9
Swarming and Music

TIM BLACKWELL

9.1. Introduction

Music is a pattern of sounds in time. A swarm is a dynamic pattern of individuals in space. The structure of a musical composition is shaped in advance of the performance, but the organization of a swarm is emergent, without pre-planning. What use, therefore, might swarms have in music?

This chapter considers this question with a particular emphasis on swarms as performers, rather than composers. In *Swarm Music*, human improvizers interact with a music system that can listen, respond and generate new musical material. The novelty arises from the patterning of an artificial swarm. *Swarm Music* is a prototype of an autonomous, silicon-based improvizer that could, without human intervention, participate on equal terms with the musical activity of an improvizing group.

Real-life swarms organize themselves into remarkable, beautiful spatio-temporal structures in a process known as self-organization. This organization is thought to arise from the instantaneous dynamics of the swarming creatures, and not by any central leadership. Swarming animals communicate with each other over long time scales through the modification of the environment in a biological process known as stigmergy. This enables cooperative behaviour such as the construction of termite mounds, despite the absence of a termite architect. Digital swarms are the software equivalent of these remarkable biological systems. A virtual swarm may be visualized, but at a more abstract level, the swarm exists as a set of local rules, or interactions, between digital entities. These rules follow the theoretical models of biological swarms.

At the heart of the answer to the question posed above is a connection between self-organization and structural levels in music, a link that suggests many possibilities for the design of creative systems. This chapter begins therefore with an account of self-organization and swarming, and develops the link to structural levels in music in Section 9.3.

Synthetic swarms, by virtue of the unpredictability of their patterning are ideally suited to improvization, and the remainder of the chapter concentrates on swarms as performing systems. The real-time interaction between people and swarms

is enabled with an analogue of stigmergy. A three component model outlines the interactions we might have with a virtual swarm, and by extension with any evolutionary algorithm. An analysis component maps external musical information into objects in the environment of the swarm. A stigmergetic interaction between swarming individuals and these objects takes place. The dynamic interactions within the swarm are described by the second component, the swarming function. The interpretation of swarming patterns into sounds is accomplished by the third component. Section 9.4 outlines the complete framework.

Section 9.5 considers the instantiation of the interactive model in the *Swarm Music* family of improvizers, and discusses the motivation for design. The following section considers live aspects of *Swarm Music*. Other performance systems that use a swarm algorithm are also summarized. Section 9.7 illustrates, by reference to system development in *Swarm Music*, a general scheme for increasing autonomy in music systems. The chapter ends with a look to the future of Swarming and Music.

9.2. Swarm Organization

9.2.1. The Science of Emergence

Self-organization (SO), the science of emergence, can, as yet, only allude to the preconditions for the emergence of large scale forms from local influences. Bonabeau and colleagues (1999) propose that SO relies on multiple interactions between component parts of a system, an ability to amplify fluctuations, and positive and negative feedback between components. Positive feedback forms the basis of morphogenesis, allowing reinforcement of new forms. Negative feedback stabilizes the system and prevents runaway. Random fluctuations play a crucial role in SO, enabling the system to find novel situations, which are exploitable through positive feedback.

The paradigmatic example of SO is the collective behaviour of social insects, for example the organization of army ants in vast foraging patterns (Burton and Franks 1985). The raid patterns of army ants contains hundreds of thousands of virtually blind individuals, a remarkable example of decentralized control (Bonabeau et al. 1999, p. 36). Recruitment to a food source through trail laying and trail reinforcement is an example of positive feedback, with stability arising from the limited numbers of foragers and the exhaustion of the food source. Random fluctuations arise in foragers through error; the occasional wayward ant who has lost a trail might find a new food source. Communication between ants, although it can take place through direct contact, is also mediated indirectly via the environment by the laying of pheromone trails. Individuals are able to exploit this information network, for example by following a trail that leads to a newly discovered food source. Although an individual can interact with its own trail, SO usually requires a minimum density of individuals who are intent on exploiting the network. The indirect and temporally adjusted environment mediated interaction is termed *stigmergy* (Grassé 1959). In a sense, stigmergy happens to humans all the time. A note

left on the kitchen table is an indirect interaction between people, influencing our actions several hours later.

Swarms, flocks, herds and shoals are familiar examples of the groupings of social animals. The organization of Atlantic herring into very huge shoals up to seventeen miles long, and with many millions of fish is a stunning example (Shaw 1975). This is particularly remarkable because it is unlikely that an individual herring can, in the murky Atlantic water and tightly packed shoal, see more than a few of its neighbours. The possibility of a leader herring coordinating this shoal is absurd, and besides, how would it orchestrate the shoal movements? It seems likely, therefore, that the shoal is an emergent entity, produced by local, de-centralized interactions.

9.2.2. Artificial Swarms

Evidence that flocks and swarms are self-organising is provided by the 'boid' animations of Reynolds (1987). The centralized approach to animations of particle systems (bees in a swarm, buffalo in a herd) is to formulate the collective behaviour as a script which each entity must obey. Swarming behaviour is not emergent because it is built into the script from the outset. However, Reynold's discovery that convincing animations can result from local, de-centralized rules has done much to support the hypothesis that swarms and flocks are self-organizing. The collective behaviour of the group is emergent because the rules concerning the parts of the swarm do not contain any notion of the whole. Additionally, de-centralization explains the scalability of natural swarms. The variation of swarm sizes over six orders of magnitude suggests that swarms must have linear complexity. Early examples of behavioural animations using the boids algorithm include bat swarms and penguin flocks in the film Batman Returns (Burton 1992) and the wildebeest stamped in The Lion King (Allers and Minkoff 1994).

Contemporary swarm algorithms follow this basic principle and can be split into three groups, although there are overlaps. The grouping is in order of faithfulness to natural swarms:

1. Bio-swarms, the most faithful, are used to develop scientific models of natural systems (for example the refined bio-swarm of Couzin et al. 2005). These swarms may be visualized, but the chief purpose is hypothesis development and testing.
2. Simulation swarms are visualizations for aesthetic and artistic purposes and do not need to accurately represent nature (Reynolds 1987; Burton 1992; Allers and Minkoff 1994). We can include musical swarms such as *Swarm Music* in this category. These swarms move in real time so that the visualisations have a sense of realism.
3. Social swarms use an information network rather than a spatial region to define a neighbourhood for interactions. Social swarms are frequently used to solve mathematical problems, as in ant colony optimization (Bonabeau et al. 1999) and particle swarm optimization (PSO) (Kennedy et al. 2001).

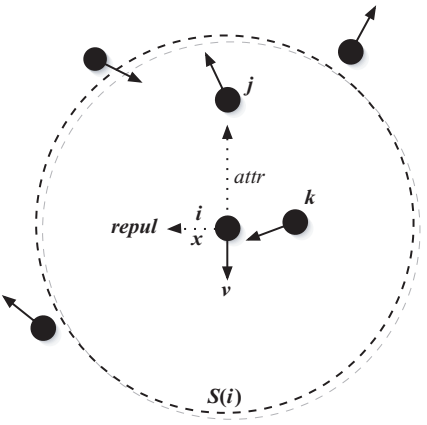

FIGURE 9.1. Swarming rules. Particle i, currently at x and moving with velocity v, is attracted to particle j and repelled from particle k. The other particles are outside i's perception, $S(i)$.

These swarms have the loosest connection to nature: the visualizations take secondary importance to the algorithmic details and in fact they can look quite unrealistic.

