Chapter 6
Live Algorithms: Towards Autonomous Computer Improvisers

Tim Blackwell, Oliver Bown, and Michael Young

Abstract A Live Algorithm is an autonomous machine that interacts with musicians in an improvised setting. This chapter outlines perspectives on Live Algorithm research, offering a high level view for the general reader, as well as more detailed and specialist analysis. The study of Live Algorithms is multi-disciplinary in nature, requiring insights from (at least) Music Technology, Artificial Intelligence, Cognitive Science, Musicology and Performance Studies. Some of the most important issues from these fields are considered. A modular decomposition and an associated set of wiring diagrams is offered as a practical and conceptual tool. Technical, behavioural, social and cultural contexts are considered, and some signposts for future Live Algorithm research are suggested.

6.1 Introduction

A Live Algorithm is an autonomous music system capable of human-compatible performance (Blackwell 2007, Blackwell and Young 2005). The context is improvised music; the Live Algorithm listens, reflects, selects, imagines and articulates its musical thoughts as sound in a continuous process. Or at least that is the dream of researchers working in this field. In practice, of course, the algorithm merely computes; an incoming stream of sampled sound acts as real-time input to a fixed and

T. Blackwell (✉)
Department of Computing, Goldsmiths, University of London, New Cross, London SE14 6NW, UK
e-mail: tim.blackwell@gold.ac.uk

O. Bown
Design Lab, Faculty of Architecture, Design and Planning, University of Sydney, Sydney, NSW 2006, Australia
e-mail: ollie@icarus.nu

M. Young
Department of Music, Goldsmiths, University of London, New Cross, London SE14 6NW, UK
e-mail: m.young@gold.ac.uk

J. McCormack, M. d'Inverno (eds.), *Computers and Creativity*,
DOI 10.1007/978-3-642-31727-9_6,

mechanical process that ultimately delivers an outgoing audio stream, rendered by a digital-to-analogue audio convertor.

The objective of a mechanical thinking brain is very far from realisation. The pending issue, therefore, is how a machine could emulate human performance convincingly enough that companion improvisers, and listeners, would accept the Live Algorithm as a contributing and creative group member with the same musical status as any other performer. The overarching aim is to extend the possibilities of music performance, achievable by the challenges of interacting creatively with a mechanical process, and by the exploitation of the pristine world of algorithmic patterning.

To achieve this, Live Algorithms should be able act responsively, proactively and appropriately without direct intervention, and contemporary methods in Artificial Intelligence, applied in real-time, offer suitable opportunities for rising to this challenge. The resulting systems, those that truly achieve this goal or at least achieve steps towards it, differ substantially to the norms of computer-as-instrument (with a fundamental reliance on human agency) and computer-as-score, in which a designers intentions are encoded as a set of rules or instructions, comparable to a musical score.

This chapter will examine several aspects of Live Algorithms and improvised music. In order to set the scene, a description of improvised music is given, specifying four attributes that we ascribe to human performers, and by implication, to a Live Algorithm: autonomy, novelty, participation and leadership (Sect. 6.2).

A formal specification is given that helps situate Live Algorithms in a wider taxonomy of computer music systems, based on a modular decomposition into analysis, patterning/reasoning and synthesis components (Sect. 6.3). A Live Algorithm is defined as a system in which these three elements are present, interconnected, absent of a human controller, and such that the above four attributes are satisfied. Several possible configurations of these modular elements (with or without a human controller) are considered, relating to a number of existing computer music practices. The core of the Live Algorithm, the patterning element, f, is considered in greater detail and a dynamical systems approach to their design is outlined.

The chapter continues with a discussion concerning the experience of performing with a Live Algorithm (or of hearing a Live Algorithm performance) in terms of the nature of performance interaction, and the possibilities of human-machine interaction in the near future (Sect. 6.4). Examples of simple behaviours that might be expected to occur are provided. Human performers operate in an extended context that lies beyond any single performance; they are subject to social and cultural forces that shape and inform their own approach to music. These social aspects and consequent implications for Live Algorithms are discussed.

