AISB'05: Proceedings of the Joint Symposium on Socially Inspired Computing (METAS 2005), B. Edmonds, N. Gilbert, S. Gustafson, D. Hales and N. Krasnogor *Editors*, 12-15 April 2005, University of Hertfordshire, Hatfield, England, pp. 7-13.

Pfeiffer – A Distributed Open-ended Evolutionary System

W. B. Langdon*

*Computer Science, University of Essex
Colchester CO4 3SQ, UK
W1angdon@essex.ac.uk

Abstract

Pfeiffer contains a population of fractals which has been evolving continuously for more than three years. The animations are developed from embryos using a Lindenmayer grammar (L-System). These open generative representations potentially allow gene duplication and the evolution of higher order genetic operators and might be a step towards the emergence of social intelligence in swarms of artificial life (alife) agents. The fitness function is simply do the snowflake patterns appeal to the users: interactive evolution (IEC). To this end, images are placed in animated snow globes (computerised snowstorms) by world wide web (www) browsers (Netscape, Mozilla, Internet Explorer, Firefox, etc.) anywhere on the planet. More than 600 people have used `http://www.cs.ucl.ac.uk/staff/W.Langdon/pfeiffer.html`.

1 Introduction

For more than three years we have been running an experiment in distributed open-ended interactive evolution in which small local populations within each user's web browser communicate via Javascript with a central server holding a variable sized global population (see Figure 2). (Initial results were reported in Langdon (2004a).) Pfeiffer is intended to show the feasibility of evolving agents on many small computers running across the Internet under the user's actions as a fitness measure. The agents are intended to be attractive and therefore they are animated in a snowstorm. Their form is given by a D0L deterministic context free L-system Pruskinkiewicz and Lindenmayer (1990) (see Figure 1), whose initial seed is a Koch fractal snow flake.

L-systems have the advantage over traditional programming in that they are inherently parallel. This is analogous to growing plant tissue (for which they first used to model) where each cell grows and divides in parallel with its neighbours and like DNA strands where, in principle, all genes can be expressed simultaneously. Karl Sims was perhaps the first person to combine L-systems with interactive evolution, e.g. Sims (1991).

The next section describe the evolutionary L-system. Section 3 summarises its usage (more details are given in Langdon (2004a) and Langdon (2004b)) while section 4 considers what lessons can be drawn. The penultimate section (5) discusses where evolutionary agents might lead us. We conclude, in Section 6.

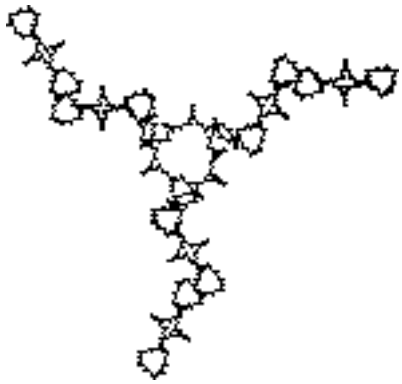

Figure 1: Example L-system fractal pattern. The picture is the phenotype developed from the 435^{th} genotype (seed) saved by users in the global population. The seed defines the L-system's initial grammar symbol as `F++F++F++F++F` and the replacement rule as `F`$\Rightarrow$`FF+FF--F`. It also specifies that start symbol be expanded four times.

2 How Pfeiffer Works

Pfeiffer (cf. Figures 2 and 3) evolves agents and displays them moving in two dimensions across the screen of a world wide web (www) browser. The visual phenotype of each agent is given by a Lindenmayer (L-system) grammar. As the agents are moved or tumble across the screen they are subject to random encounters and changes which may effect their grammar. Each time the grammar is changed the agent's new shape is drawn on the screen. The user can save pretty shapes and delete ugly ones.