Swarms which use a spatial neighbourhood typically assume that the individuals have a finite range of perception in which a given individual feels the influence of neighbours. Typically, individuals repel each other at close range, attract each other at medium range and are oblivious to each other at long range (Fig. 9.1). The attractions provide coherence, maintaining a shared neighbourhood (which may be a sub-swarm or the entire swarm) and the repulsions prevent collisions. Figure 9.1 illustrates the idea. The attractive and repulsive accelerations are the analogues of positive and negative feedback. At its simplest, a swarm algorithm considers the individual swarming participants as purely dynamic entities. These entities are represented as point particles in d-dimensional real space with dynamic state (x, v). The basic rules governing the interactions between neighbouring particles in a swarm or flock are:

1. if apart, move closer (cohesion)
2. if too close move apart (separation)
3. attempt to match velocities (alignment)

The final rule only applies collectives where there the entities move in unison, such as flocks, herds and schools. Swarming entities have more chaotic motions and drop the rule of alignment.

The dynamical update equations of swarm algorithms are discretizations of Isaac Newton's laws. The update of particle i of swarm S is

$$a_i = \frac{1}{m} f(S(i),\ \alpha) \tag{9.1}$$

$$v_i(t+1) = v_i(t) + a_i \tag{9.2}$$

$$v_i(t+1) = min(v_i(t),\ v_{\max}) \tag{9.3}$$

$$x_i(t+1) = x_i(t) + v_i(t+1) \tag{9.4}$$

where the time increment $dt = 1$ and $S(i)$ is the sub-swarm comprised of i and its neighbours. The rules 1–3 above are embodied in the particle accelerations a_i. These accelerations are computed by a force law f, which is a function of dynamic variables, neighbourhood $S(i)$, and parameters α. The mass m of the particle is usually set to unity, and the physics terms 'force' and 'acceleration' are synonymous in this context. The acceleration parameters characterise the strength of the intra-particle forces and the construction of $S(i)$, for example by specifying a radius of perception (bio and simulation swarms), or a network topology (social swarms).

Equation (9.3) is an optional speed clamp that can be used to limit particle velocity in the case of high accelerations. Some swarm implementations, especially bio and simulation swarms, use a swarming function, Eq. (9.1), that produces accelerations of fixed magnitude and clamping is never necessary. These 'steering' accelerations cause the velocity vector to rotate, and do not cause changes in speed. For example, the attraction of a particle at x_i towards a neighbouring particle at x_j might be a steering acceleration,

$$a_i = \frac{x_j - x_i}{|x_j - x_i|} \tag{9.5}$$

The calculation of a_i in Eq. (9.1) consists of a sum of attractive and repulsive terms. Particles perceive each other and other attractors with a region of perception. At long distances, particles attract, but at shorter distances repulsion will dominate. Bio-swarms use three concentric zones; the rule of cohesion applies in the outer zone, alignment applies in a middle zone and at short distances the rule of separation dominates (Couzin et al. 2002). Individuals in simulation and bio-swarms may also have a 'blind volume' in which neighbours are undetectable.

Social swarms employ an information network that is topological rather than spatial. Additionally, the particles possess a memory and so are more than merely dynamic entities. The accelerations in PSO are not constant magnitude steering vectors but are spring-like,

$$a_i = C(p_i - x_i) \tag{9.6}$$

where C is a spring constant and p_i is a good location previously visited by particle i, or by any other particle in i's topological neighbourhood. Convergence, and the stabilization of the swarm within a search space, occur through energy loss and the particle displacements become progressively smaller and the search intensifies. This energy loss is invoked by a frictional drag force. The attraction of a particle to a previous best position can be viewed as a stigmergetic interaction. Particles leave behind markers p_i at promising locations, and the markers are available to any other particle in the social network, irrespective of distance.

The music swarms that will be discussed in this chapter, employ elements of simulation and social swarms. *Swarm Music* and *Swarm Granulator* use spatial neighbourhoods and spring or steering accelerations. The particles in *Swarm Techtiles* communicate stigmergetically by depositing markers at a highly textured region of

an image. The neighbourhood is again spatial although the rule for interpretation of each particle in terms of musical parameters is social in origin.

In summary, simulation swarms and visualisations of social and bio-swarms reveal self-organizational properties: the swarm as a whole has a spatial identity with globally connected neighbourhoods, the swarm can act as a single entity (spontaneous movement of every particle in an arbitrary direction defined, for example, by a breakaway particle) and the formation of spatially separate subswarms that may later merge. The swarm rules are simple to implement—considerably simpler than trying to write top–down rules—and the behaviour does not depend on fine tuning of the acceleration parameters. The emergent organisation at the swarm level fits with the premises of SO since the algorithm incorporates positive feedback (coherence), negative feedback (separation) and complexity (many particles, stigmergetic effects, blind volumes, etc.).

9.3. Swarming and Descriptions of Music

This section establishes the link between swarming, SO and *descriptions* of music. We distinguish here the formal, music–theoretic description of music as notes, metre, dynamics, harmony, etc. and the performance itself, which is an inter-musician exchange of sonic events. The following section considers the relationship between swarming, stigmergy and the *performance* of music.

9.3.1. Levels of Description

From a music–theoretic perspective, music is commonly analysed hierarchically. For example, a work of (classical) Western art music is usually thought of as the organization of melodies, which themselves are built from phrases. The phrases are comprised of individual notes, and the whole structure is bound together by rhythm and metre. A classification loosely based on perceptual time-scales can be summarized, with suggested time-scales, (Xenakis 1989; Roads 2001);

1. Micro. This scale extends from the limit of timbre perception (tenths of a millisecond, Gabor 1947) up to the duration of notes or other sound objects.
2. Mini (note). This level includes notes and any other sound from a known or even unidentifiable source (sound objects, Schaeffer 1959) of duration tenths of a second to several seconds.
3. Meso (phrase). This level corresponds to phrases or groups of mini-events and occupies several to dozens of seconds. Melodic, contrapuntal and rhythmic relationships between objects are noticeable at this level.
4. Macro. This longer lasting duration of time encompasses form and lasts several minutes or more. Corresponding to the architecture of a composition or improvization, this level is perceived either through recollection or by knowledge of a particular macro-structure (for example, knowledge that a piece is written in sonata form).

Digital music also includes an imperceivable sample level of sound, ranging from a single digital sample at hundredths of a millisecond, up to the shortest timbred-sound. Clearly such schemes are not unambiguous, and arguably over-confine music to a rigid structure that is subservient to notation (Wishart 1966). However the analysis by levels is useful for our purpose here, which is to establish how swarming might relate to music.

9.3.2. Swarming

Imagine, rather whimsically, an abstract note-to-be as some kind of autonomous individual, able to wander at will in a 'music parameter space'. This space might be a score, or some other abstract space of musical dimensions. As it moves through this space, its characteristics—pitch, loudness, duration and onset time—will change. The note-to-be does not wander aimlessly, however; it is attracted to other note individuals, and soon groupings of notes form. Notes avoid collisions and sometimes dart away from the group. Other groupings are formed in distant regions of music space; sometimes groups collide and unite.

These swarms of melody are composed of notes that *do not know they are part of a tun*e. The notes have not been placed by a higher level imperative; rather, melody is an emergent property of the note-swarm, related to the self-organized pattern of the swarming individuals (Blackwell 2001). Collision avoidance between notes mitigates against too much repetition, which is balanced by an inter-note attraction which prevents too much variation. Observation of composed melodies shows that they occupy constrained regions of music parameter space, frequently moving step wise, suggesting a strong tensile force between notes, and with leaps for excitement, as produced, in our analogy, by random fluctuations. Examples of melodic movement are to be found in many books on composition, for example Sturman (1983).