Cumulatively, these ideas are offered as theoretical tools for the field of Live Algorithm research. Having presented them in very abstract terms, we turn to discussing a number of prototypes that we believe conform to the structure of a Live Algorithm (Sect. 6.5).

The chapter closes with an account of some possible directions for Live Algorithm research.

6.2 The Field: Creative Group Improvisation

Improvisation covers a spectrum of possibilities, from the spontaneous selection of prepared materials, which may be edited and mutated on the fly, but are made within an agreed macro-structure (for example Indian classical music and jazz), to the spontaneous creation of new micro-structures within a performance with (ideally) no previous agreements (within a genre known as "free improvisation", Bailey 1993). In the latter case, any discernible macro-structures can only emerge as the performance unwinds, since there is no single pre-defined plan or score. Other improvisational practices sit between these extremes. For example a rudimentary score might serve as a partially-defined macro-structure with opportunity for personal expression on finer levels.

6.2.1 Collective Improvisation

Free improvisation is a self-referential music. Practitioners seek to avoid any overt references to other musical genres, organisational methods or expressive devices. We propose that collective free improvisation is the ideal context for machine improvisation because the system can be considered formally, and at a first approximation, as an exchange of symbols (sonic events) between data sources (people and machines).

It has been suggested, in analogy with self-organisation in Nature, that macro-structures may emerge as a consequence of the micro-interactions between performers (Blackwell 2001), and they can do so without the agents being aware of the developing structure, or even of needing to be aware. This is exemplified by animal collectives such as vast starling flocks and herring shoals which are leaderless, decentralised and self-organising (Bonabeau et al. 1999). When group improvisation is viewed as self-organising (and this may not cover all aspects of human performance), we perceive a possibility for machine participation if the Live Algorithm can simulate inter-performer interactions.

6.2.2 The Individual (Human or Machine) in Interaction

A human performer in a collaborative improvisational context requires a basic set of capacities in order to participate appropriately. These capacities are as much to do with general aspects of the human experience and social interaction as with being a good musician. However, in improvised performance they can be specifically manifest or recognisably lacking in more or less musical terms.

6.2.2.1 Autonomy

An autonomous system, in contradistinction to an automatic system, is able to act and respond to unknowable and unforeseen inputs, and in ways that have not been

completely prescribed. Autonomy is one quality that might enable a machine improviser to become accepted as an equal partner in a group setting.

The term *agent*, as used in Artificial Intelligence, refers to a device that perceives its environment through sensors, and takes action on the environment by means of actuators (Russel and Norvig 2003). An *autonomous agent* in robotics is any embodied system that satisfies its own internal and external goals by its own actions while in continuous interaction with the environment (Beer 1995). Autonomy therefore expresses the ability of an agent to take action based on its own precepts, rather than following an inbuilt plan.

An *autonomous musical agent* would therefore base its action (musical output) in part on what it perceives (musical input). The extent to which action is based on preloaded musical responses determines the degree of automation. A system that has no input is purely generative (rather than interactive). It is similar to a closed dynamical system where the initial conditions, including any hardwired knowledge, are sufficient to determine all future states. Such a system is automatic. Any further ability an automatic system might have to make a decision or take an appropriate action in the eventuality of an unplanned input would render the system autonomous, with more autonomy resulting from a greater capacity to make such decisions or actions.

6.2.2.2 Novelty

Whether supporting, leading or subverting, a musician's contribution should endeavour to avoid the clichéd and the obvious. The ability to find a novel, but also appropriate, way of playing within a very familiar musical environment would challenge any musician, but the inability to never find novelty would mark the musician down as dull, and would inhibit the ability of the group as a whole to develop communally creative structures.

Fundamental and distinct types of creativity are described by Boden (2004). A basic creative practice is to produce new combinations of established ideas or objects, just as, for example, the amateur jazz improviser combines learned melodic patterns in response to whatever harmonic structure is presented. Algorithmic emulation of such behaviour is possible as the soloist feature on the commercial *Band In A Box*[1] demonstrates; however the results are rudimentary.