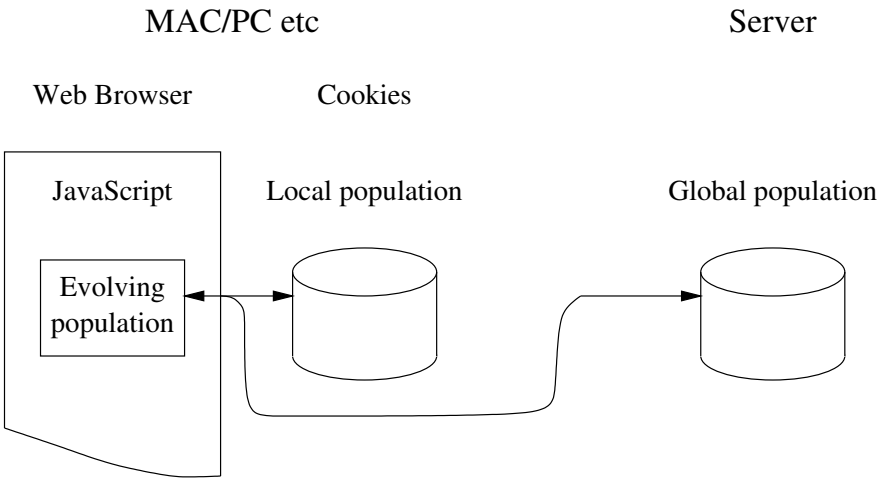

Figure 2: Overview of Pfeiffer. The user interacts via HTML and Javascript run in their web browser (left hand side). Initial seeds (chromosomes) are either retrieved from earlier runs via cookies or down loaded across the Internet via cgi programs from the global population. After evolution in the user's browser, the user may save new seeds both locally (as cookies) and in the global population.

2.1 Global and Local Populations

The active local population is stored in Javascript objects within the user's web browser. However these are lost when a new HTML page is selected. Therefore, "cookies" (if they are enabled) are used to provide off line storage of the local population.

Each time the Pfeiffer web page is down loaded, the initial value of each agent's chromosome is read from the corresponding cookie. However, if there is no cookie, the initial chromosome is down loaded from the global population across the network.

2.2 User Interaction

The primary goal of the user intervention is to use the user to provide selection pressure to drive the evolution of the agents. Passing the mouse over an agent causes its menu to be displayed. A text field allows the user to name the agent. While the pull down menu (see Figure 4) confirms the agent's identity and allows the user to: save the agent, make a copy of it (both automatically give it high fitness), delete it and close the menu. Naming an agent makes it easier for the user to track the agent he has evolved using "top ten" and "Hall of Fame" web pages, An agent "saved" by the user is stored in its cookie and appended to the global population. Once in the global population, the agent can be down loaded by other users and so distributed across the world. Cloning an agent causes an additional copy of the agent to be stored in the local population. This will often require the deletion of another, low fitness, agent. These user initiated actions exert selection pressure on the local and global populations.

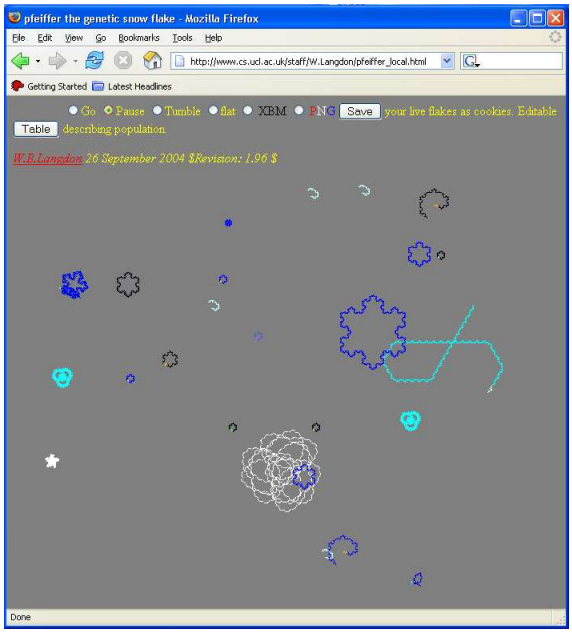

Figure 3: View of evolutionary arena as seen by user

In addition to deciding life and death, the user can influence which agents mate. Using the mouse, an agent can be picked up and moved into the path of another agent. As with saving and cloning, moving an agent implies the user prefers it and it is given high

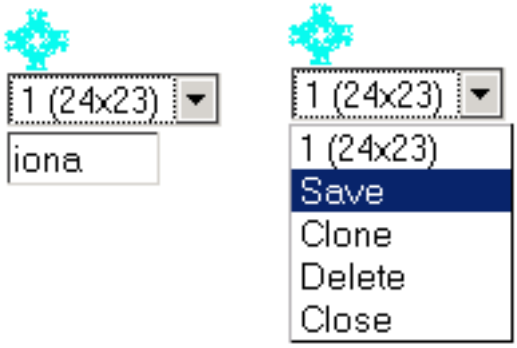

Figure 4: Example menu. Left hand, allows user to change agent's name. Right, pull down menu, allows user to save, copy or delete agent.

fitness, making it very likely to mate with the next mature agent it meets.

