ECOSYSTEM MODELS FOR REAL-TIME GENERATIVE MUSIC: A METHODOLOGY AND FRAMEWORK

Oliver Bown

Monash University, Clayton 3800, Australia

oliver.bown@infotech.monash.edu.au

ABSTRACT

In this paper I discuss a methodology for applying computational ecosystem models to sound installations and generative music systems: sonic ecosystems. I discuss the need for tools that facilitate the introduction of detailed analysis of complex multi-agent systems into a creative computing workflow, in order to create powerful ecosystemic artworks, and list a number of design requirements associated with an ecosystem-based methodology. I present a software framework and a specific sonic ecosystem model that demonstrates these ideas.

1. INTRODUCTION

Evolutionary theory has provided a rich body of ideas with which computer musicians have been able to develop new compositional strategies and algorithms. Until recently, the predominant paradigm in this domain was that of the interactive genetic algorithm, the notion that an artist could evolve pieces of software by using their aesthetic judgement as the 'selective pressure' in an artificial environment, like a pigeon breeder. More recently, however, a broader approach to the creative power of evolutionary dynamics is being embraced.

In mainstream evolutionary theory, much effort in recent years has gone towards shifting the balance of focus from the process of selection towards a more holistic view involving both *the selection on individuals by environments* and *the construction of environments by individuals*. This approach posits that novelty and complexity emerge in natural evolution through the vigourous interaction between *natural selection* and a process now widely known as *niche construction* [7]. Interests in generative and evolutionary art and music have followed suit [6], through a new paradigm which focuses on understanding and manipulating the interaction between these and other processes (such as social learning and cultural behaviour) in multi-agent systems. In this paper, I discuss tools that could help apply computational ecosystem models to sound installations and generative music systems (sonic ecosystems). I begin by discussing a number of design requirements which would facilitate the goals of this new paradigm, and then present an example of such work.

2. DESIGN REQUIREMENTS

Multi-agent models have common structures and requirements that have led to the development of a number of programming frameworks for multi-agent modelling [8]. Such frameworks provide a set of common libraries for the design of agents and environments, and also function to standardise multi-agent modelling, facilitating the interpretation of results and the repeatability and comparative utility of modelling experiments [1]. Typically, such frameworks aim to minimise the requirements for writing an agent class, the bare minimum being that agents implement a 'step' function, which defines most of the agent's behaviour. For example, the Mason framework[1], defines an interface, `Steppable`, with a single method, `step()`. Other frameworks, such as NetLogo[2], optionally provide specialised languages for designing agent behaviours.

The value of developing a general framework for ecosystem-based artworks is largely motivated by the experience of previous research in this field, which suggests the following list of design requirements for a framework for ecosystemic artworks. There is nothing in the list below that cannot be easily achieved by a competent programmer working with their preferred tools, but a framework bringing them together could contribute to a workflow that invited new creative opportunities.

2.1. Analysing the Behaviour of Systems

Multi-agent systems demonstrate the phenomenon that complex system-level behaviours can emerge from the interaction of multiple agents exhibiting only simple individual behaviour [4]. Since it is easy to design agents with a relative degree of complexity, it is not unusual for resulting multi-agent systems to require some degree of analysis before the principles underlying their global behaviour become apparent. One of the main requirements of multi-agent systems frameworks, besides providing a convenient

[1] http://cs.gmu.edu/∼eclab/projects/mason/

[2] http://ccl.northwestern.edu/netlogo/

format for designing agent behaviours, is to provide tools for handling the analysis of a system's behaviour over multiple simulation runs with different initial conditions or different global parameters (*i.e.*, to perform *parameter sweeps*).

2.2. Facilitating the Design of Live Algorithms

Blackwell and Young [2] define a live algorithm as a software system capable of interacting appropriately with a human in an improvised musical context, and offer a skeletal modular design consisting of listening, processing and sound generating modules. This very general modular deconstruction of a musical agent applies equally to the context of a musical multi-agent system which should be able to run in real-time.