Swarming can be also be inferred from the harmonic principles of consonance and dissonance (Piston 1978), endemic in the common practise of Western art music, and in contemporary popular music. Harmony can be simplistically viewed as an attraction towards the consonant musical intervals. Dissonance can occur, but the result of such a collision is a relaxation back to consonance.

Rhythmically too, we can discover the same forces; an attraction of note onsets to the subdivisions of the beat, and a repulsion away from non-metricity (unless the music is deliberately rubato, in which case the opposite rule applies).

An analogy has been suggested between musical organization at the note level, but similar principles can be construed at the meso level where a phrase may be considered as a 'unit of musical thought, like a sentence or a clause' (Piston 1978, p. 93), or at the macro level where groups of phrases produce sectional structuring, as in the exposition, development, recapitulation and coda sections of the classical sonata form, or the AABA structure of popular songs. These principles might also be applied at the micro or sample levels (Blackwell and Young 2004a, b; Blackwell and Jefferies 2005).

At each level we notice a tension between repetition and variation, a force for similarity (positive feedback) that is balanced by a repulsion (negative feedback) away from sameness. Too much similarity is boring for the listener, and too much variation can imbue the music with a feeling of disorganization (Coker 1986, p. 15). The idea from emergence is that structure at level n can arise from local interactions at level $n - 1$ and need not be enforced by top–down pressure. SO provides an appealing picture for the creation of novelty through random exploration and reinforcement, and the relationship between positive and negative feedback is compatible with our psychological expectations of music. These arguments suggest a different view of musical organization, complimenting the traditional syntactical, top-down description.

As we have seen, swarming particles move in a d-dimensional real space with a swarming algorithm f that moves the particles forward in time. Swarming patterns can be *interpreted* musically as a succession of musical/sonic events. In this picture, music is regarded as a temporal structure of meaningless level-dependent entities, since the rules governing the interactions do not derive from musical concerns. Meaning itself can only emerge, and is only apparent at, the next highest level.

9.4. Performing Swarms

9.4.1. Interactive Model

This section considers the performative, rather than the descriptive, aspects of music and self-organization. Music performance, in contradistinction to the structural analysis of music, is highly interactive and uncertain. Whether rehearsed or extemporized, unknowable features of performance enter through the unpredictability of individual interpretation, audience involvement, acoustics and other external factors. This section describes a model of performance that encompasses current computer music practise and is well suited for the development of new evolutionary and swarm-based music systems.

Improvised music is highly interactive and is the best exemplar of the parallels between performance and SO. A performance of freely improvized music is distinguished from jazz (which includes improvization within a pre-defined structure) and other compositional genres by the lack of advance planning. There is no leader, no rehearsal, no score and no written instructions. Musicians simply assemble on stage and begin playing their instruments. All musical directions, cues, initiatives and roles are therefore communicated by musical utterances, and by body language. Surprisingly, this de-centralized, potentially lawless, style of music making can produce remarkably well formed improvizations. In other words, spontaneous improvizations are capable of structuring at the macro level; the emergence of form is a consequence of the temporally local interactions between performers.

An examination of group dynamics in the light of the ingredients of self-organization—positive and negative feedback, amplification of fluctuations and

complex interactions—is revealing. There is a human tendency to conform. If the direction of an improvization is towards increasing excitement (for example by playing louder, faster and with more dissonance), there is a strong compulsion to join in and reinforce this flow. In dynamical terms, this can be regarded as an attraction towards a gestural, emotional target. This positive feedback is counterbalanced by a personal desire to innovate. In the language of dynamic systems, the musical target or attractor has a repulsive force that deflects away from exact repetition. Improvisations can include sudden changes in mood and musical direction, as if orchestrated. Dynamically, a small fluctuation caused by a random exploration can precipitate a movement by the whole group and the proto-idea is amplified. The unique constitution of the performing group and the non-linearity of the abstract performance space provides uncertain, complex, non-linear interactions. It seems therefore that a group performance has the potential to be self-organizing.

Swarms are, as we have seen, self-organizing, and might therefore implement these ideas. However, for the analogy between SO and improvization to be practically useful, the relationship between the performing group, and a computer music system running a virtual swarm, must be fleshed out. One approach is to model each individual as a particle. However particles in a swarm move in a shared space, and it is very hard to see how to define this space without giving the musicians (and the computer) precise instructions about how to interact and move. Although there is some precedent for this approach in dance (Turner 2006), this scheme is in conflict with a musicians' own perspective on what it is to improvize. Rule specification, after all, is a compositional rather than an improvizational device.

Instead, each individual carries with her/him a unique representation of music and of sound events. This representation is a product of aesthetics, experience, training, temperament and many other factors. He/she might 'hear' a sound event in a different way: as a C#, as a squeal, as the fourth note in a sequence, as angry, etc., or indeed in many of these at the same time. Ideas, as expressed in this space, evolve until an intention is formed, and new sound output produced. The representations are personal, hidden even; fellow musicians can only access external sound events, and possibly infer intention from visual cues.

The solution adopted in *Swarm Music* (Blackwell 2001) mirrors this informal account. Each individual is regarded as a sub-swarm rather than a particle. The sub-swarms move in secret, hidden spaces; external sound events are parameterized as objects in the environment of each sub-swarm. Interaction between sub-swarms is now possible through a stigmergetic mechanism. Events at micro, mini and meso levels are parameterized according to the internal representations available to any individual. These parameterizations constitute 'sound objects' which populate the internal spaces of each individual, whether human or machine. To the participant, these objects act rather like messages, influencing stigmergetically the flow of one's own internal states. Collaboration and self-organization between the sub-swarms can still happen, but unlike natural systems, each subswarm/individual moves in a distinct space, Figure 9.2.

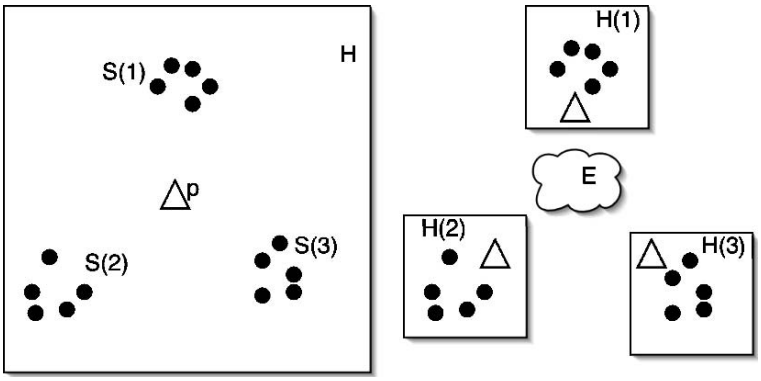

FIGURE 9.2. In this diagram, particles are blobs and attractors are triangles. The left diagram shows three sub-swarms $S(1-3)$ swarming in a space H around an attractor p. The right hand diagram depicts the interactive model. Here the sub-swarms move in separate spaces $H(1-3)$. Each space is replete with an image of the sound object, E.