6.2.2.3 Participation

An improviser has to be able to support ongoing musical activity by making contributions that do not detract from but rather enhance the current musical direction. This would be a very hard characteristic to pre-program in a top-down manner:

[1] http://www.pgmusic.com/.

how to ascertain and characterise a musical direction, and which of many possible contributions might enhance the current musical mood? However an algorithm specification does not necessarily require a top-down structure. Participatory activity should be recognisable both to human performers and listeners. The extent and character of the participation might be evident in apparent types of behaviour (some examples are discussed in Sect. 6.4.1); musical processes that allude to social modes of interaction. The wider challenges in achieving true human-machine participation are explored in later in this chapter, from musical, social and cultural perspectives.

6.2.2.4 Leadership

Another attribute of an improvising musician is the capacity to undertake a direct leadership role. In other words to attempt to change the musical direction, to invoke a new musical centre. In improvised music such roles may be fuzzy and interchangeable, and never explicitly agreed upon, but at any given time there is a balance to be struck between responsiveness and proactive intervention.

6.2.3 Relationship of the Four Attributes to Creativity

Perhaps the most familiar model of a creative process is the "exploration of a conceptual space", i.e. explorative behaviour within constraints, whether explicitly defined or not. In freely improvised group performance, it is characteristic for timbral, textural and gestural events, however individually novel, to be consistent with a shared, emerging aesthetic. This could be viewed as a participatory exploration of a musical space. In algorithmic terms, an iteration through a set of parameters or the navigation of system state to distant areas of state space can be viewed as an exploration of the potentialities of the formal computer code.

Boden's most demanding level of creativity is the notion of a transformation of conceptual space itself (Boden 2004). It is very challenging to think of any algorithmic process that could produce brand new interpretations of its own output. However, the ability to intervene proactively seems a necessary pre-condition of transformational creativity. We believe that live algorithmic music in which leadership from any party is possible offers such a prospect, i.e. to change our expectations about how humans and machines can engage in collective performance, and the consequent nature of the creative outcomes. Collective improvisation already offers a powerful approach to transformational creativity, in that the group does not possess a single shared conceptual space but a set of distinct individual conceptualisations interacting in a shared creative activity: different participants can influence these collaborators. Individual understanding of the music is a continually evolving interaction.

6.3 Theoretical Considerations

This section takes a formal approach to Live Algorithms. First, a design methodology is outlined, then it is shown how this methodology can categorise computer music systems and be applied as a conceptual tool for describing Live Algorithm behaviour. The impact of Artificial Intelligence on Live Algorithm design is considered and a dynamical systems approach is described in detail.

6.3.1 P, Q and f

Our PQf architecture, originally presented in Blackwell (2001) and described in this section, identifies three modules which may be combined in various ways to implement computer music systems. The modules are: P (listening/analysis), Q (performing/synthesis) and f (patterning, reasoning or even intuiting). The purpose is two-fold. The modules represent basic functionalities (the actual software might not be cleanly divided but the functions of conceptual parts of the system remain well defined) and their wiring diagram (Fig. 6.1) helps us to explore what systems are possible, a possible architecture for the development of any particular system and a taxonomy of established practice. Secondly, the modules represent actual software components. The development and distribution of separate modules and a language for inter-module communication would enable rapid evaluation of computer music systems, saving much effort, and encouraging novel combinations.

6.3.2 Definition of a Live Algorithm

A Live Algorithm is defined as a system in which these three modules are present, interconnected, absent of a human controller, and such that the above four characteristics (autonomy, novelty, participation and leadership) are ascribable attributes of the system.

6.3.3 Architecture

Naively speaking, P is to ears as f is to brain as Q is to voice, but in humans these compartments are themselves conceptually ambiguous. The boundaries between the modules can be reconsidered according to different ideas about perception, cognition and production (including the conceptual status of cochleas, hands and instruments). The same re-evaluation can occur in novel computer music systems. For example, in an extreme analysis $P = adc$, $Q = dac$, $f = internal\ dsp$ where adc and dac are converters between analogue (a) and digital (d) representations and dsp stands for any digital signal processing module.

