. In fact, von Neumann was well aware of these concerns and warned that by restricting attention to the *logic* of self-reproduction, "one has thrown half the problem out the window, and it may be the more important half" [von Neumann 66].[11]The other half of the problem to which he is referring is that of the molecular phenotype [Pattee 95a]. Burks says of the kinematic model that "von Neumann intended to disregard the fuel and energy problem in his first design attempt. He planned to consider it later, perhaps by introducing a battery as an additional elementary part" [Aspray & Burks 87] (p.485). Another major difference between biological organisms and von Neumann's self-replicators is the capacity of the former, but not the latter, for self-maintenance in the face of environmental perturbations. Alvy Ray Smith has pointed out [Smith 91] that some of the CA-based self-reproduction models developed by von Neumann and more recently by others are very non-biological in other ways as well; for example, reproduction in CA models does not occur by development from an 'egg', most models suffer from 'overcrowding' such that an individual self-replicator can only reproduce once (or a small number of times) before it runs out of space in which to place its offspring, etc.

A good review of subsequent work on self-reproduction in CAs is presented in [Sipper 98].

---

[10] The general constructive machine **A** of this design is often referred to as a 'universal constructor'. However, this term should be used with caution; from the above description of the architecture it is clear that **A** can build any machine **X** *that can be described upon a tape* $\phi(\mathbf{X})$. For cellular automaton models it can be proved that there are some configurations that the universal constructor cannot build (e.g. [Moore 62], [Myhill 63]). These are referred to as 'Garden of Eden' configurations, as the only way they may exist is if they are programmed in as the initial state of the space at time zero.

[11] Quoted from [Pattee 88] (p.69).

Much of this more recent work (e.g. [Langton 84], [Ibáñez *et al.* 95], [Tempesti 95], [Perrier *et al.* 96], [Morita & Imai 97]) has the kind of 'non-biological' character mentioned above. In fact, as pointed out by Barry McMullin, much of this work does not even seem to share von Neumann's concern with the evolution of increased complication, but addresses the 'problem' of self-reproduction in and of itself as if this was all that von Neumann was concerned with [McMullin 92a]. For example, Chris Langton described a much simpler self-replicator in a CA, by dropping von Neumann's requirement for universal construction [Langton 84]. Langton, like von Neumann, asserts that the self-replicators must treat their stored information in two different manners: "*interpreted*, as instructions to be executed (translation), and *uninterpreted*, as data to be copied (transcription)" (*ibid.* p.137, original emphasis), but he only insists on these requirements in order to avoid the problem of 'trivial' self-reproduction[12] rather than from any concerns relating to evolvability. The resulting automaton, referred to as 'Langton's loop', is very fragile in that it cannot in general withstand perturbations and mutations, and is certainly not capable of heritable viable mutation. McMullin describes Langton's work as a "cruel (though of course unintentional) parody" of von Neumann's [McMullin 92a] (p.181). This 'myth' that von Neumann was concerned with self-reproduction in and of itself has by no means been dispelled even now; for example, an introduction to a paper at a recent European Conference on Artificial Life reads "The first steps in [developing a universal theory of evolution] were the cellular automata of von Neumann, *who only studied self-replication.* Among other [*sic*], Langton and Sipper *also* addressed the question of evolution." [de Dinechin 97] (p.560, emphasis added).

As already mentioned, these studies do not generally consider the ability of the automaton to actively maintain its own structure in the face of environmental perturbations. This deficiency has certainly been recognised for a long time (e.g. [Arbib 69] and, more recently, [McMullin 92a]), but very little work has so far been done to create more robust, self-maintaining, self-reproducing CAs (but see Section 3.2.3). Indeed, to anti-

[12] 'Trivial' self-reproduction, at least in the sense being used by Langton, occurs when reproduction of a particular sort of configuration happens purely due to the rules of the system rather than to anything explicitly encoded in the configuration itself. For example, a CA with a transition rule such as "if this cell is empty (in the quiescent state) and one of its neighbouring cells is in state $\mathcal{A}$, then change the state of this cell to $\mathcal{A}$" is an environment in which the state $\mathcal{A}$ trivially self-replicates. I suspect that Langton's work was the seed for the preoccupation of many artificial life researchers over the past decade with the 'problem of trivial self-reproduction'. However, I believe that this is not the most relevant distinction with respect to evolvability, as I will argue in Chapter 7.

cipate the discussion presented in Chapter 7, it may be that artificial life platforms will have to model the material grounding of self-reproducing entities in some way, rather than being purely logical models, if strong selection pressure of self-maintenance is to exist. Some extensions to the cellular automata framework in this direction have been reported, such as the brief description by Myhill of a model in which machine components moved discretely in a two-dimensional environment [Myhill 64]. Von Neumann's original 'kinematic model' also falls into this category [Aspray & Burks 87] (p.374). However, the majority of work still models purely logical self-reproduction, where the reproducing entities are configurations of states with no material grounding; in such systems, no collection of 'raw materials' is required for reproduction, and from this it follows that there is no competition for raw materials between individual reproducers. It is likely that only when such considerations are included in these models can we expect there to be selection pressure for self-reproducers with the ability of self-maintenance, potentially leading to the evolution of autopoietic organisms.

A short time after von Neumann's death in 1957, Lionel Penrose developed a series of mechanical models of self-reproduction, which bear some similarity to von Neumann's kinematic model [Penrose 59]. Around the same time, a couple of other simple physical models of self-reproduction were also reported ([Jacobson 58], [Morowitz 59]), but Penrose provided the most discussion of his work. Furthermore, he also considered a self-reproduction scheme which was more like the relatively simple template-reproduction mechanism of the nucleic acids (DNA and RNA) than von Neumann's full self-reproduction architecture [Penrose 62]. Penrose mentions a number of deficiencies with this model (e.g. it was hard to get fully-formed copies of molecules to unbind at the right time, and there was no consideration of energy requirements), but was able to draw a number of conclusions about the properties required of a molecule which can form precisely self-replicated chains. He identified three necessary properties: (1) the molecules must have good alignment properties; (2) there must be a mechanism for strong permanent bonding down the chain (in order to form the copy); and (3) there must be a mechanism for cross-linkage between the original and the replica (which appeared to Penrose to be necessary to ensure stability). Although Penrose's approach was not particularly rigorous, his conclusions remain one of the few contributions to the 'logic' of template-reproduction to this day. This topic will be discussed further in Chapter 7.

The only other significant contribution to the subject, to my knowledge, is the formal system described by Douglas Hofstadter in his book *Gödel, Escher, Bach*. Hofstadter called the system 'Typogenetics', and explains that he "tried to capture some ideas of molecular genetics in a typographical system" ([Hofstadter 79] p.504). In this system, a strand of letters encodes a sequence of operations (such as copying, cutting etc.) which are to be performed upon the strand itself. Typogenetics was later developed and analysed by Harold Morris in his PhD dissertation [Morris 88], in which he showed that it was possible to produce several types of strand which were capable of self-reproduction. Louis Varetto has also reported work with self-reproducing strands in Typogenetics [Varetto 93]. In the state in which the system is presented, each strand can only inspect itself (i.e. there is no notion of a strand competing for resources with other strands in an environment), and the evolutionary potential of the self-reproducing strand is also unclear. These are both significant deficiencies from the point of view of using the system as a model of biological evolution (Hofstadter again was more interested in the process of self-reproduction in its own right). Modifications could conceivably be made to correct these—indeed, some were suggested by Varetto (*ibid.*)—but I am unaware of any implementations of such modifications.

The reproduction of structures in Penrose's and Hofstadter's models can be seen as the result of a self-inspection process; atomic subcomponents of the structure are inspected and copied, and these copies are then joined together to create a replica of the whole structure. More elaborate schemes for reproduction by self-inspection have been described in [Laing 77] and [Ibáñez *et al.* 95]. The relationship between these architectures and that proposed by von Neumann will be discussed in some detail in Section 7.2.3.

The next significantly different, substantial implementation of a system supporting self-reproducing components was reported by Steen Rasmussen and colleagues in the early 1990s.[13] They conducted a series of experiments using a general approach they named 'Coreworld'[14] (e.g. [Rasmussen *et al.* 90], [Rasmussen *et al.* 91]), inspired by an earlier

[13] Although Eigen and Schuster's work in the 1970s should also be mentioned. In the context of prebiotic evolution, they discussed ways in which the amount of selectively maintainable information in a system could be increased beyond the capacity of an individual replicator (see, for example, [Eigen & Schuster 77]). They introduced the 'hypercycle' (a cyclically catalytic group of replicators) as an functional organisation which could achieve this.

[14] Specific configurations of Coreworld were named Venus I, Venus II, and Luna.

computer game developed by A.K. Dewdney called 'Core Wars' [Dewdney 84].[15] In this approach, a one-dimensional address space is initialised with assembler-level instructions taken from a set of ten (or fewer). A number of execution pointers are distributed through the address space, determining which instructions get executed at any given time. Each address also has a certain 'resource' concentration associated with it, and execution of instructions is also conditional upon sufficient resources being available in the local neighbourhood. Noise is introduced into the system by new instruction pointers being introduced at random with low probability, and also by instructions being 'mutated' into randomly-chosen different instructions in certain circumstances. Although this work was concerned primarily with the emergence of self-organisation and cooperative structures (described in Section 3.2.3), experiments were conducted in which the memory was initialised by inserting a single copy of a hand-written self-reproducing program, and letting it reproduce. However, it was found that the self-reproducers soon started to malfunction and die out due to the noise in the system and to copying themselves on top of each other (but Rasmussen and colleagues still found this a useful method by which to create a biased initial distribution of instructions).

A major advance in the use of computer platforms such as Rasmussen's Coreworld to study the evolution of self-reproducing individuals was made by biologist Tom Ray with his 'Tierra' system ([Ray 91], [Ray 94b]). In his work, Ray places central importance upon the notion of self-replication:

> "Self-replication is critical to synthetic life because without it, the mechanisms of selection must also be pre-determined by the simulator. Such artificial selection can never be as creative as natural selection. The organisms are not free to invent their own fitness functions. Freely evolving creatures will discover means of mutual exploitation and associated implicit fitness functions that we would never think of. Simulations constrained to evolve with pre-defined genes, alleles, and fitness functions are dead-ended, not alive." [Ray 91] (p.372).

Ray's major development was to design an instruction set for his self-replicators which was robust to mutations and therefore evolvable, while at the same time retaining

[15] The ancestry of this approach can be traced back even earlier (e.g. [Bratley & Millo 72], [Burger *et al.* 80]), although these examples were presented in a fairly informal manner.

the property of computational universality.[16] The basic idea is that a virtual operating system is provided, complete with this robust and simple machine language, together with an address space (memory) of fixed size. The robustness of the language is achieved chiefly through the use of relative or template-driven[17] addressing in branching instructions (rather than absolute addressing), and by avoiding the use of explicit memory addresses as operands to instructions. In such an environment, the proportion of functional programs in the space of all possible programs is greatly increased, and the chances of mutating a functional program and coming up with another program that also works are much higher.

An evolutionary run commences with the introduction of an *ancestor* program into the otherwise empty memory. The ancestor is a hand-written self-replicator which produces another copy of itself in the computer's memory when it is run. At each iteration of the system, each program in the computer's memory is allowed to execute a certain number of instructions. By only running a limited number of instructions from each program at each time, the system does not run into the halting problem,[18] which is a potential problem for other kinds of program evolution system. A small element of stochastic behaviour is associated with the execution of the machine instructions, e.g. an ADD instruction which usually adds one to its operand may occasionally add zero or two instead, or a COPY instruction may sometimes mutate a byte of the data it is copying. The programs are also subject to point random mutations at a given low rate. In either of these ways, as a run proceeds variations of the ancestor program begin to appear. If a variation retains the ability to produce a copy of itself, then it too may be retained in the population of programs over a number of generations. As the available memory begins to fill, a 'reaper' operation is performed to kill off a number of the programs. Programs which perform operations which cause their flag to be set[19] are killed off quicker than others (by being advanced up the 'reaper queue'), but otherwise the order in which programs are killed off is largely determined by their age.

---

[16] A proof that the language has this property is presented in [Maley 94].

[17] With the template-driven branching scheme, a branching instruction is followed by a template (a sequence of bits). The operating system will search for the nearest matching template in the rest of the code and move the instruction pointer to that position.

[18] That is, some programs might never terminate.

[19] Examples include issuing a `jmp` instruction with a template pattern for which no match can be found, and attempting to write to a memory address for which the program does not have write access.

A number of interesting results have been obtained from such evolutionary runs, for example the appearance of "parasites"—short pieces of code which run another program's copying procedure in order to copy themselves. Hyper-parasites (parasites of parasites) have also been observed, along with a number of other interesting ecological phenomena [Ray 91].

It may be true to say that most of the interesting behaviour that has been seen to evolve has done so because facilities for these behaviours were engineered into the original language specification. For example, the fact that parasites emerge is a little less surprising when the mechanisms of template-driven branching are considered. I will return to this issue in Chapter 7. However, what these studies have demonstrated is that it *is* possible to build a robust operating system in which non-trivial self-replicating computer code can evolve.

The original goal of Tierra was to "parallel the second major event in the history of life, the origin of diversity" [Ray 91] (p.373). The event being referred to is known as the *Cambrian explosion*, and marks the first and sudden appearance in the fossil record, some 550 million years ago, of a great diversity of macroscopic multicellular organisms. Ray explains that "rather than attempting to create prebiotic conditions from which life may emerge, this approach involves engineering over the early history of life to design complex evolvable organisms, and then attempting to create the conditions that will set off a spontaneous evolutionary process of increasing diversity and complexity of organisms" (*ibid.*).

Ray admits that his early experiments with Tierra didn't actually achieve this goal, and suggests that the system is better viewed as "an artificial world which may roughly parallel the RNA world of self-replicated molecules (still falling far short of the Cambrian explosion)" (*ibid.*).

In analysing the broad patterns that emerge from evolutionary runs on Tierra, Ray notes that the observed evolutionary innovations fall into two broad categories: 'ecological solutions' and 'optimisations' [Ray 97]. The former include adaptations involving other programs in the environment (e.g. parasitism, immunity, etc.), and the latter are improvements within an individual organism to make the reproduction algorithm more efficient. Some experiments in which individual programs are allowed

to create parallel processes have been reported ([Thearling & Ray 94], [Thearling 94], [Thearling & Ray 96]), and in these cases optimisations include the use of parallelisation to increase the speed of reproduction (but only when the system was inoculated with a two-process parallel ancestor). Parallelisation is achieved by each process owning its own instruction pointer, but these all refer to a single copy of the program's code. Speed-up is achieved by each parallel process copying a different section of this code to create an offspring.

More recently, Ray and his colleagues have been striving towards evolving *differentiated* parallel programs, where some of the processes may be performing other tasks (e.g. sensory), rather than all being dedicated to reproduction. These attempts have so far been largely unsuccessful, although the most recent milestone has been to create an environment in which a hand-written differentiated ancestor (comprising 10 parallel processes, two of which are devoted to reproduction, and the remaining eight to sensing the environment) is at least maintained over prolonged periods of evolution, rather than being degraded into a purely reproductive organism [Ray & Hart 98]. However, this partial success has only been achieved by adding some high-level operations to the system, such as insertion, deletion and crossover operators, and new forms of mutation. Ray writes that the inclusion of these operators represents "a change in the philosophy towards this kind of genetic operation. These operations are being imposed by the Tierra operating system, and are not under the control of the creatures themselves. The reason for the shift is that the primary goal and focus of the work is now the attainment of complexity increase. This is a very big, and probably very difficult objective. For this reason I am willing to try anything that might aid in achieving the goal. Among the rewards of achieving the goal, should be creatures that do many more things for themselves, which should make the imposed genetic operators seem to be a fairly trivial sin."[20]

Comparative studies using different instruction sets have also been reported [Ray 94a]. The results suggest that the evolvability of the system is very sensitive to the underlying genetic system (i.e. small changes to the instruction set lead to very different results).

A number of similar systems have been produced by other researchers. These in-

[20] From an online report on Tierra, available at `http://www.hip.atr.co.jp/~ray/tierra/netreport/netreport.html#Philosophy`.

clude 'Avida', developed by Chris Adami and Titus Brown [Adami & Brown 94], and 'Computer Zoo', written by Jakob Skipper [Skipper 92]. Programs in both of these systems live in a two-dimensional environment. This notion was introduced mainly to provide a form of genetic isolation (to promote heterogeneity), and also to facilitate efficient implementation on parallel hardware. Although the details of these systems differ somewhat, the general mode of operation is the same. Some attempts have also been made to extend the representation of individual programs into two dimensions (i.e. each instruction of a program is physically located at a unique position on a 2D grid), but these have generally proved too brittle to support prolonged evolution (e.g. [Davidge 92], [Maley 93], [de Dinechin 97]).

John Koza has proposed a system of self-replicating programs using a slightly different approach [Koza 94]. Koza's programs are boolean trees whose nodes may be standard logical functions (specifically, AND, OR and NOT) or a number of extra functions which not only produce a boolean result but also have side effects which act upon the program itself or upon other programs in the computer's memory. For example, there is an ADD function which will search the rest of the memory for a program matching a given template, and, if one is found, will have the side effect of substituting this new program in the position where the ADD instruction resided in the original program.

One result of this approach of using boolean functions with side effects is that a program may perform a boolean calculation as well as having the side effect of producing another copy of itself somewhere else in memory. Koza encouraged his programs to perform specific boolean calculations by imposing a fitness function on the system—a program has a good chance of proceeding to the next generation only if it is able to self-replicate *and* performs well on the target boolean function.

In this respect, Koza's system is very similar to Avida, where programs can gain more CPU-time by successfully completing user-specified tasks. In this important sense, Avida and Koza's system differ from Tierra and the other related platforms; they have an *externally specified* function (or set of functions) which the programs are encouraged to perform *as well as* having to be able to reproduce. In this way, there are effectively two fitness measures, the former being completely unrelated to a program's ability to self-replicate.

Although this technique could conceivably be used to evolve programs which perform useful tasks (although it has not been demonstrated that it can scale to larger problems), it restricts the capacity of the system for open-ended evolution. By concentrating on applying selection pressure through the abiotic environment (i.e. by the externally defined functions), the programs are all evolving towards solving the same static problem rather than facing an indefinite variety of environments. Although biological organisms are themselves constrained by their abiotic environment, it is widely agreed (see Chapter 2) that the selection pressure for the most impressive forms of open-ended evolution (such as evolutionary arms races, symbiogenesis, and sexual selection) comes from the ecological relationships *between* organisms (i.e. the biotic environment). It is likely that only those systems in which organisms are able to *interact* with each other, and in so doing to directly affect each other's chances of survival, will display truly open-ended evolution. This issue will be discussed further in Chapter 7.

3.2.2 Open-Ended Evolution

The work reviewed in the previous section focussed on self-reproduction. This is clearly related to the topic of open-ended evolution (as defined in Section 2.5), and indeed that was also a major concern to many of the researchers mentioned, such as von Neumann and Ray. In this section, other important contributions to the topic of open-ended evolution will be reviewed. In many cases, the decision to review the work in this section rather than in the previous one is fairly arbitrary, and is based largely upon the relative emphasis given to the two topics by the authors of the work.

Nils Aall Barricelli was, to my knowledge, the first person to actually run artificial evolution experiments on computers [Barricelli 57], [Barricelli 62], [Barricelli 63]. He conducted a series of experiments, starting in 1953, to investigate the evolution of life (and not, as many others after him, purely to investigate the use of evolution as an optimisation technique). Barricelli explicitly discussed in his work what it would take in addition to reproduction and random variation in order for us to consider his evolved organisms 'alive'. In fact, he decided that the term 'life' was too poorly defined to be of use in the context, as mentioned in Section 2.1.3. Barricelli introduced the concept of symbiogenesis[21] in his work as an additional requirement for his organisms (which he

[21] According to Barricelli, the symbiogenesis theory claims that if genes are to evolve into "relatively

then called 'symbioorganisms'), and, instead of asking whether these symbioorganisms were alive, asked the reciprocal question of "whether the objects we are used to call[ing] living beings are a particular class of symbioorganisms" [Barricelli 63] (p.7).

The basic model used by Barricelli can be considered as a one-dimensional cellular automaton, where each state persists from one time step to the next depending upon the state of other cells in certain neighbouring positions. In this way, cooperative configurations of states can arise. Among the phenomena that Barricelli observed in this system are: self-reproduction of certain collections of states (i.e. symbioorganisms); crossing of material between two symbioorganisms; spontaneous formation of symbioorganisms; parasitism; self-maintaining symbioorganisms; and, evolution [Barricelli 62]. Barricelli's method and results are similar in a number of ways to those of many of the more recent studies described in the previous section, such as Rasmussen's and Ray's. The fact that these experiments were run using the computing equipment available in the 1950s makes them all the more impressive. It is highly regrettable that Barricelli's work has been largely forgotten or ignored by today's artificial life community,[22] although there are a few signs that it is at last beginning to achieve some recognition (e.g. [Dyson 97], [Fogel 98]). Although many might argue over the central role that Barricelli gave to the process of symbiogenesis in his work, his experiments are strengthened by the fact that he at least *proposed* an explicit list of the assumptions behind the model, and also by the simplicity of the model he developed. In these respects, his work is of more scientific value than much of the more recent work on these subjects.