9.4.2. Live Algorithms

The model of performance as a self-organizing system suggests ways that machines might interact autonomously, rather merely automatically or manually, with people. Autonomy implies that an interacting system can support group activity, as well as introduce novel elements, and all without the presence of an operator. The model sketched in the preceding section suggests that internal state flow, as generated by a swarm simulation, can act as an 'ideas generator'. Interaction with the real world is effected by forming an image, as an attractor for example, of external events in the state space of the system. This image informs, but does not govern, state flow. State flow, and hence output, is not contingent on input: the system is capable of making contributions in periods when the group is silent and is capable of silence when the group is active. Self-organization around attractors is a supportive activity and the amplification of spontaneous fluctuations away from an attractor gives rise to novelty.

The idea that interaction involves state change rather than parameter selection is an important aspect in the design of 'live algorithms' (Blackwell and Young 2005). A live algorithm is an autonomous music system capable of human-compatible performance. Several live algorithms have been developed; the Voyager system of Lewis (2000), Al Biles' GenJam (2006) and Francois Pachet's Continuator (Pachet 2004) are notable examples. Many issues surrounding machine interaction are covered in Rowe (2004). The proposed architecture for live algorithms builds on the interactive model of Section 9.4.1. A major advantage of the interactive model is that knowledge of collaborators' internal states are not necessary. This circumvents the difficulty of modelling, in a live algorithm, human intentionality and lessons the problems humans might have in interacting with an algorithm whose logical process depart greatly from human experience.

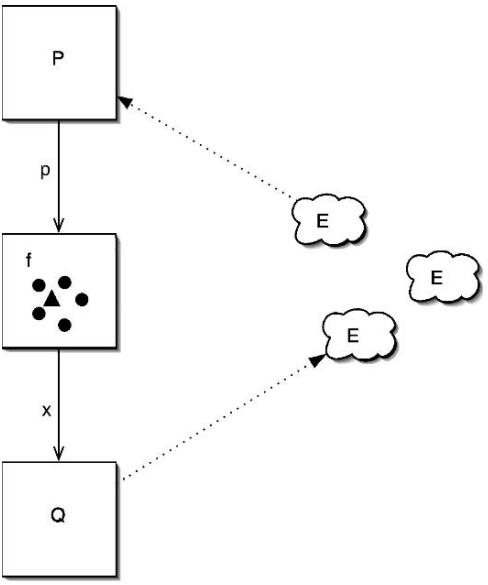

FIGURE 9.3. Modular structure of a live algorithm showing analysis (P), synthesis (Q) and patterning (f) modules. In this figure, a swarm provides spatio-temporal patterns as it self-organizes around an attractor (triangle). Q converts swarming configurations into musical patterns E.

A modular structure for live algorithms has been proposed by this author and Young (2004b, 2005). This architecture is shown is shown in Figure 9.3. External sound objects E are parameterized as internal images p by an interpretative, analytical module P. P corresponds to our ability to interpret incoming sound in terms of internal representations. A patterning, ideas engine f transforms internal states x in an internal space H. This module represents the restless flow of ideas that an improvizer might have, ideas that are guided, but not determined by, inputs p. Many possible choices of patterners f exist, including neural networks, evolutionary algorithms and swarm simulations. A third module, Q, re-interprets internal x as external sound. This involves a mapping onto synthesizer controls q. Q is a synthesis module, for example a MIDI sythnesizer or a granular synthesizer, and represents the conversion of volition into action. This architecture is general enough to subsume contemporary computer music practices such as (manual) live electronics and live coding and the automated process of algorithmic/generative music (Blackwell and Young 2005).

Since interaction with internal states can only occur if the state space contains an image of the environment, and participation with the environment can only happen if system state is mapped to sound, the live algorithm architecture is minimal. Systems of arbitrary complexity can be built by layering and cross-wiring between modules. However, all interactive systems (where interaction is defined as state change) must reduce to this PQf architecture. Since analysis (P), synthesis (Q) and generative (f) algorithms are individually the subject of much current research,

it is hoped that much progress in live algorithm research can be made by connecting pre-existing units.

9.4.3. Autonomy

The swarming function f can be written as

$$x(t+1) = f(x(t),\ v(t),\ p(t),\ \alpha) \tag{9.7}$$

where $\{x,\ v\}$ are dynamic variables, $p = P(E)$ is the image of the environment and α is a list of undetermined parameters, for example maximum velocity, spring constants and radius of perception. The α's can be thought of as controls, presets or algorithmic constants. They can be adjusted in real time by an operator as in the practices of live electronics and live coding. Potentially the α's, along with the choice of representation, will have a huge affect on the musicality of the system, governing many features of the output. It is important to distinguish system characteristics from autonomy. Live algorithms, just like humans, may be quite idiosyncratic, and this would be an advantage in an improvized context, but this need not affect their ability to interact. The α's might be interdependent, $\alpha_1 = \alpha_1(\alpha_2,\ \alpha_3, \ldots)$ and/or contextual $\alpha = \alpha(x,\ v,\ p)$ and often the α's are descriptions at the next higher musical level. The challenge for the designer of an autonomous system is to find a self-regulating, contextual condition for each undetermined parameter α_j so that the system is flexible, adaptable to the musical context and does not require any tuning by hand. One solution for determining an α and increasing system autonomy in *Swarm Music* is presented in Section 9.7.

9.4.4. Visualizing the Algorithm

Figure 9.3 does not depict a feed-through system. The arrows show direction of parameter flow, not ordering, and each module is intended to operate concurrently. The state flow $x(t) \to x(t+1)$ can be run as a simulation, i.e. a visualization shows entities moving at realistic speeds. A visualization serves as an embodiment of the algorithm, and gives clues on system behaviour to participating musicians (and to the audience). This visualization will only be useful to us if it proceeds at a comprehensible pace, and does not include too much information. In a sense, the visualization aids overall transparency of the system; visual cues are important for person–person interaction, and their value cannot be underestimated in machine–human interaction too.

The requirement that the algorithm is running a simulation of a real, or an imagined, natural system means that the update loop must contain a sleep function that links the iterative time t to real time τ. For example, the desired velocity of the particle across the screen is a function of the clamping velocity, $v_{\max}$, and the nominal update time interval $\Delta\tau$. A sleep function can halt the update loop at each iteration in order to preserve $\Delta\tau$ and ensure that states move at a fixed speed. Without such a consideration, the algorithm will run as fast as a CPU will allow, tying the algorithm to a particular machine, and making behaviour inconsistent.

9.5. Swarm Music

9.5.1. Overview of Live Algorithms Based on Swarming

The interactive model of Section 9.4.1 and the live algorithms architecture of Section 9.4.2 has been implemented in three systems, *Swarm Music* (Blackwell and Bentley 2002), *Swarm Granulator* (Blackwell and Young 2004a) and *Swarm Techtiles* (Blackwell and Jefferies 2005). In each case, the internal states x are particle positions in a swarm and f is the swarming function, Eq. (9.7). The systems differ, however, in representational levels and on the interpretation of the internal space H.

The space in *Swarm Music* is spanned by parameters salient at mini (note) and meso (phrase) levels. *Swarm Granulator* has an internal representation at the micro (granular) and *Swarm Techtiles* operates at the sample and micro level. In both *Swarm Music* and *Swarm Granulator*, attractors p are parameterizations of the input stream and are placed directly in an otherwise featureless H. Swarm particles are drawn towards any attractors in their zone of perception, and particle positions are interpreted one by one as synthesizer parameters. The flow of the swarm through H therefore corresponds to a melody (*Swarm Music*), or a stream of texture (*Swarm Granulator*).