There are several fundamental wirings of the three modules, with or without a human controller (Fig. 6.1), that can be used to form a taxonomy of computer music systems. The figure shows the controller placed to the left of the system (parameter domain) and the audio environment, Ψ, to the right of the system. Musicians, operating in the sonic domain (to the right of the system in the figure) contribute directly to Ψ.

P and Q are established subcomponents of music systems. The novel aspect of a Live Algorithm derives from the inclusion of a patterning/reasoning module, f, which has neither audio input or output, but is a more general purpose algorithm which could be applied equally in non-computer music contexts. In general f embodies a computational process with input and output parameter streams. In Live Algorithm terms, f is a generative unit, the machine equivalent of ideas and imagination. This function is key to enabling the system to demonstrate the capabilities of autonomy and novelty.

Each wiring is already in common use in various computer music scenarios. These are described in the following in each case, and their potential for Live Algorithms research is discussed.

P performs analysis of incoming audio (Fig. 6.1A). Its human-equivalent function is *listening*. In the figure, Ψ is the musical environment; Ψ_{in} (Ψ_{out}) are incoming (outgoing) audio streams. (Alternatively, an incoming sound wave could be digitised by an analogue-to-digital converter. Such a converter would be regarded as part of P itself.) P processes incoming samples, producing analysis *parameters*. These parameters seek to describe the audio, in music theoretic terms (events, pitch, duration), as spectral data, in timbral terms such as smoothness and roughness, or in other high level descriptors. P therefore emits a stream of parameters at a slower rate than the signal rate. In *Music Information Retrieval*, the data is used for the automatic scoring of performance. Figure 6.1A could represent a possible performance scenario in which a musician can inspect analysis parameters in real-time, most likely via a graphic display. This set-up may be used to supplement the sonic information the musician already has access to. Figuratively, the musician (functioning as a controller) is placed to the left of the P module to emphasise that system interaction is via parameters, and not by audio (in which case the musician would be placed to the right of Ψ). Reliable algorithms for machine listening are of considerable importance but the problem is very challenging when the audio source is a combination of several instruments. Machine listening is the subject of a large research effort within the DSP community.

P itself does not perform any function other than analysis. If some further purpose is intended, the analysis parameters are fed into an algorithmic module, f, as depicted in Fig. 6.1B. For example, if the information is to be used to access similar excerpts from a music database, f would perform the similarity measure and the look-up.

Note that links between modules are not directional to indicate that parameters might be passed in either direction. For example, a subcomponent of f might require a finer level of analysis from P, and could therefore send an instruction to P to that effect. The bi-directionality of system components means that the division into P,

