Self-organised music

TIM BLACKWELL† and MICHAEL YOUNG‡

†Department of Computing, ‡Department of Music, Goldsmiths College, University of London, New Cross, London SE14 6NW, UK
E-mail: t.blackwell@gold.ac.uk, m.young@gold.ac.uk

Self-organisation, as manifest, for example, by swarms, flocks, herds and other collectives, is a powerful natural force, capable of generating large and sustained structures. Yet the individuals who participate in these social groups may not even be aware of the structures that they are creating. Almost certainly, these structures emerge through the application of simple, local interactions. Improvised music is an uncertain activity, characterised by a lack of top-down organisation and busy, local activity between improvisers. Emerging structures may only be perceivable at a (temporal) distance. The development of higher-level musical structure arises from interactions at lower levels, and we propose here that the self-organisation of social animals provides a very suggestive analogy. This paper builds a model of interactivity based on stigmergy, the process by which social insects communicate indirectly by environment modification. The improvisational element of our model arises from the dynamics of a particle swarm. A process called *interpretation* extracts musical parameters from the aural environment, and uses these parameters to place attractors in the environment of the swarm, after which stigmergy can take place. The particle positions are reinterpreted as parameterised audio events. This paper describes this model and two applications, Swarm Music and Swarm Granulator.

1. INTRODUCTION

Many animals exhibit remarkable collective behaviour. Social insects gather in large numbers – swarms – to forage and build nests. The ability of flocking birds to coordinate their motion in order to avoid obstacles and to rapidly change direction of flight is well known to us all. Surprisingly, this collective behaviour does not necessarily derive from central organisational control or leadership, but arises from the local behaviour and interaction of (relatively) simple organisms. Indeed, the ability of swarms, flocks, etc., to shrink and grow in population (there appears to be no upper limit to population size), suggests that their computational complexity is linear. In other words, each swarm member is only aware of other members in its immediate neighbourhood. A dramatic example is to be found with the huge shoals of migrating herring, sometimes up to seventeen miles long and with millions of members; it is hard to conceive of any centralised method of communication that can account for this collective behaviour (Reynolds 1987). Indeed, Reynolds demonstrated that the behaviour of flocks, schools and swarms can arise merely from local interactions between the entities. There is no need for global coordination. The rules for an individual in a flock or school can be summarised as: try to move close to your neighbours, avoid collisions, and try to match your velocity (i.e. speed and direction) with your immediate neighbours. (The rules for swarms are even simpler since the requirement for velocity matching is dropped.)

Compelling evidence that entities that are interacting through local rules can self-organise into large spatio-temporal structures comes from computer animations of animal behaviour (Reynolds 1987; Bonabeau, Dorigo and Theraulaz 1999). A common theme of these investigations is the desirability of decentralised organisation, from the perspective of stability and adaptability. Theories of global forms emerging from local interactions have also been applied to the study of human systems such as economics and traffic flow (Resnick 1997). *Self-organisation* is believed to be based on four components: positive feedback, negative feedback, amplification of fluctuations and multiple interactions (Bonabeau *et al.* 1999). In addition, an important mechanism for self-organisation, known as *stigmergy* was proposed by Grassé (1959) to account for the nest building activity of termites. This mechanism has subsequently shown to be of a more general nature and many examples are explored in Bonabeau *et al.* (1999). A stigmergetic interaction is a form of indirect interaction, occurring when one individual modifies a feature in the environment that another individual responds to at a later time. In Grassé's example, the nest building of termites is coordinated by the structure of the nest and not by the workers themselves; a new configuration is produced when the activity of one worker stimulates another (possibly different) activity by another worker. Latterly, these ideas have been applied to various systems, including cooperative transport by a swarm of robots (Kube and Zhang 1994).

Natural systems interact with an uncertain, changing, complex environment. The result of this interaction is the high degree of diversity found in the natural world; one design does not suit all. A top-down, rigid

Organised Sound **9**(2): 123–136 © 2004 Cambridge University Press. Printed in the United Kingdom. DOI: 10.1017/S1355771804000214

design may achieve good results in a certain context, but it may struggle if the environment changes. Flexible, dynamic and adaptable systems, however, change with the environment, developing novel solutions to problems; they form an interesting model of human creativity (Bentley and Corne 2001). It is pertinent, therefore, to view another system – the organised sounds produced by human groups, commonly known as music – from this perspective. The analogy can be made at a formal level. If we consider, for example, musical tones to be simple individuals interacting with neighbouring tones through simple rules, then the question is: can self-organisation lead to higher-level structures (which in this case would correspond to melodies and rhythms)? This idea was tested in early versions of Swarm Music, a MIDI-based system where sound events are represented as interacting particles moving in a three-dimensional Euclidean space. The coordinates of each particle position correspond to pitch, amplitude and time interval between events. The swarming behaviour of these particles leads to melodies that are not structured according to familiar musical rules, but are nevertheless neither random nor unpleasant (Blackwell 2001, and section 5.2.2).

An analogy can also be found in freely improvised performance. An uncertain and complex musical environment may itself stimulate innovation where a more controlled environment of preconceived ideas (i.e. top-down design) may falter. Structure arises from the temporally local interaction of individuals who continuously alter the environment, perhaps without prior knowledge or clear intention. The analogy with the natural systems of social animals is immediate. The group dynamic is, at least in part, stigmergetic.

The point here is not that musicians are simple organisms but that it is possible, at some level of description, to explain complex collective behaviour from the assumption of relatively simple interactions (Bonabeau *et al.* 1999). It is entirely possible, therefore, that computer systems can take part in free improvisations, contributing and responding to the musical environment: all we need do is to define suitable interactions between machine and humans, and ensure that the system implements the components of self-organisation referred to above. In the following, we shall call such a system an 'artificial improviser'. This does not imply that an artificial improviser would operate as a human might but merely that it is engaging in low-level stigmergetic interactions with a dynamic musical environment.

In order to build an artificial improviser, a mechanism for stigmergy must be found. A very transparent mechanism was used in Swarm Music. The external environment consists of MIDI events (including audio events that have been parsed to MIDI parameters) emanating from external performers (humans or other swarms). These events are placed as **attractors** in the Euclidean space of the swarm. Particles are drawn towards these attractors, and the ensuing organisation of the swarm around the attractors produces a melodic stream influenced but not wholly determined by these events. The process of attractor placement and conversion of particle positions into output events is called **interpretation**.

The innumerable timbral qualities of the musical environment are, however, missed in a MIDI parameterisation, and these qualities are also likely to be important for humans. This paper addresses this issue by presenting a generalised scheme for an artificial improviser, abstracted from Swarm Music, but with far wider application. One application of this general scheme is Swarm Granulator, where swarms of interacting sound grains produce self-organising timbres and textures.

It is the purpose of this paper to present a working model of creative interaction with an artificial improviser. The model is split into two fundamental processes: interpretation and swarming. Our interpretative model for interaction is proposed in section 3. Interpretation is a multiple stage process of parameterisation that determines the whole mechanism for performance interaction. An overview of swarming and how swarm events are generated is presented in section 4. This section presents a detailed specification, through a number of particle update rules, for the implementation of a **particle swarm**. Practical implementation of the two applications already mentioned (Swarm Music and Swarm Granulator) is explained in section 5. However, it is first worth considering briefly the very wide topics of improvisation, interaction and musical organisation, and how their complexities can be addressed.

2. IMPROVISATION, INTERACTION AND MUSICAL ORGANISATION

2.1. Improvisation and emergence

Uncertainty surrounds any improvised performance. The extent of this uncertainty is contingent on the presence (or absence) of *a priori* agreements, whether explicit or tacit. This is expressed as the 'degree of improvisationality' by Sawyer (2003). 'Ritualised' or highly controlled improvised performance can be distinguished from freer creative music-making in a number of ways, not least by the cultural value ascribed to its practice, and the extent of its apparent 'ossification'. Aspects of performance practice are highly significant: an emphasis upon collective (rather than individual) improvisation suggests a less controlling approach, as does the avoidance of recourse to notation or other pre-existing materials. Such factors constrain or afford opportunities for creative input.