Swarm Techtiles uses elements from social and simulation swarms and operates between sample and micro-levels. Particles fly over a landscape of 'woven sound' (a warp-weft mapping of incoming samples onto pixels), searching for optimum regions of local texture. Particles communicate stigmergetically by leaving markers at regions of high image texture, and produce sonic improvizations by unweaving small image tiles into sound. Swarm Granulator and Swarm Techtiles are described in detail in a review of swarm granulation (Blackwell, forthcoming).

9.5.2. Interpretation

Swarm Music has developed from a four to a seven dimensional system. Four dimensions are occupied by mini (note) level parameters and the other three dimensions correspond to phrase level parameterizations. A screen shot from *Swarm Music*, Fig. 9.4, shows the first three dimensions of an N-particle swarm.

The listening module, P can receive either audio or MIDI. Digital audio is converted into MIDI messages by an inbuilt event and pitch detector which relates average event energy in decibels to MIDI 'velocity', and the dominant frequency of a fast Fourier spectrum to MIDI note number (middle C = 60, C# = 61 etc.). Otherwise, a MIDI source is plugged directly into P.

P extracts note loudness a and pitch f from the MIDI message. Additionally, P keeps track of five other features. All seven axes are specified in Table 9.1. Axis seven has only recently been incorporated in *Swarm Music* and is reported here for the first time. These features become the seven components of the attractor p. There are as many attractors as there are particles, and attractors are replaced in turn, so the system only as a memory of the last N events (this constitutes a

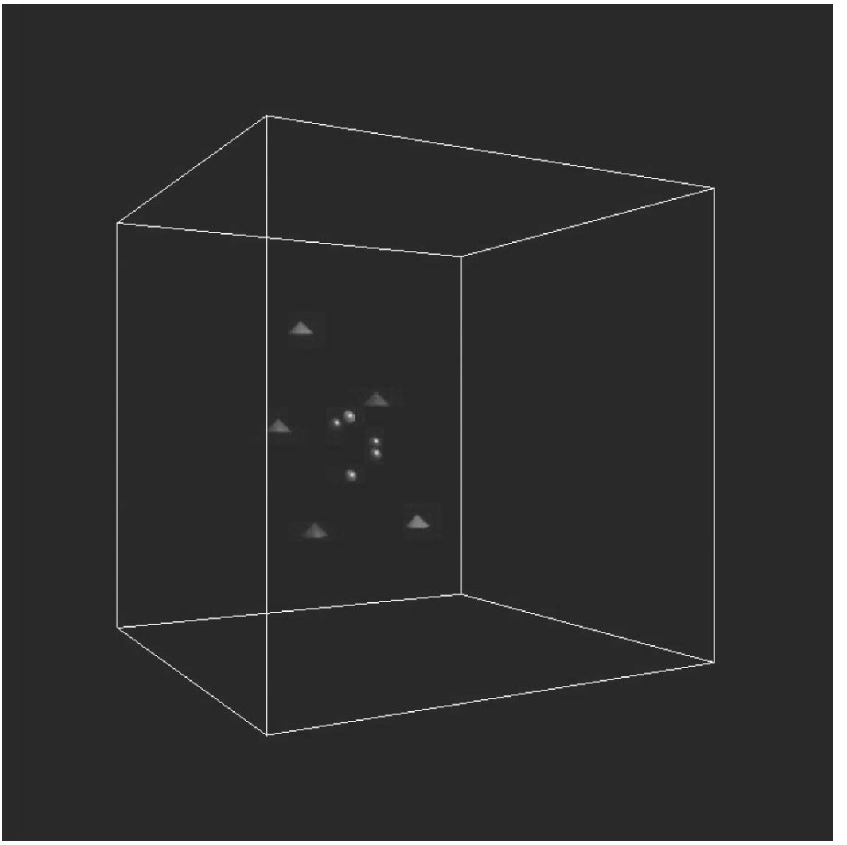

FIGURE 9.4. A five particle swarm. Particles are depicted as spheres and attractors as cones. The mappings into the three dimensions of this visualization are: loudness → out-of-page; onset time interval → left–right and pitch→ up–down

single phrase in *Swarm Music*) that it has heard. The attractors, which act like pheromones to the swarm, rapidly evaporate.

Apart from the four note-level axes, 1–4, *Swarm Music* incorporates three phrase-level dimensions, allowing for swarming in a subspace of phrase param-

TABLE 9.1. The seven dimensions of *Swarm Music*

Axis	Description	Symbol
1	Event energy/note loudness	a
2	Time interval between events	Δt
3	Event pitch	f
4	Time duration of events	Δt_{event}
5	Number of simultaneous events in a phrase	n_{chord}
6	Number of ascending or descending pitches in a phrase	n_{seq}
7	Similarity between successive phrases	s

eterizations. The fifth axis is chord number. Each incoming phrase is examined for the number of coincident, or near coincident, events and this number becomes the fifth component, p_5 of the new attractor. The sixth dimension is the number of consecutive ascending note-numbers (ranging from $-N$ to $+N$, with negative values indicating descending sequences) over the phrase. The seventh dimension represents the similarity of two adjacent phrases with a similarity measure. The similarity s is a value in the unit interval with $s = 1$ for a perfect N note match (by note number only) between the last two phrases. A similarity of zero means that there were no matches.

The swarm has N particles and these are interpreted, by the synthesis module Q, as a set S_N of N notes. Each note is described by four parameter 1–4, $\{a,\ \Delta t,\ f,\ \Delta t_{\text{event}}\}$. The loudness a of each note in S_N is determined by the first component, x_1 of each particle's position. Onset time interval (in the absence of chords) between notes, pitch and note duration correspond to components x_{2-4}.

Phrase descriptions are the properties of a group of notes and not of an individual. Similarly, the phrase descriptions for S_N must be a property of the swarm as a whole. The swarm centre of mass,

$$\bar{x} = \sum_{all\ particles} x \tag{9.8}$$

is a convenient measure of swarm configuration. Q uses components $\bar{x}_{5-7}$ of the centre of mass to modify the phrase S_N. If the chord number, $n_{\text{chord}} = \bar{x}_5$ is larger than 1, then the Δt's of the first n_{chord} notes of S_N are set to zero. This will ensure that they will sound simultaneously. The first $n_{\text{seq}} = \bar{x}_6$ notes of S_N are sorted by pitch. (The system also allows sorting by any of the other three note level parameters $\{a,\ \Delta t,\ \Delta t_{\text{event}}\}$.) The final phrase parameter, $\bar{x}_7$, is unusual because it does not affect S_N; rather it adjusts a parameter in the swarming module f. This is discussed in detail below.

9.5.3. Design

The design of a swarming system for music requires two major decisions, namely representation and dynamics. Representational issues govern the interpretations of particle state and the design of P and Q. The choice of dynamics (the swarming function f) is seemingly independent of representation, but ultimately they must be related because different particle dynamics might be more or less appropriate for a given representation. The appropriateness of a dynamics to a representation is the personal choice of the algorithm designer; there is no prima facie guide to representation and dynamics, since the design of a creative system is not logically determined.

Interpretation of the swarming patterns must be accomplished by a mapping of the state of each particle onto a musical/sonic parameter, which in turn is rendered by a synthesizer. This general scheme allows for mappings of any complexity (or simplicity). Since the mappings are essentially arbitrary, some guiding principle

is needed, at least to get started. The principle of transparency has been suggested (Blackwell and Young 2004b): the interpretative mapping should be comprehensible to the audience, and to collaborating musicians, so that the relationship between the particle movements and the output is clear. The swarm itself may be visualized in order to negotiate the digital divide between the workings of the algorithm and the output.