In addition, Barricelli conducted some experiments in which the individual symbioorganisms were also decoded into a strategy for playing a simple game (i.e. they had a phenotype) [Barricelli 63]. When two symbioorganisms were competing to reproduce into the same space, they played the game according to their individual strategies, and the winner was allowed to reproduce. This is similar in some ways to the dual selection pressures in Avida and Koza's work reported above, but has the advantage (from the point of view of open-ended evolution) that the additional selection pressure is de-

---

higher forms of life", they must only be able to reproduce through a symbiotic relationship with other genes [Barricelli 57] (p.145). A group of genes that collectively reproduces in this way (a 'symbioorganism') can be considered a special case of a hypercycle [Eigen & Schuster 77], in which each component in the group is absolutely *required* for the reproduction of the next component, rather than just acting as a catalyst.

[22] His papers contained many ideas on subjects which have since become popular research areas, such as the use of real DNA for performing computations ([Barricelli 63] pp.121–122).

termined by other organisms within the system, rather than being externally defined. Barricelli met with mixed results in these experiments ([Barricelli 63], [Barricelli 72]), partly because symbioorganisms which had developed good strategies for playing the game often became infected by parasites. I suspect that Barricelli's system was more open to these types of phenomena than are systems such as Avida, because the individual cells within a symbioorganism reproduce by themselves rather than *en masse*. As such, they behave more as individually 'selfish genes' [Dawkins 76], and the symbioorganisms as a whole should therefore not necessarily be expected to be good at finding 'optimal' solutions to given problems.

There has been some more recent work within the artificial life community on the subject of symbiogenesis. Some of the issues involved in incorporating symbiogenesis (and symbioses in general) in artificial life models are discussed in [Daida *et al.* 96], and studies of the general conditions under which symbioses may occur are reported in [Bull & Fogarty 95]. Some practical implementations, incorporating the ideas of symbiogenesis, have been successful at solving particular problems (for example, [Ikegami & Kaneko 90], [Numaoka 95]), but as organisms in these systems have a limited behavioural repertoire and are solving a specific, externally defined task, their capacity for truly open-ended evolution is restricted.

Gordon Pask also describes some early work with artificial life models [Pask 69], although the evolutionary potential of his automata is limited by the rather small set of actions which they may perform. A rather more interesting system was developed by Michael Conrad and Howard Pattee [Conrad & Pattee 70]. In this model, individual organisms, with a genotype representation and a phenotype obtained by interpreting the genotype as instructions (cf. Barricelli's game-playing symbioorganisms), compete in a one-dimensional world for the possession of 'chips' which they use for self-repair and reproduction. It is closely related in many ways to Ray's later Tierra model, but with a smaller number of instructions representing a limited set of possible interactions between organisms, rather than a computationally complete instruction set as in Tierra (although even in Tierra there is still only a limited number of types of interaction between organisms). Also, it has a notion of conservation of matter, lacking in Tierra, to model ecosystem interactions. Having said this, there is only one type of matter in the model (a 'chip'), and it has a fairly arbitrary connection to the structure

of an organism. For example, an organism's genome is represented as a string of 'states' rather than a string of matter—an organism's store of chips is only used to determine when it can repair itself and when it can reproduce. The major consequence of disassociating the structure of the organism from the 'matter' in the world is that the structure must therefore be predefined and is not able to evolve, whereas, had it been embedded in the material world, new organism structures could emerge from new organisations of the matter. This problem of predefining a non-material structure for organisms is shared by Tierra and other models described in this and the previous section, and will be discussed further in Chapter 7. The organisms in Conrad and Pattee's model had the potential to engage in symbiotic relationships with other organisms (by sharing chips), and also to reproduce sexually. The results showed that symbiosis was often selected for, but sexuality generally was not. However, in summarising the results, the authors conclude that:

> "It is evident that the richness of possible interactions among organisms and the realism of the environment must be increased if the model is to be improved ... One point is clear, that the processes of variation and natural selection alone, even when embedded in the context of an ecosystem, are not necessarily sufficient to produce an evolution process ... Experience with the present model re-enforces our feelings that the most profound and significant processes of evolution—the innovations, the origins of new hierarchical levels of organization—are still outside the scope of this type of program and remain to be discovered." [Conrad & Pattee 70] (pp.407–409).

Many people claim that these processes remain outside the scope of artificial life models even today (e.g. [Stewart 97], [Bedau 98a]).[23]

In the mid-1970s, John Holland proposed a collection of models he collectively referred to as the 'α-Universes' as a suitable environment in which to study the spontaneous emergence of self-reproducing systems [Holland 76]. The design is similar in many ways to Conrad and Pattee's model, but was influenced by an analogy with the spontaneous emergence of life in a 'primordial soup' of biochemical molecules. A feature shared

[23] But see [Mayer & Rasmussen 98] for a recent study in which emergent hierarchical structure has been successfully simulated.

by both models (and by some of Barricelli's work) is that individual components are acted upon by certain predefined operators (the 'physics' of the world), but (adjacent collections of) components can also be interpreted as encodings of further operators (phenotypes, or 'emergent operators' in Holland's terminology), giving the models a more open-ended quality. Also, as in Rasmussen and colleagues' work discussed in the previous section, the configurations which are reproduced in the α-Universes are not individual structures, but *collections* of structures. In Barry McMullin's detailed analysis of the α-Universes [McMullin 92a], he suggests that these collective structures might alternatively be regarded as autopoietic (roughly, self-producing and self-maintaining) organisations. However, McMullin observes that "the 'higher-level', properly autonomous entities, are not, in general, self-reproducing in any sense, and are *certainly* not genetically self-reproducing in the von Neumann sense of permitting an open-ended growth in complexity" [McMullin 92a] (p.269, original emphasis).

Holland's original work was based upon a mathematical analysis of phenomena that he expected to emerge in the system. Fifteen years later, the α-Universes were implemented as a computer program by McMullin ([McMullin 92a], [McMullin 92b]). He found a number of problems with the design that were not anticipated by Holland and which meant that it did not produce the 'life-like' behaviour that he postulated. As many of the problems were ultimately due to components in the world being unable to control their local environment and maintain their own structure, McMullin has subsequently gone on to investigate software implementations of autopoiesis [McMullin & Varela 97].[24]

After the α-Universes, Holland developed the 'Echo' model of complex adaptive systems ([Holland 95], [Hraber *et al.* 97]). Echo places more emphasis on ecological interactions and exchange of resources than do most of the other models reviewed. In particular, in Echo Holland takes the view that it is the 'market' that emerges from exchanges of resources between individual agents that is the source of much of the interesting behaviour of a complex system. Individual agents are modelled at a fairly high level, with a predefined structure. In the basic model, agents can participate in a limited set of interactions with other agents (e.g. to exchange resources), and reproduce auto-

[24] Interestingly, Barricelli also predicted that his symbioorganisms would need to develop means of controlling their local environment (such as a membrane) if they were to evolve past a certain level of complexity [Barricelli 63] (pp.122–124).

matically when they have collected sufficient raw materials [Holland 95]. Interactions are governed by each agent's collection of 'tags'—short strings of symbols which are encoded on the agent's genome (or 'chromosome' in the Echo terminology). Each tag is used for a specific set of interactions, where the outcome of the interaction is partially determined by comparing corresponding tags in the interacting agents in some predefined way. Various extensions to this basic model are also described, to add features such as multicellularity, metabolism and selective mating (*ibid.*). The design is based upon a core set of principles which Holland believes are common to all complex adaptive systems. One feature that distinguishes Echo from other models reviewed in this chapter is that it has been successfully used for several ecological studies (e.g. [Schmitz & Booth 96], [Hraber & Milne 97]).

I have now discussed what I consider to be the most important practical examples of artificial life models aimed at self-reproduction and open-ended evolution. To end this section, I will briefly mention a number of other studies which have also made relevant contributions.

In my discussion of Avida and Koza's system in the previous section, I made the distinction between selection pressure that originates from other programs (organisms) within the environment, and that which originates from externally defined functions. This distinction was emphasised by Norman Packard, who used the terms *intrinsic* and *extrinsic* adaptation respectively [Packard 88]. He argues (as have many others mentioned in this section and the last) that models with intrinsic adaptation are more appropriate for modelling biological evolution. Packard described a model which he used to study evolutionary dynamics, which is distinguished by its simplicity. He says "I make every attempt to strip down most of the complexity of real biological systems, with the aim of discovering a minimal model that displays evolutionary behavior" (*ibid.* p.142). However, as his organisms only have two genes, the relevance of the model in the current context of studying open-ended evolution is limited.

Larry Yaeger has described a system called PolyWorld [Yaeger 94], which is in many ways the antithesis of Packard's minimal approach. PolyWorld models many features of biological life, such as a simple 'metabolism', a nervous system and vision. Yaeger describes it as an attempt to evolve Artificial Intelligence through the evolution of nervous systems in an ecology. In PolyWorld, organisms controlled by (genetically

determined) neural networks move around a two-dimensional environment, collecting energy, fighting and mating. Yaeger's model is one of the very few artificial worlds in which distinct species of organisms have evolved and coexisted.[25] Unfortunately, the results of PolyWorld have not been analysed in sufficient detail to enable many useful scientific conclusions to be drawn from it, and the complexity of the model does not help in this respect.

Early attempts at allowing variable length genomes in an artificial life simulation are described by Robert Collins in his PhD thesis [Collins 92], which extends previous work by David Jefferson, Charles Taylor and colleagues (e.g. [Taylor *et al.* 88], [Jefferson *et al.* 91]). In this work, an organism's genome encodes a neural network which controls its behaviour. By allowing variable length genomes, larger networks are able to evolve, which are capable of producing more complex behaviour. Along the same lines, Inman Harvey has worked on extending the theory and the design of the standard genetic algorithm to allow open-ended evolution by permitting the length of genomes to increase over time [Harvey 93]. The approach is called SAGA, and, like Collins *et al.*'s work, is concerned with extrinsic adaptation (i.e. situations where an external fitness function is applied). This method has proved useful for a number of practical purposes, mostly concerned with evolutionary robotics, but it is not clear how relevant it is for many of the systems I have described in the last two sections. I suspect that some of the assumptions upon which Harvey based his analysis of the need to extend the 'schema theorem' of genetic algorithms for open-ended evolution (*ibid.* Chapter 6) might not be valid for these models. In particular, it is not clear how the notion of fitness used in the analysis can be related to models with intrinsic adaptation. Furthermore, Harvey's analysis assumed a system with low epistasis, and it is questionable whether models such as Tierra meet this requirement.

Finally, it is worth noting that some of the most spectacular examples of artificial evolution that have been produced to date model *co*-evolutionary processes of one form or another (e.g. [Hillis 90], [Sims 94a], [Sims 94b], [Miller & Cliff 94], [Cliff & Miller 96], [Floreano *et al.* 98], [Nolfi & Floreano 98]). In these studies, the success of organisms in one population depends upon the success of organisms in another, coevolving population. However, these studies have all been geared towards producing organisms which

[25] Another is Lindgren's model of a population of individuals playing a variation of the iterated Prisoner's Dilemma [Lindgren 91].

are good at performing a particular task. To this end, the coevolving organisms are still generally competing in some pre-specified (extrinsically defined) game, and they are not given the potential for truly open-ended evolution in which they could develop entirely *new* games to play.

3.2.3 Self-Organisation and Origin of Life Models

In this final review section, I will briefly mention some work on topics such as self-organisation, the origin of life, self-maintenance and autopoiesis. This work is not so relevant to the experimental portion of the thesis reported in Chapters 4–6, but we will come back to it in Chapter 7.

In contrast to the usual procedure for Tierra-like systems of inoculating the environment with hand-written self-reproducing programs, some models have been modified to look at the *spontaneous emergence* of self-reproducing individuals (e.g. [Koza 94], [Pargellis 96]). Similar work has also been reported within the framework of a cellular automata model [Chou & Reggia 97]. However, the details of these models are so far removed from anything in the real world that I doubt that they can really tell us anything of scientific value about the origin of life.

Walter Fontana and colleagues have developed a fundamentally more interesting approach with their 'artificial chemistry' models (e.g. [Fontana 91], [Fontana *et al.* 94], [Fontana & Buss 96]). They argue that a formalism is needed in biology (and other areas) for constructive systems (i.e. those where the components are objects whose structure can change as the result of interactions). This should be coupled with classical dynamic systems approaches to form a *constructive dynamic systems theory*. Such a theory would help us to understand dynamic systems in which new operators can emerge as the system evolves. Some of their latest work in developing such a theory is described in [Fontana & Buss 96]. The relevance of this to the study of open-ended evolution is clear.

Rasmussen and his colleagues have described a similar approach to studying self-organisation ([Rasmussen *et al.* 90], [Rasmussen *et al.* 91]), but based upon a formalism similar to the von Neumann machine (as described in Section 3.2.1) rather than

the λ-calculus and other formalisms used by Fontana and Buss.[26] They successfully obtained emergent cooperative structures, but emphasised that the details of such structures depended heavily on the details of the system. However, they do suggest that a number of more general conclusions may be drawn, which may have analogies in prebiotic chemical systems [Rasmussen *et al.* 91] (pp.245–246). These include: that "there are certain relations, which need to be fulfilled, between system size, available executions per system update, and initial conditions before the systems are able to support complex cooperative dynamics"; that "functional stability to perturbations is a *product* of evolution and not a property of the details of the underlying programmable matter"; that "cooperation emerges as a natural property of the functional dynamics in systems with constructive dynamics"; that "simplifying the instruction set below a certain level of complexity inhibits the emergence of higher-order cooperative structures";[27] that "it seems easier to create a reproductive system without genes. The emerging cooperative structures have several properties in common with autocatalytic sets found for catalyzed cleavage-condensation reactions in polymer systems"; and that "the more low-level the living process is, the more fuzzy the organism-environment distinction appears".

The approach of Rasmussen *et al.* can be related to work by Kauffman, Bagley, Farmer and colleagues on artificial chemistries (e.g. [Kauffman 86], [Bagley & Farmer 91]). However, Fontana and Buss claim that their approach (mentioned previously) is more general than these, because it is based upon a *theory of object construction* (e.g. [Fontana & Buss 96], Section 3). Other recent approaches to building constructive dynamical systems include those suggested by Wolfgang Banzhaf, Peter Dittrich and colleagues (e.g. [Banzhaf 94], [Dittrich & Banzhaf 98]), by Shinichiro Yoshii and colleagues (e.g. [Yoshii *et al.* 98a], [Yoshii *et al.* 98b]), and by Yamamoto and Kaneko [Yamamoto & Kaneko 97]. Banzhaf, Dittrich and colleagues, for example, have described experiments with a catalytic self-organising reaction system of binary strings. Their most recent work involves the decoding of these strings as programs which determine how one string reacts with another, which is a similar concept to the idea of emergent operators in Holland's α-Universes.

---

[26] Rasmussen *et al.* also discuss the relationship between these and other universal formalisms [Rasmussen *et al.* 91] (pp.243–244).

[27] This is similar to von Neumann's conclusion that "complication is degenerative below a certain minimum level" [von Neumann 49] (p.482).

As a general comment, most of these models do not have any spatial structure (the reactions occur in a well-stirred tank), which might have important consequences for self-organising systems. For example, with no spatial structure there can be no notion of *individuality* in the organisations which emerge. Also, the models do not necessarily have conservation of matter. However, these are not fundamental limitations, and the models could be modified fairly straightforwardly. Indeed, such modifications have been discussed by some of these authors themselves (e.g. [Fontana & Buss 96], [Dittrich & Banzhaf 97]).

Finally, a handful of computer models of autopoiesis have been described (for example, [Varela *et al.* 74], [Zeleny 77], [McMullin & Varela 97]), but these have so far met with limited success at achieving sustained autopoietic organisation.

3.3 Methodology and Design Issues for Artificial Life Platforms

In this final section of the chapter, I will attempt to pull together some of the lessons learned in the work reported in the previous sections by highlighting some of the important methodological and design issues involved in creating an artificial life platform.

3.3.1 Methodology

Some of the more important methodological issues concerning the scientific use of artificial life techniques were discussed in Section 3.1.3. The key point is that simulations should be based upon explicit theories and assumptions if they are to be of scientific value ([Pattee 88], [Noble 97], [Taylor 98]).

As as example, the emergence of parasites and similar phenomena in Tierra [Ray 91] might appear to be suggestive of a close parallel between this system and biological evolution. However, the appearance of parasites in Tierra turns out to be dependent on some fairly specific aspects of the system's design, rather than on any particular general principles.[28] Furthermore, closer analysis reveals that it only requires a *single mutation* of the original ancestor program to produce a parasite; this was reported in Ray's

[28] More precisely, there *may* be some general principles concerning parasitism that Tierra shares with Nature, but the point is that no-one has explicitly stated what these might be, and Tierra was therefore not designed to be a particularly good test of any such principles.

original paper [Ray 91] (and has recently been reiterated in [Channon & Damper 98]), but has been overlooked in most subsequent reports of parasitism in Tierra. These facts suggest, if nothing else, that claims concerning the relationship between artificial life models and the real world should be treated with caution.

I think that Tierra and many (but certainly not all) of the other recent contributions to artificial life can be regarded as constituting an 'exploratory stage' of the subject, in which the potential of the new techniques has been investigated. This is a useful stage, and indeed is a normal aspect of any experimental science [Cohen 95]. However, further progress towards scientific knowledge must be based upon the development and testing of properly formulated theories.

3.3.2 Self-Reproduction

Most of the work reported in this chapter has relied upon concepts of self-reproduction in one form or another. However, different studies have modelled self-reproduction in different ways and at different levels. Ray has insisted on the importance of self-reproduction in artificial life models of evolution (see Section 3.2.1), but von Neumann's analysis shows us that a variety of different issues are involved in this concept (e.g. the method by which reproduction is achieved, the robustness of the reproducers to mutations, the heritability of variations, etc.). There is a risk of confusion if one talks simply about self-reproduction, without referring to these more specific issues.

A great deal more work is required before we can properly understand the evolutionary consequences of design decisions relating to each of these more specific issues. Von Neumann's architecture was designed specifically so that the self-reproducing automata would have the potential of participating in an evolutionary process in which more and more complicated automata arose. I am not aware of such a detailed analysis of other sorts of reproduction, although it has been suggested that collectively-autocatalytic sets generally do *not* have great evolutionary potential (see, for example, [Maynard Smith & Szathmáry 95]). It has also been suggested that reproduction by self-inspection (discussed on p.51) has less evolutionary potential than genetic reproduction (e.g. [von Neumann 66] pp.121–123, [McMullin 92a] pp.191–193), although little has actually been proved.

For truly open-ended evolution, we might also want to consider how one sort of reproduction might *evolve* from another sort.[29] I will return to discuss these issues in much more detail in Chapter 7, with the benefit of experience gained from running an extensive series of experiments with a Tierra-like platform (reported in Chapters 4–6). The main point I wish to make here is that the evolvability of the system clearly does depend upon the type of reproduction employed, and upon other issues involved in the concept of self-reproduction, even if we do not yet fully understand these dependencies.

3.3.3 Representation

In addition to consideration of the general scheme of reproduction, we also have to tackle issues concerning the representation that should be used for the reproducers and for the environment in which they exist. Rasmussen and colleagues have presented a useful survey of different universal formalisms which may be employed, and point out that a spectrum of other formalisms exist in between them [Rasmussen *et al.* 91].

There is also a choice between procedural and functional representations. Tierra and related systems have used procedural languages with considerable success, although some have argued that functional languages have many desirable properties (see, for example, [Fontana 91]).

Another issue concerns the level at which the model is pitched. Von Neumann argued that a level must be found which is neither too high nor too low:

> "If you choose to define as elementary objects things which are analogous to whole living organisms, then you have obviously killed the problem, because you would have to attribute to these parts those functions of the living organism which you would like to describe or to understand ... One also loses the problem by defining the parts too small, for instance, by insisting that nothing larger than a single molecule, single atom, or single elementary particle will rate as a part. In this case one would probably get completely bogged down in questions which, while very important and interesting, are entirely anterior to our problem." [von Neumann 49] (p.479).

[29] Maynard Smith and Szathmáry discuss ways in which the evolutionary potential of hypercycles can be enhanced through compartmentation, and suggest that this might eventually lead to the evolution of a genetic system [Maynard Smith & Szathmáry 95].

Similarly, Yaeger points out that the computational power that researchers currently have readily available is not sufficient to enable one to start off at the level of subatomic particles and expect to observe ethological behaviours [Yaeger 94] (p.268). As an example of what is presently achievable, Bernd Mayer and Steen Rasmussen have recently described a system in which emergent third-order structures (micelles) appeared from organised second-order structures (polymers), which in turn appeared from organised first-order structures (monomers)—the monomers being the fundamental level of representation in the simulation [Mayer & Rasmussen 98]. This work (which employed a two-dimensional lattice world of dimensions 100×100 units) pushes currently-available computational power to the limit. Finally, Harvey argues that the level of representation should be as low as possible, so as not to introduce the prejudices of the designer or to restrict the evolutionary potential of the system [Harvey 93] (p.48).