Freely improvised music is concerned only with the creative contribution of participants, and is

deliberately and self-consciously uncertain. Inevitably, performers will contribute according to their own experiences, prior learning, practices and habits (whether individual or culturally determined), but in a group setting they may well have little idea of what will proceed before the first sound begins. The listeners' experience is comparable. Such music-making is in marked contrast to 'ritualised' genres such as Indonesian gamelan or Indian classical music. (Jazz, in all its diverse idioms, is harder to classify.) In existing freely improvised repertory, from Stockhausen's *Aus den Sieben Tagen* to the diverse work of AMM, the emphasis is not upon the top-down organisational strategies of Western classical music but upon collective acts; interactional processes take precedence. Creative input is constrained chiefly by the social experience of participants. These experiences are explored within the context of jazz (and all its stylistic and technical precepts) by Berliner (1994). A much more general theory of interactive behaviour is offered by Katovitch (1986) and further explored by Bastien and Hostager (1992). This theory of social interaction within a group is described as 'becoming situated'. Shared goals are recognised and pursued by group members, who both assume and cast roles. This might result from knowledge of 'suprapersonal social facts' (Katovitch), that is, shared historical assumptions. More interestingly, a brand new shared history evolves as the cooperative experience develops; as players may become aware of the appropriateness of their response to others' contributions, they may also appraise their own ability to initiate behaviour from others.

Comparable issues are explored in Sawyer's 'interactional semiotics' of group improvisation. This is predicated on the notion that ensemble behaviour is greater than the sum of its parts. During a collective performance, any appropriate individual contribution is *pragmatic*, meaning that it can be assumed to embody the tacit agreements established by other contributions made up to that point. This is known as 'indexical presupposition', a term used by Silverstein (1993). The same contribution *projects*; it invites consequences; it has an indexical relationship to future possible interactions (that is, it represents 'indexical entailment'). An emphasis upon entailment helps again to distinguish free improvisation from more restricted or ritualised idioms.

At the core of these ideas is the notion of emergent structure as proposed by Meade (1932). Here emergence describes the spontaneous evolution of structure and meaning in social activities, such as conversation or improvised music-making. Meade writes, 'The emergent when it appears is always found to follow from the past, but before it appears it does not, by definition, follow from the past' (quoted in Sawyer 2003: 12). This is an eloquent description of free improvisation, which can only cultivate structure from the 'bottom-up'. Viewed as a whole, an ensemble can be seen to evolve self-organising behaviours, creating the illusion of certainty. Emergent organisation might even be found in the generalised structural functions of composed music, for example the 'complementary, counteractive and cofunctioning relationships' identified by Berry (1976: 7).

Such theories offer comparably convincing and complex models of interactive improvisation, at least in so far as humans experience it. However, they can only serve as metaphors for interaction that are complicated by a proactive artificial improviser. A precise algorithmic model is needed. Computer–human integration has often been modelled using the prosaic language of parameter mapping. This describes how human inputs can be parameterised, transformed and mapped onto control functions for a synthesizer, for example. The synthesizer produces a response; the human might listen and decide what to do next. Performance (gestural) behaviour is frequently the source of parameter information; alternatively, data can be obtained from the analysis/processing of live audio.

The theoretical categories of parameter mapping have been widely considered (Hunt *et al.* 2000; Arfib *et al.* 2002). Our system described below could possibly be likened to an implicit or generative mapping mechanism in which only general mapping behaviours can be anticipated, not real output values. However, the terminology of 'mapping' cannot adequately describe the integration of an artificial improviser into an ensemble that is becoming socially 'situated'. The artificial voice is as provocative and enabling as any other; it directly participates in the emergence of group organisation. In our performance model, the computer's contributions are as significant indexically as any act of a human performer. The machine must therefore possess strategies for interaction with real-life musicians.

2.2. Organisational levels

If self-organisation is possible within a musical context, it raises the question: organisation at what level? Music is often described in terms of layers of structure; for instance, building up from fundamental parameters (pitch, duration, loudness) to the composite and conceptual (melody, texture, form). Here we present merely a working method of organisational levels, leaving aside many complex issues.

Levels can be understood in terms of perceptual time scales (Roads 2001). The lowest relevant timescale pertains to the micro(structural)-level. Grains are measured in milliseconds; heard individually, they may sound like clicks or fragments from a recognisable source. Gabor's theories posit the fundamental nature of grains (Gabor 1947). We may on occasion be aware that sound comprises discreet

events. However, there is no direct analogy between micro-level analysis (e.g. FFT or wavelet analysis) and the parameters of granular synthesis.

The time-scale closest to our immediate experience is the sound event level or mini-(structural) level (Xenakis 1981); a time-scale measured from a few seconds down to perhaps 1/10th second. Such events might be considered to have static elements, in other words, fixed characteristics suggestive of the 'musical note'. This traditional view of the sound event has been attacked by Wishart, who criticises the hegemonic three-dimensional 'lattice' of pitch, duration and fixed instrumental timbre (1996). His criticism highlights the conceptual confinement of music when represented by Western notation. Often, sounds are much more complex, because they result from a physical agency or are heard to signify such an agency (Smalley 1992). So, although sounds may have identifiable static elements, they also contain properties that evolve over time. The term 'dynamic morphology' (Wishart 1996) emphasises the gestural nature of a sound event, which could be determined by attack/decay characteristics, fluctuations in amplitude, pitch and timbre. Granulation tends to produce such material, so the alternative vocabulary of 'sound mass' (Varèse 1971) and 'cloud' (Roads 2001) is particularly relevant.

The third relevant time-scale is the meso(structural)-level; the level at which sound events are experienced in relation to one another, rather than individually. Mesostructure arises from permutations at the lower sound-event level. For example, our attention might be drawn to particular aspects of the mini-level, overall pitch or loudness of events.

Perhaps less usefully, the meso-level can be thought to divide up the higher macrostructural level. This latter scale encompasses an entire piece or performance, and its characteristics determine overall musical form. In improvised music, the macro-level can only be described with the benefit of hindsight and reflection, once the complex interactions that cause structure to emerge are complete. In fact, structure may be thought to emanate upwards through all these levels: the character and behaviour of grains determines the timbral qualities (dynamic morphology) of sound events. In turn, significant aspects of sound events recur and evolve to form structural patterns at the meso-level, and so on. A self-organising system might wish to emulate this upwards progression, and a swarming improviser using granular synthesis has the potential to do this. One difficulty arises because the horizons between these time-scale/structural levels are rarely clear in practice, and are especially problematic in the idioms of electroacoustic and freely improvised music (i.e. when timbre and gesture are especially significant concerns). Wishart's discussion of the differences between a 'sequence' (i.e. a complex pattern of events) and 'texture' (a single event with complex characteristics) illustrates the point (Wishart 1994).

In order to create a useful working model we must first imagine a scenario in which real-life musicians work with a machine improviser. At any given moment in a performance, musician A will be aware of the sounds making up the musical environment, and (s)he may be aware of the source of those sounds (resulting from performers A, B, C . . ., etc., and our computer improviser). Musician A must decide, consciously or otherwise, how to listen analytically. Such listening can refer to perceived sources, structural levels, or to indexical properties, perhaps. Does the sound have prominent timbral features? Is it interesting because it is higher in pitch, or quieter than earlier sounds? Is the group as a whole gradually becoming less rhythmically active? Listening techniques would not be objective; musician A will have a whole set of personal and cultural values that affect his/her judgement. Musicians A and B may have had a heated argument just before the performance started. In any case, the musician is entirely free to respond accordingly, and (s)he may wish to complement or contradict the perceived qualities or direction of the music. Unlike the human players, the artificial improviser is not subjective, but behaves according to its hidden algorithmic design. To resolve these complex differences, we need a simple algorithmic model that allows us to design a workable system.

3. INTERPRETATION

3.1. Motivation

A simple process known as interpretation was proposed in the first version of Swarm Music (Blackwell 2001, 2003) which integrates all group contributors, whether carbon or silicon-based. This process has now been integrated into a more general model of interaction. This interpretative model forms the theoretical basis of our current implementations (a new version of Swarm Music and Swarm Granulator).

3.2. The interpretative model

Figure 1 depicts interaction between two systems (human or machine-based) A and B. System A is listening to audio Y emanating from B. A is also producing an audio output X. If A is interacting with B, then X must in some way depend on Y and this influence is denoted $X(Y)$. A similar meaning is attached to $Y(X)$.

This picture, however, hides much. Human systems will be quite selective about which parts of the audio environment they will use to inform their own output, and this is desirable for silicon improvisers too. Interactivity merely implies that A is influenced by B,

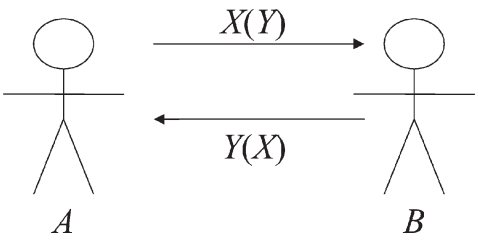

Figure 1. A simple model of interaction. *A* is depicted here to be in interaction with another individual, *B*, but *B* could equally well represent a sub-group of the ensemble. A similar point applies if the diagram is read from *B*'s perspective.

and this influence can be quite weak. In general, we can say that *A*'s musical output will depend strongly on many personal, hidden variables h_A. For humans, h_A would correspond not only to operational rules but also to variables not easily quantifiable such as instrumental training, previous improvisational encounters and stylistic influences. For silicon-based improvisers, h_A corresponds to the details of internal generative algorithms. Even without interaction, then, *A* would still produce an output, so the notation can be extended: $X = X(Y, h_A)$.