2.3 Generating the Phenotype

The system is able to display the results of arbitrary L-systems. In the original system (and even today in some browser) this is beyond Javascript. Therefore it was necessary to generate the graphics on a server and down load them into the user's browser (see Figure 5). In this mode of operation, each new seed is passed to the server. It is interpreted as a Lindenmayer grammar. This generates a series of drawing instructions, which are immediately obeyed. The resulting picture is compressed and converted to .GIF format and passed back to the user's browser for display. Because of the data compression, this takes only a few seconds. However the delay could cause problems due to the agent's genotype and phenotype becoming out of step Langdon (2004a). Therefore the new version of Pfeiffer processes L-systems and graphics generation in the user's browser. However both systems are active (for compatibility with less able browsers).

2.4 Genetic Representation

Each agent seed is a variable length linear text string. The default seed grows into the Koch snow flake  . The default seed is the 56 character string `v=60&str=F++F++F & it=2 & sc =5 & rules=('F','F-F++F-F')` (this can be replaced by the user).

The string is split by & characters into parameters. They are are processed left to right. Thus if any parameter is repeated, the second "gene" is "dominant".

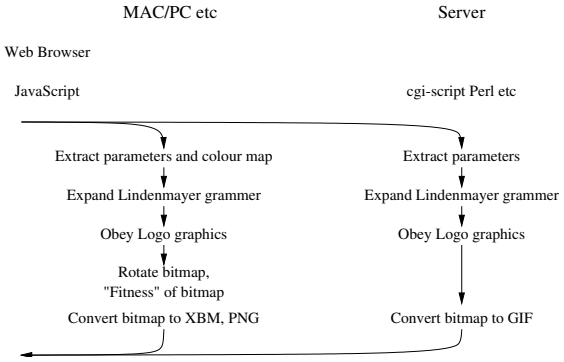

Figure 5: Mapping genotype to phenotype. In this development process the genotype (the L-system plus associated parameters) is converted to a graphic. The chromosome may either be passed to our server, interpreted and a .GIF file returned (right hand side) or interpreted locally (left). The local version avoids Internet delays, allows colour, and 3-D effects but is less portable.

Five parameters are recognised. They are `v` (angle), `str` (start string of grammar), `it` (depth of recursive expansion), `sc` (side, in units of 0.2 pixels) and `rules` (grammar replacement rules). Each substring formed by splitting the seed at the `&` is further split by `=`. If the first part of the substring exactly matches one of the parameter names then its value is set to the text between the first and second (if any) `=`. If a parameter is missing, its default is used. The defaults come from the Koch snow flake, they are `v`=60, `str`=`F++F++F`, `it`=2, `sc`=5 and `rules`=`('F', 'F-F++F-F')`. When `rules` is parsed characters such as `(` and `)` are removed. In our Koch example this means the single substitution rule is: `F`⇒`'F-F++F-F'`. The use of the defaults is effectively the same as if the default text where inserted at the start of every seed (albeit protected from genetic operators).

Once parameters have been decoded the L-system is interpreted. First the start string `srt` is expanded `it` times. At each step every character which matches the left hand symbol of a rule is replaced by the corresponding right hand side. Note any letter can potentially match a rule, not just those used by the turtle graphics, allowing indirect rules. The expansion yields a potentially very long string. To avoid infinite or very long recursions, time outs are applied.

The string is interpreted as a series of "turtle" drawing instructions. Except for 3-D instructions, predefined graphics and increasing the line width, all of the turtle instructions given in Pruskinkiewicz and Lindenmayer (1990) are supported. The graphic is

displayed to the user moving across the main screen, see Figure 3. The local version, also supports a "tumbling" effect in the animation. On the server .GIF format is used. Some browsers (e.g. Mozilla) support colour PNG format files, whilst others (e.g. Microsoft) only support monochrome XBM format.