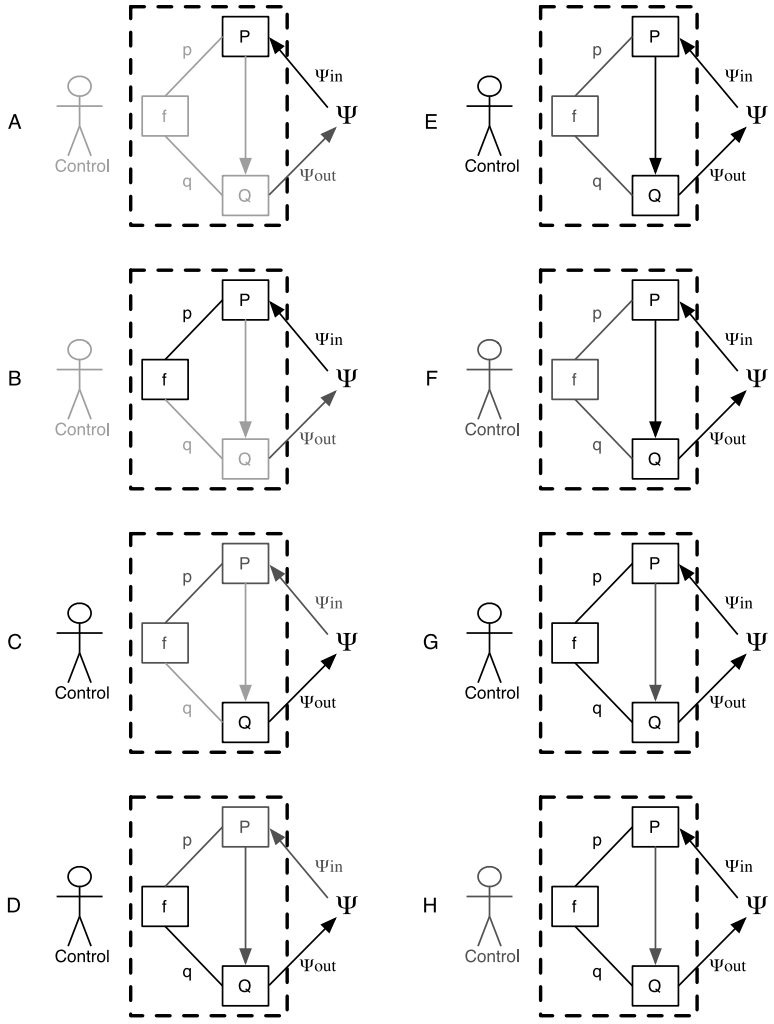

Fig. 6.1 PfQ "wiring diagrams" for different computer music applications, a non-exhaustive set of possibilities. An optional human software controller is depicted to the left of the modular decomposition; the shared audio environment, denoted Ψ, and placed to the right of the system, represents all utterances from instrument musicians and other computer music systems. The diagram shows eight common computer music systems: **A**—Audio analysis; **B**—Audio analysis with added functionality as provided by f (e.g. real-time score generation); **C**—Audio synthesis; **D**—Generative (Algorithmic) music; **E**—Live computer music involving a musician-controller who is able to monitor and adjust the functioning of a largely automatic and reactive system (such systems have become the accepted practice in live computer music settings); **F**—Reactive system as used, for example, in Sound Installations; **G**, **H**—prototype and actual Live Algorithmic systems. The ideal, wiring **H**, runs autonomously without the presence of any human control. In practice, some human attention is often required, as depicted in **G**

f and Q is to some degree arbitrary; in practice the separation is distinct since each module serves a different fundamental purpose.

Figure 6.1C shows a synthesis unit, Q, outputting audio. *Audio synthesis* is a well studied area of computer music and there are many available techniques, ranging from the rendering of sampled sound and the emulation of actual instruments (e.g. by physical modelling), to the design of new synthetic sounds using purely algorithmic techniques with no obvious analogue in the domain of physical instruments (for example, granular synthesis). The figure shows the possibility of synthesis control by an operator in real-time (but not sonically, since that would involve a P module). This is the archetypal computer music application, the computer-as-instrument; real-time control of an *electronic instrument* by parameter manipulation (the instrument might be controlled via a mouse and keyboard, or by a hardware device such as a MIDI[2] keyboard, a pedal or some other controller). Figure 6.1C might also represent live sound diffusion, in which a performer can alter parameters such as spatial position and volume in the playback of a pre-recorded composition.

Figure 6.1D shows the attachment of a module f to Q. f provides a stream of synthesis control parameters. This is the typical realisation of *generative/algorithmic music*, depicted here with a human controller, but equally possible without. This is the computer-as-proxy. f automatically generates musical information occupying a compositional functionality. There are many possibilities for f, and this represents a major branch of musical activity. It might be understood to represent any and all forms of rational process underlying pre-compositional musical methods, but contemporary, computational applications are most apposite. Two modern examples are Xenakis's development of computer-based stochastic techniques (Xenakis 2001), and the work of Cope (1992) who has developed generative systems that produce music by reference to a database of extant compositions.

One important source of potential f's are algorithms from complex and dynamical systems science. There is a wide and inspiring palate of patterning algorithms. Example include: cellular automata, generative grammars, L-systems, evolutionary algorithms, flock and swarm algorithms, iterated function systems (fractal generators), chaotic dynamics, real time recurrent neural networks and particle systems (see, e.g. Flake 1998, McCormack et al. 2009).