Looking now at how *Y* can influence *X*, it is clear that *A* must attach meaning (interpret) the input *Y*. Then a response can be prepared based on the useful information that *A* has inferred. The complexities surrounding the many hidden processes (experience, volition, aural ability, ideology . . .) will be ignored. Interpretation by *A* of the aural environment will here be represented simply as P: $Y \rightarrow p$, where p represents some of the information that *A* can infer from *Y***.** In this mathematical notation, *P* is a function, mapping audio input *Y* to an internal representation p. However, to use the language of the section 2.1, *P* embodies the meanings that can be understood to emerge from presupposition and entailment.

We split the preparation of output into two functions, *F* and *Q*. *F* represents *A*'s internal processing activities (i.e. how *A* processes internal representations of the environment). *Q* represents how *A* uses this information to actually deliver a sound, for example, through playing an instrument or via control of some synthesizer. To be specific, *F* is the process $F(h_A)$: $p \rightarrow q$, where q is an intermediate state. *Q* takes the intermediate state and prepares an output Q: $q \rightarrow X$. The model therefore depends on three functions, *P*, *F* and *Q*, which loosely correspond to listening, reflecting and responding:

$$Y \xrightarrow{P} p \xrightarrow{F(h)} q \xrightarrow{Q} X \tag{1}$$

P and *Q* are interpretative functions since they are responsible for the conversions of external audio *Y* to the internal representation p, and from the internal representation q back to audio *X*. Expression (1) is very general, since *F* deals only with internal representations p and q. However, if we wish to utilise stigmergy and self-organisation, *F* will need to implement the components of self-organisation referred to in the introduction (positive and negative feedback, amplification and multiplicity). One choice for *F*, therefore, is to use a *swarming function* $\boldsymbol{f}$;

$$\boldsymbol{q}_i = \boldsymbol{f}(\{\boldsymbol{x}_i\}, \{\boldsymbol{p}\}, c). \tag{2}$$

In this swarming equation (2), $\{\boldsymbol{x}_i\}$ denotes the positions of individuals within the swarm and $\{\boldsymbol{p}\}$ are features of the environment, e.g. attractors. *P* therefore modifies the internal environment of the swarm in response to changes in the external audio environment. Self-organisation of the swarm $\{\boldsymbol{x}_i\}$ around $\{\boldsymbol{p}\}$ leads to correlations in output q and hence, via mappings *Q*, to correlations in audio output *X*. The hidden variables are $h = (\{\boldsymbol{x}\}, c\}$, where the behaviour of the swarm is parameterised by constants c. A full explanation of the swarming equation will be given in the next section.

Although this formalism requires an internal representation p of external sounds, p does not need to be dynamic. For example, a fixed configuration of p will lead to non-interactive improvisations in some fixed region of the space of outputs, corresponding to the patterns that the swarm makes around p in *N*-dimensional Euclidean space. (Such a space, denoted $\boldsymbol{R}^N$, shares the same properties of the three-dimensional space that actual swarms move in, but the dimensionality may be greater or less.) The influence of *Y* can also be weak; *P* can ensure that huge changes in *Y* only lead to small changes in p. Or indeed, *P* could be highly sensitive to small environmental changes.

From the perspective of interpretation, the internal processes – swarming – merely transform the input parameterisation into modulating numbers; the details (representations and rules) of the transformations are hidden. Interpretation involves listening at some structural level (*P*) and responding at the same, or different level (*Q*). The interpretation can be 'transparent' with *P* equal to the inverse of *Q*, $P = Q^{-1}$ (implying a like-for-like improvisational strategy), but this is not the only option. All that matters is that the interpretative functions *P* and *Q* are *transparent enough* for interacting humans to grasp and use during performance – a similar point, of course, applies to successful human–human *interaction* (although interactivity alone is not a necessary condition for a successful improvisation).

3.3. Dynamic refinements

So far, only a static view of the interpretation has been presented. For example, the formalism does not depict

that audio streams X and Y comprise discrete audio events (on time scales down to the inverse Nyquist frequency) and that p and q represent a sequence of internal representations. The term 'audio event' is used here merely formally to indicate windowing of the audio streams at time scales $\Delta\tau$, and is not meant to imply any level-dependent semantics. However, the choice of interpretation functions P and Q certainly is a level-based activity. Assuming that a musical level can be parameterised at a time scale $\Delta\tau$, P and Q can be drawn from a set of level-dependent functions, labelled by an index α, $\{P_\alpha(\Delta\tau), Q_\alpha(\Delta\tau)\}$. Suppose that an audio event $e(\tau, \Delta\tau)$, commencing at time τ and of duration $\Delta\tau$, is projected out of the stream Y by a projection operator $E(\tau, \Delta\tau)$, $e(\tau, \Delta\tau) = E(\tau, \Delta\tau)Y$. Then the interpretative functions can be decomposed as

$$P_\alpha(\Delta\tau) = P_\alpha E(\tau, \Delta\tau) \tag{3}$$

where P_α is a member of a suite of interpretative functions which operate on a generic event e. For example, the extraction of amplitude is a generic function, applicable to events on any timescale.

Equation (1) is a general statement of stigmergy. P invokes a change to environmental variables p experienced by a swarm, and Q embodies the swarm's response, which is the output of environment variables q. However, stigmergy in nature is an indirect interaction, which means that the swarm's response happens after some time delay. Therefore, the rate of flow of information through equation (1) in the interpretative model is very important because, for example, if the attractors in (2) are placed in the swarm's environment immediately after events are projected from Y, then $X(Y)$ is potentially highly coupled in time, which will almost certainly lead to stereotyped improvisations. Alternatively, it will be hard for listeners to perceive any interaction at all if $X(Y)$ is loosely coupled in time. In practice, a human improviser will memorise recent information p, and will be quite selective about what elements of p to use in the future. The $p(\tau)$ should therefore be delayed before being sent to F (i.e. how A processes internal representations of the environment) by

$$p(\tau + \tau_{delay}) = P_\alpha E(\tau, \Delta\tau)Y. \tag{4}$$

Some examples of interpretative schemes are shown in figure 2.

4. SWARMING

4.1. Overview

This section defines a particle swarm and links the movement of this swarm to the swarming function of

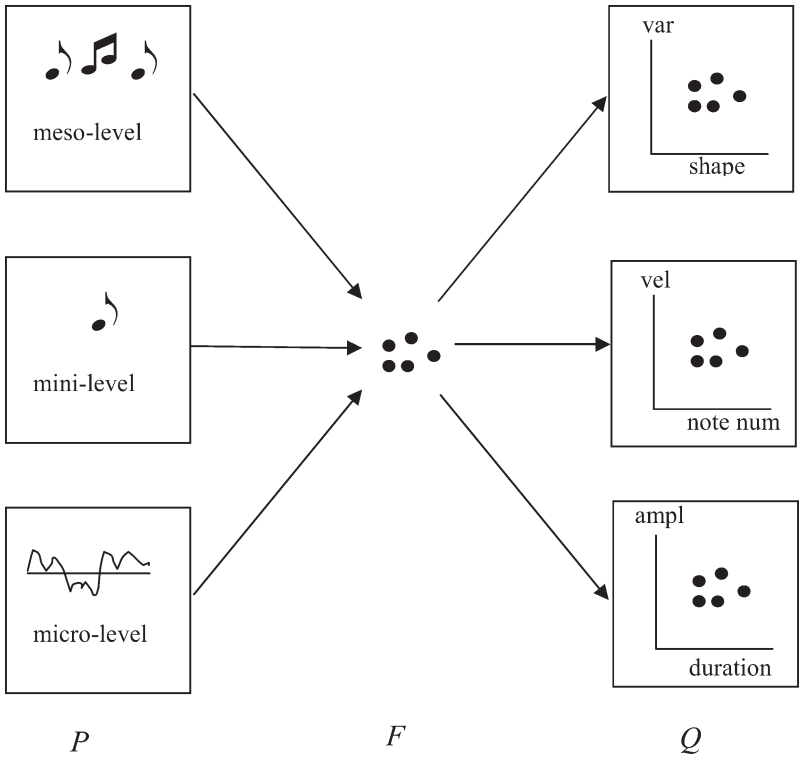

Figure 2. Possible interpretative schemes. In this example, P interprets at the micro-, mini- or meso-level, sending parameters to F which is depicted as a swarm of particles. The action of Q can be thought of as projecting particle coordinates onto axes denoting level-dependent parameters. At the micro-level, these axes might include grain amplitude and grain, at the mini-level they may include MIDI note number and velocity, and at the meso-level, phrase shape and variance.