2.5 Example

Suppose the initial seed (chromosome), sent to the server, is the 59 characters `v=60 & str=F++F++F3t5F+r c sc=5 & rules = ('F','F-F++F-F')`. After removing spaces we are left with 51 characters This splits (at `&`) into two parameters `v`=60 and `str`=`F++F++F3t5F+rcsc=5`. Defaults are given for missing parameter values (`it` and `sc`), while `rules` is fixed. So the grammar is `v`=60, `str`=`F++F++F3t5F+rcsc`, `it`=2, `sc`=5 and `rules`=`('F','F-F++F-F')`. Note how the original value of `sc` had been corrupted but is restored by the default. The start string is expanded by the server twice (`it`=2) to give the final image.

Iteration		Size	Expansion	Line segments
0		3×3	`F++F++F3t5F+rcsc`	4
1		6×7	`F-F++F-F++F-F++F-F++F-F+ +F-F3t5F-F++F-F+rcsc`	16
2		15×17	`F-F++F-F-F-F++F-F++F-...`	64

3 Global Usage of Pfeiffer

From December 2001 to July 2004, excluding the author, 44,605 L-systems have been evolved and interpreted (3,522 locally and the rest and down loaded) by 1077 user sessions from 384 different sites. Usage of the system has been highly non-uniform. The length of individual sessions is approximately log-normal. However activity also varies in time reflecting the system's development, particularly support for additional browsers (Mozilla, Netscape 6 and Microsoft, July 2003) and local interpretation and graphics generation (Jan 2004) (see Figures 7 and 8).

All the phenotypes created during a two month trial period are given in Langdon (2004b).

4 Discussion

As pointed out in (Hemberg et al. 2001, page 165) traditional interactive evolution often runs into the problem of human fatigue, i.e. people being unable to continue acting as the "fitness function" after a few

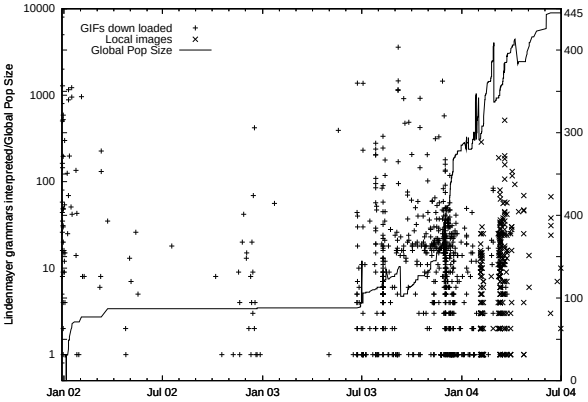

Figure 8: Usage of Pfeiffer. Crosses refer to creation of agents by individual users (note log scale), whilst the line shows the evolution of the global population (linear scale).

minutes. This normally severely limits both population size and number of generations. For example, in the approximately 250 papers described by Takagi (2001), typically populations contain only 9 or 16 individuals and no more than 10–20 generations are used. I.e. typically interactive evolutionary computation (IEC) runs have up to only 100 to 300 fitness evaluations. In contrast, the global population of Pfeiffer has been grown from about 100 to 514 today (January 2005) and 46,000 images have been presented to ≈ 600 people. Pfeiffer continues to attract users after more than three years of operation.

It appears Takagi (2001) most interactive evolutionary computation tries to make the user "optimise" and so tries to minimise varation between people. In contrast Pfeiffer tries to encourage variation, to produce many appealing patterns rather than a single optimal one. Again most IEC computer graphics (although by no means all) are static, while here they are animated.

The simple text string representation is certainly highly robust and flexible. Its compactness makes global distributed on line operation feasible.

The L-system (with supplied defaults and a robust interpreter) allows a huge degree of redundancy to evolve. The "fitness landscape" clearly contains a huge degree of "neutrality" Babajide et al. (1997); Shipman et al. (2000); Smith et al. (2002); Yu (2003) and evolution is using it.

L-systems readily allow evolution of many plane figures (but are not general purpose). Many new fractal like patterns have been readily evolved using them.