2.3. Creating Flexible Agents and Configurations for Multiple Contexts

Both agents and their environments may be usefully embedded in disparate contexts. For example, a multi-agent evolutionary simulation may produce an evolved population, from which one individual is used on its own in a live performance. Or a simulation may need to be run in non-real-time many times to perform a parameter sweep to find the ideal settings for a real-time installation that runs with a visual accompaniment.

2.4. Probing the System Interactively

As an extension to the general requirement of extracting data from a multi-agent simulation, it is often useful to interactively explore a simulation. For example, if the simulation was an instantiation of Conway's *Game of Life* [5], it is informative to explore the system's behaviour by being able to pause the model and manually edit its state. However, any interactive interface should be implemented separately from the agents and world themselves, and deactivated for efficient real-time playback.

2.5. Working with a Well-Integrated System

A critical requirement is an effective development environment where disparate elements can be brought together, in particular, a multi-agent modelling framework and an audio framework. This is a practical goal in a number of respects. The most immediate benefit is minimising the dependency on communication protocols between applications, which inevitably require a greater workload and early commitment to potentially bad design decisions.

2.6. Facilitating New Forms of Extensibility

Above all, research into sonic ecosystems should be a creative exploratory process based on powerful tools that facilitate an understanding of the system being developed. Thus the overarching aim of the above goals is to facilitate new ways of handling complexity. Developing generative software, especially within the ecosystem paradigm, can easily lead to opaque behaviour [3]. The ecosystem paradigm necessitates a creative development cycle where designs are analysed, and analysis informs changes to design: the more you know about what your system is doing the better.

3. TOWARDS AN INTEGRATED ENVIRONMENT

The overall goal of building a framework for ecosystemic artworks is therefore to produce a general purpose audio and computer music library, integrated into a modelling framework. The modelling framework should resemble existing multi-agent modelling frameworks, such as Repast, Mason and NetLogo, but must be more specifically geared towards evolutionary and ecosystemic models, and easily adapted for multiple target applications, such as batch processing, interactive exploration and real-time performance.

The model discussed below uses a framework that has been designed with these principles in mind. The framework encourages a distinction between the model itself, consisting of a configuration of schedulers, listeners and agents (usually one scheduler and a population of agents), and various elements external to the model, such as analysis tools and visualisation and sonification elements, which are typically implemented as listeners, and can be easily attached and detached from the main simulation. The modularity and nested structure of the format for the main model is also designed to encourage ecosystem-specific forms of extensibility, such as the future integration of two populations of agents in a common environment, or the embedding of agents and their environments as subsets of bigger environments.

The configuration of a model for different contexts is shown in Figure 1. A set of elements towards the centre of the model (described below) is highlighted in both the batch and real-time modes. This is the core of the simulation, consisting of a scheduler managing a population of agents. In this model, agents inhabit a sonic environment and listen to and generate sound. The scheduler also handles the audio system which can either be run in real-time or non-real-time.

4. A SONIC ECOSYSTEM MODEL

This section describes an ecosystem model and generative music installation work which exemplifies a number of the issues discussed in Section 2. The model aims to animate the sounds of a man-made acoustic environment, establishing a digital wilderness into which these sounds are 'released' and allowed to evolve interdependent relationships: a sort of biologically-inspired Musique Concrète.

The model explores the role of feedback between acoustic and computational elements by treating the installation