section 3. Particle swarms ultimately derive from the virtual flocks of Reynolds' original animations (Reynolds 1987), but the flapping animated 'boids' are replaced with point-particles in N-dimensional Euclidean space, $\boldsymbol{R}^N$. The particles change their positions by the application of simple forces or accelerations. Reynolds established convincing flock animations with just three accelerations: a spring-like attraction towards the centroid of neighbouring particles, a collision-avoiding acceleration and a velocity-matching acceleration. Typically, particle swarms use similar accelerations, except that a swarm does not implement velocity matching of near-neighbours. Particle swarms may also use attractions to special positions in space, known as targets or attractors (Blackwell 2001, 2003). For the improvising systems proposed here, the attractors derive from external audio events, and are positioned in $\boldsymbol{R}^N$ according to the interpretative function P. The particle accelerations determine the development of the swarm in time; they are presented in some detail in section 4.2. Section 4.3 explains how particle positions are converted into swarm events. These swarm events, which reflect the shape of the swarm in Euclidean space, are mapped to actual audio events by Q.

4.2. Particle swarms

A swarm S of M particles is denoted $S=\{\tilde{P}_k\}$, $k=1 \ldots M$. A particle $\tilde{P}_k$ is specified by two N-dimensional vectors, $\{\boldsymbol{x}_k, \boldsymbol{v}_k\}$, where $\boldsymbol{x}_k$ and $\boldsymbol{v}_k$ are the particle position and velocity. Additionally, a particle has access to attractor k at position $\boldsymbol{p}_k \in \boldsymbol{R}^N$. Two vectors $\bar{\boldsymbol{x}}$, $\bar{\boldsymbol{p}} \in \boldsymbol{R}^N$ describe global properties of the swarm and the attractors. The swarm position $\bar{\boldsymbol{x}}$ lies at the centroid of the particle positions $\{\boldsymbol{x}_k\}$ and the swarm attractor $\bar{\boldsymbol{p}}$ lies at the centroid of the particle attractors $\{\boldsymbol{p}_k\}$.

A time development operator $U(t-1, t)$: $S(t-1) \rightarrow S(t)$ moves the swarm forward in time, updating each particle in turn, $\tilde{P}_k(t-1) \rightarrow \tilde{P}_k(t)$. A sweep through the whole swarm, updating each particle in turn, is counted by an index i. The discrete time counter t is defined by $t = Mi + k$ (note that this is not the same as real time τ used in section 3.3 and in section 4.3 below). U is specified by a set of rules, equations (5)–(12). Equations (5)–(6) show the general form of the update. The three accelerations towards the swarm centroid, the attractor centroid and the avoidance acceleration are summed. k's velocity is updated by adding the total acceleration, $\boldsymbol{a}_k(i)$ to the current velocity of particle k (5 and 6). The position update for particle k, equation (7), is then performed by simply adding the updated velocity to the current position.

$$\boldsymbol{a}_k(i) = \boldsymbol{a}_{swarm,\,k} + \boldsymbol{a}_{attractor,\,k} + \boldsymbol{a}_{avoid,\,k} \tag{5}$$

$$\boldsymbol{v}_k(i) = \boldsymbol{v}_k(i-1) + \boldsymbol{a}_k(i) \tag{6}$$

$$\boldsymbol{x}_k(i) = \boldsymbol{x}_k(i-1) + \boldsymbol{v}_k(i) \tag{7}$$

Five scalar constants $c=\{v_{\text{clamp}}, \tilde{q}, m, d_{core}, d_{limit}\}$ parameterise U. v_{clamp} is a clamping or limiting velocity which sets an upper limit to the effect of the accelerations. If the result of calculation (6) is to update the velocity to a magnitude $|\boldsymbol{v}|$ that is larger than v_{clamp}, then $\boldsymbol{v}$ is rescaled by a factor $\frac{v_{clamp}}{|\boldsymbol{v}|}$ so that its magnitude is v_{clamp}. $\tilde{q}$ is a particle charge which sets the scale for the collision-avoiding accelerations. m is the particle mass; accelerations are inversely proportional to m. d_{core} is a small distance used to shape the inter-particle repulsion and d_{limit} is a perception limit. The perception limit is defined so that a particle at $\boldsymbol{x}$ is only aware of other particles and attractors within a box $B_{limit}(\boldsymbol{x}) = [-d_{limit}, d_{limit}]^N$ centred on $\boldsymbol{x}$.

Equations (8)–(12) explain the calculation of $\boldsymbol{a}_k$ in some detail – the time arguments have been dropped for simplicity.

$$\boldsymbol{a}_{swarm,\,k} = m_k^{-1}\,\delta(\bar{\boldsymbol{x}}, \boldsymbol{x}_k)\,(\bar{\boldsymbol{x}} - \boldsymbol{x}_k) \tag{8}$$

$$\boldsymbol{a}_{attractor,\,k} = m_k^{-1}\,\delta(\bar{\boldsymbol{p}}, \boldsymbol{x}_k)\,(\bar{\boldsymbol{p}} - \boldsymbol{x}_k) \tag{9}$$

$$\boldsymbol{a}_{attractor,\,k} = m_k^{-1}\,\delta(\boldsymbol{p}_k, \boldsymbol{x}_k)\,(\boldsymbol{p}_k - \boldsymbol{x}_k) \tag{9a}$$

$$\boldsymbol{a}_{avoid,\,k} = m_k^{-1} \sum_{l=1, l\neq k}^{M} a_{k,l}\,\delta(\boldsymbol{x}_k, \boldsymbol{x}_l) \tag{10}$$

$$a_{k,l} = \frac{\tilde{q}_l\,\tilde{q}_k}{|x_{core}|^2}\,\frac{x_k - x_l}{|x_k - x_l|},\ |x_k - x_l| < d_{core} \tag{11}$$

$$= \frac{\tilde{q}_l\,\tilde{q}_k}{|x_k - x_l|^2}\,\frac{x_k - x_l}{|x_k - x_l|},\ \text{otherwise.}$$

$$\delta(\boldsymbol{y}, \boldsymbol{x}) = \begin{cases} 1 \text{ if } \boldsymbol{y}_k \in B_{limit}(\boldsymbol{x}_k) \\ 0, \text{ otherwise} \end{cases} \tag{12}$$

Equation (8) is a linear spring-like acceleration towards the swarm position. Equation (9) is a similar linear acceleration towards the swarm attractor. Equation (9a) is an alternative to (9), whereby each particle is attracted to its own attractor, rather than the centroid of the attracting group. This increases the diversity of the swarm for some attractor configurations since a particle may feel conflicting pulls towards the swarm, and towards an outlying attractor. From a musical perspective, (9a) will make the swarm as a whole more responsive to new directions in the musical environment, so that output X more closely follows input Y. Equation (10) is a collision-avoiding acceleration between particle k and any other particle within $B_{limit}(\boldsymbol{x}_k)$. This repulsion is specified in equation (11).

The delta function $\delta(\boldsymbol{y}, \boldsymbol{x})$ occurring in these equations is defined by equation (12); it ensures that accelerations between a particle at $\boldsymbol{x}$ and a position $\boldsymbol{y}$ are only calculated if $\boldsymbol{y}$ is within $\boldsymbol{x}$'s perception, i.e. is in the box $B_{limit}(\boldsymbol{x})$.

Equation (11) shows that inter-particle repulsions $a_{k,l}$ are Coulombic repulsions between particles that are within the perception limit of each other, and are equal to a constant for separations less than the d_{core}. The core distance d_{core} limits accelerations which could otherwise become arbitrarily large for very small separations in the denominator of the inverse square law. $\bar{\boldsymbol{x}}$ and $\bar{\boldsymbol{p}}$ are always updated immediately before each particle update.

The particle attractors $\boldsymbol{p}_k(\tau)$ are updated in real time τ in a separate process to the particle update thread. These attractors do not arise from the swarm, but are placed in the swarm's environment in response to external audio stimulus. Storing the attractors in a buffer for a time delay τ_{delay} before placing them in $\boldsymbol{R}^N$ can also control the rate of flow of information through the system.