The principle of transparency urges the design to be as simple as possible, even to the extent of a literal interpretation of music descriptions. *Swarm Music* was originally intended as a note-level improvizer, and notes have loudness, pitch and timing corresponding to dimensions 1 to 4. The interpretation of these dimensions is very transparent. If a particle were to find itself at an attractor at p, it would output the same MIDI-parameterized notes that the system captured. In fact, due to the finite kinetic energy and the erratic particle movements, the swarm arranges itself *around* the attracting group, and outputs a melody that has a resemblance in rhythm, pitch sequence and loudness to the captured phrase.

In terms of the visualisation, a literal interpretation might be a map of pitch to height (x_3-axis, towards the top of the screen) and loudness to closeness to the viewer (x_1-axis, 'out' of screen). The mapping in each case is linear. The temporal parameters of note onset time and note duration are harder to map. One idea is to use the velocity of the particles as an indicator of rhythm, but this is problematic for two reasons. Firstly, particles in swarm simulations usually fly at a set speed, as determined by a velocity clamping which occurs immediately after velocity update, Eq. (9.3). *Swarm Music*, and optimisation swarms, use spring like forces,

$$a_i^{\text{attr}} = C \sum_{\text{all perceived attractors}} (p - x_i) \tag{9.9}$$

but *Swarm Music* uses a stiff spring constant C so that clamping is nearly always employed, and only steering occurs. The second problem with possible interpretations of velocity is that self-organization would have to take place in the $2Nd$-dimensional phase space of position and velocity. However, there is little, if any organization in velocity for a swarm, rather the organization is revealed in the sequence of spatial patterns. Whilst velocity organization does occur in flocks, it arises by virtue of the velocity aligning term in the dynamics and is not emergent.

Swarm Music, *Granulator* and *Techtiles* therefore derive their temporal interpretations from the spatial configuration of the particles. In *Swarm Music*, the x_2-axis is calibrated in beats per minute ($\sim \frac{1}{\Delta t}$); each particle's position along this axis is interpreted as the time interval between the onset of this particle's note and the immediately proceeding one. Spatially coherent swarms, where each particle has a similar x_2, will yield regular rhythms, and widely scattered particles or sub-swarms will produce a high diversity of onset times. A similar scheme is used for the x_4 component, note durations.

9.6. Experience

9.6.1. Performance

An important aspect of *Swarm Music* is the use of performance variables as part of the generative framework. Human performers will invariably 'interpret' a score, since a complete set of performance characteristics cannot be specified. For example, a musician can, in performance, vary tempo and rhythm, as well as dynamics (changes in loudness). Variations can happen at any structural level. *Swarm Music* could be used as a score generator by saving output MIDI events to file. However, *Swarm Music* is better exploited as an improvizer in partnership with a human(s). The system is able to quickly respond to incoming musical gestures with swarming melodies and rhythms. There is no notion of fixed tempo; rather, rhythms and dynamics are constantly changing due to the swarming motion of the particles, yet there is always a connection to the external sonic environment because of the mapping from incoming sounds to attractors. The system moves freely with the improvization, appearing to interact responsively with a partner (Fig. 9.5).

Another reason for the perceived musicality of *Swarm Music* is the use of spring forces to determine particle accelerations. Typically, spring forces produce oscillatory motion, with the period of oscillation governed by the strength of the spring. The update rule, Eq. (9.1), is a sum of attracting spring forces, Eq. (9.9), and Coulomb repulsions between neighbouring particles,

$$a_i^{\text{repul}} = K \sum_{\text{all perceived particles}} \frac{(x_i - x_j)}{(x_i - x_j)^3} \tag{9.10}$$

where K is a constant. Although particle motion is subject to irregular fluctuations due to the disturbances caused by the positioning of new attractors, the finite step size of the update, and the Coulomb repulsions, a remnant of oscillatory motion remains. This motion produces swings to loudness, pitch, note duration and rhythm and are a characteristic of the system. It is expected that live algorithms, just like human improvizers, should be idiosyncratic (Blackwell and Young 2005).

9.6.2. Other Examples of Swarming in Music

This summary reviews three other examples of music systems employing swarms and flocks. These systems represent alternative approaches to swarm simulations: visualizations, sonifications and non-sonic interaction. Each system is viewed from the perspective of the PQf architecture.

Visualizations of music in terms of swarms and flocks has been explored by various workers. An early example is Rowe and Singer (1997); the behaviour of a boid animation is controlled by acoustical information supplied by musicians. The flocks do not themselves produce sounds however; in the language of PQf, the system consists of analysis module P and swarming function f.

Sonifications of swarms have also been attempted. Spector and Klein (2002) were inspired by *Swarm Music* to add musical events to their swarm and flock

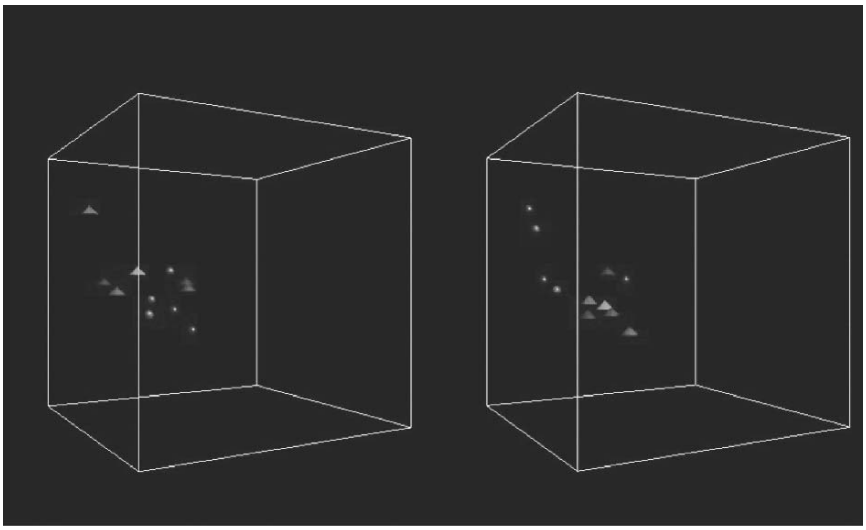

FIGURE 9.5. Improvisation with a 2-swarm. The left swarm (swarm A) has spontaneously began to move along the x_1 and x_3 axis (towards the bottom right-hand corner of H). The image of this movement in the right swarm (swarm B) can be seen in the distribution of attractors which mirror the positions of particles of swarm A. It is impossible to say if the swarm B will follow swarm A's initiative; attractors may be placed in the top right-hand corner of H_A, reflecting the positions of swarm B, and this may draw swarm A back

simulations, implemented in the BREVE simulation system. Notes are associated with certain events within the system, for example, feeding. Different instrument timbres are associated with each of the three species, and gradual musical transitions occur as each species enjoys a period of feeding. This is an example of sonification of a flock of agents, although the interpretation module Q depends on agent behaviour and not directly on flock spatial patterning. The authors report that in an extension of their system, spectrum and dynamics information from recorded music was used to alter constants in the swarm update formula although few details are given. The shift to live music would presumably be easy to make so that this system would comprise a full PQf architecture, although it is not apparent how transparent it would be.