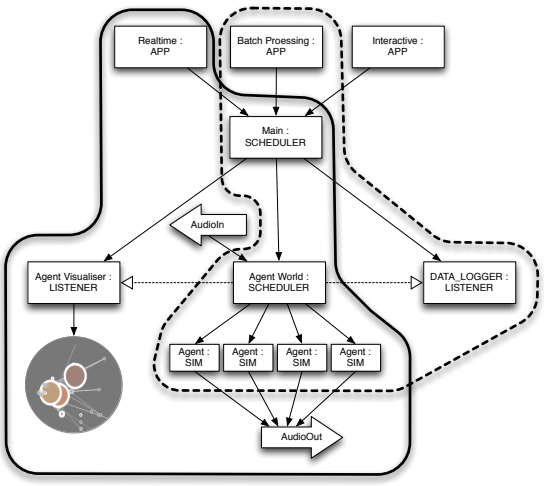

Figure 1. Different model configurations. In the real-time configuration (solid line), the simulation is run from with the RealTime application, with real-time audio and visualisation but no analysis. In the batch processing configuration (dashed line), the simulation is run from within the Batch-Processing application, with non-real-time audio and analysis but no visualisation.

space as an acoustic environment to which a population of virtual agents is exposed, and defining the central resource of these agents as the 'available space' in each frequency bin of an ongoing spectrogram. Sounds from the local environment are also used as source material for the agents to play back. Agents get fitter by 'inhabiting' bands of this frequency space that have less noise in them. One implication of this is that any sound made in the room will deplete the system's available resources: a large noisy audience or background environment creates a harsher environment with lower carrying capacity than a quiet empty room. Consistent sounds such as humming fans and traffic noise establish the long-term structure of the environment.

For sonic ecosystems, the design and study of resource models is an important part of the creative process. In this model, the design of the resource model is based on a *catch twenty-two* situation in which the greater the contribution an agent makes to the sound that is being made in a given frequency band, the greater the split that agent gets of the available resource. This sets up a trade-off in which the ideal amount of noise to be made depends on the likely behaviour of other agents. An agent on its own would do best making the tiniest squeak (maximising the available space), but two agents making tiny squeaks would, over time, compete to make louder and louder squeaks, since the gain from being louder than one's competitor outweighs the gain for being quiet.

The agent is fed analysis data drawn from its environment, and possesses a sample player and a neural network decision-making unit. The sample player plays samples taken from a bank of sounds. At each time-step, each agent's decision network receives the power spectrum data from the environment and outputs an action: either start playing, modify the sound, or stop playing. The start and modify actions are accompanied by a set of modifier commands that control the pitch, gain, loop start, loop end, and panning of the sound, and each such command consists of a destination value and a duration over which to perform the transformation. The decision network has internal nodes which act as simple 'memory' variables and are modified by the network according to its output.

These various contributors to an agent's behaviour are determined by a set of evolutionarily variables: the weights of the decision network, the list of actions and their various sample control commands, and the sample used by the sample player.

Reflecting the design requirements discussed above, this model was created with the expectation that it would undergo further development through exploratory study. I discuss a simple batch run of the model to investigate the effects of certain global model parameters on behaviour. A sweep was performed across parameters controlling the overall space of the environment, the proportional size of the fitness gained by agents, and the relative distribution of the available space across frequency bands. For each parameter set, a configuration file, a record of the evolutionary data, and a recording of the generated audio was gathered. Having looked at the analysis data and recordings, any configuration could then be loaded into the real-time application. The recorded evolutionary data was used to quickly see which parameters led to populations with more than one species. As shown in Figure 2, it is easy to notice from a time-based plot of an evolutionary parameter that, in this case, the population has diversified. The graph shows the value of a specific parameter of the decision network for each agent at the time of its death (decision network parameters do not vary over agents' lifetimes). This separation into two 'species' corresponds to a sonic ecosystem in which two distinct sonic processes can be heard unfolding. Evolutionarily, this means that under the conditions of the model, the two species were able to coexist for some time, before one died eventually out. This illustrates the potential compositional use of the model.