The representation of the environment is another issue, and one which has received little attention so far in the artificial life literature. Organisms in Tierra, the α-Universes and many other systems described in this chapter exist in a one-dimensional world; von Neumann's self-reproducing automata, Avida and others are two-dimensional, and several people have tried modelling three dimensions. An important point to note is that there are qualitative differences in the properties of spaces of different dimensionality (e.g. concerning random walks, [Feller 67] pp.359–363). Von Neumann suspected that three dimensions, or a Riemann surface (a multiply-connected plane), might be required for his kinematic model of self-reproduction, although he did not prove it [von Neumann 49] (p.485). Whether or not this is true, though, it is clear that the kinds of phenomena that might evolve in an artificial life model will be constrained by the dimensionality of the model.

More broadly, the basic laws governing components in the system and how they interact, and the spatial structure of the system, will all contribute crucially to how the components evolve. The general idea that more complicated environments lead to more complicated organisms is fairly widely accepted, but we are really only just beginning to investigate the dependencies which exist between these fundamental aspects of an evolving system.

Chapter 4

Design Details of the Cosmos Platform

Despite the potential problems associated with Tierra and similar platforms discussed in the previous chapter, it was decided that it would be worthwhile to develop and investigate a system built according to the same principles. Reasons for doing this included the following:

- To develop a better appreciation of the issues involved in designing such a system.
- To enable an extensive analysis of the behaviour of such a system to be conducted and reported.
- To observe possible differences in behaviour between the new system and existing Tierra-like platforms, arising from differences in design.
- To get a better idea of how serious some of the potential problems mentioned in the previous chapter might be, and how they might be resolved.

Issues such as these are relevant because the approach pioneered by Tom Ray with Tierra is fairly widely used, and its validity is fairly widely accepted, within the artificial life community. A new Tierra-like platform, called Cosmos,[1] was therefore developed. The design details and relevant implementational issues are discussed in the rest of this chapter.

[1] 'Cosmos' is an acronym for COmpetitive Self-replicating Multicellular Organisms in Software.

4.1 Cosmos Design Philosophy

The basic approach employed in Cosmos to model an evolutionary process is the same as in Tierra. However, many of the design details are different, reflecting the slightly different goals motivating the two systems. One of the original goals of Cosmos was that it should be able to support self-replicating programs with some of the features possessed by simple *cellular* biological organisms, such as mechanisms for communication and response to environmental stimuli (which may potentially promote coevolution between organisms), and mechanisms for regulating the genome (which may promote the evolution of differentiated programs).

Before continuing, clarification should be given of some of the terms that will be used when describing Cosmos. Biological terms will often be used, as these tend to be somewhat more concise than the associated terms relating to computer architectures. While these biological terms suggest the analogy that was in mind when Cosmos was designed, the analogies are certainly not exact; many simplifications and modifications obviously have to be made when designing such a system. With this is mind, the meanings attached to some biological terms in the present context are listed in Table 4.1.

Term	Meaning in context of Cosmos
Genotype	The instructions that make up a program (the host code within a cell).
Genome	The structure within a program which stores the program's instructions. In the current context, the terms genome and genotype are used more or less interchangeably.
Phenotype	The action (behaviour) of a program as its instructions are being executed.
Organism	A single program, which may be unicellular or multicellular.
Cell	A single process in an organism. This term encompasses the host code and any foreign code that may be present, together with associated working memory, buffers, registers and other structures.
Unicellular	An organism containing a single cell/process (in other words, a serial program).
Multicellular	An organism containing multiple cells/processes (in other words, a parallel program).

Table 4.1: Definitions of Biologically-Related Terms Used for Describing Cosmos.

Perhaps the most significant difference between Cosmos and Tierra is that programs in Cosmos cannot directly read the code of their neighbours. Cells can only communicate with each other (within or between organisms) by message passing (described in Sec-

tions 4.3.7 and 4.6.1). Apart from this intercellular communication, each cell only has read, write and execute access within its own cell boundary.

Among the other important differences between Cosmos and Tierra are a number of features in Cosmos intended to encourage the evolution of diversity and complexity[2] in the competing programs, rather than just the optimisation of their ancestral algorithms. The most important of these are the energy token allocation system, described in Sections 4.3.5 and 4.5.2, and the regulator system of *promoters* and *repressors* which governs the execution of a program's code, described in Section 4.3.3. The regulator system is closely linked to the programming language in which the self-replicators are written, introduced in Section 4.4. Further differences between Cosmos and Tierra are discussed in Section 4.10.

4.2 Preliminary Issues

Before going into the details of program representation and behaviour, a few words should be said about some general features of Cosmos.

4.2.1 Representation of Information

The underlying representation of many of the components of the Cosmos system is the BitString. Four different types of BitString are used: BitStrings, InfoStrings, WritableInfoStrings and EnvironmentalInfoStrings. These are defined as follows:

BitString A vector of binary digits (i.e. a string of 0s and 1s).

InfoString Like a basic BitString, but also has a *type* associated with it (an integer i in the range $0 \leq i \leq 15$), and a pointer to the current read/write position along the string. A string of bits belonging to an InfoString cannot usually be altered after its initial creation—it can only be read. The only exception is that an InfoString may be *mutated*, which entails one or more of its its being flipped at random.

WritableInfoString An InfoString in which the bit string can be written to as well as read from.

[2] Primarily, hierarchical object complexity (see Section 2.3.2). In other words, many of the design features of Cosmos were intended to promote the evolution of multicellular organisms from unicellular ones.

EnvironmentalInfoString An InfoString that has an *intensity* (a non-negative real valued number) associated with it.

4.2.2 Spatial Structure

The shared space in which the organisms reside is a two-dimensional grid, divided into discrete squares.[3] Each cell in the population is associated with a particular square at any given time. This environment can be configured to wrap around, or not to wrap around, in each dimension. More information about the environment in which the cells live is given in Section 4.5.

4.2.3 Time Slicing and the Top-Level Algorithm

The Cosmos operating system simulates the parallel execution of a large number of programs. As Cosmos is actually implemented on a serial machine, a form of time slicing is required to achieve this (i.e. at each time slice, a small number of instructions are executed for each program, one at a time). The top-level algorithm that implements this procedure is described in Section 4.7. At each time slice, it must be decided how many instructions are to be executed for each program. Possibly the most obvious strategy is to execute a fixed number of instructions for each program. However, from an evolutionary point of view, this would introduce selection pressure for small programs because, all else being equal, longer programs would take a larger number of time slices to reproduce. This may or may not be desirable. The decision of how many instructions to run for each program at each time slice is therefore governed by a couple of parameters which can be tuned by the user. Specifically, a program of length L bits is allowed to execute N instructions per time slice, determined by the formula:

$$N = \texttt{et_value_constant} * L^{\texttt{et_value_power}}$$

N is rounded down to an integer value. This allows considerable flexibility: for example, if `et_value_power` is set to 0.0, then each program executes `et_value_constant` instructions per time slice, regardless of length; if `et_value_power` is set to 1.0, then the allocation is linearly proportional to program length, so evolutionary selection is size-neutral (all else being equal). Further details of time slicing are given in Section 4.3.5.

[3] The system has been designed to deal with arbitrary n-dimensional environments, but the current implementation requires some minor revisions to allow this.

4.2.4 Naming of Organisms

For the purpose of analysis of the system's behaviour, individual organisms are given names according to their genotype. The name is composed of a number followed by a string of (usually four) upper-case alphabetic characters. The number is the length of the genome (expressed as a number of bits) in the organism's initial cell. The character string is a unique identifier for that particular genome. Ancestor organisms inoculated into the system at the start of the run are named with the character string AAAA. If an offspring has an identical genotype to its parent, it will share the same name. If the offspring has a different genotype, then it is given a new name (the operating system keeps track of which names have already been issued, to avoid duplication). For example, the first organism to appear in the system that differs from the inoculated ancestors will be named with the character string AAAB. Should all character strings up to ZZZZ have been issued for organisms of a particular length, an extra A is added to the string (so the next organism of that length with a different genotype to its parent will be named with the character extension AAAAA).

4.3 The Structure of an Individual Cell

4.3.1 Overview

The basic structure of a single cell is shown diagrammatically in Figure 4.1. Each cell is a process running on the (virtual) Cosmos operating system. A cell has its own program code, working memory, stack, registers and various other structures. The major features of the cell are explained in the rest of this section.

4.3.2 The Genome

The Genome is an InfoString (i.e. a BitString with an associated type), containing encoded instructions that the cell can execute.[4] Which sections of the genome are translated into instructions and executed is determined by the action of promoters and repressors (see Section 4.3.3). After a cell has been created, its genome cannot usually be altered, except by the action of mutation (see Section 4.5.7).

[4] As I am usually referring to the contents of the Genome, rather than to the structure itself, when I use the term 'genome', I will use the standard typeface from now on.

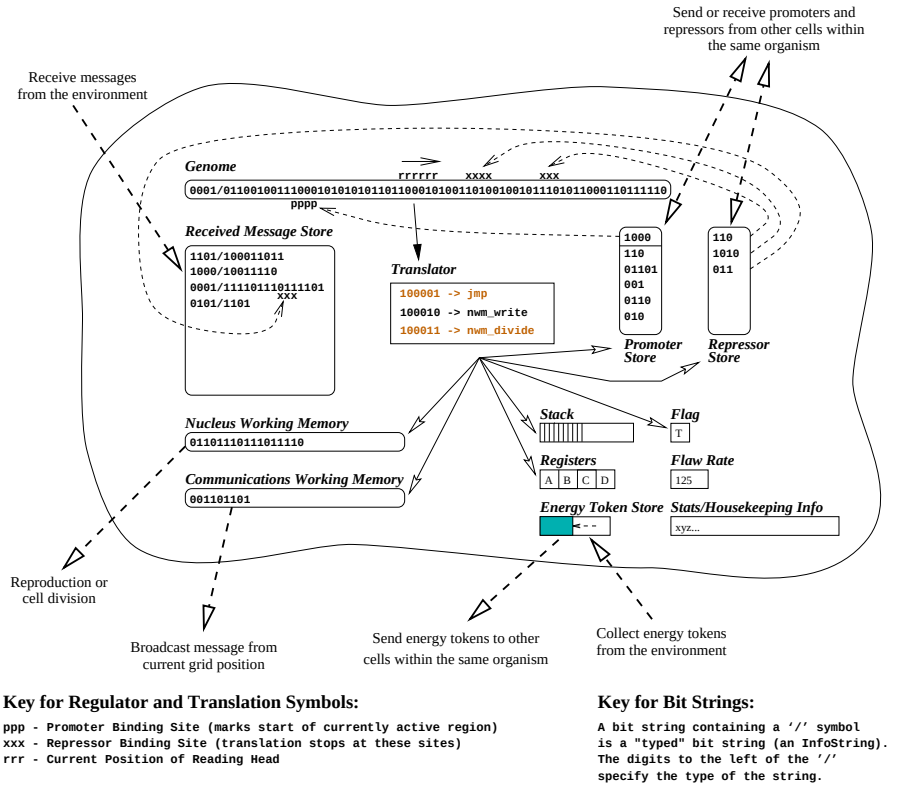


Figure 4.1: The Structure of a Cell in Cosmos.

4.3.3 Regulators: Promoters and Repressors

The translation of the genome is governed by regulators. These are (usually short) BitStrings, and come in two distinct types; promoters and repressors. The cell has a separate store for each of these two types of regulator, and each store can contain a number of regulators of the appropriate type. Regulators may be added to the Promoter Store and Repressor Store in two ways: either by the cell creating a new regulator (by executing an appropriate `reg_create` instruction),[5] or, in the case of a multicellular program (see Section 4.3.9), by the cell being sent a regulator from a neighbouring cell. A cell can also remove regulators from its Promoter Store and Repressor Store, by executing an appropriate `reg_destroy` instruction.

[5] See Section A.2 for an explanation of the instruction set.

Promoters and the Promoter Store

The Promoter Store is an ordered list of promoters. Only the promoter currently at the top of the list is active at any given time. The active promoter specifies the position along the genome[6] at which translation will begin. When a new promoter becomes active, a search is made along the genome for a pattern of bits that matches the promoter bit string.[7] If a matching region is found, the promoter is said to have *bound* to that region, and translation of the genome begins from the first bit to the right of the binding region. If no binding site is found for the active promoter, or when the translation of the current section of genome is terminated (e.g. when the Read position reaches the end of the genome, when it reaches a repressed region, or when a `stop` instruction is encountered), the active promoter is deactivated and placed at the bottom of the list in the Promoter Store, and the promoter which is now at the top of the list becomes active.

Repressors and the Repressor Store

The Repressor Store is a list of repressors, but, unlike in the Promoter Store, any or all of the repressors on the list may potentially be active at the same time. When a new repressor is added to the store, a search is made for a binding site on the genome,[8] in a similar way as for the active promoter. If a binding site is found, the repressor is said to be bound to the corresponding area of the genome, and that area of the genome is said to be repressed. If, during translation of the genome, the read position moves onto a repressed site, translation ceases at that point and the current promoter is deactivated.

4.3.4 The Translator

The process of translating the genome into executable instructions is illustrated in Figure 4.2. As the read head moves along the genome, it passes the string of bits that it reads to the Translator. The Translator has a table that maps bit strings to instructions in the programming language of the cells. As soon as the incoming string

[6] Or on eligible InfoStrings in the Received Message Store. See Section 4.3.7 for details.

[7] The search begins at the current Read position on the genome, and proceeds outwards in both directions simultaneously.

[8] Or on eligible InfoStrings in the Received Message Store. See Section 4.3.7 for details.

of bits matches an entry in this table, the Translator executes the associated instruction and the read head is moved along the genome to the next unread bit. In the current implementation, the map of bit strings to instructions is hard-coded into the Translator, all instructions are encoded by bit strings of equal length (six bits), and all 64 possible six-bit codes have an entry in the table (which means that in some cases, two different six-bit codes encode the same instruction). Any binary string of length six is therefore guaranteed to decode to a valid instruction. This hard-coded mapping is defined in the system input file `genetic_code.ini`, described in Section A.5.1.

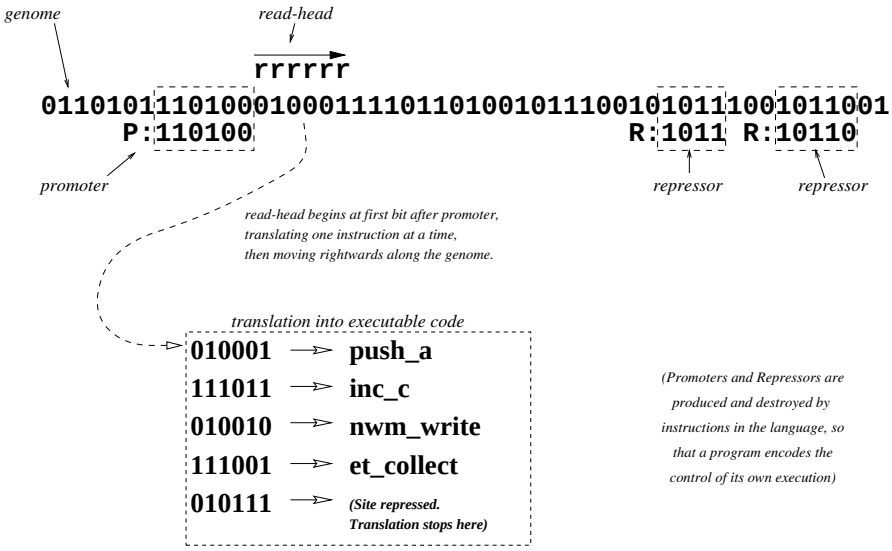


Figure 4.2: Translation of the Genome.

In future experiments with the system, the hard-coded mapping from bit strings to program instructions may be replaced by a mapping which can vary from one cell to the next, and which can evolve.

4.3.5 The Energy Token Store

A large number of cells may exist concurrently within Cosmos. In order to run the code of all of these cells, the processor must time slice between each cell, as described in Section 4.2.3. In that section a formula was given which shows how many instructions a cell with a genome of a given length is allowed to execute at each time slice. However, for the cell to actually execute this number of instructions, it must pay one *energy token* to the processor for each instruction it executes. A cell has a store of energy tokens (which it collects from the environment as described in Sections 4.5.2 and 4.5.3). Furthermore,

a cell's Energy Token Store may be leaky, in which case a number of energy tokens are lost from the store at the end of each time slice, in addition to any that were used to pay for the execution of instructions. The leak rate of the store is determined by the parameter `ets_leak_rate_per_timeslice`, described in Section A.1.

Cell Death

If the number of tokens in this store falls below a particular threshold (defined by the global parameter `ets_lower_threshold`, described in Section 4.8), the cell dies. Additionally, when the maximum number of cells allowed in the system (as defined by the parameter `max_cells_per_process`) has been reached, the processor will kill off a number of cells which have the smallest number of stored energy tokens,[9] in order to make room for new cells. It is therefore essential that a cell maintains a reasonable level of energy tokens in its store. (There is one other way in which a cell may die—it can terminate itself by executing the `kill` instruction.)

When a cell dies, any energy tokens remaining in its Energy Token Store are distributed to the local environment. More information about energy tokens is given in Section 4.5.2.

4.3.6 Cell Division and Reproduction

It has already been mentioned that a cell only has read, write and execute permission within its own boundaries. Considering that the primary function of the cells is to make copies of themselves in other areas of the system's memory, this may seem like an odd restriction. However, the mechanism of cell division and reproduction employed in Cosmos was inspired (albeit fairly vaguely) by the process of cell division in biological organisms.

The Nucleus Working Memory. Each cell has an area called the Nucleus Working Memory, which is just a WritableInfoString. The cell can compose arbitrary bit strings in this area,[10] but in the normal operation of a self-replicating program, it would construct

[9] In this situation, the choice of which cells to kill is actually stochastic, with the level of a cell's Energy Token Store determining the probability of its being killed.

[10] The only restriction is that there is a maximum length to which these strings are allowed to grow, defined by the global parameter `info_string_size_limit`. This is to prevent the situation in which

a copy of its genome here. Thus, rather than directly writing instructions one at a time to a new area of memory (as in Tierra, for example), a Cosmos cell copies its genetic information into its own Nucleus Working Memory. When the genome has been copied in this way, the cell may issue a `nwm_divide` or a `nwm_split` instruction. These have the effect of transferring the contents of the Nucleus Working Memory into a new cell, which will be placed at a nearby grid position. The former instruction creates a cell which is completely separated from the parent cell (i.e. a new child organism), whereas the latter creates a cell which will remain a member of the same organism (i.e. an extra process in a parallel program: see Section 4.3.9).

In either case, upon division the contents of the Energy Token Store, Promoter Store and Repressor Store are divided equally between parent and child cell. The other main structures of the new child cell (i.e. the Nucleus Working Memory, the Received Message Store and the Communications Working Memory)[11] are initially empty.

4.3.7 Inter-Organism Communication Structures

Two major cell structures remain to be explained; these are the Received Message Store and the inter-organism Communications Working Memory. These two structures are both concerned with communications between organisms. The former is used to store incoming messages from other organisms, and the latter is used to compose messages to be sent out to other organisms.

The communications aspect of these structures is described in more detail in Section 4.6.1, but the part they play in the functioning of the cell is explained here.

The Communications Working Memory. The Communications Working Memory, like the Nucleus Working Memory, is a WritableInfoString (with a limited maximum length) which a cell can use to compose arbitrary sequences of bits. A cell can then issue a `cwm_send` instruction to broadcast the contents of the Communications Working Memory into the environment (explained in Section 4.5.5). The Communications Working Memory does not directly affect the functioning of the cell in any other way.

---

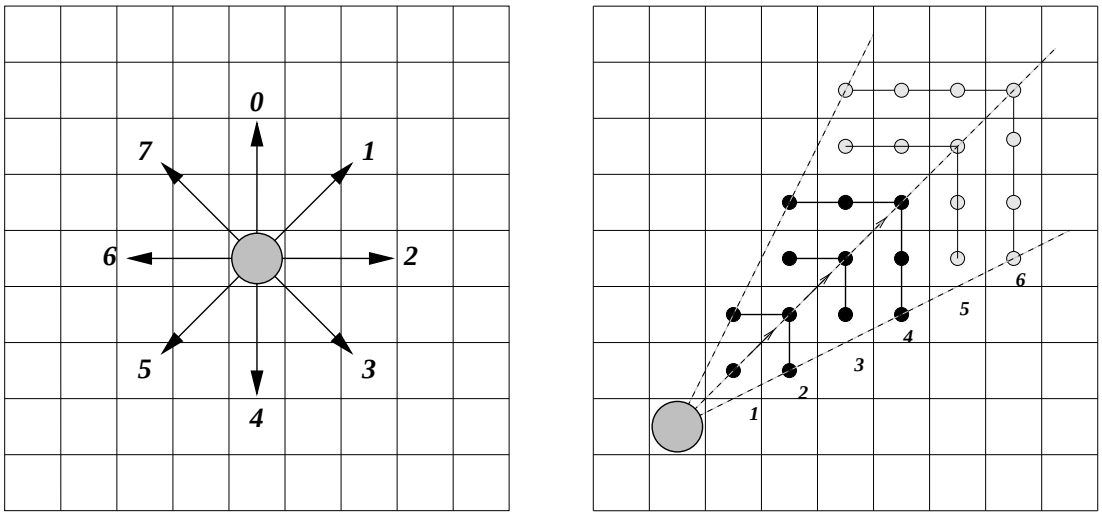
a program evolves which gets stuck in an infinite loop writing to the Nucleus Working Memory, eventually using up all of the memory in the system.