First parent

```
v =l-72&strF+'+F+4+F+F&&st F+2& 'c s 5tetulF+ =-F Fe ,- F&F(Fl+&=
```

Second parent

```
v =l=72&&strF '+F+4+F+F&&st F+2& 'c s 5tetulF+ =-F Fe,,- F&F(Fl++=
```

Offspring, replaces first parent

```
v =l-72&strF+'+F+4+F+F&&st F+2& 'c s | 5tetulF+ =-F Fe ,- F&F(Fl+&=
```

Figure 6: Example crossover. Length of first parent 67, first cut point at 37, remove 10 characters, insert 11 characters. 68 characters in offspring.

Figure 7: Usage of Pfeiffer up to April 2004. Red lines connect each user's country to the central server. Heaviest use has been from UK, USA and Canada, but users have also come from the far and middle east, India, Europe, Latin American and South Africa.

5 Future: Breeding "Intelligent" agents

Our agents are very limited. We feel they need to be able to evolve to react to their environment. They need to be able to evolve to predict their environment. Of course this makes requirements of both the agent and the environment. Also, perhaps crucially, each agent needs to be able to effect the environment, and predict what those effects will do for it (and for others). While L-systems have been mainly used (as we have done here) to create static structures, they can describe networks. Those networks could contain sensory, processing and active elements Hornby and Pollack (2002) and/or use cultural evolution–mental simulation, imitation and knowledge-based operators–such as used by the vehicles of Gabora (1995). Gruau (1994) describes another indirect approach to evolving artificial neural networks (ANNs). While Stanley and Miikkulainen (2003) surveys developmental evolution in computer science.

There is a strand of thought in which intelligence came from a co-evolutionary struggle between members of the same species Ridley (1993). If true, can intelligence arise in isolated agents? Or are interacting/communicating agents needed?

A problem with simulated worlds has been hosting sufficient complexity so as to be challenging but still

allowing agents to be able make predictions about what will happen next and what will happen to me or to others if I do this. The Internet hosts tides of data. This data is not random. It ought to be possible to harness it to give a suitable virtual environment.

We have fallen for the usual trap of constructing a two dimensional world (on the computer screen). However is there any hope of evolving artificial life (and thereby artificial intelligence) in two dimensions? Obviously three dimensions are sufficient but computer simulations offer many dimensions (N$\gg$3).

6 Conclusions

Lindenmayer grammars can be used as the basis for a distributed interactive evolutionary system and produce interesting fractal like patterns. Many new patterns have been evolved Langdon (2004b), some exploiting the L-system to produce some regularities and re-use of motifs. It is feasible to represent individual agent's genetic material (seed/chromosome) with a variable length text string without defined fixed semantic fields and using crossover at the character level. The representation allows a huge degree of redundancy to evolve. The "fitness landscape" clearly contains a huge degree of "neutrality" and evolution is using it. This loose representation allows the location etc. (as well as the meaning) of the L-system to evolve. Gene duplication, translocation and other genetic operations could be supported by such a general representation.

In terms of harvesting spare CPU cycles, the project confirms it can be done using Javascript and user's web browser. The project does hint at some successes. World wide distributed evolution is clearly feasible. Perhaps more importantly one can recruit users (which are much more valuable than their CPUs) to assist in guided evolution. Finally animated tools are an attractive way to spread interest in artificial evolution, intelligence and life.

Acknowledgements

I would like to thank anonymous reviewers and Liane Gabora. I would like to thank Birger Nielsen for use of his perl Lindenmayer interpreter `http://www.246.dk/lsystems.html`, David Corney, Emma Byrne, Song Hu, Joao Oliveira, Mel Slater, Jorge Tavares, Ernesto Costa, Francisco Pereira, Sara Silva, Maarten Keijzer Jano van Hemert and several anonymous referees. Internet sites containing javascript source code and documentation have been very useful. Special thanks to `http://www.altan.hr/snow`, `http://devedge.netscape.com`, `http://www.mozilla.org`, `http://javascriptsource.com`, Roger E. Critchlow Jr. `pnglets`, Jukka Korpela `forms/extraspace.html`, Dan Steinman `mouseevents.html`, Peter-Paul Koch `js/events_compinfo.html`, Phil Reeve, and David Blackledge for `AnalogClock.html`.

Finally I would like to thank the users of Pfeiffer.