Further batch processing, analysis and modification in this manner can be used to look for situations in which multiple species coexist in the same sonic environment. This is likely to require, in particular, more detailed analysis and modification of the resource model, distribution of the available space, and the adaptive intelligence of agents (*i.e.*, the decision network) to help them find strategies for coexisting with kin. The analysis described above is a first step towards a model-specific test suite for seeking interesting dynamics

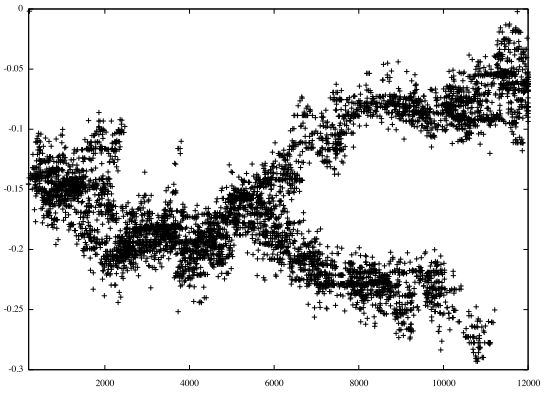

Figure 2. Change over time of one evolutionary variable (the strength of one of the weights of the decision network) across the population. The graph clearly shows the population diverging into two distinct forms, with one of the forms dying out towards the end of the simulation.

and structures. As well as the ecosystemic scenario itself, this could involve other scenarios such as specific unit-tests for evolved agents; taking the agents from the ecosystem to a lab.

It is also clear from the above method that the entire ecosystem model can itself be turned into a unit of selection, with its simulation parameters used as genetic variables. Likewise, individual agents could be extracted and used as live algorithms. This kind of flexibility with respect to evolving objects has exciting potential, although they are not an immediate area of focus.

5. DISCUSSION

This paper has introduced some issues involved in ecosystem-based music in terms of a set of design requirements. A framework built with these design requirements in mind was used to study the behaviour of an ecosystem-based sound installation. Both the framework and the installation presented here are works in progress, but illustrate the creative potential involved in an ecosystemic approach to generative music, and the value of building a well-integrated development environment that addresses these goals. The current work only partially address this set of requirements, and needs to be developed and tested further in live installation environments[3].

6. ACKNOWLEDGEMENTS

Aidan Lane, Jon McCormack, Alan Dorin, Benjamin Porter and Alice Eldridge made valuable contributions to the ideas in this paper. This research was funded by the Australian Research Council under Discovery Project grant DP0877320.

[3] At the time of writing, the installation had just completed its first performance at Club Transmediale, Berlin.

7. REFERENCES

[1] R. Axelrod, "Advancing the art of simulation in the social sciences," in *Handbook of Research on Nature Inspired Computing for Economy and Management*, J.-P. Rennard, Ed. Hersey, PA: Idea Group, 2005.

[2] T. Blackwell and M. Young, "Live algorithms," http://www.timblackwell.com/, 2005.

[3] E. Di Paolo, J. Noble, and S. Bullock, "Simulation models as opaque thought experiments," in *Articial Life VII: Proceedings of the Seventh International Conference on Articial Life*, M. A.Bedau, J. S. McCaskill, N. H. Packard, and S. Rasmussen, Eds. Cambridge, MA: MIT Press, 2000, pp. 497–506.

[4] D. Fliedner, "Six levels of complexity; a typology of processes and systems," *Journal of Artificial Societies and Social Simulation*, vol. 4, no. 1, 2001.

[5] M. Gardner, "Mathematical games: The fantastic combinations of john conway's new solitaire game "life"," *Scientific American*, vol. 223, pp. 120–123, October 1970.

[6] J. McCormack, "Artificial ecosystems for creative discovery," in *Proceedings of the 2007 Genetic and Evolutionary Computation Conference*, D. Thierens *et al.*, Eds. ACM, New York, 2007, pp. 301–307.

[7] F. J. Odling-Smee, K. N. Laland, and M. W. Feldman, *Niche Construction: The Neglected Process in Evolution*, ser. Monographs in Population Biology. Princeton, USA: Princeton University Press, 2003, no. 37.

[8] S. F. Railsback, "Agent-based simulation platforms: Review and development recommendations," *Simulation*, vol. 82, no. 9, pp. 609–623, 2006.