[11] The function of these latter two structures is explained in Section 4.3.7.

The Received Message Store. Inter-organism messages take the form of BitStrings. When they are being composed in the Communications Working Memory they are WritableInfoStrings, when they are broadcast in the environment they are converted to EnvironmentalInfoStrings, and when they are received by others cells into their Received Message Stores, they become plain InfoStrings.

A cell can issue a `rms_receive` instruction to receive messages which have been broadcast from nearby grid positions. These messages (which are EnvironmentalInfoStrings), like all InfoStrings, have a type (a number between 0 and 15) associated with them, and the value of a cell's **dx** register at the time that it issues a `rms_receive` specifies which type of messages are to be received. In addition, the search in the environment for EnvironmentalInfoStrings of the specified type only proceeds in a certain direction; starting from the grid position of the cell that issued the instruction, the search emanates in one of eight directions, specified by the low three bits of the **cx** register (see Figure 4.3(a)). The search proceeds one grid square at a time, covering all grid squares in the specified eighth of the area around the cell until a certain number of grid squares have been searched (defined by the global parameter `rms_receive_search_area`). For example, Figure 4.3(b) shows a cell searching in direction 1. If `rms_receive_search_area` is set to 12, say, then the grid positions marked with black dots will be searched. The search emanates from the cell along a series of wavefronts—the grid position on wavefront 1 is searched first, followed by those on wavefront 2, then 3, then 4. At this point, 12 positions have been visited, so the search stops. Any EnvironmentalInfoStrings of the specified type found in this area are copied into the cell's Received Message Store as InfoStrings. (A cell may extend the reach of a search by re-issuing an identical `rms_receive` instruction from the same grid position within a certain time limit after the first one. This time limit is specified by the global parameter `max_time_for_msg_receive_reinforcement`. If a cell does this, the search will continue outwards from the last grid position searched previously. In the example of Figure 4.3(b), the grid positions marked with gray dots, on wavefronts 5 and 6, will be the next 12 positions searched in this situation.)

The host cell may process these received messages, using the `str_switch` and `adr` instructions to set the **ax** register to an address within a message, and using the instruction `mov_ic` to sequentially read the message.



(a)
A cell can search for messages in one of 8 directions

(b)
A cell searching for messages in direction 1. See text for details.

Figure 4.3: Searching for Communications with the `rms_receive` Instruction.

Messages in the Received Message Store are normally treated as passive structures which may be inspected by the host code, but this is not always the case. As already mentioned, each message in the store has an associated type. The host code of the cell—the genome—being an InfoString, also has a type associated with it.[12] If any message in the Received Message Store happens to be of the same InfoString type as the cell's genome, then it may potentially be used as additional genetic material, and translated into executable instructions. In other words, promoters and repressors may bind to it in just the same way as they can bind to the genome. If the active promoter does indeed bind to a message in the Received Message Store, translation begins along it just as it would on the genome. A cell has several lines of defence against such parasitism, which are mentioned in Section 4.6.1.

A situation where the execution of code from messages in the Received Message Store may be particularly common is when the parameter `neighbouring_genomes_readable` is set to `yes`. In this case, whenever a new promoter becomes active in the cell (see Section 4.3.3), rather than trying to first find a binding site on the cell's genome, or even on eligible messages already resident in the Received Message Store, the cell first

[12] The type of the cell's genome cannot be directly altered, and is passed on to children when the cell splits or divides. However, it is subject to mutation like any other part of the cell (see Section 4.5.7). Therefore, it is possible for organisms with different genome types to emerge in the system.

imports copies of the genomes of any immediately neighbouring cells, one by one, into its Received Message Store. Note that this importation occurs automatically, without the host cell having to issue a `rms_receive` instruction, and without the neighbouring cell having to make a copy of its genome and issue a `cwm_send` instruction. Each imported message (the copy of the neighbouring cell's genome) is checked for a binding site for the new promoter. If a site is found, the message remains in the Received Message Store, and the cell starts executing instructions from it, starting in the position immediately following the binding site (see Section 4.3.3). If no binding site is found, the cell deletes the imported message from its Received Message Store and imports the genome of the next neighbouring cell, if there are any remaining. Only after all the neighbouring genomes have been checked in this way will the cell consider searching for a binding site on existing messages in the Received Message Store, and finally on the cell's genome itself. This mechanism was incorporated into the system in an effort to simulate the ability of programs in Tierra to read the code of neighbouring programs [Ray 91].

4.3.8 Other Structures

There are a number of other structures associated with a cell, which are mentioned briefly here.

Registers There are four (16 bit) registers. The registers **ax** and **bx** are used primarily for storing and manipulating addresses, whereas the registers **cx** and **dx** are used for arithmetic. The main use of the **ax** register is to store addresses returned by the `adr` instruction. This instruction looks for a specified bit string along the genome (or other eligible InfoString), and, if found, returns the address of the first bit of the matching area into the **ax** register. The address is simply the (zero-based) position of the bit from the left of the genome. The `mov_ic` instruction can be used in conjunction with `adr` to read an instruction from the genome, at the address pointed to by the **ax** register, into the **cx** register. Details of these instructions are given in Section A.2. (There is actually a slight complication involved with the use of `adr` and `mov_ic`; these instructions do not only work with the genome, but can also be used on InfoStrings in the Received Message Store, as already mentioned. Each cell actually keeps a pointer called the ADRStringPointer, which normally points to the genome. However, it can be

changed to point to one of the InfoStrings in the Received Message Store by the use of the `str_switch` (or similar) instruction. The `adr` and `mov_ic` instructions always work on the InfoString currently pointed to by the ADRStringPointer.)

Flag There is one flag, used mainly to signal unusual or error conditions in the execution of some instructions.

Stack Each cell has a single stack, with a limited maximum capacity (defined by the global parameter `stack_size_limit`). Instructions are included in the language for pushing numbers onto the stack and for popping numbers from it.

Flaw Rate Each cell has a parameter which defines the frequency with which flaws occur in the execution of instructions (see Section 4.5.7). This flaw rate is subject to mutations (Section 4.5.7), so it may evolve over time.

Statistics and Housekeeping Information There are various other minor structures associated with a cell, mostly concerned with keeping statistics of the cell's lineage and activity (for future analysis) and with keeping track of various activities within the cell. These structures are not explained in detail here, but some are mentioned in passing throughout the rest of this chapter where appropriate.

4.3.9 Parallel Programs (Multicellular Organisms)

It has already been mentioned that the design of Cosmos was guided by an analogy to cellular biological organisms (Section 4.1). In order to model not just unicellular organisms, but also multicellular ones, Cosmos has been designed to support parallel programs—an analogy to multicellularity. Furthermore, it allows programs to dynamically create new parallel processes as they are running, as an analogy to the growth of a multicellular organism from a single celled origin.

All programs in Cosmos are instances of the Organism[13] class. An Organism may contain one or more Cells (each Cell being essentially an individual process). There is therefore no fundamental difference in the representation of serial and parallel programs; a serial

[13] A capital 'O' is used here to emphasise that we are talking about the specific implementation details. However, as the Organism class encapsulates the functionality of an organism, the two terms can be used interchangeably. Therefore, in the rest of the document I shall just use the term organism (with a small 'o'). The same applies for cells and the Cell class.

program is just an Organism which has only one Cell, while a parallel program is an Organism with more than one Cell.

Topology of a Multicellular Organism

In a parallel program, each cell has a specific position in the environment (just like any other cell). The only restrictions on the placement of cells within a parallel program (beyond those defined for all cells by the global parameters) are that every cell within the organism must be adjacent to (i.e. occupy one of the eight neighbouring grid positions) at least one other cell owned by the organism, and that two cells within the same organism cannot share the same grid position. The topology of an organism is important in terms of its intercellular communications, as any given cell can only exchange regulators and energy tokens with immediately adjacent cells within the organism. By means of this transfer between cells in a multicellular organism, the behaviour of any cell is affected by the behaviour of its neighbours. See Section 4.6.1 for more details.

As a parallel program develops, an individual cell can actually change its position relative to its neighbours, using the `migrate` instruction. This gives a cell the opportunity of interacting with different neighbouring cells throughout the life of the program.

Energy Transport

As mentioned above, a cell in a multicellular organism can pass energy tokens from its store to its neighbouring cells, using the `et_transport` instruction. In this way, it is possible for a multicellular organism to develop specialised cells that collect energy tokens from the environment and distribute them throughout the rest of the organism, leaving other cells free to specialise in other tasks if necessary.

Fission

It has already been said that all of the cells comprising a multicellular organism are restricted to being located in such a position that they are in contact with (i.e. in an adjacent grid position to) at least one other cell in the organism. However, as individual cells within a multicellular organism can die at different times (in the ways described in Section 4.3.5), it is possible to get a situation where a collection of cells that was

once connected as a multicellular organism breaks into two or more unconnected groups of cells because of the death of one of more cells in the middle of the structure (see Figure 4.4). If such a situation arises, the separate sub-groups of cells each now become separate organisms in their own right. Cell division and organism fission are therefore two distinct ways in which a new organism may be created.

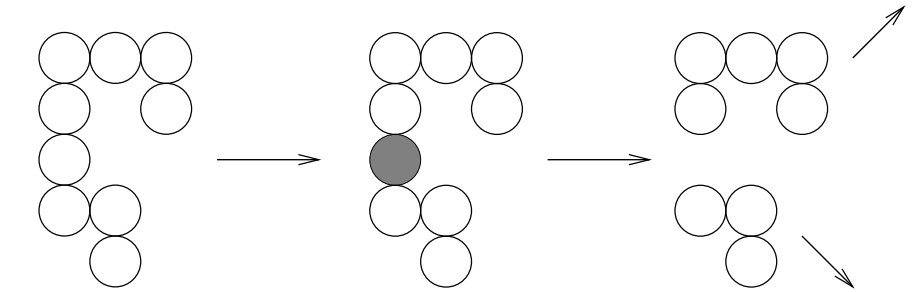


Figure 4.4: An Example of Organism Fission.

The Cost of Multicellularity

The cost of being part of a multicellular organism is governed by the global parameter `multicellularity_penalty_factor`. That is, for each cell in a multicellular organism, this parameter represents the number of energy tokens that are deducted from that cell's Energy Token Store at each time slice for each additional cell with which it is in contact. For example, if a cell is adjacent to two other cells belonging to the same organism (i.e. there are two cells with which it can exchange regulators and energy tokens), then at each time slice, twice the amount of energy tokens as specified by `multicellularity_penalty_factor` are deducted from that cell's store. This parameter therefore defines how expensive it is for a cell to maintain a connection with one other cell in a multicellular organism.

Organism Death

An Organism is composed of one or more cells. In Cosmos there is no specific idea of an organism, as a whole, dying—rather, an organism dies when the last of its constituent cells dies.

4.4 The Programming Language and Representation

REPLiCa,[14] the programming language in which the self-replicating programs are written, is based upon the Tierran language [Ray 91], with some changes and additions to support the extra functionality of Cosmos. Like Tierran, REPLiCa has been designed to be robust, in the sense that there is little syntactical structure to a program, so that any random collection of REPLiCa instructions will form a valid program that will do *something* (maybe not anything sensible, but it will not cause the system to crash). The REPLiCa instruction set is listed, with annotations, in Section A.2.

One big difference between Tierran and REPLiCa is in the mechanism for control flow branching and jumping. Tierran uses a system of template-driven jumping (see [Ray 91] for details). REPLiCa does not have jumps of this kind; rather, jumps may be accomplished in two different ways. The first, primarily for single jumps rather than loops, is just by the creation of an appropriate promoter to bind to the desired jump destination, either followed by the deletion from the Promoter Store of the currently active promoter (using the `reg_destroy` instruction), or by the issuing of a `stop` instruction—both of which have the effect of stopping the execution of the current section of code and activating the new promoter.[15] The second way by which (local) jumps may be performed is by the use of the `set_jmp` and `jmp` instructions. Each cell contains a pointer called the LocalJumpPointer which, if set, points to a position on the genome (or currently active InfoString in the Received Message Store). When a `set_jmp` instruction is executed, this pointer is set to the address of the next instruction. When a `jmp` instruction is executed, control passes to the instruction pointed to by the LocalJumpPointer (if it is set, otherwise no jump is performed). The LocalJumpPointer can be cleared with the `clr_jmp` instruction.

The translation of the bit-string representation of a program on the genome, and the control of execution of the program by promoters and repressors, illustrated in Figure 4.2, has already been explained in Sections 4.3.2–4.3.4.

---

[14] 'REPLiCa' is an acronym for Robust Evolvable Programming Language for Cosmos.

[15] Of course, when programs are evolving, especially when we are considering parallel programs, there may be more than one promoter in the Promoter Store at one time. However, here we are describing how a human might design a program that performs a jump—evolution would probably go about designing a program in a very different way.

4.5 The Environment

4.5.1 The Grid

As mentioned in Section 4.2.2, cells in Cosmos live in a discrete two-dimensional spatial environment (the 'grid'). At the start of each time slice, a number of energy tokens are deposited to each position on the grid (see Section 4.5.2). Cells can collect these energy tokens by using the `et_collect` instruction (see Section 4.5.3). If energy tokens are scarce at a cell's current location (or indeed for any other reason), the cell (to be precise, the whole organism) may move around the grid (see Section 4.5.4). For multicellular organisms, each cell must occupy a different grid position, i.e. all organisms are 'flat' (cells cannot pile on top of each other in the same grid position). However, cells from different organisms *can* occupy the same grid position. What this means is that all organisms are flat, but they can 'slide over' each other, and in this sense the environment is two-and-a-half dimensional.

4.5.2 Distribution of Energy Tokens

At the start of each time slice sweep across all of the cells in the population (in the routine `DistributeEnergyTokens`, described in Section 4.7), the Cosmos operating system releases a certain number of energy tokens into the environment. These tokens are then available to be collected by cells, by the use of the `et_collect` instruction. At the end of each time slice sweep (in the routine `AttenuateEnvironmentalEnergy`, also described in Section 4.7), the operating system takes a number of energy tokens away from each grid position. In the current implementation, different grid positions may receive different numbers of energy tokens at the beginning of each time slice sweep (determined by the various distribution schemes described below), but all positions have the same number of energy tokens removed at the end of each time slice sweep (specified by the parameter `number_of_energy_tokens_per_grid_pos_per_sweep`, if they have that number available). If the number of energy tokens received by a grid position in a time slice sweep exceeds the number removed from it, and they are not collected by cells during that sweep, the excess tokens remain there for future collection. A grid position may therefore sometimes accumulate a relatively large number of energy tokens (up to a maximum limit defined by the global parameter `max_energy_tokens_per_grid_pos`)

if there is not much demand for them by cells in the locality.

The distribution of energy tokens across the grid may follow a number of different patterns, defined by the global parameter `energy_distribution_scheme`. At present, four such patterns are defined: `land`, `sea`, `mixed` and `random`. Note that the total number of energy tokens distributed to the environment at each time slice sweep is always specified by the product of the parameter `number_of_energy_tokens_per_grid_pos_per_sweep` with the number of squares in the grid. The different distribution schemes determine how many of these tokens are distributed to individual squares. The different schemes work as follows:

Land Each grid position receives a constant number of energy tokens from one time slice to the next. In the current implementation, there is one extra parameter, `x_delta`, associated with this sort of energy distribution, which defines the gradient of the distribution from the left-hand side of the grid to the right-hand side. See Figures 4.5(a) and 4.5(b) for examples of this type of distribution.

Sea In contrast to `land` distribution, for `sea` distribution each grid position receives a varying number of energy tokens from one time slice to the next. During each time slice, energy tokens are distributed to grid positions which are located under a 'wave'—a vertical band which moves one position to the right after each time slice: see Figure 4.5(c). Grid positions which are not located under a wave in the current time slice receive no energy tokens for that time slice. In the present implementation there are two parameters associated with this method; `wave_width` and `number_of_waves`. The former specifies the width, in grid positions, of a single wave, and the latter specifies how many waves are to be fitted in to the grid from left to right (the waves are evenly spaced across the grid).

Mixed This is a mixture of `land` and `sea` distributions, with the top portion of the grid receiving energy according to the `land` distribution, and the bottom portion according to the `sea` distribution. The relative sizes of these top and bottom portions of the grid are determined by the global parameter `land_fraction`. An example is shown in Figure 4.5(d).

Random Energy tokens are distributed in packets with size determined by the global parameter `energy_distribution_random_chunk_size` to randomly chosen grid

positions, until the correct total number of energy tokens have been distributed. An example is shown in Figure 4.5(e).

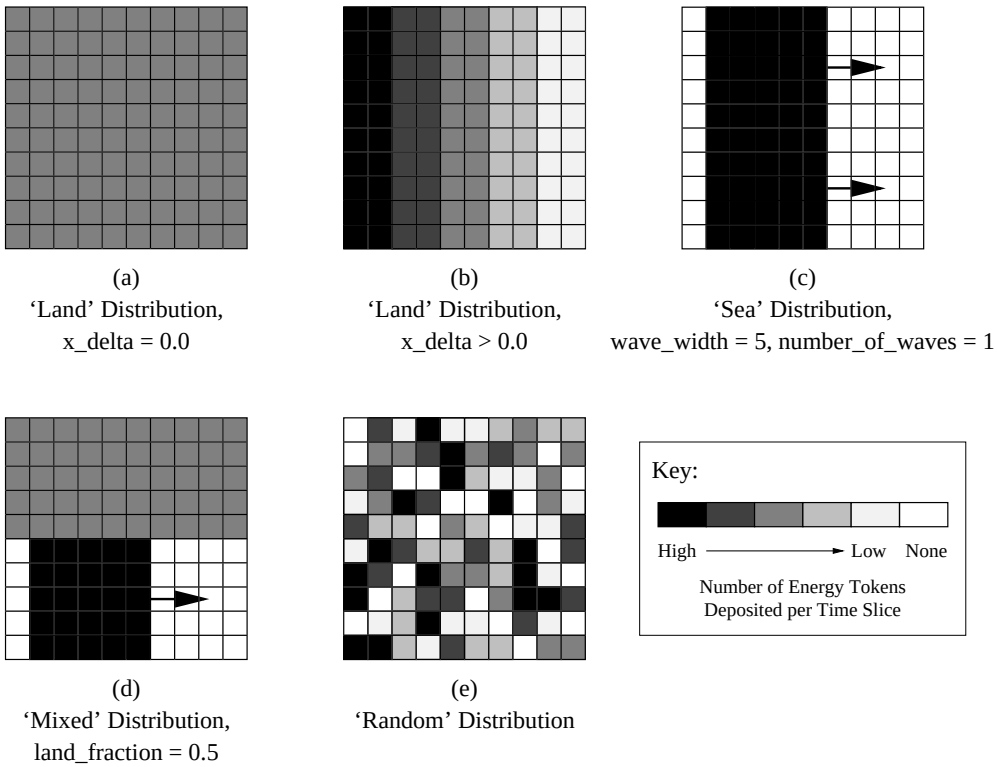


Figure 4.5: Different Patterns of Energy Token Distribution.

A multicellular organism may also pass energy tokens between its cells (using the `et_transfer` instruction), leading to the possibility of some of the cells specialising in energy token collection and distribution of these tokens to the other cells in the organism.

With such a system of CPU-time allocation, programs may potentially evolve which operate on a wide variety of time-scales. For example, very short programs may exist which quickly grab just enough energy tokens to make a copy of themselves, while much more complicated programs may coexist which gather large numbers of tokens over long periods of time, and reproduce at a much slower rate.

When a cell dies, any unused energy tokens are passed back to the local environment (where they may be collected by other organisms). This mechanism provides potential selection pressure for the evolution of organisms that kill other organisms in order to collect the energy tokens thus released into the environment. This could

happen if, for example, an organism transmitted EnvironmentalInfoStrings containing the `kill` instruction, which another organism subsequently received and executed (see Sections 4.3.7 and 4.6.1 for further details of how this would work).

4.5.3 Collection of Energy Tokens

In the present implementation a choice of two energy collection schemes, `shared` and `private`, is provided. The global parameter `energy_collection_scheme` determines which scheme will be used.

Shared Energy. Under this scheme, when a cell issues an `et_collect` instruction to collect energy tokens from the environment, it first tries to collect spare tokens from its current grid position. However, if the grid position does not contain sufficient energy tokens, the cell then looks for other cells at the same grid position or in one of the eight neighbouring grid positions. If other cells exist in one of these nine locations, energy tokens will be extracted from the Energy Token Store of one (or more) of these (at random) until the cell has obtained the normal quota of energy tokens for one execution of `et_collect` (as defined by the global parameter `number_of_energy_tokens_per_collect`).

Private Energy. With this scheme, if the local grid position does not contain enough energy tokens for an `et_collect`, the cell just takes what is there, but does not attempt to gain additional energy tokens from neighbouring cells.

4.5.4 Moving around the Grid

A cell does not have to remain in its original grid position, but can move around by using the `move` instruction. The contents of the **cx** register at the time the instruction is issued determines in which direction the cell will try to move (the low 3 bits specify a direction from 0 to 7, as indicated in Figure 4.3(a)).