Equations (8)–(12) differ slightly from earlier versions of Swarm Music. In particular, the spring constants determining the strengths of the attractions have been set at unity because the parameter with the dominating effect on output is v_{clamp} (Blackwell 2003). The Coulomb repulsion also differs slightly in this version because the spatial dimensions are decoupled, as specified in equation (10). The update rules are merely N copies of a one-dimensional dynamical system. In the earlier version, the components were coupled through the Coulomb repulsion which was a function of the Euclidean distance $|\boldsymbol{r}|$ between particles. Dimensional coupling can still take place, but it must be handled by the interpretative functions (see section 5.2 for a discussion on the musical effect of dimensional decoupling).

4.3. From particles to events

The swarmer is a software module that implements a particle swarm. The function of the swarmer is to use a particle swarm to provide a series of 'swarm events' $\boldsymbol{q}(\tau)$. This is accomplished with two timing functions which operate on two components of a particle position vector. The other components of the position vectors provide event parameters. The timing information for event start and end is therefore internally derived from the swarm thus allowing for *temporal* organisation of events due to *spatial* organisation of the particle positions. In order to explain how the swarmer can do these things it is helpful to consider the swarming equations (5)–(12) in terms of a state machine. The state of this machine is the current swarm configuration $S(t)$ and the transition function consists of M time-development operators,

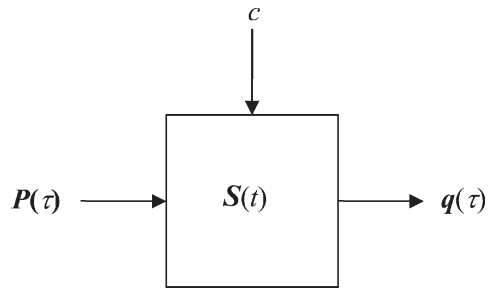

Figure 3. Swarmer as a state machine. The machine has inputs $\boldsymbol{p}(\tau)$ and c, an output $\boldsymbol{q}(\tau)$, and current state $S(t)$.

$\boldsymbol{U} = (U_1, U_2 \ldots U_M)$, where each U_k updates particle k according to the rules (5)–(12). The swarmer as a state machine is illustrated in figure 3.

In order to extract temporal information from the spatial information $\boldsymbol{x}$, two timing functions are introduced, g_1 and g_2. The swarmer pauses for a time $\delta\tau_t = g_1(\boldsymbol{x}(t).\boldsymbol{e}_1)$ upon generation of state $\boldsymbol{x}(t)$, where $\boldsymbol{e}_1$ is the unit vector along direction **1**. The required output from the swarmer is a stream of swarm events $\boldsymbol{q}_1, \boldsymbol{q}_2, \boldsymbol{q}_3 \ldots$, commencing at times $\tau_1, \tau_2, \tau_3 \ldots$ Events, however, must have a beginning and an end. A second timing function $g_2(\boldsymbol{x}(t).\boldsymbol{e}_2)$ extracts an interval $\delta\tau_{t,\,event}$ from $\boldsymbol{x}_k$, where $\delta\tau_{t,\,event}$ is the duration of event $\boldsymbol{q}_t$, i.e. $\boldsymbol{q}_t$ ends at $\tau_t + \delta\tau_{t,\,event}$. The event timing information $\delta\tau_t$ and $\delta\tau_{t,\,event}$ is packaged in the N-dimensional event vector $\boldsymbol{q}_t = (\delta\tau_i, \delta\tau_{i,\,event}, \boldsymbol{x}_k.\boldsymbol{e}_3, \boldsymbol{x}_k.\boldsymbol{e}_4 \ldots \boldsymbol{x}_k.\boldsymbol{e}_N)$. The connection between the real time τ and discrete time index t is

$$\tau_t = \sum_{s=1}^{s=t-1} \delta\tau_s. \tag{13}$$

Note that the focus on discrete events does not limit the generality of the swarmer. Event frequencies can be increased right up to the Nyquist frequency of the digital-to-analogue converter in order to provide near-continuous modulation of an output stream. Any module accepting swarm events is not, of course obliged to use $\delta\tau_t$, $\delta\tau_{t,\,event}$, although it will certainly need to store $\delta\tau_{t,\,event}$ if it is to parameterise audio events, because the module will need to know when to terminate the audio event triggered by the reception of $\boldsymbol{q}_t$, and audio events will overlap if $\delta\tau_{t,\,even} > \delta\tau_t$.

In summary, the swarmer produces a set of discrete swarm events $\boldsymbol{q}_t$ at times τ_t. These events are generated by particle movement which is determined, in turn, by a set of particle update rules. These rules calculate an acceleration for each particle due to nearby attractors and other particles. The strength of these accelerations is determined by a set of particle update constants. The swarm attempts to organise itself around the attractors, which themselves are dynamic. The functionality of the swarmer can be conveniently written using timing functions $\boldsymbol{G}_k = (g_1, g_2, I_3 \ldots I_N)$, where

each component of $\boldsymbol{G}_k$ acts on component $\boldsymbol{x}_k.\boldsymbol{e}_a$ of particle k, and I_a is the identity function. The swarming equation (2) of section 3.2 can then be expressed as

$$\boldsymbol{q}_t = G_k \bullet U(\{\boldsymbol{p}_k(\tau)\}, c)S(t-1). \quad (14)$$

4.4. Self-organisation and interpretative swarms

It is worth considering whether particle swarms in interpretative interaction with external musical sources (this is the 'improvising system') might self-organise. According to Bonabeau *et al.* (1999), the necessary ingredients are positive and negative feedback, multiple interactions and the amplification of fluctuations.

The improvising system does have the possibility of positive feedback; attractors representing external events Y lead to swarming around $\boldsymbol{p}$, and a correlated output $X = Q(\boldsymbol{q})$. This reinforcement of events in the audio environment will grow if the external collaborators continue the loop. However, the external improvisers may be inclined not to perpetuate this musical direction, and indeed the inter-particle repulsion has a similar damping effect, producing swarm events which will be interpreted at some distance from the musical attractor. Hence the improvising system contains negative feedback. In addition, the repulsions and chaotic motion of the particles introduce fluctuations, which may be amplified by the external collaborators, or by the swarm itself through the inter-particle coupling to the swarm centroid. There are multiple direct interactions between the particles (equations (8) and (10)) and multiple indirect interactions between the particles and external events (mediated by stigmergetic attractor placement, equations (1) and (9)). Furthermore, colonies of swarms known as multi-swarms (described below) implement multiple stigmergetic internal interactions. These multiple interactions, in partnership with the other three ingredients, therefore have the potential to lead to the self-organisation of musical events and the spontaneous generation of structure.

5. APPLICATIONS

5.1. System overview

The working model of self-organising swarm-based improvisers has been explained in sections 3 and 4. This section explains how this theoretical scheme can be fleshed out to provide realisations of self-organising swarm-based improvisers. Figure 4 is a modular diagram for our artificial improviser.

One possibility for the implementation of F is the use of multi-swarms. A multi-swarm is a colony of particle swarm S_i interacting stigmergetically (Blackwell 2003). Particle positions from one swarm are added to the attractor buffers of the other swarms, where they join attractors emanating from the external audio environment. In order to provide different external behaviour, swarm events from each swarm can be interpreted using separate functions Q_i.

Each swarm can be thought of as a separate musical individual. In this case the Q_i would be very different,

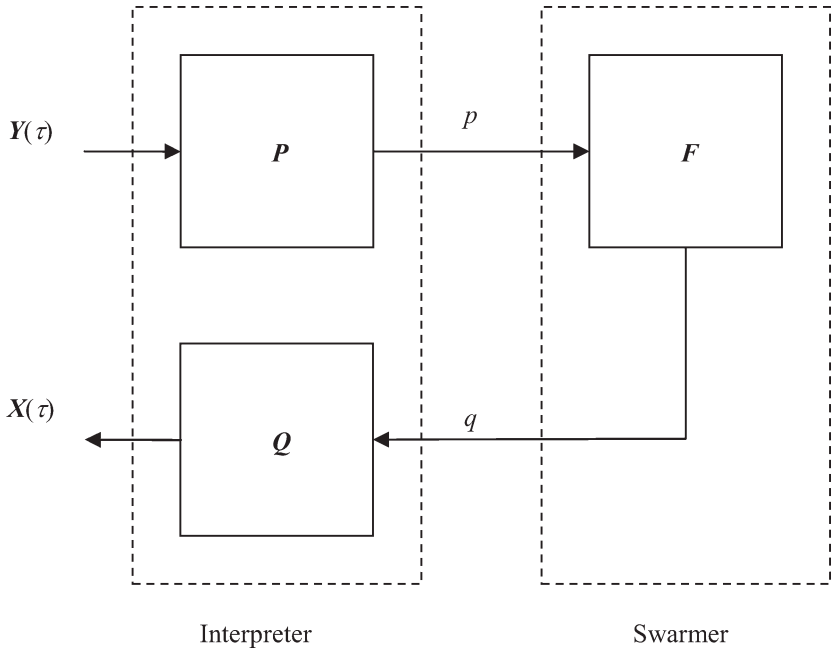

Figure 4. Modular view of the system. A separate software module, a swarmer and interpreter, is dedicated to each process (swarming and interpretation).

leading to quite distinct output streams X_i. This is almost equivalent to numerous versions of systems comprising a single swarm, except that inter-swarm attractor placement in the multi-swarm takes place *before* interpretation. The coupling between swarms in a multi-swarm is therefore stronger than between separate swarms because swarm events do not have to compete with audio events from other external systems or musicians.