Clearly not interactive, fQ offers some potential for variation. For example, if some of the algorithm parameters (i.e. in f) are pseudo-random, or influenced by some other data stream, it could function as a device with variable and less predictable sonic output. An fQ compositional system could be used in performance in which musicians play alongside the system. In this case the interaction is severely limited, as the players role is to only to follow. This scenario was established in the now largely defunct genre of "live performer and pre-recorded tape" that developed through the 1950s to 1970s, although such practices are still evident in many commercial musical contexts. An intriguing contemporary possibility is the real-time manipulation of an algorithm (and synthesis parameters). Such a system is used by

[2]Musical Instrument Digital Interface, an interface and protocol standard for connecting electronic musical instruments and devices.

live laptop performers (e.g. *live coders*) who manipulate or even create small algorithmic units—f's—in real-time.

Figures 6.1E and 6.1F show a direct parameter mapping between analysed source and synthesiser control. If the functions P and Q are mutual inverses, a mapping between them is trivial, $Q = P^{-1}$, and the system is a (possibly distorted) musical mirror. Alternatively, the relationship between P and Q and the mapping between them may be so complex that a participating musician is challenged to find any obvious correlation. These systems can only be a vague prototype of a Live Algorithm as they are only automatic, in the sense defined earlier; the initial conditions that establish the mapping remain unaffected by context and new data. If there is direct intervention by a user, as in 6.1E, the system as certainly not autonomous. These systems cannot achieve autonomy because they are overwhelmingly reactive (for example it may not play in the absence of sound or pause in the presence of sound). Any attempt to move beyond this basic feedthrough device requires algorithms that do more than provide a huge switchboard of parameter connections, i.e. an f module (see below).

Systems E and F may be used in certain *sound installation* contexts, including situations where Ψ represents sonic and non-sonic environments (including movement, brightness, etc.). Although the system is primarily a parameter map, a musician could monitor synthesis data that may inform his/her choice of synthesis parameter settings. This scenario is the accepted practice in live computer music, and accounts for the vast majority of music produced in a live setting with computers. Typically a software development system such as Max/MSP is used to implement P and Q functionality (although they are infrequently broken down into actual software modules), with a visual display in the form of a "patch" and soft controls such as sliders and dials to allow real-time (non-sonic) interaction with the system. E might be considered as an enhanced instrument.

Systems 6.1G and 6.1H are syntheses of the basic A–F types. The most significant elements are that they incorporate both analysing and performing within the sonic domain, and establish a loop across the sonic and computational domains by incorporating a functional module f. The ideal Live Algorithm operates autonomously (system H); in practice, and until a true Live Algorithm is realised, some degree of intervention is desirable (system G). Further exploration of these challenges is presented in the sections below. Before this, we need to consider the fact that all these processes occur in real-time, and also musical time, in which sonic events have a past, present and future.

6.3.4 The Live Algorithm from the Outside

The Live Algorithm is, from the point of view of fellow performers, a black box. We consider the functionality of the system as a whole in terms of the most primitive description, the flow of data into and out from the device. Such a study points at possible performance measures that can be utilised in Live Algorithm design.

In Fig. 6.1, the analysis parameters p and q are marked alongside the appropriate links. In principle parameters could be passed in either direction, but at a simple level we may consider that f receives p as input, and emits q in a continuous process. Moments of silence between events are then represented by streams of constant parameters. The task of P is to deliver a parameter stream $\{p\}$ to f, and that of Q is the sonification of an output stream $\{q\}$. P and Q act as transducers that enable f to interact with the external environment. The process can be formally represented as

$$\begin{aligned} \Psi_{out} &= Q\big(f\big(x, P(\Psi_{in})\big)\big) \\ &\equiv F(\Psi_{in}) \end{aligned}$$

where x is any internal state of f and it is assumed that $\{\Psi_{in}\}$ does not include any of the system's own outputs. F is the *observable* function of the Live Algorithm. f itself is hidden. Performers only have access to each other's F; inferences about private desires, goals, etc. in other words, performers f's, are made on the basis of these observations.