However, movement is complicated by the fact that a cell may be part of a multicellular organism (in which other cells are also trying to move, possibly in different directions). The organism must move as a whole, so what actually happens is that the issuing of

a `move` instruction by a cell is actually a *vote* to move in a particular direction rather than an instruction that has immediate effect. At each time slice, an organism counts up all of the movement votes from its constituent cells, and decides how to move as follows:

A normalised total movement vector $\mathbf{A}$ is calculated by summing all the individual votes of cells within the organism:

$$\mathbf{A} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{a}_i \tag{4.1}$$

where n is the total number of cells in the organism, and $\mathbf{a}_i$ is the unit vector of movement (in one of eight possible directions) specified by cell i (or $\mathbf{0}$ if the cell did not issue a `move` instruction during the current time slice).

A 'multiple movement factor', M, is then calculated. This factor determines the extent to which two or more cells moving in tandem within an organism are more efficient than would be expected by simply summing their individual movements. M is defined as:

$$M = (m - 1)L + 1 \tag{4.2}$$

where m is the number of cells within the organism that individually issued a `move` instruction; and L is the 'constant of leverage' when two or more cells move at the same time ($L \geq 0$). L is defined by the global parameter `movement_leverage_factor`.

Movement is further complicated in the situation where the organism is overlapping (or partially overlapping) another organism on the grid. In this case, there is a 'friction' term F which slows the organism down as it attempts to move over other cells. This term is defined as follows:

$$F = \frac{o}{n} \tag{4.3}$$

where o is the number of cells in the organism which share a grid position with cell(s) from other organisms. The friction factor can actually be turned on or off with the global parameter `apply_friction_factor`. If it is turned off, F is effectively set to zero.

The total movement that the organism attempts to make, $\mathbf{X}$, is therefore specified by

$$\mathbf{X} = M(1 - F)\mathbf{A} \tag{4.4}$$

The organism moves from its current position by the distance and direction given by **X**, unless it reaches the edge of the grid, in which case it stops at that point (if the grid boundary does not wrap).

4.5.5 Inter-Organism Communications

If a cell broadcasts an inter-organism communication using the `cwm_send` instruction (as mentioned in Section 4.3.7), the contents of its Communications Working Memory is packaged into an EnvironmentalInfoString structure (with an initial intensity specified by the global parameter `envinfostring_initial_intensity`, and a type specified by the low four bits of the **dx** register). This EnvironmentalInfoString is deposited in the environment in the same grid position as the cell, where it can be detected by other cells (by using the `rms_receive` instruction, described in Sections 4.3.7 and 4.6.1).

Each grid position in the environment can hold one EnvironmentalInfoString of each of the 16 possible types. If a string of the same type already exists in the grid position when a `cwm_send` message is issued, the existing string is deleted and replaced by the new one.

At each time slice sweep (in the `AttenuateMessageIntensities` routine, described in Section 4.7), the intensity of each EnvironmentalInfoString is attenuated according to the following equation:

$$I_{n+1} = k(I_n)^p \tag{4.5}$$

where I_n is the intensity at time n, and k and p are constants defined by the global parameters `envinfostring_decay_constant` and `envinfostring_decay_power` respectively. When the intensity of any string falls below a certain threshold (defined by the global parameter `envinfostring_lower_threshold`), the string is deleted.

There is one additional feature associated with these EnvironmentalInfoStrings, whereby a cell can reinforce the intensity of a message that it has already sent. If the cell reissues the `cwm_send` instruction within a given number of time slices (determined by the parameter `max_time_for_msg_send_reinforcement`), while still in the same grid position, and it has not written anything else into its Communications Working Memory in the meantime, then the intensity of the existing EnvironmentalInfoString is incremented

by a small amount.[16]

4.5.6 Environmental Information

As well as carrying specific inter-organism communications (mentioned in Section 4.3.7 and explained in more detail in Sections 4.5.5 and 4.6.1), the environment also carries summary information about itself. These messages are transmitted (in the form of EnvironmentalInfoStrings) by the environment itself, one at each grid position, and may be intercepted by cells in exactly the same way as they intercept other inter-organism communications. The messages contain the following information (represented in a binary encoding):

- The number of cells at that grid position
- The total number of free energy tokens at that grid position

All of these messages behave just like any other EnvironmentalInfoStrings in the environment; the only distinguishing feature is that they are all given an InfoString type of 15. (There are no restrictions about organisms using the same type number for their own communications.) They may be picked up by any cell using the `rms_receive` instruction.

4.5.7 Mutations and Flaws

Little has so far been said about the role of mutation in Cosmos. Mutation is a vital process from the evolutionary point of view, as it provides a continual source of genetic novelty for selection to work upon. Mutations occur naturally throughout the system at a low rate, and may affect most of the structures within the cell (i.e. the Genome, the Received Message Store, the Nucleus Working Memory, the Communications Working Memory, the Promoter Store, the Repressor Store, the flaw rate, the stack, the registers and the flag). For structures which are based upon BitStrings, mutations are governed by the global parameter `mutation_period`, which specifies the probability of an individual bit within the structure being flipped. For structures based upon integer

[16] To be precise, the magnitude of the increment is kI^p, where I is the current intensity, and k and p are constants defined by the global parameters `envinfostring_decay_constant` and `envinfostring_decay_power` respectively.

numbers (the flaw rate, stack and registers), mutations occur at the same rate as for BitStrings, but the details are slightly different. For the flaw rate, a mutation causes a random increment or decrement in the current value within predefined limits.[17] For the stack, a mutation will, with equal likelihood, either cause a random number to be pushed onto the stack, or the top number to be popped off it. For registers, a mutation will cause the register's current value to be replaced by a random value. Mutations also affect the cell's flag at the same rate, causing the flag's state to be inverted.

In addition, variety may also be introduced into an organism by the flawed execution of instructions in its genome.[18] When a flaw occurs (which happens at a rate defined by an individual cell's flaw rate, as described in Section 4.3.8), the instruction which is about to be executed, rather than just being executed once, will either be executed twice (successively) or not at all. (The choice is random, with both events occurring with equal likelihood.) The effect of a flaw is therefore that instructions may occasionally produce abnormal results, such as an `inc_a` instruction adding 2 to the value of the **ax** register instead of 1.

Despite this distinction between mutations and flaws, the net results are the same. If the error affects what gets written to the Nucleus Working Memory of a cell just before it issues a `nwm_divide` instruction, then it will be passed on to the child organism and become a permanent addition to the gene pool. On the other hand, if the error does not affect the contents of the Nucleus Working Memory (even indirectly), and it does not affect the regulators that get passed on to any offspring, then it will only affect the current organism and will not be inherited by child organisms. From an evolutionary point of view, only the former scenario is important.

4.6 Actions and Interactions

The methods available to cells and organisms for interacting with the 'physical' environment and with other cells and organisms have already been discussed: issues such as

[17] To be precise, the flaw rate can change by plus or minus n parts per thousand, where n is determined by the parameter `flaw_period_max_change_per_thou`.

[18] Tierra features both mutations and flaws (although the mechanisms for flaws is somewhat different) but in subsequent work by Chris Adami and Titus Brown with their Avida system the authors suggested that flaws played only a minor role in evolution compared to mutations [Adami & Brown 94]. Informal observations from preliminary runs of Cosmos suggested that flaws in the execution of instructions significantly increase the rate at which useful mutants are produced.

the collection of energy tokens from the environment, and moving around the grid, were explained in Section 4.5; intercellular communications (i.e. the transfer of energy tokens and regulators) have been mentioned in Sections 4.3.9 and 4.5.2; and inter-organism communications have been mentioned (from the point of view of the mechanisms involved) in Sections 4.3.7 and 4.5.5. In the present section, more will be said about some higher-level effects and implications of both types of communication.

4.6.1 Implications of Intercellular and Inter-Organism Communications

The general philosophy governing the design of the communication facilities in Cosmos was to provide the organisms with as rich an environment as possible. In particular, the inter-organism communications instructions allow organisms to exchange arbitrary messages. The idea is that, as in nature, many possibilities for communication are provided by the 'physics' of the system. The question of whether these possibilities are realised or not is left to the evolutionary process.

Intercellular Communications

As mentioned in Section 4.3.9, a cell which is a member of a multicellular organism can communicate with other cells in the organism by sending regulators from its Promoter Store and Repressor Store (using the `reg_transport` instruction). In this way, the execution of code in a particular cell may be influenced by many other cells in the organism, because regulators which are sent from one cell to another will influence which sections of code get executed in both cells. Therefore, although each cell in a multicellular organism has the same genome (assuming there are no somatic mutations), each cell may be executing different parts of this genome at any given time.

As a cell within a multicellular organism can only exchange regulators and energy tokens with its immediate neighbours, organisms adopting different shapes will have different capacities for internal communication and regulation. Within an organism, cells can also actively switch neighbourhoods by migrating to a different position (using the `migrate` instruction). If multicellular organisms do evolve in any runs of Cosmos it will be of interest to see what sorts of shapes they adopt, and how much variety in shape exists across the population.

Inter-Organism Communications

The mechanisms for inter-organism communications were introduced in Sections 4.3.7 and 4.5.5. A cell can broadcast an arbitrary message using the `cwm_send` instruction, and receive other messages from the environment—sent from cells in other organisms, cells within the same organism, or from the environment itself (Section 4.5.6)—using the `rms_receive` instruction. Allowing organisms to exchange arbitrary bit strings has little direct biological analogy. Rather, it is an attempt to equip the organisms with some communication channels in much the way that biological organisms can communicate using channels such as light, sound etc.

Once messages (InfoStrings) have arrived in a cell's Received Message Store, they may be read by the host code (using `str_switch`, `adr`, `mov_ic` and related instructions), and messages of the same type as the genome of the host cell may even be treated as executable code, as described in Section 4.3.7. This allows for genetic information to be exchanged between organisms in a manner analogous to the direct exchange mechanisms employed by lower biological organisms such as viruses and bacteria.

If the foreign code is detrimental to the performance of the host cell, the host may be expected to evolve measures to prevent the foreign code from being executed. This can be achieved in a number of different ways, such as by using a different type number for its own genome (which may come about by mutation), by removing the foreign code from the Received Message Store (using the `str_remove` instruction), or by not receiving the foreign code in the first place. If, however, the foreign code is beneficial to the host, then it may be expected that the host will evolve to copy this code into its Nucleus Working Memory so that it will become incorporated into the host genome in future generations. The system is even flexible enough to allow for the possibility of the evolution of sexual reproduction (see Section 6.7 for details of a hand-written sexual organism).

4.7 The Top-Level Algorithm

A pseudo-code listing of the top-level algorithm is shown in Figure 4.6. Most of it should be self-explanatory. The `Inoculate` routine constructs a number of self-replicating programs and places them at specified positions on the grid (governed by the para-

```
Inoculate
currentTimeSliceSweep = 1
while (stopping criteria not met)
{
      DistributeEnergyTokens
      AttenuateMessageIntensities
      ExecuteCellTimeSlices
      PerformOrganismLevelOperations
      if ((currentTimeSliceSweep MOD mutation_application_period) = 0)
           ApplyMutations
      if ((currentTimeSliceSweep MOD overcrowding_check_period) = 0)
           CheckOvercrowding
      if ((currentTimeSliceSweep MOD env_info_broadcast_period) = 0)
           BroadcastEnvironmentalInfo
      ExportData
      AttenuateEnvironmentalEnergy
      currentTimeSliceSweep = currentTimeSliceSweep + 1
}
```

Figure 4.6: The Top-Level Algorithm.

meters `ancestor`, `number` and `placement`). The stopping criteria for the main loop may be to run for a given number of time slices (if the parameter `limited_run` is set to `yes`) or to run indefinitely (only stopping if and when all programs on the grid have died out). `DistributeEnergyTokens` places a number of energy tokens in each grid position, as described in Section 4.5.2. `AttenuateMessageIntensities` refers to the intensities of any EnvironmentalInfoStrings that currently exist in the environment. `PerformOrganismLevelOperations` checks, for each organism, whether a fission has occurred by the death of one of more cells within it (see under "Fission" in Section 4.3.9), subtracts energy tokens for each cell in a multicellular organism depending on how many neighbours the cell has (see under "The Cost of Multicellularity" in Section 4.3.9), and finally calculates and performs any movement of the organism from the contributions made by individual cells (Section 4.5.4). `CheckOvercrowding` checks whether the current population of cells on the grid exceeds the limit specified by the global parameter `max_cells_per_process`. If so, a fraction of the population (specified by the parameter `population_cutback_on_overcrowding`) is killed off. The choice of which cells to kill in this situation is stochastic, but is based upon how much energy each cell has stored in its Energy Token Store. `BroadcastEnvironmentalInfo` generates an environmental message of each grid position, as described in Section 4.5.6. `AttenuateEnvironmentalEnergy` removes a number of energy tokens from each grid

position, as described in Section 4.5.2.

4.8 Global Parameters

The Cosmos system as described contains a considerable number of global parameters. These are listed and described in Section A.1. The number of parameters is much larger than in most other artificial life platforms, but this is largely because other platforms often have many features which are hard-coded in a fairly arbitrary way. In contrast, Cosmos was designed to allow the user a great degree of control over the system's configuration.

4.9 Input and Output Files

The configuration of an individual run is specified in a number of files which Cosmos reads when the run commences. These files contain details of non-default parameter settings, of the mapping between instructions in the REPLiCa programming language and the binary encoding used to represent them in a cell's genome, and of user-defined ancestor programs. Full details of these input files are given in Section A.5.1.

The core Cosmos system is a stand-alone application. In order to allow the analysis of an evolutionary run, a number of log files containing information about the different organisms are written by the system during the run. The files may then be used to produce graphs and statistics about the run. These output files are described in Section A.5.2.

4.10 Major Differences between Cosmos and Tierra

In this section the main areas in which Cosmos differs from Tierra are highlighted. A fuller explanation of why some of these differences were incorporated into the system can be found in [Taylor & Hallam 97]. In the following, the extension of standard Tierra to deal with parallel processes, as described in [Thearling & Ray 94] and [Thearling 94], is referred to as 'Parallel Tierra'.

Cellular Structure. An individual program (or more precisely, an individual process, which may be serial or parallel) in Cosmos has many more structures associated with it than do programs in Tierra. Tierran programs just have the list of instructions, a program pointer, registers and a stack. In contrast, Cosmos programs also have all of the structures explained in Section 4.3. The idea was that they should incorporate some of the features (e.g. regulators, translation machinery, and areas where new strings may be constructed) observed in cellular biological organisms. The programs must rely largely on communications to interact with the outside world, and cannot directly read the code of their neighbours.

Regulator System. The regulator systems of Cosmos (promoters and repressors: see Section 4.3.3) have no equivalent in Tierra. They were designed specifically to allow cells in a multicellular organism to be able to influence which sections of code were being executed in neighbouring cells, thereby promoting cell differentiation and specialisation. The design of the regulator systems was inspired by the processes of chemical signalling between cells, and the use of promoter sequences and repressors within cells, in biological organisms.

CPU-time Allocation and Energy Tokens. In Cosmos, each cell has to pay one energy token for every instruction it executes. Cells must collect these tokens from the environment, and store them in their Energy Token Store. A cell dies when the number of tokens in its Energy Token Store falls below a threshold (defined by the parameter `ets_lower_threshold`). Furthermore, if the population size exceeds a threshold (defined by the parameter `max_cells_per_process`), cells are killed off stochastically, but those with fewer energy tokens in their Energy Token Store have a greater chance of being killed. A cell can therefore exert considerable influence over its own longevity, via its success at collecting energy tokens from the environment.

In contrast, programs in Tierra have little control over their longevity. As individual Tierran programs have no notion of energy levels, a separate 'reaper queue' mechanism is employed to govern cell death. Programs can move up the queue if they cause error conditions during execution (see Section 3.2.1), but in general the probability of death increases with age [Ray 91]. The reaper queue therefore effectively imposes an upper

limit on the lifespan of programs, whereas there is no theoretical upper limit in Cosmos.

Additionally, the energy token scheme in Cosmos introduces the idea of a competition for the available energy, which is missing in Tierra. Furthermore, if the parameter `energy_collection_scheme` is set to `shared`, cells may extract energy tokens from their neighbours. In this situation, a cell is a potential energy resource for other cells, and, if environmental energy were scarce, it would become advantageous for a cell to kill its neighbours by draining their energy. If cells could defend themselves against such attacks, some sort of coevolutionary process might arise from such interactions.

Read, Write and Execute Privileges. Tierran programs only have write access within their own 'cell membrane' (apart from when they are in the process of creating a daughter cell, when they also have write access to a specific additional chunk of memory, which has been allocated by the Tierra operating system). A similar situation exists in Cosmos. However, Tierran programs have read and execute privileges for *all* areas of instruction memory, so that they can directly examine the code of other programs, and even execute this code. Cosmos cells, on the other hand, only have direct read and execute privileges within their own cell membrane, and must rely on the system's communication facilities to interact with other cells (see Section 4.6.1). This restriction in Cosmos is related to the guiding analogy of the biological cell, which cannot directly read the genetic code of a neighbouring cell.

Exchange of Messages and Genetic Information. The Cosmos mechanisms for the direct exchange of arbitrary messages (which may, for example, be copies of genetic information) have no parallel in Tierra. This difference is linked to the differences in read, write and execute privileges described in the previous point.

Division Process. This point is related to the previous two. As a Cosmos cell only has write access within its own cell membrane even when it is composing a copy of itself, this copy must first be composed within the parent cell (in the Nucleus Working Memory). The copy is then issued *en masse* to a new memory location.

In Tierra, a cell is first allocated a new block of memory, then writes a copy of itself into this memory, and finally 'divides', signalling that the block of memory is now a

new organism in its own right.

There is not a great deal of difference between the two mechanisms, but an advantage of the Cosmos method is that it allows an organism to *reproduce* (i.e. to create a child organism) and to *grow* (i.e. create a new cell which remains a member of the multicellular organism) using exactly the same technique.

In contrast, Parallel Tierra includes a `split` instruction which adds an additional CPU to the processor structure of the program. This mechanism is natural for a parallel machine architecture with a shared program space, as used with Parallel Tierra. In Cosmos memory is not shared across cells, so that a multicellular program must actually copy itself from one cell to another in order to run in parallel. With this type of architecture, it seems preferable that the bulk of such copying work should be performed by the cells themselves rather than by the Cosmos operating system (but see the further discussion on this topic in Section 7.2.3).

Additionally, having very similar mechanisms for growth and reproduction of organisms is arguably more analogous to the way that multicellular biological organisms may have evolved.

Local Competition. One of the problems that has been observed with the process of evolution in Tierra is that it suffers from premature convergence due to global interactions between cells [Adami & Brown 94].

Chris Adami and Titus Brown sought to overcome this problem in their Avida system by giving each of the cells a location on a two dimensional toroidal grid. Cells can only interact with other cells occupying nearby grid positions, thereby slowing down the rate of propagation of evolutionary changes throughout the total population and promoting heterogeneity.

Cosmos addresses this problem by placing organisms on a grid (as in Avida), and by restricting cells to only be able to communicate and interact with other cells within a certain distance on the grid (but see the further discussion on this topic in Section 7.2.3).

Binary Representation. In Tierra, programs are directly represented as lists of instructions. In Cosmos, the program code is represented as a binary string (specifically,

an InfoString), and a translation process is required to produce the executable code. One consequence of this design is the possibility of the evolution of ultra-compact programs which use the same section of bit string to encode multiple sequences of instructions in different reading frames (as is observed in some biological organisms; [Matthews 91] p.144). Another consequence is that it would be easy to modify the system in order to study the evolution of the genetic code itself (i.e. the mapping from bit strings to program instructions).

Size of Instruction Set. The REPLiCa instruction set is about twice as big as that of the Tierran language. Many of the instructions can certainly be removed without having a great impact on the things that programs can do (e.g. the self-replicator listed in Section A.3.1 only uses 17 different instructions). If the genetic code were allowed to evolve, then unused instructions might be expected to be removed from the code by natural selection, allowing common instructions to be represented multiple times.

Memory Model. Cosmos uses a distributed memory model of parallelism, in contrast to the shared memory model of Parallel Tierra. In other words, each cell in a multicellular organism in Cosmos has its own copy of the program code, of the other cellular structures, and of the CPU state information (registers, instruction pointer, etc.). This distributed memory model, together with the Cosmos regulator system, should promote the emergence of differentiation in parallel programs. However, little work has so far been conducted with parallel programs in Cosmos, so it is not yet known how effective this approach really is.

Memory Addressing Scheme. For reading from and writing to structures within cells, Cosmos uses a local addressing scheme for each structure (i.e. the first bit of the Genome, of the Communications Working Memory, and of the messages in the Received Message Store, are all treated as address zero within that particular structure). Cells have no knowledge of their memory location (or that of other cells) in the global addressing scheme of the system. This is in contrast to Tierra, which uses a global addressing scheme. The only ways that cells can interact with each other are therefore by communication; by physical contact, such as by extracting energy tokens from each other (which is possible when the parameter `energy_collection_scheme` is set

to `shared`—see Section 4.5.3) and slowing down passing organisms (see Section 4.5.4); and, for cells within a multicellular organism, by the exchange of regulators and energy tokens.