Alternatively, the multi-swarm can be compared to a single musical personality. The swarm events q_i from each swarm can be interpreted and mixed by the Q_i before outputting a single stream X. This is useful when X can be conceptualised as the superposition of parameterisations,

$$Q = \sum Q_i, \quad X = \sum X_i. \tag{15}$$

Another design choice rests with the parameters c of equations (5)–(12). Although some simplifications have been made, there are still five free parameters which determine responsiveness ($v_{\max}$, m), diversity (particle charge, d_{core}) and awareness (d_{limit}). Additionally, these parameters can be different for different swarms in the multi-swarm, and for different directions in $\boldsymbol{R}^N$.

In general, the coordinate space of the swarmer itself has design implications. Apart from fixing the dimensionality of mappings P and Q, the extent of the coordinate space needs to be decided. Although it would be possible to allow particles to move without restrictions in $\boldsymbol{R}^N$, it is more manageable if motion is restricted to a finite subspace, $\boldsymbol{T}^N \subset \boldsymbol{R}^N$. Furthermore, the behaviour at the boundaries of $\boldsymbol{T}^N$ must be specified for any implementation. In the two applications discussed below, $\boldsymbol{T} = [0, 128]$ and particles simply reflect at the boundary of $\boldsymbol{T}$.

Further important choices have to be made for the interpretive functions P and Q which in many ways define the character of the system. Swarm Music, for example, is MIDI based; Q operates at the ministructural level, turning swarm events into MIDI events which are rendered by a synthesizer. Furthermore, Swarm Music is transparent: $P = Q^{-1}$. On the other hand, Q for the Swarm Granulator operates at the microstructural level, mapping swarm events to grain events which are rendered by a granular synthesizer.

It has previously been remarked in section 4.2 that $\boldsymbol{p}(\tau)$ can be delayed before being placed in $\boldsymbol{T}^N$. This delay, τ_{delay}, might be fixed, or might depend on the rate of flow of information into the system. The second option can be achieved by storing $\boldsymbol{p}$ in a queue of fixed size. When the queue is full, attractors are taken from the head of the queue and placed in $\boldsymbol{T}^N$ upon arrival of new attractors at the tail of the queue. τ_{delay} therefore depends on the length of the queue and the rate of flow of new attractors to the queue's tail. The queue is a simple implementation of a memory buffer, and this is important for stigmergetic interactions which are not instantaneous.

5.2. Swarm Music

The current version of Swarm Music is now six-dimensional. In previous versions, the interpreter parsed incoming audio/MIDI for interval between events (component **1**), event pitch (component **2**) and loudness (component **3**). The same parameters were used for interpretation back into audio, so the system operates purely at the mini-structural level. This version has been extended by adding event duration (i.e. $\delta\tau_{i,\ event}$, component **4**) and two meso-level parameters, chord number and sequence number.

Attractor component **5**, $\boldsymbol{p}.\boldsymbol{e}_5$, corresponds to the number of pitches sounding within a short (user-specified) time $\delta\tau_{chord}$. An attractor component is then placed in this dimension, and a swarm chord number at the start of each iteration is derived from the swarm centroid, $n_{chord} = \bar{\boldsymbol{x}}.\boldsymbol{e}_5$. n_{chord} can vary between 1 and M, the size of the swarm; if $n_{chord} > 1$, $\delta\tau$'s for the first n_{chord} swarm events are set to zero.

The number of simultaneous chord tones is a mesostructural property of the swarm as a whole. The sequence number $n_{sequence}$, which can vary between $-M$ and M, is another meso-level parameter and is determined by the component of the swarm centroid along this dimension, $\bar{\boldsymbol{x}}.\boldsymbol{e}_6$. The sequence number has the effect of forcing the first $|n_{sequence}|$ swarm events at each iteration particles to be sorted into ascending/descending pitches for $n_{sequence} > (<) 1$.

The interpreter extracts another two parameters. These are mode and tonic. To begin, the interpreter assumes a mode (major, minor, pentatonic, diminished, wholetone or chromatic) and tonic. Then, thirds and fifths of incoming MIDI events are inspected for agreement. A number of disagreements are tolerated before change is instigated to a mode that includes the errant note. A note histogram is also maintained to determine a tonic. Pitch interpretation of $\boldsymbol{q}$ is then dependent on the current mode and tonic. This simple algorithm allows for harmonic interaction and the results can be quite unpredictable, but emergent organisation is not produced by swarming. Instead, the modal and tonic parameters are passed directly to Q as parameters, $Q = Q(p_{tonic}, p_{mode})$. The actual pitch of the interpreted note from particle k is then determined by splitting the **2** axis into equal intervals based on the current parameterisation, and finding the interval corresponding to component $x_k.\boldsymbol{e}_2$. For example, if Q is parameterised in C Major, the **2** axis, which runs from 0 to 128, is split into intervals of width $\frac{12}{7} \approx 1.7$ so that middle C corresponds to components in the interval [60, 61.7], D to components in the interval [61.7, 63.4], and so on.

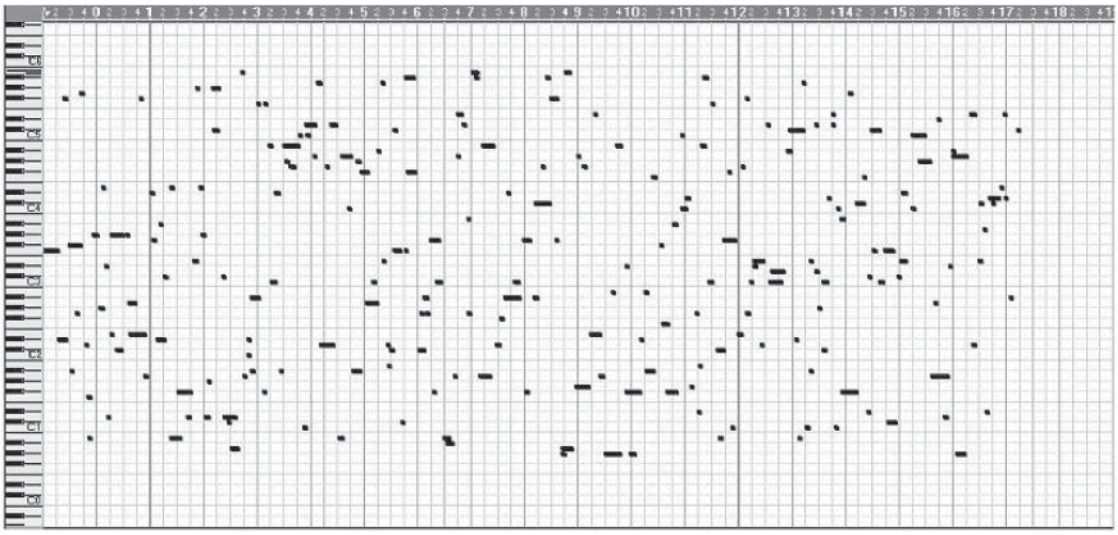

Figure 5. Particle positions, interpreted as MIDI events, for two five-particle swarms. The particles are randomly distributed at each update.

It was noted in section 4.2 that the current version of Swarm Music implements dimensional decoupling. This is preferable from a musical perspective since there is no *a priori* reason to correlate components. For example, if pitch and loudness components were correlated, then swarming around a new attractor that only differed from the previous attractor by a change in the pitch component would lead to a change in the dynamic output.

5.2.1. Swarm Music in performance

Swarm Music has performed with human improvisers at a number of events, including the Music and the Mind Festival, University College London, 2003 (Taylor 2003) and at The Big Blip Science/Arts Festival 2003 (Garland-Jones 2003) where a five-particle 2-swarm took part in a piano duet with one of the authors. Some performer feedback on early Swarm Music performances has been reported in Blackwell (2003). Young (as pianist at the Big Blip concert) writes: 'You were definitely aware of a response, and a performance loop emerging. Extremes of material seemed to work best – soft chords played slowly would soon change the kind of material coming from the swarm, after fast loud single lines for instance. The question was how to respond again – join in and reinforce the conditions the swarm 'wants', or always keep moving on to something new?'