References

1 Aderonke Babajide, Ivo L. Hofacker, Manfred J. Sippl, and Peter F. Stadler. Neutral networks in protein space, a computational study based on knowledge-based potentials of mean force. *Folding & Design*, 2:261–269, 20 August 1997. ISSN 1359-0278.

2 Liane Gabora. Meme and variations: A computational model of cultural evolution. In Lynn Nadel and Daniel L. Stein, editors, *1993 Lectures in Complex Systems*, pages 471–485. Addison Wesley, 1995. URL `http://cogprints.org/531/00/mav.htm`.

3 Frederic Gruau. *Neural Network Synthesis using Cellular Encoding and the Genetic Algorithm.* PhD thesis, Laboratoire de l'Informatique du Parallilisme, Ecole Normale Supirieure de Lyon, France, 1994. URL `ftp://ftp.ens-lyon.fr/pub/LIP/Rapports/PhD/PhD1994/PhD1994-01-E.ps.Z`.

4 Martin Hemberg, Una-May O'Reilly, and Peter Nordin. GENR8 - A design tool for surface generation. In Erik D. Goodman, editor, *2001 Genetic and Evolutionary Computation Conference Late Breaking Papers*, pages 160–167, San Francisco, California, USA, 9-11 July 2001. URL `http://www.ai.mit.edu/projects/emergentDesign/genr8/lateGecco.pdf`.

5 Gregory S. Hornby and Jordan B. Pollack. Creating high-level components with a generative representation for body-brain evolution. *Artificial Life*, 8(3):223–246, 2002. ISSN 1064-5462. URL `http://www.demo.cs.brandeis.edu/papers/hornby_alife02.pdf`.

6 W. B. Langdon. Global distributed evolution of L-systems fractals. In Maarten Keijzer, Una-May O'Reilly, Simon M. Lucas, Ernesto Costa, and Terence Soule, editors, *Genetic Programming*,

Proceedings of EuroGP'2004, volume 3003 of *LNCS*, pages 349–358, Coimbra, Portugal, 5-7 April 2004a. Springer-Verlag. URL http://www.springerlink.com/openurl.asp?genre=article&issn=0302-9743&volume=3003&spage=349.

7 W. B. Langdon. Lindenmayer systems fractals evolved by Pfeiffer 10 September – 9 November 2003. Research Note RN/04/13, University College, London, 19 July 2004b. URL ftp://cs.ucl.ac.uk/genetic/papers/rn0413.pdf.

8 Przemyslaw Pruskinkiewicz and Aristid Lindenmayer. *The Algorithmic Beauty of Plants*. Springer-Verlag, 1990.

9 Matt Ridley. *The Red Queen, Sex and the Evolution of Human Nature*. Penquin, 1993. ISBN 0140167722.

10 Rob Shipman, Mark Shackleton, and Inman Harvey. The use of neutral genotype-phenotype mappings for improved evolutionary search. *BT Technology Journal*, 18(4):103–111, October 2000. ISSN 1358-3948. URL ftp://ftp.cogs.susx.ac.uk/pub/users/inmanh/bt.pdf.

11 Karl Sims. Artificial evolution for computer graphics. *ACM Computer Graphics*, 25(4):319–328, July 1991. URL http://web.genarts.com/karl/papers/siggraph91.html. SIGGRAPH '91 Proceedings.

12 Tom Smith, Phil Husbands, Paul Layzell, and Michael O'Shea. Fitness landscapes and evolvability. *Evolutionary Computation*, 10(1):1–34, Spring 2002. URL http://www.cogs.susx.ac.uk/users/toms/Papers/Smith_EvolutionaryComputation2002.pdf.

13 Kenneth O. Stanley and Risto Miikkulainen. A taxonomy for artificial embryogeny. *Artificial Life*, 9(2):93–130, 2003. URL http://nn.cs.utexas.edu/downloads/papers/stanley.alife03.pdf.

14 Hideyuki Takagi. Interactive evolutionary computation: Fusion of the capabilities of EC optimization and human evaluation. *Proceedings of the IEEE*, 89(9):1275–1296, September 2001. ISSN 0018-9219. Invited Paper.

15 Tina Yu. Neutral evolution in evolutionary computation. In James Foster, editor, *GECCO–2003 Genetic and Evolutionary Computation Conference: Tutorial Program*, pages 627–643, Chicago, 13 July 2003.