4.11 Summary

In this chapter I have described the design details of the Cosmos platform. Cosmos is an implementation of a virtual parallel computer that can simulate the concurrent execution of several thousand programs (limited only by the amount of memory available). The general design was influenced by Ray's Tierra platform [Ray 91], but there are some fairly significant differences between the two systems (as listed in Section 4.10). The following chapter is devoted to a detailed description and analysis of the system's behaviour during a single evolutionary run, where the memory (i.e. the grid) is inoculated with a number of hand-written self-reproducing programs. The results of a wide variety of further experiments with Cosmos are described in Chapter 6.

Chapter 5

Cosmos Experiments 1: Detailed Analysis of a Standard Run

In this chapter, the results of a standard Cosmos run (i.e. using the default parameter settings) are described. First of all, the various analysis and visualisation techniques that will be used are explained (Section 5.1). After this, the results of the run are analysed in some detail (Section 5.2). In the next chapter, results of many other runs, designed to explore the parameter space of the system, will be presented.

5.1 Analysis and Visualisation Techniques

One of the reasons that is often given for pursuing research with artificial evolutionary systems is that they allow us to study evolution in carefully controlled conditions, and to record as much data as we wish during the course of an evolutionary run. However, the very fact that we can record anything we wish to can sometimes lead to problems. Without care, it is very easy to drown in megabytes of data. In this section, we consider some of the questions we might want to ask concerning the behaviour of an evolutionary system, and describe the measures that were used in the Cosmos runs to provide answers to these questions.

There is a broad distinction that can be made between measures which relate to individual organisms within the population, and measures which relate to the population

as a whole. We will use both types when analysing the Cosmos runs in this chapter.

5.1.1 Individual-based Measures

In any population of self-replicating entities which are competing against each other for resources required for replication (e.g. energy and materials), there are three factors which determine the rate at which any particular type of replicator, i.e. any particular S-lineage (see Section 2.5), will spread throughout the population [Dawkins 76]. These are the life-span (*A-longevity*) of the individual A-replicators, the rate at which they replicate (their *A-fecundity*), and the error rate[1] of the replication process (their *A-fidelity*). These three factors can be thought of as defining the orthogonal axes of the fundamental space in which evolution is occurring. Adaptive (i.e. non-neutral) evolution would be expected to move the A-replicators' positions in this space either to greater A-longevity, increased A-fecundity or increased A-fidelity (or some combination of these).

In biological systems, the fundamental units of evolution are the (L-)genes, and it does not necessarily follow that the evolution of whole organisms will follow this pattern. For example, evolution of an L-gene towards increased L-fecundity across a group of organisms carrying an A-gene instance of the L-gene does not necessarily entail increased A-fecundity of *all* of the organisms (or A-genes). In Cosmos and similar systems, however, each program as a whole is an encoding of a self-replication algorithm. As such, it can only replicate *as a whole*; the subsections of a program are highly epistatic, so evolution cannot in general vary one part of the program independently of other parts while retaining the functionality of the whole. Furthermore, replication is asexual, so there is generally no exchange of genetic material between individuals when an offspring is produced. It is therefore reasonable, at least to a first approximation, to treat the whole program as a single A-gene and to expect programs as a whole to evolve along these three axes as described above.

A number of measures were therefore chosen to track changes in each of these three factors through an evolutionary run. The plotting technique used for each of these measures was as follows: for time slice windows of equal width from the start to the

[1] Precisely, what matters is the rate at which *selectively significant* errors occur, i.e. errors which exclude the offspring from that particular S-lineage.

end of the run, we plotted the value of the measure for programs that died within that time slice window. For the plots for all three of these factors the data is pruned by only plotting values for individual programs of types which achieved a concentration of at least five individuals at some time during the run (as determined by the parameter `species_count_threshold_for_recording`). The data was further pruned by only recording information for 1 in every 50 eligible programs (as determined by the parameter `morgue_record_period`). In the plots, the darkness displayed at any point reflects the number of individual programs taking that particular value at that particular time (i.e. the more programs, the darker the plot).

For *A-longevity*, we looked at the age at death of each program. An example plot can be seen in Figure 5.7.

For *A-fecundity*, we looked at two measures: the number of time slices between the first and second successful replication of each program (the *replication period*) (this could obviously only be applied to programs that successfully replicated at least twice in their lifetime), and the length of programs. The length of a program is an indirect measure of fecundity—all things being equal, a longer program will take longer to replicate than a shorter program, as the longer one has to copy more instructions. An example plot for replication period can be seen in Figure 5.4, and for length, in Figure 5.2.

For *A-fidelity*, we looked at two measures: the flaw rate,[2] and the proportion of the total number of offspring produced by an individual that were unfaithful (i.e. less than 100% accurate). Example plots of these two measures can be seen in Figures 5.6 and 5.11 respectively.

In addition to the measures of A-longevity, A-fecundity and A-fidelity, we also used a visualisation technique developed by Mark Bedau and colleagues [Bedau & Brown 97]. The fundamental idea behind this technique is:

> "to identify those genotypes[3] that make a difference in the evolutionary process. Generally, we consider a genotype to 'make a difference' if it continues to be active in the evolving system... In [Tierra-like models] the

[2] Actually, we looked at the inverse of flaw rate, the flaw period. This is the expected number of successful instruction executions in a program between successive flaws.

[3] Note we are looking at genotypes here, i.e. groups of identical programs, rather than individual programs.

relative adaptive significance of a genotype is reflected by its concentration in the population. Relatively well adapted genotypes will have a relatively high concentration in the population, and relatively poorly adapted genotypes will we correspondingly scarce. Thus, we here define the [*cumulative evolutionary activity counter*] $a_i(t)$ of the i^{th} genotype at time t as its concentration integrated over the time period from its origin up to t, provided it exists:

$$a_i(t) = \begin{cases} \int_0^t c_i(t)dt & \text{if genotype } i \text{ exists at } t \\ 0 & \text{otherwise} \end{cases}$$

where $c_i(t)$ is the concentration of the i^{th} genotype at t. A genotype's [cumulative evolutionary activity counter] reflects its adaptedness (relative to the other genotypes in the population) throughout its history in the system." [Bedau & Brown 97][4]

To summarise the evolutionary activity of all the genotypes throughout the history of evolution in the system, we proceed as follows:

"The values of the activity counters of each [genotype] in the system over all time can be collected in the *component activity distribution*, $C(t, a)$, as follows:

$$C(t, a) = \sum_i \delta(a - a_i(t)),$$

where $\delta(a - a_i(t))$ is the Dirac delta function, equal to one if $a = a_i(t)$ and zero otherwise. Thus, $C(t, a)$ indicates the number of [genotypes] with activity a at time t." [Bedau *et al.* 98]

The visualisation technique is simply to graph these component activity distribution functions. This can be simplified by just plotting a point in (t, a) space whenever $C(t, a) > 0$.

These activity distribution functions (also referred to as "activity wave diagrams") provide a concise visual record of the appearance, competition and death of genotypes throughout an evolutionary run. Bedau and Brown also discuss how they can be interpreted in more detail, to reveal features such as periods of random drift among

[4] The rewording at the indicated places in this extract is to make it consistent with later changes in the terminology that Bedau *et al.* use to describe these techniques.

selectively neutral variants [Bedau & Brown 97]. Example activity wave diagrams can be seen in Figures 5.13–5.15.

The activity wave diagrams indicate which genotypes played important roles during the evolutionary run. The final individual-based analysis technique we used was to actually look at the code of significant genotypes, and to make comparisons between them. In particular, by looking at the genotypes which were abundant at the end of a run and comparing them to the genotype of the ancestor, we can get some idea of which parts of the ancestor were vital for its reproductive success (and therefore were more or less unchanged at the end of the run), and which parts were more redundant or less efficient (and how evolution managed to improve them).

5.1.2 Visualisation of Spatial Distributions

A number of 'movies' were recorded during the runs to show how the spatial distribution of the programs changed over the run. Each movie recorded a different piece of information about each program in the population (as well as the program's position on the grid), at various times throughout the run. This information included the program's length, age, size (number of cells, to indicate whether the program was serial or parallel), whether it had executed any foreign code during its lifetime (imported into its Received Message Store), whether it had moved during the last time slice, the amount of energy in its Energy Store, and, for each grid position, the amount of free environmental energy available at that time slice. By replaying these movies, we can see how the spatial distribution of these measures changed throughout the run.

5.1.3 Population-based Measures

In addition to the individual-based measures, a number of collective measures were also used. The population size throughout the run was recorded, as was the population diversity (the number of *different types* of program—different genotypes—in the population).

Four summary measures based upon the component activity distribution function, described in the previous section, were also used: Cumulative Activity, Mean Cumulative Activity, New Activity and Mean New Activity. These measures were developed by

Mark Bedau and colleagues ([Bedau & Packard 91], [Bedau *et al.* 97], [Bedau *et al.* 98]) to highlight between them the significant adaptive events occurring during an evolutionary run.

Cumulative Activity, $A_{cum}(t)$, is a measure of the continual adaptive success of the genotypes in the system at a given time. It is defined at time t as the sum of the activity counters, $a_i(t)$, of all genotypes (i) in the population at that time. The Mean Cumulative Activity ($\bar{A}_{cum}(t)$, defined as $\frac{A_{cum}(t)}{D(t)}$, where $D(t)$ is the diversity—the number of *different* genotypes—at time t) is the cumulative activity per genotype.

New Activity, $A_{new}(t)$, is a measure of the rate at which *new* adaptively significant genotypes are appearing in the system at time t. It is defined as follows:

> "Adaptive innovations correspond to new components flowing into the system and proving their adaptive value through their persistent activity. Let a_0 and a_1 define a strip through the component activity distribution function, $C(t,a)$, such that activity values a in the range $a_0 \leq a \leq a_1$ are among the lowest activity values that can be interpreted as evidence that a component has positive adaptive significance. Then, one reflection of the rate of the evolution of adaptive innovations is the new evolutionary activity, $A_{new}(t)$, which sums the evolutionary activity per component with values between a_0 and a_1:
>
> $$A_{new}(t) = \sum_{i, a_0 \leq a_i(t) \leq a_1} a_i(t)\text{"} \qquad \text{[Bedau } et \text{ } al. \text{ 98]}^5$$

Finally, Mean New Activity, $\bar{A}_{new}(t)$, is the new activity per genotype, defined as $\frac{A_{new}(t)}{D(t)}$. For more discussion of these measures, and of the reasons for defining them as they are, see [Bedau *et al.* 98] and [Bedau & Packard 91].

---

[5] Again, this formula has been changed from the original text to make the notation more consistent. In the original paper, $A_{new}(t)$ is defined as

$$A_{new}(t) = \frac{1}{D(t)} \sum_{i, a_0 \leq a_i(t) \leq a_1} a_i(t)$$

(i.e. as above, but divided by $D(t)$). This is what will be referred to here as *Mean* New Activity, $\bar{A}_{new}(t)$.

5.1.4 A Neutral Shadow

In order to gauge the extent to which the observed dynamics of Cosmos are due to the adaptive properties of individual genotypes rather than to any other features of the system, we also ran a "neutral shadow" of the system (as suggested by Bedau et al. [Bedau *et al.* 98], [Bedau & Brown 97], [Bedau *et al.* 97]).

The neutral shadow works as follows. During a standard run of the system, various details are recorded at each time slice (if the parameter `record_neutral_model_data` is set appropriately). These details include the number of organism births during that time slice, the number of these new births which resulted in new genotypes (through the action of flaws or mutations), the number of cell divisions (to form parallel processes), the number of cell deaths, and any movements made by individual cells.

The idea of the neutral shadow is to re-run the system using this recorded data, but in such a way that that no particular organism or genotype has any adaptive advantage over any other. Nominal "programs" exist at grid locations, reproduce, and die. At each time slice, a number (specified by the data recorded from the real run) of these programs are chosen *at random* to reproduce and to mutate. (Cell divisions in the real run to form parallel programs are treated as reproductions in the neutral shadow, so that all programs are unicellular.)[6] Similarly, a number of cells are chosen at random to be killed, and to move around the grid, determined by the numbers recorded from the standard run. In this way, we get the same number of events occurring in the neutral shadow as in the standard run, but these events happen to randomly chosen individuals. By applying the same measurement techniques to the neutral shadow as we do to the standard run, we can get an idea of the "raw" dynamics of the system, independent of the adaptive success of any particular genotype.

The results of the neutral shadow are discussed alongside the results of the standard run in the following sections.

[6] Therefore, at any given time, the number of *cells* in the neutral shadow is always the same as in the standard run, although the number of *organisms* may be lower if the standard run contains some multicellular (parallel) programs.

5.2 Detailed Analysis of a Standard Run

In this section, the results of a single run of Cosmos using the default parameter settings, together with its neutral shadow, will be analysed in some detail.

As mentioned in the previous chapter, there are many parameters associated with Cosmos, some of which control fairly minor or obscure aspects of the system. Because of this, it would be impractical to conduct a thorough investigation of the entire parameter space. Instead, many of the more obscure parameters were held constant throughout all of the experiments reported in this chapter. These are listed in Section B.1, along with their associated values.

The parameters which were varied between experiments (together with some other major parameters such as `grid_size`) are listed in Table 5.1. This table also shows the default values of these parameters, as used in the run reported in this section.

As shown in the table, the run was initialised by inoculating 100 ancestor programs into the system. The size of the world was 40 x 40 squares, and the ancestors were distributed evenly throughout this area.

The ancestor used for this run (and for nearly all of the other runs) is shown schematically in Figure 5.1 (a full listing is given in Section B.2). The basic operation of the ancestor is as follows. First, it looks for a template pattern at the start of its genome to work out the memory address of its first line of code. Next, it searches for a template pattern at the end of the genome, to work out the address of its last line of code. The ancestor then enters into a loop, copying instructions one at a time, from the beginning of the genome to the end, into the Nucleus Working Memory. When this copying is complete, the cells divides (creating a new organism), and the ancestor attempts to move into a neighbouring grid position.

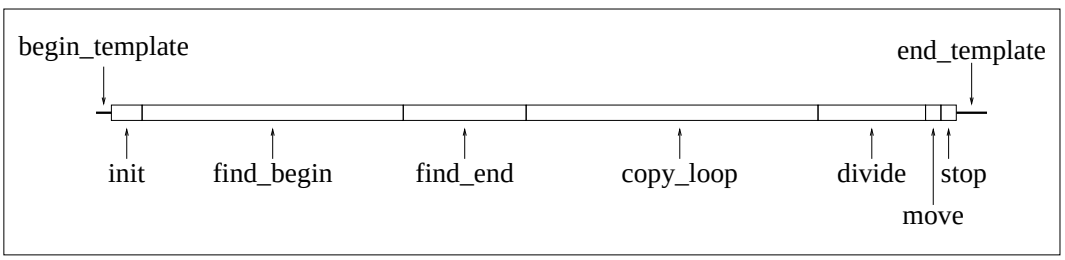


Figure 5.1: Schematic of the Standard Ancestor, 348AAAA.

Parameter	*Value*
inoculation	
ancestor	user_defined
number	100
startinfo	
rng_seed	834981793
termination	
limited_run	yes
number_of_timeslices	1000000
environment	
grid_size	40
horizontal_wrap	yes
vertical_wrap	yes
max_cells_per_process	2500
number_of_energy_tokens_per_grid_pos_per_sweep	30
max_energy_tokens_per_grid_pos	100
energy_collection_scheme	shared
energy_distribution_scheme	land
x_delta	0.0
cell	
et_value_constant	0.025
et_value_power	1.0
number_of_energy_tokens_per_collect	10
max_energy_tokens_per_cell	100
neighbouring_genomes_readable	no
mutation	
apply_mutations	yes
mutation_application_period	10
mutation_period	1000000
apply_flaws	yes
default_flaw_period	250000
io	
record_neutral_model_data	yes
visualisation_recording_on	yes
visualisation_record_energy_only	no

Table 5.1: Default Values for Major Parameters in Cosmos.

The duration of the run was one million time slices. As discussed in Section 5.1.4, data for the neutral shadow was recorded during this run, and then fed into the neutral model to give us an idea of the non-adaptive characteristics of the system.

5.2.1 Program Length

The change in program length over the run is shown in Figure 5.2. A number of interesting points are apparent from this graph. The most obvious one is that there is a trend for increasing program length over the course of the run. The length of the ancestor programs was 348 bits, but by the end of the run, virtually all of the programs in the population were of length 390 bits.[7] Recalling from the previous chapter that each instruction is represented by 6 bits, this increase in length corresponds to an extra 7 instructions being added to the ancestor.

Another point of interest in the graph is that, at any given time during the run, the population tends to be composed of organisms of only a single length.[8] The only exception to this occurs during a transition phase from programs of one length to programs of a greater length. During these phases, progams of two, or, occasionally, three different lengths coexist, but the shorter programs soon die out, to be completely replaced by the longer ones.

This result is in contrast to those of similar runs on Tierra, where shorter "parasite" programs evolved and coexisted with the longer programs (as described in Section 3.2.1). As discussed in Section 3.3.1, the non-appearance of parasites was expected, because Cosmos does not allow cells to execute the code of other cells.

The increase in program length, on the other hand, was unexpected. During each timeslice, CPU-time was allocated to each program (if it had enough energy) according to its length, so that, all else being equal, selection was size neutral (see Section 4.2.3). However, the picture is complicated by the fact that the program has to cycle once round the "copy loop" to copy each instruction. In the ancestor, the copy loop accounts for 19 of the 58 instructions, or approximately one third of the total number. The total

[7] There were in fact a handful (7) of programs of different lengths, but these are not recorded in Figure 5.2, because of the data pruning methods described in Section 5.1.1.

[8] Again, the data pruning techniques discussed in Section 5.1.1 mean that information is not displayed in these graphs for genotypes that are represented by only a very few individual programs. Such individuals typically account for about 1% of the total population.

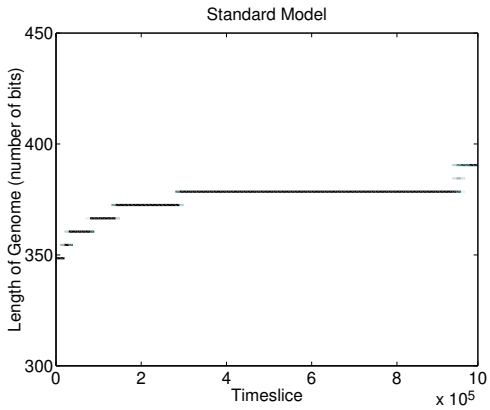


Figure 5.2: Standard Run: Length.

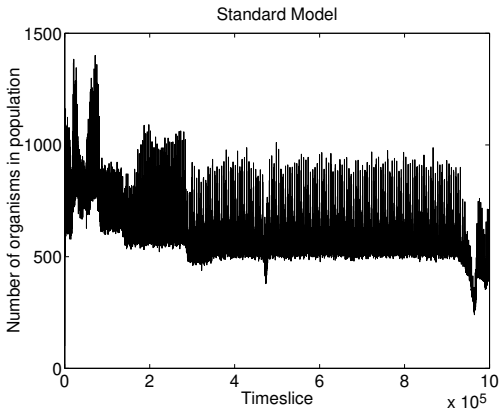


Figure 5.3: Standard Run: Population Size.

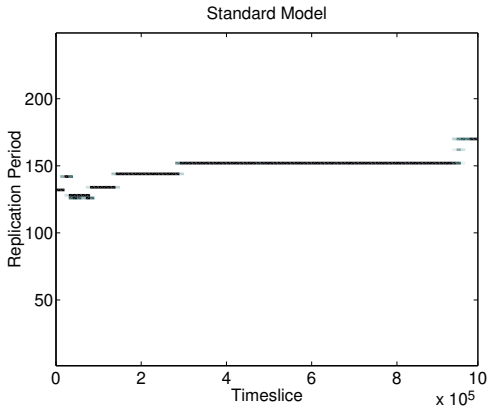


Figure 5.4: Standard Run: Replication Period.

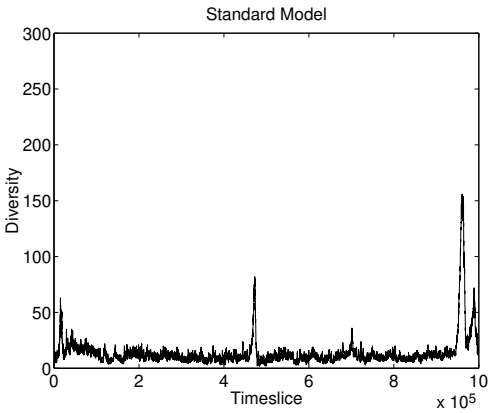


Figure 5.5: Standard Run: Diversity.

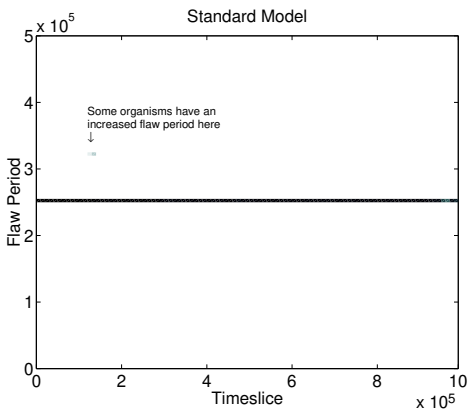


Figure 5.6: Standard Run: Flaw Period.