Non-sonic interactions with swarms may proceed through physical gestures, rather than by music. Unemi and Bisig (2005) have developed an interactive boid simulation that acts as a virtual instrument. The boids move in a 3D space, with boid coordinates interpreted as pan, pitch and loudness. Users interact with the flocks by making physical movements which are captured by a camera. The user can change the instrumentation, melodic and rhythmic patterns of the flock in a process not dissimilar to conduction. The synthesis Q and f modules of this system bear much in common with *Swarm Music*, but since their P only accepts visual information, the system would not serve as a live algorithm.

9.7. Autonomy

Swarm Music has a user interface enabling direct access to many system parameters. The parameters α of the swarming function, Eq. (9.1), for example spring constants and maximum speeds along each dimension, can be controlled in real time. Interpretative parameters in Q such as the size of each axis can also be manipulated; pitch interpretation of particle position might be placed in the range MIDI 60 to 95, note onset times between $\frac{1}{120\,BPM}$ and $\frac{1}{60\,BPM}$, loudness between MIDI 64 and 127 etc. These real-time adjustments enable swarm 'conduction', a term that refers to Morris's conducted improvizations of groups and orchestras through a vocabulary of signs and gestures (Morris 2006). In a sense, conduction regards an entire orchestra as an instrument. This centralized control, of course, departs markedly from emergence through local interactions. A user may directly influence the swarm and its interpretation manually, and this has a considerable affect on the output, but the system is not operating as a live algorithm.

Swarm Music began as a four dimensional system operating solely at the note (mini) level. Live experience with the system showed that hand-tuning of f and Q often occurred during improvizations. Intervention at the interpretative stage is equivalent to adjusting phrase-level characteristics of the system. However, in the interests of autonomy, meso and macro level characteristics should be emergent rather than controlled. Luckily, a mechanism to transform (controllable) parameters into variables is suggested by the PQf architecture.

Any interpretative action can become autonomous by extending the dimensionality of the system. A P_{new} must be written that listens for the required characteristic in E (Figure 9.2). P_{new} parametrises this feature of E and maps to an attractor in H. Swarm interpretation must also be extended so that particle position components in the new dimension are correctly interpreted by Q_{new}, ideally for transparency with $Q_{\text{new}} = P_{\text{new}}^{-1}$. The first conduction controls to be automated in this way were chord number and pitch sort number, n_{chord} and n_{seq}. The conceptual mapping between the environment and the internal spaces is shown in Figure 9.6

Further live experience with the six dimensional system revealed that the particle speed control had a big impact on system performance and was frequently adjusted by the operator. The speed control is v_{max} in Eq. (9.3). Small v_{max} means small particle displacements leading to small changes in the output phrase. This sounds like a variation of a theme or an idea. At $v_{\text{max}} = 0$, the swarm is stationary and

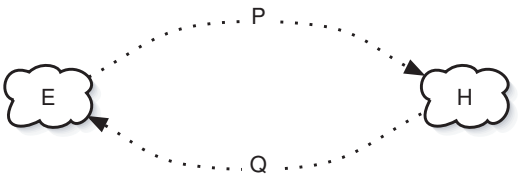

FIGURE 9.6. Interpretative functions P and Q map from the external environment, E to the internal space H of the live algorithm

the output riffs; large $v_{\max}$ increases the energy of the particles so they fly further from the attractors and the musical output is more diverse.

In a big advance towards autonomy, the speed control has recently become automated. *P* listens for similarity between incoming phrases, and sets the $v_{\max}$ attractor component along axis 7 according to a similarity measure. A simple matching algorithm is currently used. *P* hears a sequence of notes $\{\ldots, e_i, \ldots, e_j\}$, ending on the current (most recently received) note e_j. Denote an *N* note phrase$\{e_i, \ldots, e_j\}$, $j = i + N - 1$ by $\{i \to j\}$. The similarity $s(\{i \to j\},\ \{k \to l\})$ between a sequence $\{i \to j\}$ and an earlier *N* note sequence$\{k \to l\}$, can be defined as

$$s(\{i \to j\},\ \{k \to l\}) = \frac{1}{N} \sum_{n=0}^{N-1} c(e_{i+n},\ e_{k+n}) \tag{9.11}$$

where the correlation between notes, $c(e_i,\ e_j)$, can be defined to lie in the interval $[0,\ 1]$. A simple measure of note similarity is the absolute value of the number of steps between e_i and e_j, normalized to unity. Another measure might set $c(e_i,\ e_j)$ to one if $e_i = e_j$, and to zero otherwise. In order to look for the re-occurrence of an *n* note sequence, $n \leq N$, in the last two *N* note phrases (the repeated pattern may have intervening notes), it is necessary to compute $s(\{i \to j\},\ \{k \to l\})$ for $k = i - 1,\ i - 2, \ldots, i - N$. The maximum of the *N* comparisons will then certainly reveal a match if there is one. This defines the overall similarity

$$s = \underset{k}{max}\ s(\{i \to j\},\ \{k \to l\}))$$

(Note that identical computations arising from earlier phrase comparisons in Eq. (9.11) do not need to be performed so the computation of *s* has linear complexity.)

Suppose for the sake of argument that *P* has heard a high similarity over the last few phrases; perhaps the human partner is playing riffs. *P* sets the seventh component of *p* to $p_7 = (1 - s)\ X$ where *X* is the linear box size, $H = [0,\ X]^7$. The swarm will be consequently be attracted to a region of *H* where particle positions x_7 are high. *Q* calculates a speed limit from the swarm centre of mass according to

$$v_{\max} = \bar{x}_7 \frac{V}{X} \tag{9.12}$$

where *V* is a maximum speed limit, and modifies Eq. (9.3) accordingly. This will ensure that particle motion is small or zero even, and the output is also riffing, or slowly evolving. The problem with this scenario is that, should $\bar{x} = 0$, the swarm becomes frozen and incapable of movement, even if later attractors have small *s* values! If $\bar{x}$ is finite but small, it may take the swarm a very long time to move across *H* towards the new attractor. The solution implemented in *Swarm Music* is to ensure that *Q* clamps all $v_{\max}$ components *except* the seventh (similarity) component. $v_{\max 7}$ itself remains fixed and finite, allowing movement in this dimension. Particles can now move towards p_7, shifting the swarm centre of mass, and increasing particle speed and diversity.

9.8. Outlook

What use do swarms have in music? This chapter has answered this question by arguing that

1. Theoretic descriptions of music use a hierarchy of levels n, where each level corresponds to a perceptual time-scale
2. Composing music is a centralized, top-down process: $n \rightarrow n-1 \rightarrow n-2$
3. Self-organization (SO) is an emergent process, observed in natural systems, producing high level structure from low level interactions: $n \rightarrow n+1 \rightarrow n+2$
4. By analogy with SO, the interaction of musical objects at any level might produce, without implicit composition, new structure at higher levels
5. Improvised music is a de-centralized activity exhibiting an emergence of form through the low-level interactions of performers
6. Swarms are an exemplary, paradigmatic model of SO
7. Swarms might be used in music to self-organize musical objects at any level (sound granules, notes, phrases) into structures at a higher level
8. A model of interaction based on stigmergy has led to the design and implementation of swarm music systems that can interact with people in an improvized setting *as if they were musicians*

At the heart of the *Swarm Music* family of systems is a swarming module f. The function of f is to provide an almost limitless stream of spatial patterns. Analysis modules map the external sonic environment into the internal space of the system where interaction between system state and the external image can take place. A synthesis module interprets system state as sound.