The system can be autonomous in performance, but an operator may also intervene. The user interface allows real-time control of each parameter c, the size of $\boldsymbol{T}^N$ and the mode/tonic interpretation can be overridden. The effect on the swarm is not unlike the conducted improvisation techniques of John Zorn (Bailey 1992: 75–8) or Butch Morris' 'conduction' (http://www.conduction.us). Some Swarm Music improvisations, including conductions, can be downloaded from the website http://www.timblackwell.com.

5.2.2. Cooperation amongst swarms

The next three figures illustrate the cooperative effects between the two swarms in a 2-swarm. Each figure shows a Cubase edit screen. In these plots, time τ runs from left to right, and pitch vertically, so that the figures show three parameters of each event, $\delta\tau$, $\delta\tau_{event}$ and MIDI note number n_{MIDI}. Figure 5 shows the output of a five-particle 2-swarm, where the particles are repositioned at random within $\boldsymbol{T}^N$ at each particle update. Figure 5 shows that $\delta\tau$, $\delta\tau_{event}$ and n_{MIDI} vary uniformly within $\boldsymbol{T}^N$ over strips $(\tau, \tau + \Delta\tau)$. Figures 6 and 7, however, show plots for each swarm where the particles are interacting through equations (5)–(12). Correlations between the swarms can be easily seen in each dimension. Rather than randomness, the plots show that $\delta\tau$, $\delta\tau_{event}$ and n_{MIDI} occupy smaller regions of $\boldsymbol{T}^N$ for each strip. Two further features of figures 6 and 7 are worthy of comment. The swarms interact stigmergetically so that the positions of particles in Swarm A become attractor positions in Swarm B and *vice versa*. The result is that a movement of the note component of the centroids of each swarm show a similar pattern: over the first 4–5 time units there is centroid movement down in pitch, followed by a small fluctuation up and then downwards again. Then, for time units 5–18, the plots show joint movement up in pitch with both swarms descending again at $\tau = 12$. It is impossible to say which swarm provoked this cooperative behaviour, and which swarm followed.

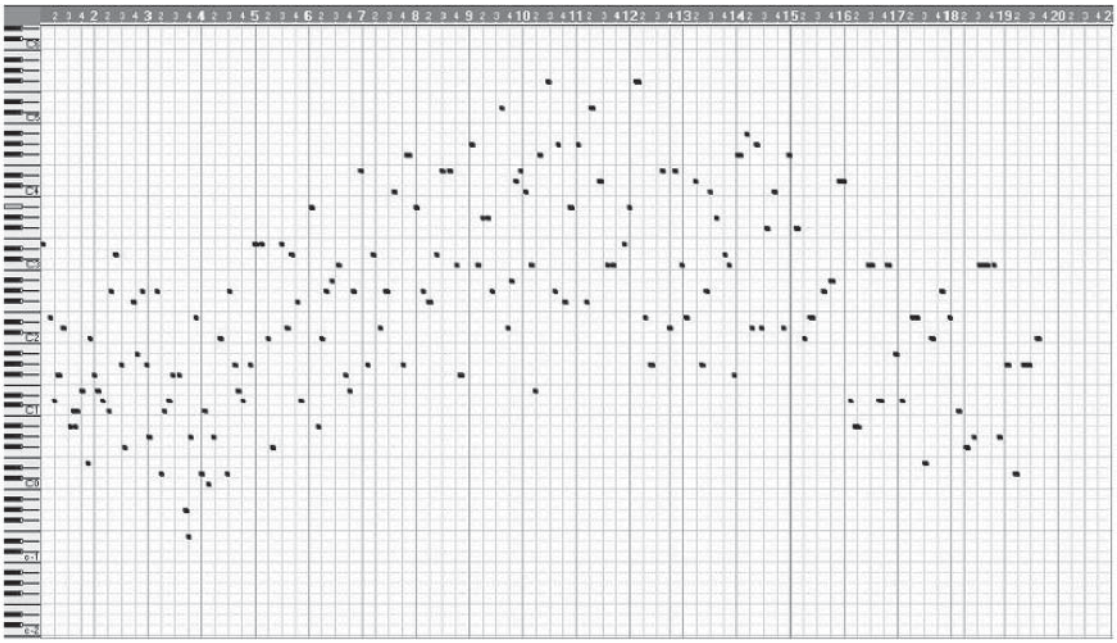

Figure 6. Particle positions, interpreted as MIDI events for one swarm (A) in an interacting 2-swarm. The swarms implement the full update equations (5)–(12).

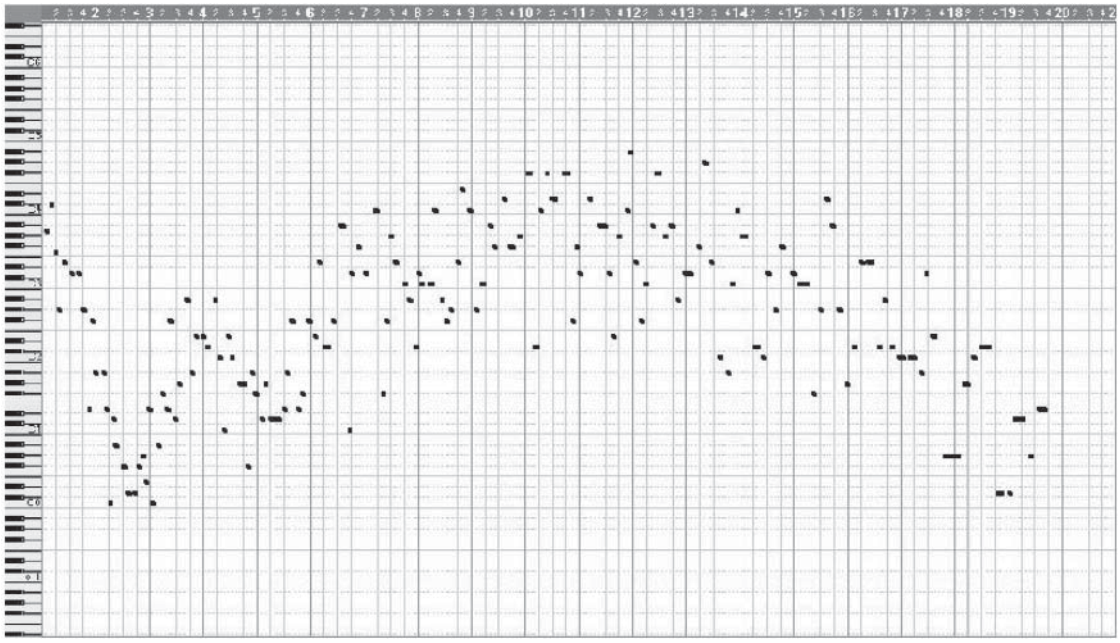

Figure 7. Particle positions, interpreted as MIDI events, for the second swarm of the interacting 2-swarm depicted in figure 6. The horizontal time axis is the same in both plots.

The 2-swarm is acting as a single entity where cause and effect are not helpful descriptors.

5.3. Swarm granulator

The overall system has three modules; interpreter, swarmer and granulator. In granulation, or granular synthesis, grains are generated by multiplying an envelope (window) of given amplitude, duration and shape with a waveform. Synthesis is achieved by iterating grains either synchronously or asynchronously. The result is a stream of sound with potentially very diverse timbral characteristics. Many grain-event level parameters affect these perceptual features; the

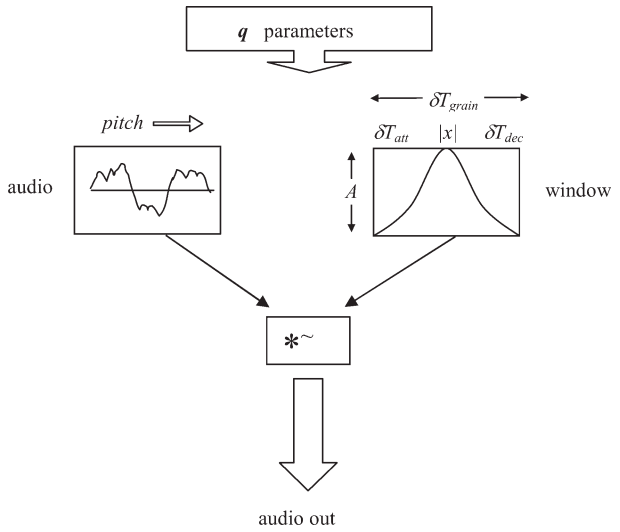

Figure 8. A modular view of the granulator, showing grain parameterisation from the swarm events $\boldsymbol{q}$.

various approaches to this technique are explored in detail by Roads (2001).