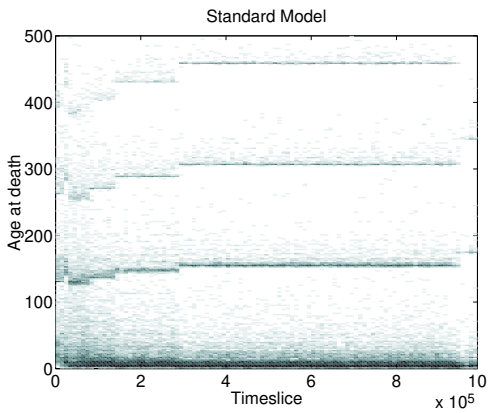


Figure 5.7: Standard Run: Age at Death.

number of instructions, I, that must be executed to replicate the program is therefore approximately:

$$I = N_{non_loop} + N_{loop} * N_{total} \tag{5.1}$$

where N_{non_loop} is the number of instructions not involved in the copy loop ($58 - 19 = 39$), N_{loop} is the number of instructions in the copy loop ($= 19$), and N_{total} is the total number of instructions ($= 58$). In this case, I is therefore $39 + 19 * 58 = 1141$ instructions. (This is not exactly right, because there are some conditional instructions, and the templates at the beginning and end of the program do not actually get executed, but it is near enough.) Now, during this run, the parameter `et_value_constant` was set to 0.025, so at each timeslice the ancestor (of length 58 instructions, or $58 * 6 = 348$ bits) was allowed to run $0.025 * 348 \simeq 8$ instructions (rounded down to a whole number). The total number of timeslices required for an ancestor program to replicate itself is therefore approximately 1141 / 8 $\simeq$ 142 timeslices.

Similar calculations can be made for programs of different lengths. If we assume that the size of the copy loop is always one third of the size of the whole program, the results we get are as shown in the left-hand side of Figure 5.8. In this case, longer programs still take longer to replicate, so we would still expect selection for shorter programs. (The graph looks somewhat erratic because of the effects of rounding down the number of instructions per time slice to an integer value, etc.) If however we assume that the size of the copy loop is constant no matter how long the program, the results are somewhat different (see the right-hand side of Figure 5.8). In this case, the assumption of length-neutral selection is more realistic.

As an increase in program length was observed, we might therefore expect the extra instructions to have been added outside of the copy loop. However, as the allocation of CPU-time is also dependent on a program having enough stored energy, it is possible that the observed increase in program length might be associated with the evolution of programs that collect and store more energy from the environment. In this case, it might even benefit programs to add more energy collection instructions inside the copy loop, as each one will then be executed during every iteration of the loop. The true cause of the increase in program length will be investigated in Section 5.2.7.

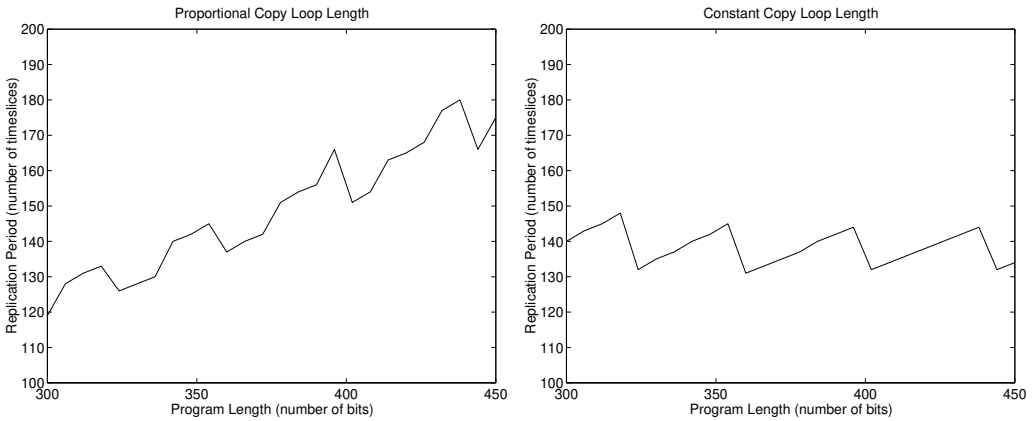


Figure 5.8: Theoretical Relationship between Program Length and Replication Period. *Left*: Size of copy loop increases proportionally with size of program. *Right*: Size of copy loop is constant.

5.2.2 Replication Period

The replication period (defined for programs that replicated faithfully at least twice during their lifetime as the number of time slices between the first and second faithful replication) is a more direct measure of a program's fecundity than is program length. From Figure 5.8 we can see that, if the length of the copy loop is increasing in proportion to the length of the program as a whole, we would expect trends in replication period during a run to roughly track trends in program length (because, generally, longer programs take a longer time to replicate in this case). However, if the length of the copy loop remains fairly constant as the program length as a whole increases, we would not expect such an association.

The observed change in replication period over the run is shown in Figure 5.4. As can be seen, the general pattern is very similar to that of change in program length (Figure 5.2), which suggests that the length of the copy loop is increasing as the programs as a whole get longer. This, together with the observation that the replication period increases during the run, suggests that there is some other factor playing an important role in determining the fitness of the programs. As suggested earlier, this factor could be a program's ability to collect energy. We will look into this in Section 5.2.7.

5.2.3 Age at Death

The graph of the age at death of programs in the population should show us whether they are evolving along the longevity axis. Figure 5.7 shows the graph for this run. There are a number of interesting points to note about this graph. First, there does appear to be some evolution towards increased longevity. On top of this, there is considerable structure in the distribution of ages at which organisms die. This is interpreted as indicating that the cycle of births and deaths in the population is highly synchronised throughout the run. The figure shows that the majority of programs live for some multiple of roughly 130 time slices at the beginning of the run, with fewer programs surviving for each successive multiple. At the end of the run, we still see the periodic structure, but the period has increased to about 160 time slices. These period lengths correspond very well with the replication period of the programs (see Figure 5.4).

A likely explanation is that the observed periodic structure arises as a consequence of the population experiencing a ceiling effect. Each time the population size reaches the ceiling, a number of programs die, creating space for the remaining programs to reproduce. Once this reproduction stage occurs, the population size is soon at the ceiling again, so the cycle repeats. The extinctions triggered by the population size hitting the ceiling are therefore periodic, resulting in the observed distribution of ages, with most organisms surviving for an integral multiple of the period of this cycle. The maximum number of programs (more accurately, cells) allowed to coexist in this run was 2500 (specified by the parameter `max_cells_per_process`). The actual population size never approached this limit (see Figure 5.3, discussed in the next section), so this was not the source of the ceiling effect. The other possible limiting factor is the total amount of energy distributed in the environment at each time slice. As discussed in the next section, it seems likely that this factor did indeed create a ceiling effect on population size, and therefore explains the observed distribution in Figure 5.7.

5.2.4 Population Size and Diversity

The graphs of population size and of diversity are shown in Figures 5.3 and 5.5 respectively. No multicellular (parallel) programs evolved during the run, so the number

of cells in the population was always equal to the number of organisms, as shown in Figure 5.3.

One of the points of interest in the population size graph, as mentioned in the previous section, is the fact that the limit on the number of cells in the population imposed by the parameter `max_cells_per_process` was never reached. This suggests that the population size was instead being limited by the amount of energy available in the environment. Figure 5.3 also shows a very high frequency oscillation in the population size, convolved with the lower frequency trends. This is consistent with the explanation offered in the previous section, that the birth and death cycle in the population was highly synchronised, producing a pattern of growth and decline determined by a ceiling population size related to the total amount of energy available in the environment.

To check whether the energy available in the environment really did impose a population ceiling lower than that specified by the parameter `max_cells_per_process`, a number of test runs were conducted. The only difference between the runs was the number used to seed the random number generator at the start of each one. Nine runs were conducted. In these runs, mutations and flaws were switched off, so that no evolution occurred. This allows us to see how the population size varies in a population of the ancestor programs, divorced from any effects due to competition from newly evolved programs. The graph of population size for one of these runs (which all lasted for 100,000 time slices), is shown in Figure 5.9. The figure clearly shows that the maximum population size achievable in practice is indeed well below the limit of 2500 set by `max_cells_per_process`; in fact, it is less than half of this value. Even with mutations and flaws switched off, slight differences arose between the different runs because, at each time slice, the programs in the population are processed in random order. However, the corresponding graphs for the other runs were all very similar; the maximum population size reached at any point in any of the runs was 1280 individuals.

Looking at the larger trends in the population size, we can see that they seem to be negatively correlated to the trends in replication period (Figure 5.4). This observation is consistent with the earlier suggestion that the programs are getting longer because they are collecting more energy. The more energy each program stores on average, the less is available in the environment for use by other programs, so the lower the maximum population size achievable. We will look into the question of whether the

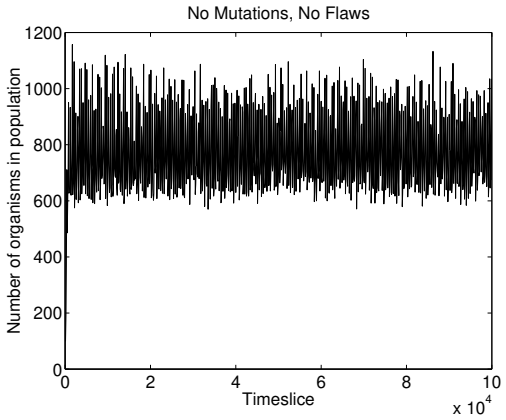


Figure 5.9: Population Size Limit Imposed on Ancestor Programs by the Default Rate of Introduction of Energy into Environment.

programs were evolving to collect more energy in Sections 5.2.7 and 5.2.8.

Looking at the diversity of the population throughout the run (the number of different types of program at any given point), we see that it is generally very low (Figure 5.5). Examination of the activity wave diagram (Figure 5.13, discussed in Section 5.2.6) reveals that there there is usually just one dominant genotype in the population at any given time (except at the very beginning of the run), and the concentration of other varieties is low.

There are interesting spikes in the diversity graph, one at roughly time 473,000, and another at 960,000. These spikes immediately precede dips in the population size (Figure 5.3). Inspection of the raw data files produced during the run reveals that these spikes correspond to the appearance of a number of longer genotypes, of lengths mostly in the region of 500–600 bits. None of these was ever represented by more than a single program in the population, suggesting that these were not viable self-replicators. As the numbers of these programs never exceeded the threshold defined by the system parameter `species_count_threshold_for_recording`, they do not show up in Figure 5.2 (as discussed in Section 5.1.1). As no further data was recorded for these programs, not much more can be said about them. However, it is reasonable to guess that they stored large amounts of energy, leading to the observed drop in population size.

To get an idea of the extent to which the diversity of the population is due specifically to the adaptive success of the individual genotypes rather than to other non-adaptive

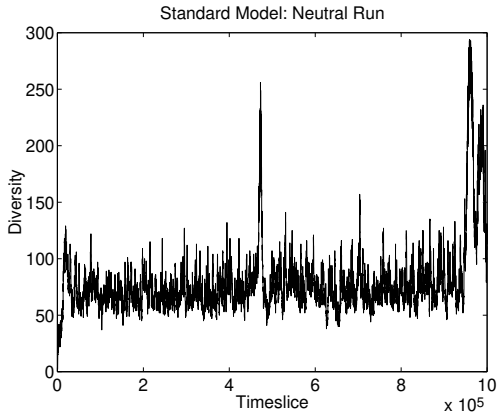


Figure 5.10: Standard Run, Neutral Shadow: Diversity.

dynamics in the system, we can look at the diversity graph of the neutral shadow. (Recall that the neutral shadow replays the exact sequence of births and deaths recorded in the real run, but acts upon random individuals. The population size in the neutral shadow is therefore always identical to the real run.) The neutral diversity graph is shown in Figure 5.10. This graph shows that the diversity in the neutral shadow was considerably higher than in the standard run, which suggests that the low diversity observed in the standard run was indeed due to the adaptive success of one or a small number of genotypes at outcompeting other varieties.

5.2.5 Flaw Period and Proportion of Unfaithful Replications

The two measures used to track evolution of the programs along the A-fidelity axis were: the flaw period of individual programs, and the proportion of the total number of offspring produced by an individual that were unfaithful (less than 100% accurate). The corresponding graphs are shown in Figures 5.6 and 5.11 respectively. In the graph of unfaithful replications (Figure 5.11), the vast majority of the points lie at zero, indicating that nearly all programs which reproduced did so faithfully. In order to see whether there was any trend for programs which did *not* always reproduce faithfully, the data from Figure 5.11 is reproduced in Figure 5.12 with the exception that points lying at zero infidelity have been omitted. The bar on the right of these two figures shows the scale (i.e. the mapping between the darkness of the plot at any point in the graph and the number of programs which had that particular infidelity at that particular time). Note that the scale in Figure 5.12 is two orders of magnitude smaller

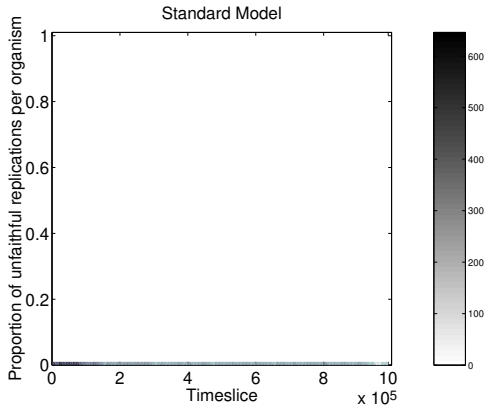


Figure 5.11: Standard Run: Unfaithful Replications.

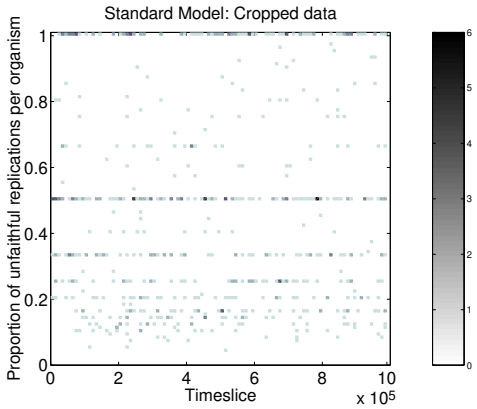


Figure 5.12: Standard Run: Unfaithful Replications (omitting points with zero infidelity).

than in Figure 5.11.

It is clear from these graphs that there was little change in either of these measures throughout the run. In other words, the programs are not evolving along the A-fidelity axis. From the analysis in the previous sections, it would appear that the parameter choices for this run create a selection pressure that predominantly favours evolution along the longevity and fecundity axes.

5.2.6 Activity Measures

The activity wave plots for the standard run and its neutral shadow are shown in Figures 5.13 and 5.14 respectively. These two figures are plotted to the same scale. A magnified version of the neutral shadow plot in shown in Figure 5.15.

The first point to note is that in the neutral shadow, compared to the standard run, there are *no* significant waves at all. In other words, no individual genotypes achieved significant continued adaptive success in this run. Even if we look at the magnified plot of the neutral shadow (Figure 5.15), we see that the pattern of activity is very different to the standard run (Figure 5.13). In the neutral shadow, there are many more activity waves than in the standard run, but they generally survive for a much shorter duration. These observations were expected (because operations are executed on randomly-chosen individuals in the neutral shadow), and they suggest that the significant activity observed in the standard run (Figure 5.13) *is* due to the adaptive

success of the individual genotypes concerned.

Looking at the activity waves of the standard run, we see that the ancestor genotype (348AAAA) is fairly quickly replaced by other genotypes. The first variant to enjoy notable continued adaptive success is genotype 366AADW, which first appears at time 76,100. At time 130,100 a new genotype, 372AAFT, appears, and gradually outcompetes 366AADW. By time 146,400 only a single program of genotype 366AADW remains, although this program manages, somewhat impressively, to survive for a further 91,700 time slices, living right up to time 238,100. At time 274,700, genotype 378AALB appears on the scene. This genotype quickly displaces 372AAFT, and remains the dominant genotype in the population right up until time 955,100, under 5000 time slices short of the end of the run. During its long reign, 378AALB is challenged a small number of times, most notably by the variants 378ANGE and 378ASNT. These variants are both of length equal to the dominant genotype, and both eventually lose their challenge and are driven to extinction. Having seen off these challenges, genotype 378AALB remained the dominant type of program for a short while longer. However, at time 925,700 a new genotype, 390AAGX, is born. Within 25,000 time slices, it drives 378AALB to extinction, and at the end of the run 390AAGX is the final dominant genotype.

The graphs of Cumulative Activity, Mean Cumulative Activity, New Activity and Mean New Activity, for the standard run and its neutral shadow, are shown in Figures 5.16–5.23.

Looking at the graph of Cumulative Activity in the standard run (Figure 5.16), we can see that it reflects the shape of the activity wave plot (Figure 5.13). This is because the value plotted in Figure 5.16 is the sum of the activities of all the individual genotypes at a given point in the run (i.e. the sum of all the waves present at any given moment in Figure 5.13). As a single wave tends to dominate the activity wave plot at most points during the run, the activity value of that genotype largely determines the shape of Figure 5.16. The diversity of genotypes in the standard run is usually very low (Figure 5.5), so the graph of Mean Cumulative Activity (defined at any given time as Cumulative Activity divided by Diversity), also has a similar shape (Figure 5.18).

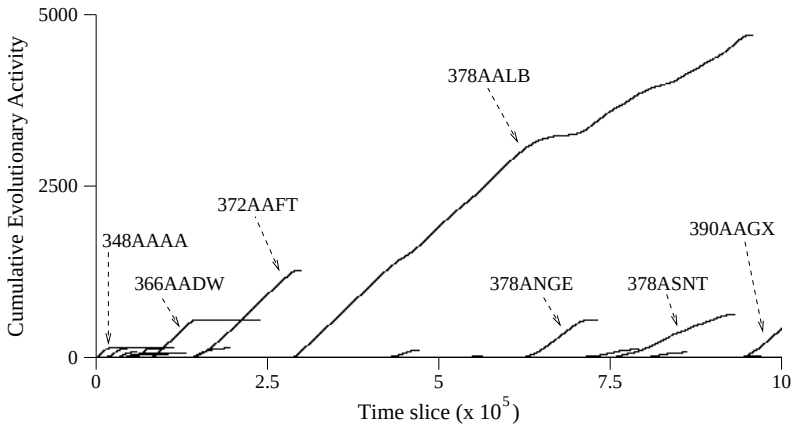


Figure 5.13: Standard Run: Activity Waves Plot.

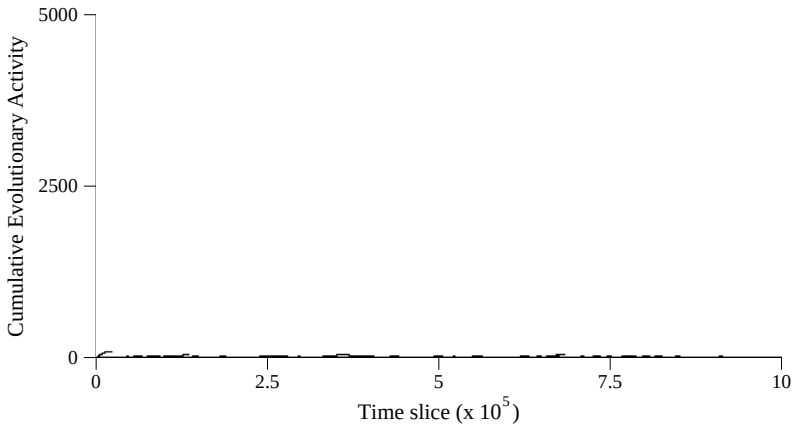


Figure 5.14: Standard Run, Neutral Shadow: Activity Waves Plot.

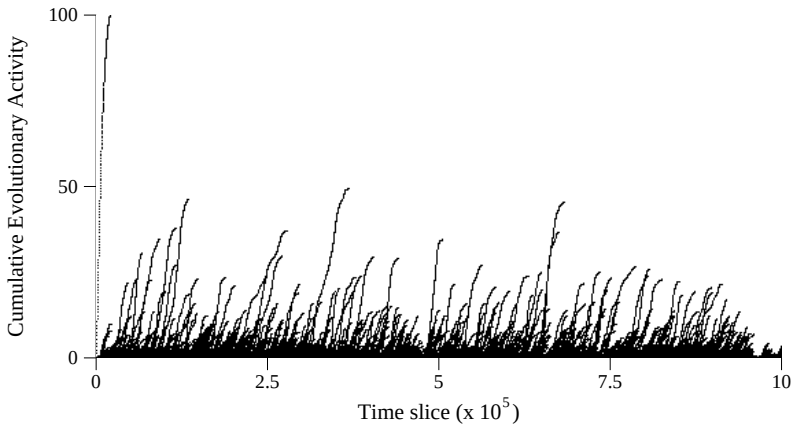


Figure 5.15: Standard Run, Neutral Shadow: Activity Waves Plot (magnified).

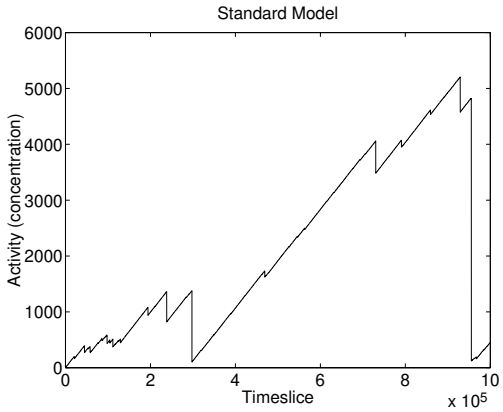


Figure 5.16: Standard Run: Cumulative Activity.