This three component architecture can be readily adapted to include other patterning algorithms by substitution for f. Natural computation provides many examples of possible patterners, for example, evolutionary algorithms and neural networks. Other examples of possible f's include chaotic and non-linear systems from the field of dynamical system, multi-agent systems from artificial intelligence and many models from artificial life.

One aim of this research effort is to develop autonomous music systems (live algorithms). A swarm inspired interactive model based on stigmergy is proposed here, although of course other approaches may also be profitable. The goal of live algorithms research is not to replace human music making with an automatic machine; rather it is to augment human experience through the development of new, algorithmic ways of playing music. The desire is to find artificial music that is different from human expression, yet comprehensible. This overarching principle of transparency should be foremost in the design of algorithmic systems. The virtue of swarm systems is that a visualisation of internal process is already in a form that is understandable to us.

It is impossible to predict how live algorithms research might proceed, but a few observations are pertinent. To start, the description of music into separate levels is an activity of classification much loved by computer scientists and music

theoreticians. Human performers, whilst acknowledging this system, perhaps see granularities[1] rather than levels. Granularities do not exist in a hierarchy, but co-exist in a network of relationships. Features at any granularity may inform choices at any other granularity; no granularity is uppermost. Furthermore, performers always have the option of merging, deleting, re-configuring and even spontaneously inventing new granularities during the course of a performance. Granularity can be incorporated within the PQf architecture by remarking that state variables x in the state machine $f(x, \alpha)$ at one granularity can be mapped to parameters α of another granularity. In this way, emergence can propagate through the network. Section 9.7 outlines the general scheme.

Artificial Intelligence might also have much to offer. AI provides reasoning, a top-down activity, and learning, an activity based on memory. Advances may be made by combing a swarm-like system with a deductive mechanism that develops a degree of top-down structuring; the self organizer becomes an organizing self. The individuals in swarm systems do not possess any memory and so cannot learn. However, some type of memory is present in the system as a whole (swarm plus environment). Future swarm music systems might exploit this by including long-lived pheromone trails.

Machine consciousness is another fertile are for exploration (Holland 2003). The defining feature of a 'conscious algorithm' is the ability to self-model. An artificial improvizer, if endowed with such a facility, would be able to compare its own contributions with those of other participants. Such comparisons might involve an aesthetic function, as well as reference to past experience. The research issue is not plagued by questions of whether or not artificial improvizers are actually conscious; the idea is to see what other algorithms can be useful to the overall aim.

Potentially, a biologically inspired system might be able to negotiate the criticism that computer music cannot produce 'interesting' music without human intervention. This is due to its perceived inability to break rules (Miranda 2001, p. 206). Rules are a feature of top-down organization. A self-organizing system might produce appealing music, not so much by breaking rules, but by allowing new rules to spontaneously emerge. Swarm simulations are simple to implement and provide a complete model of self-organization. They are therefore a natural choice for exploring the potential of performing machines.

References

Allers, R. and Minkoff, R (Dirs.) (1994). The Lion King (USA).

Biles, A. (2006). In A. Biles and E. Miranda (Eds.), *Evolutionary Improvisation. Evolutionary Computer Music.*, Springer-Verlag, Berlin.

Blackwell, T.M. (2001). Making Music with Swarms. MSc thesis, University College London.

[1] I am grateful to Professor Mark d'Inverno for suggesting this term.

Blackwell, T.M. (2003). Swarm music: Improvised music with multi-swarms. In *Proc. the 2003 AISB Symposium on Artificial Intelligence and Creativity in Arts and Science.* pp. 41–49.

Blackwell, T. (Forthcoming). Swarm Granulation. In Machado, P. and Romero, J. (Eds.), *The Art of Artificial Evolution: A Handbook.* Springer-Verlag, Berlin.

Blackwell, T.M. and Bentley P.J. (2002). Improvised music with swarms. In *the 2002 World Congress on Evolutionary Computation*. pp. 1462–1467.

Blackwell, T.M. and Jefferies, J. (2005). Swarm techtiles. R. Rothlauf et al. (Eds.), *Evo Workshops 2005, LNCS 3449.* Springer-Verlag, Berlin. pp. 468–477.

Blackwell, T.M. and Young M.W. (2004a). Swarm Granulator. Raidl, G. R. et al. (Eds.), *EvoWorkshops 2004, LNCS 3005.* Springer-Verlag, Berlin. pp. 399–408.

Blackwell, T.M. and Young M.W. (2004b). Self-organised music. *Organised Sound 9*(2): 123–136

Blackwell, T.M. and Young M. (2005). Live algorithms. *Society for the Study of Artificial Intelligence and Simulation of Behaviour Quarterly* **122**: 7.

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

Burton, T. (1992). (Dir.) Batman Returns USA/UK 1992.

Burton, J.L. and Franks, N.R. (1985). The foraging ecology of the army ant. *Ecol. entomol* **10**: 131–141

Coker, J. 1986. Improvising jazz. New York: Simon and Schuster

Couzin, I., Krause K., Ruxton G. and Franks N. (2002). Collective memory and spatial sorting in animal groups. *J. theor. Biol.* **218**: 1–11

Gabor, D. (1947). Acoustical quanta and the theory of hearing. *Nature* **159** (4,044): 591–4.

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

Holland, O. (2003). *Journal of Consciousness Studies* **10**: 4–5.

Kennedy, J., Eberhart, R.C. and Shi, Y. (2001). *Swarm Intelligence.* Morgan Kaufmann, San Francisco.

Lewis, G. (2000). Too many notes: Computers, complexity and culture in voyager. *Leonardo Music Journal* **10**:33–39.

Miranda, E. (2001). *Composing Music with Computers*. Focal Press, Oxford.

Morris, L. (2006). Available online at http://www.conduction.us (accessed 18 March 2006).

Pachet, F. (2004). Beyond the cybernetic jam fantasy: *The continuator. IEEE Computers Graphics and Applications*, January/February 2004.

Piston, W. (1978). *Harmony, 5th ed.* Norton, New York.

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

Roads, C. (2001). *Microsound*. MIT Press, Cambridge, MA.

Rowe, R. (2004). *Machine Musicianship.* MIT Press, Cambridge, MA.

Rowe, R. and Singer, E. (1997). Two highly-related real-time music and graphics performance systems. *Proc. Int'l Computer Music Conference*, pp. 133–140.

Schaeffer, P. (1959). The interplay between music and acoustics. *Gravensaner Blatter* **14**: 61–69.

Spector, L. and Klein, J. (2002). Complex adaptive music systems in the BREVE simulation environment. In Bilotta et al. (Eds.), *Workshop Proceedings of the 8th Int'l Conf. on the Simulation and Synthesis of Living Systems.* University of New South Wales, Sydney, pp. 167–23.

Shaw, E. (1975). Fish in schools. *Natural History* **84**(8): 4046.
Sturman, P. (1983). Harmony, melody and composition. Longman, Singapore.
Turner, J. (2006). http://www.janeturner.net/current.php (accessed March 14th 2006).
Unemi, T. and Bisig, D. (2005). Playing by interaction among two flocking species and a human. In *Proceedings of the Third Int'l Conf. on Generative Systems in Electronic Arts,* Melbourne, Australia, pp. 171–179.
Wishart, T. (1996). *On Sonic Art* (revised ed.). Harwood, Amsterdam. Academic.
Xenakis, I. (1989). Concerning time. *Perspectives of New Music* **27**(1): 84–92.