Our implementation uses a dual cross-platform system; interpretation and granulation on a 500 MHz Apple G4 and swarming, written in Java, runs on a 1.7 GHz PC. The two machines communicate using our own implementation of the Open Sound Control protocol for Ethernet communication (http://www.cnmat.berkeley.edu/OpenSoundControl/). Granulator and interpreter modules (see figure 8) are written in Max/MSP, with objects from the Granular Toolkit by Nathan Wolek (http://www.nathanwolek.com) and analysis objects including Miller Puckette's *fiddle~*.

Analysis P operates at the sound-event level (e.g. with *fiddle~*) and Q operates at the micro-(grain) level, and a transparent mapping is made from extracted parameters to grain parameters. Specifically, P extracts four sound-event parameters; pitch, amplitude, duration and duration between successive sound-events. Amplitude data is extracted continuously when events are detected. Q determines audio buffer transposition (=pitch), amplitude (A), duration (δT_{grain}), time between successive grains (δT) and grain attack and decay time (δT_{att} and δT_{dec}). The swarmer, therefore, operates in ($N = 6$)-dimensional coordinate space with the attractor components representing grain attack and decay times fixed.

An incoming sound stream from performers is repeatedly recorded into a *buffer~* object. Grains are produced by looking up the audio buffer and shaping the result with a Hanning window. The generation of grains is not periodic; their start and end points are controlled by swarm particles. This contrasts to conventional implementations of granular synthesis, such that $\delta T_{grain} = \delta T_{att} + \delta T_{dec} + x$, where x is unknown at the onset of the grain. The grain amplitude window is often distorted as a result, with an elongation at the loudest point. The audio buffer entry point is determined by the operator. Prior to this, the buffer contents are analysed such that only regions below an acceptable amplitude level are filtered out. This procedure moderates the somewhat unpredictable nature of this type of granular synthesis, more accurately described perhaps as 'granular reconstruction' (Wishart 1994).

The result is a stream of audio varying from sparse and irregular bursts to highly dense clouds of sonic material, slowly or rapidly evolving in pitch and amplitude range and in timbre. The characteristics of this are clearly dependent on the recorded audio. To widen the potential range of textural density, three simultaneous grain streams are used, i.e. the swarmer implements a 3-swarm.

Swarm Granulator has performed with three acoustic musicians at The Big Blip. For this event, each swarm had ten particles and, for additional texture, each swarm had different parameter settings c. Some accounts of the performance from the musicians' perspective are reported in Blackwell and Young (2004), and some excerpts for the concert are available on the website http://www.timblackwell.com

6. DISCUSSION

The inspiration behind the work presented here is the compelling analogy between self-organisation in nature and improvised music. We suggest here that the emergence of structure in improvised performance can be understood from the perspective of self-organisation. The paper outlines a model of how these ideas can be embodied in an artificial improviser. This interpretative model uses the idea of indirect environment-mediated interaction (stigmergy); attractors, which are parameterisations of external events, are placed in the environment of a particle swarm. The organised patterns of the swarm around the attractors provide parameters which modulate audio output. Two systems have been implemented, one based on swarming grains, and one on swarming MIDI events. These systems have been tested in performance where they have pro-actively engaged with human performers.

Both systems are still being developed. In particular, the interpretative functions, which operate at specific musical levels, are the subject of ongoing research. The ultimate aim is for a single-level, unified system which is transparent to both the performers and the audience. The ideal form of transparency might begin with the extraction and mapping of micro-level parameters (e.g. by wavelet analysis). Currently, parameters are extracted at mini- and meso-levels. There is a tantalising possibility that interpretation could take

place only at the smallest perceivable level, the micro-level, and that musical structure at every level upwards could arise through self-organisation.

Since organisation at higher and higher levels would be expected to take place with diminishing frequency, it could be that a hybrid multi-level approach is preferable. In which case, interpretation of dynamic elements such as timbral change, vibrato and attack/decay characteristics should be added to the static parameters so far implemented. Spectral characteristics of incoming audio can be explored, for instance by FFT analysis, enabling the automation and subsequent swarming of buffer entry points (which in turn determine the waveform used for grains).

Ultimately, the integration of Swarm Granulator and Swarm Music into a single system that listens, swarms, and modulates output at the micro-, mini- and meso-levels, could lead to a formidable artificial improviser.

REFERENCES

Arfib, D., Couturier, J. M., Kessous, L., and Verfaille, V. 2002. Strategies of mapping between gesture data and synthesis model parameters using perceptual spaces. *Organised Sound* **7**: 127–44.

Bailey, D. 1992. *Improvisation: its Nature and Practice in Music*. London, UK: The British Library Sound Archive.

Bastien, D. T., and Hostager, T. J. 1992. Cooperative as communicative accomplishment: a symbolic interaction analysis of an improvised jazz concert. *Communication Studies* **43**: 92–104.

Bentley, P. J., and Corne, D. W. 2001. *Creative Evolutionary Systems*. San Francisco, CA: Morgan Kaufmann.

Berliner, P. F. 1994. *Thinking in Jazz: the Infinite Art of Improvisation*. Chicago and London: The University of Chicago press.

Berry, W. 1976. *Structural Functions in Music.* Englewood Cliffs, NJ: Prentice-Hall.

Blackwell, T. M. 2001. *Making Music with Swarms*. M.Sc. thesis, University College London.

Blackwell, T. M. 2003. Swarm Music: improvised music with multi-swarms. In *Proc. of the 2003 AISB Symp. on Artificial Intelligence and Creativity in Arts and Science*, pp. 41–9.

Blackwell, T. M., and Bentley, P. J. 2002. Improvised music with swarms. In *Proc. of the 2002 Congr. on Evolutionary Computation*, pp. 1,462–7.

Blackwell, T. M., and Young, M. W. 2004. Swarm Granulator. Applications of Evolutionary Computing, *Proc. EvoWorkshops 2004 (LNCS 3005)*, pp. 399–408. Springer-Verlag.

Bonabeau, E., Dorigo, M., and Theraulaz, T. 1999. *From Natural to Artificial Swarm Intelligence*. New York: Oxford University Press.

Gabor, D. 1947. Acoustical quanta and the theory of hearing. *Nature* **159**: 591–4.

Gartland-Jones, A. 2003. http://www.atgj.org/TheBigBlip/

Grassé, P. 1959. La reconstruction du nid et les coordinations inter-individuelles chez Bellicosi-termes natalensis et Cubitermes sp. La theorie de la stigmergie: essai d'interpretation des termites constructeurs. *Insect Societies* **6**: 41–83.

Hunt, A., Wanderley, M., and Kirk, R. 2000. Towards a model for instrumental mapping in expert musical interaction. In *Proc. of the Int. Computer Music Conf., ICMC 2000*, pp. 97–118.

Katovitch, M. 1986. Temporary stages of situated activity and identity activation. In C. Couch, S. Saxton and M. Katovitch (eds.) *Studies in Symbolic Interaction: The Iowa School*. Greenwich, CT: JAI Press.

Kube, C. R., and Zhange, H. 1994. Collective robotics: from social insects to robots. *Adaptive Behavior* **2**: 189–218.

Meade, G. H. 1932. *The Philosophy of the Present*. Chicago: University of Chicago Press.

Resnick, M. 1997. *Turtles, Termites, and Traffic Jams: Explorations in Massively Parallel Microworlds*. Cambridge, MA: MIT Press.

Reynolds, C. 1987. Flocks, herds, and schools: A distributed behavioural model. *SIGGRAPH '87* **21**(4): 25–34.

Roads, C. 2001. *Microsound.* Cambridge, MA: MIT Press.

Sawyer, R. K. 2003. *Group Creativity: Music, Theater, Collaboration*. Mahwah, NJ: Lawrence Erlbaum Associates.

Silverstein, M. 1993. Metapragmatic discourse and metapragmatic function. In J. A. Lucy (ed.) *Reflexive Language*. New York: Cambridge University Press.

Smalley, D. 1992. The listening imagination: listening in the electroacoustic era. In J. Paynter *et al.* (eds.) *Companion to Contemporary Musical Thought*. London: Routledge.

Taylor, P. 2003. Music and the mind. *Musician*, June: 21.

Varèse, E. 1971. The liberation of sound. In C. Boretz and E. Cone (eds.) *Perspectives on American Composers.* New York: Norton.

Wishart, T. 1994. *Audible Design.* York, UK: Orpheus the pantomine Ltd.

Wishart, T. 1996. *On Sonic Art* (revised edition). Amsterdam: Harwood Academic.

Xenakis, I. 1981. Concerning time. *Perspectives of New Music* **27**: 84–92.