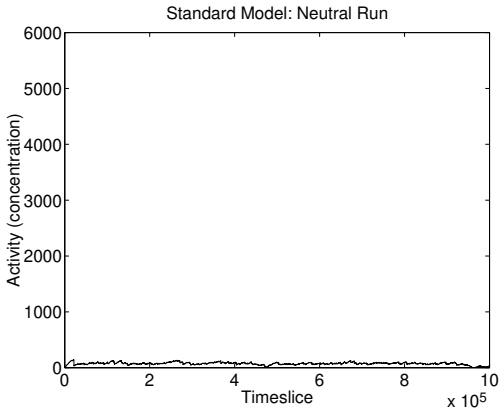


Figure 5.17: Standard Run, Neutral Shadow: Cumulative Activity.

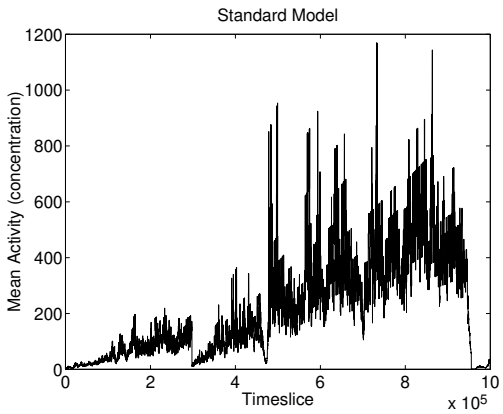


Figure 5.18: Standard Run: Mean Cumulative Activity.

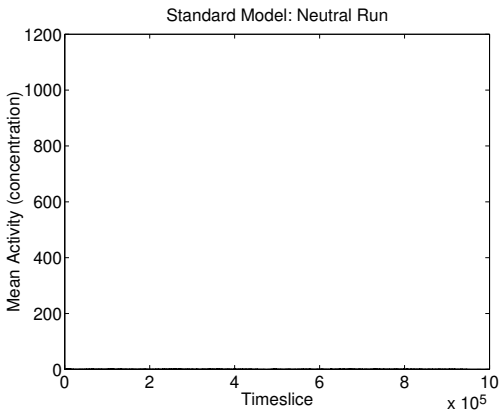


Figure 5.19: Standard Run, Neutral Shadow: Mean Cumulative Activity.

The graphs of Cumulative Activity and Mean Cumulative Activity for the neutral shadow (Figures 5.17 and 5.19) show again (as we saw in the activity wave plot of Figure 5.14) that there was no significant cumulative evolutionary activity in the neutral shadow compared to the standard run.

The graphs for *New* Activity, reflecting the introduction of new genotypes with adaptive significance, are shown in Figures 5.20–5.23. The parameter a_0, which determines the lowest level of activity that is taken to indicate adaptive significance, was set at the lowest level such that the neutral shadow produced zero new activity throughout the run. The value chosen for a_0 was 150, and for the upper activity limit, a_1, the value chosen (somewhat more arbitrarily) was 300. When applying the same thresholds to the

standard run (Figures 5.20 and 5.22), we can see that this measure clearly indicates the points during the run at which significant new genotypes were introduced (c.f. Figure 5.13).

In [Bedau *et al.* 98], a scheme is presented for classifying evolutionary systems into (at least) three distinct classes, depending on whether adaptive evolutionary activity is absent (class 1), bounded (class 2), or unbounded (class 3). The classification of a system is determined by its diversity D, mean cumulative activity $\bar{A}_{cum}$, and mean new activity $\bar{A}_{new}$.[9] If the system has bounded D, zero $\bar{A}_{cum}$ and zero $\bar{A}_{new}$, it belongs to class 1 (no adaptive evolutionary activity). If it has bounded[10] D, bounded $\bar{A}_{cum}$ and positive[11] $\bar{A}_{new}$, it belongs to class 2 (bounded adaptive evolutionary activity). If it has unbounded D, bounded $\bar{A}_{cum}$ and positive $\bar{A}_{new}$, it belongs to class 3 (unbounded adaptive evolutionary activity). Bedau and colleagues claim that the biosphere (at least with consideration to the Phanerozoic fossil record), belongs to class 3, whereas all artificial evolutionary systems they have studied belong to classes 1 and 2. From this perspective, a major challenge for the field of artificial life is to produce an artificial evolutionary system which exhibits class 3 dynamics.

This classification scheme is not perfect, as we can never be sure whether the results of a limited run of a system (which might have led us to suppose, for example, that D is bounded) will hold true for a much longer run (e.g. D might actually turn out to be unbounded). However, it is at least a step in the right direction, towards a rigorous classification of evolutionary dynamics. According to the scheme, and based upon the results reported in this section, Cosmos would be classified as class 2—the same as various other artificial evolutionary systems. Despite its weaknesses, the scheme does at least indicate that the type of evolution evidenced by the fossil record may be

[9] As explained earlier, Bedau calls this measure simply the new activity, A_{new}, rather than mean new activity. The name has been changed here to make it consistent with the naming of other measures.

[10] The definition of (un)boundedness given in [Bedau *et al.* 98] is: The function $f(t)$ is unbounded *iff*

$$\lim_{t \to \infty} \left(\frac{sup(f(t))}{t} \right) > 0$$

where $sup(.)$ is the supremum function.

[11] Similarly, the definition of a positive function is given in [Bedau *et al.* 98] as follows: The function $f(t)$ is positive *iff*

$$\lim_{t \to \infty} \left(\frac{\int_0^t f(t)dt}{t} \right) > 0$$

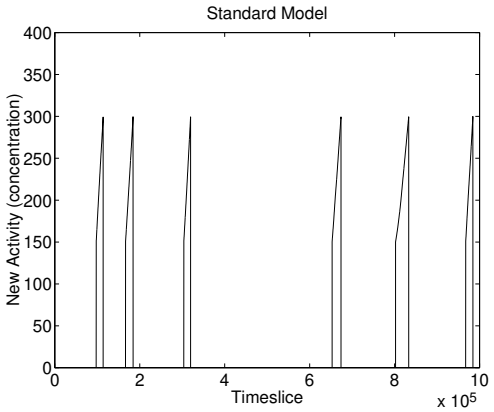


Figure 5.20: Standard Run: New Activity.

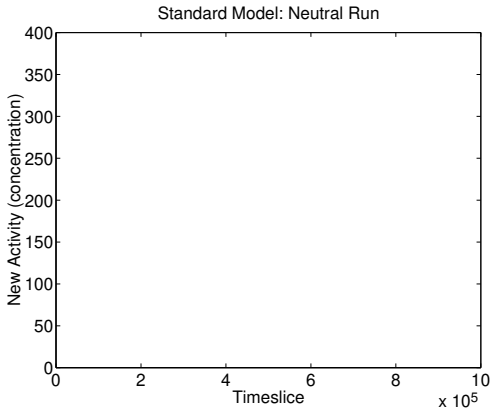


Figure 5.21: Standard Run, Neutral Shadow: New Activity.

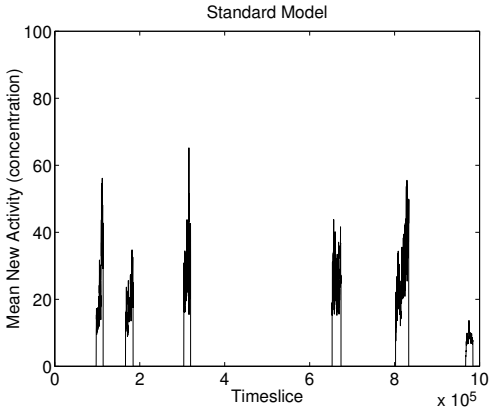


Figure 5.22: Standard Run: Mean New Activity.

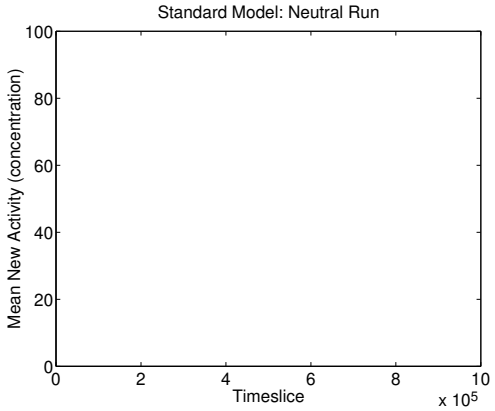


Figure 5.23: Standard Run, Neutral Shadow: Mean New Activity.

qualitatively different to that observed in our artificial systems.

5.2.7 Analysis of Significant Genotypes

By analysing the output data from the run, it is possible to reconstruct the phylogenetic tree (the evolutionary history) of the significant genotypes during the run. The reconstruction for the standard run is shown in Figure 5.24.

An interesting point to note about the phylogeny is that some of the ancestors of the genotypes that were significant throughout the course of the run did not themselves enjoy much success. An extreme example is genotype 360ABFL, from which all of the subsequent adaptively significant genotypes are descended. Despite its significance as

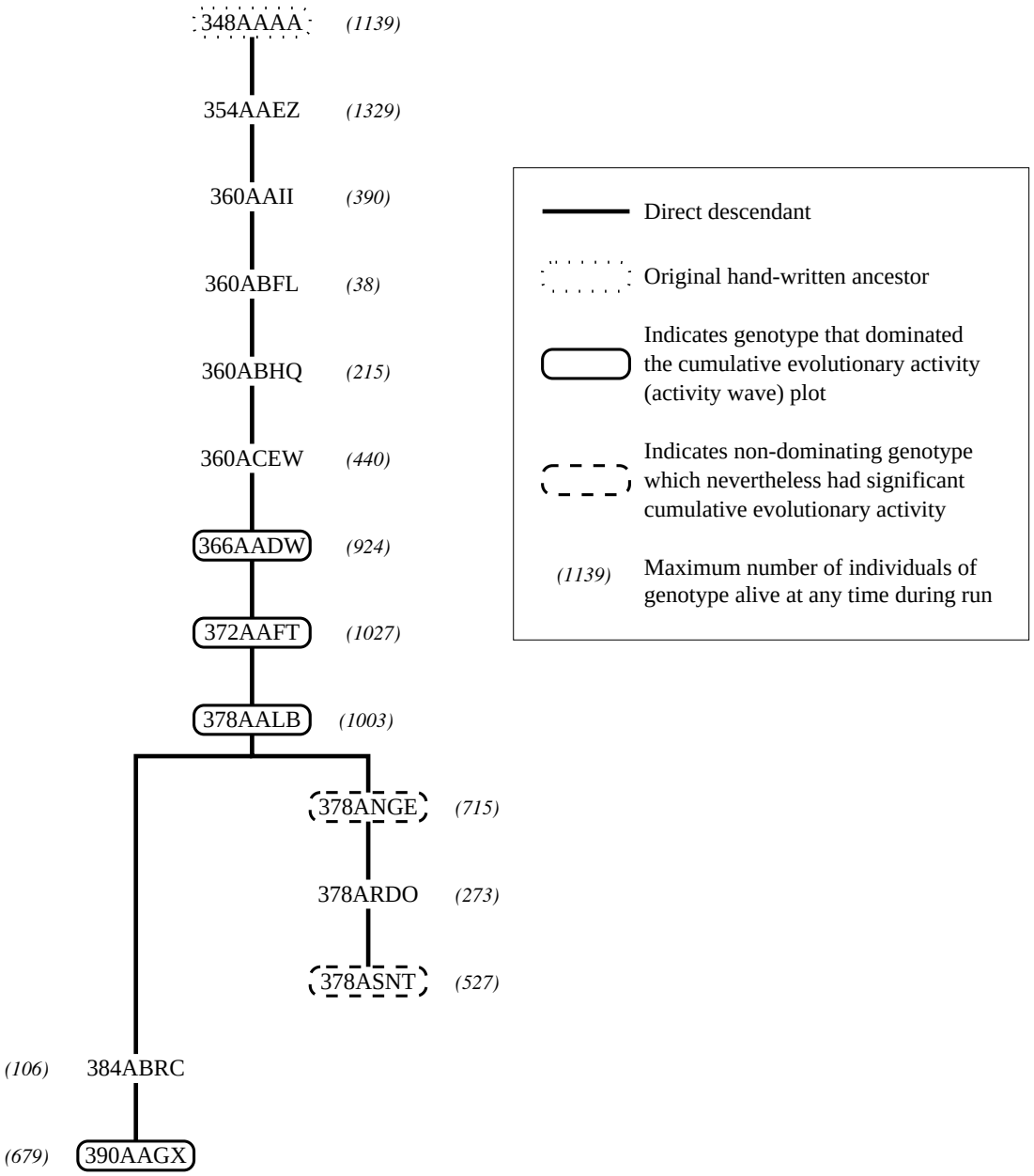


Figure 5.24: Standard Run: Phylogeny of Significant Genotypes.

genotype	ancestor	begin		find_begin		find_end		copy_loop		divide		end	
		3		17		8		19		7		4	
		+	−	+	−	+	−	+	−	+	−	+	−
366AADW	348AAAA	0	0	0	0	0	0	3(3)	1	0	0	3(0)	2
372AAFT	366AADW	0	0	0	0	0	0	1(1)	0	0	0	0	0
378AALB	372AAFT	0	0	0	0	0	0	1(1)	0	0	0	0	0
378ANGE	378AALB	0	0	0	0	0	0	0	0	0	0	1(0)	1
378ASNT	378ANGE	0	0	0	0	0	0	0	0	0	0	2(0)	2
390AAGX	378AALB	0	0	0	0	0	0	2(2)	0	0	0	0	0

Each row presents data on the comparison of a genotype (first column) with its ancestor (second column). Columns 3–8 refer to subsections of the genome (c.f. Figure 5.1; the subsections labelled begin_template and init have been grouped into a section labelled begin here, and similarly the subsections move, stop and end_template have been grouped together as end). In the row below the subsection name is the length of that subsection (in number of instructions) in the original ancestor 348AAAA. The column marked + shows how many new instructions were *added* to this subsection during the evolution from the indicated ancestor. The column marked − shows how many instructions were *removed*. Numbers in braces () in the + column show how many of the added instructions were the energy collection instruction (`et_collect`).

Table 5.2: Location of Differences Between Significant Genotypes and Their Ancestors.

an ancestor, the number of programs of genotype 360ABFL existing at any one time was *never* more than 38 (in a total population of over 800), and the genotype only existed for about 4000 time slices in total. Had circumstances been slightly different, the genotype might have died off before giving rise to its descendant 360ABHQ, or indeed 360ABFL might never have arisen at all. This indicates that the role of chance might play a large part in determining the outcome of a single evolutionary run. We will look into this issue in detail in Section 6.1.

A summary of the sources of innovation from ancestor to descendent for the most significant genotypes is presented in Table 5.2. The table reveals that the only sections of the genome which vary between the genotypes are the copy_loop and end sections.

Looking at the changes in the copy_loop section, we see that *all* of the additions were extra `et_collect` instructions. Comparing the final dominant genotype 390AAGX with the original hand-written ancestor 348AAAA, *seven* extra `et_collect` instructions have been added within the copy_loop section. If we compare the other subsection that experienced changes, the end section, we see that it only actually grew by one instruction from 348AAAA to 390AAGX (there were three additions and two removals). These findings therefore confirm the prediction made in Sections 5.2.2 and 5.2.4 that the programs are evolving to collect more energy, and that the extra energy collection instructions are being placed within the copy loop, so that they are executed at each iteration of the loop.

The changes made in the `end` section included the deletion of the `move` instruction in genotype 366AADW (the genome of the ancestor 348AAAA is shown in Section B.2). None of the subsequent dominant genotypes re-inserted this instruction, so the ability to move around the grid apparently did not convey an adaptive advantage in this run. The `stop` instruction was also deleted (both this and the `move` instruction were replaced by instructions which had no effect in the program), with subsequent genotypes taking advantage of the fact that when program execution reached the final instruction in the genome, the `stop` instruction was implicitly executed anyway. The extra instruction added at the end of the `end` section (in the transition from genotype 348AAAA to 366AADW) was an extra copy of the final instruction in the program (which actually formed part of the end template). This extra instruction appeared as a consequence of the `if_not_fl` instruction in the copy loop being replaced by an `et_collect`. The removal of the `if_not_fl` meant that the program did not check whether it had already reached the position marked by the end template before copying the next instruction to its offspring, so the final section of the end template was copied twice before the copy loop was exited. Apparently, the inclusion of an extra `et_collect` in the copy loop outweighed the penalty of having an extra, redundant instruction at the end of the program.

5.2.8 Spatial Distribution

By replaying the 'movies' recorded during the run (Section 5.1.2) we can see how the spatial distribution of the programs, and of various specific aspects of the programs and the environment, changed throughout the run.

As the run settled down, the spatial distribution of the programs became more regular, with many programs placed such that all of their immediately neighbouring grid positions were empty (compare the positions of programs in Figure 5.25, taken at time slice 5,000, with those in Figure 5.26, taken at 910,000).

The fact that the programs were evolving to collect more energy from the environment is demonstrated by the much higher average energy levels of programs in Figure 5.26 compared to those in Figure 5.25.

As the population tended to be dominated by a single genotype throughout the run,

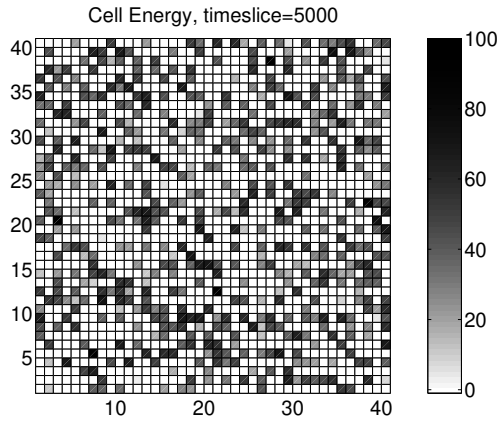


Figure 5.25: Standard Run: Cell Energy Distribution, Time Slice 5,000. (White squares are empty.)

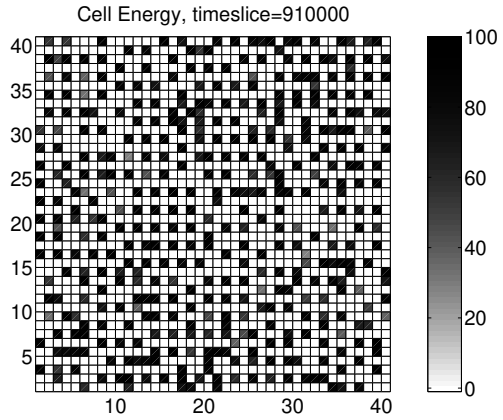


Figure 5.26: Standard Run: Cell Energy Distribution, Time Slice 910,000. (White squares are empty.)

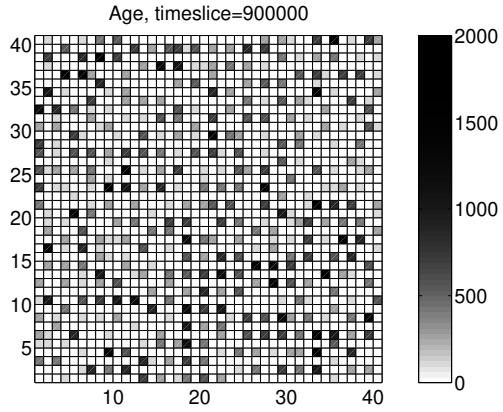


Figure 5.27: Standard Run: Cell Age Distribution, Time Slice 900,000. (White squares are empty.)

the movie of spatial distribution of program length is uninteresting. The distribution of ages of programs across the grid was fairly even, as shown in Figure 5.27.

No multicellular (parallel) programs emerged during the run. Also, because the system parameter `neighbouring_genomes_readable` was set to `no`, neither did any programs evolve which executed foreign code. After the `move` instruction included in the original genotype 348AAAA had been lost in genotype 366AADW, only an occasional variant rediscovered it subsequently, but this was never again taken up by most of the population.

5.2.9 Summary of Results

The summarise the results of this run, the programs are evolving towards increased A-longevity (Figure 5.7), but in doing so their A-fecundity decreases (i.e. their replication period increases: Figure 5.4). The programs do not evolve along the A-fidelity axis (Figures 5.6, 5.11, 5.12). The predominant evolutionary innovation is the accumulation of extra energy collection instructions within the programs' copy loops, which increases a program's chances of survival relative to its competitors (Section 5.2.7). No multicellular (parallel) programs evolved, nor did any parasites or any other sort of program that used code from neighbouring programs (Section 5.2.8). In other words, the evolution observed in this run was steady microevolution, with no spectacular macroevolutionary innovation. Analysis of the activity measures reveals that this run exhibits Class 2 dynamics, according to Bedau *et al.*'s classification scheme, which groups it together with various other artificial life platforms, and distinct from the evolutionary dynamics of the biosphere (Section 5.2.6). Reconstruction of the phylogenetic tree of the significant genotypes reveals that chance events may play a major role in determining the outcome of such a run (Section 5.2.7). We will look into this issue in detail in the next chapter.