The Live Algorithm must therefore decide when and what to transmit with regard to the history of inputs and outputs, $\{\Psi_{in}\}$ and $\{\Psi_{out}\}$, the internal dynamic state x and any static parameterisation of f, P or Q (which may include data on previous performances).

The task of finding an f that might satisfy the requirements of a Live Algorithm is formidable. One way forward is to explore possibilities on an ad hoc basis (as is common) and in the lack of any formalised objective this is the only available means of development. The development of a performance measure for a Live Algorithm, however, would suggest a more systematic methodology. The search for a performance quantifier could commence by investigating human practice. If such a metric could be established, it would guide development of fs; there is even the possibility that a Live Algorithm could become its own critic and learn from experience.

Although the input streams are continuous, we may suppose for the purpose of this discussion that inputs and outputs can be coarse-grained. For example, audio samples can be grouped into a phrase of notes. The continuous streams could then be considered as sequences of discrete events; the precise definition of what might be a meaningful event, and the duration of the sequences is not important for the following analysis. The streams can then be split into past, current and future and comparisons can be made between them:

Ψ_{out} contains elements

1. of past Ψ_{in}'s (referential)
2. of current Ψ_{in} (reactive)
3. of future Ψ_{in}'s (pre-emptive)
4. of past Ψ_{out}'s (consistent)
5. of future Ψ_{out}'s (planning)
6. not found in past or current Ψ_{in}'s (independence)
7. not found in past Ψ_{out}'s (exploratory).

In order to participate in a group improvisation, the Live Algorithm has to convince the other participants that it is listening to and engaging with their contributions. As with conversation, this is achieved by the assimilation of inputs into outputs. Participation can therefore be measured by the degree of reference and reaction.

A novel contribution has to be surprising in some way to those engaged in the musical conversation. Novelty can be measured by the extent to which an output is independent (of inputs) and is self-exploratory.

The concept of leadership is very subtle: what can the system do so that the environment is persuaded into participation? In order to lead, a player has to embark on a musical direction that is picked up by the other participants. Copy cat scenarios such as $\Psi_{out} = \{A, B, A \ldots\}$; $\Psi_{in} = \{B, A, B \ldots\}$ should not be considered to involve a leader. Leadership therefore requires both pre-emption and novelty.

We would naturally regard any musical human partner as autonomous. A central problem of Live Algorithm research is to persuade human musicians that a machine partner is making valid contributions, contributions that should be respected, and not ignored or regarded as a malfunction. This is a problem of perceived autonomy. Operationally we suppose that a non-heteronomous and non-automatic system is *autonomous*, where a system with no referential or reactive elements is defined as *automatic* and a system that is fully determined by its environment, i.e. has no independent elements, is *heteronomous* (i.e. subject to external control). Autonomy is a relative term, with degrees of autonomy ranging in extent form marginal to very tight coupling with the environment. In this definition of autonomy, Ψ_{out} is not entirely determined by either the history $\{\Psi_{in}\}$ *or* the internal state x alone. An autonomous system sits between heteronomy and automation.

Bertschinger et al. (2008) point out that in order to avoid heteronomy, different actions must be possible given the same environment. In our formulation, future actions are contingent on both histories, $\{\Psi_{in}\}$ and $\{\Psi_{out}\}$, so that a specific Ψ_{in} occurring at t_1 and again at $t_2 > t_1$ would, in general, be followed by a different response Ψ_{out} since the histories $\{\Psi_{in}\}_{t_1}$, $\{\Psi_{out}\}_{t_1}$ and $\{\Psi_{in}\}_{t_2}$, $\{\Psi_{out}\}_{t_2}$ will generally differ.

A Live Algorithm that deploys stochastic methods (or at least some degree of statistical uncertainty) could also supply different actions given the same environment: completely random behaviour should not be considered autonomous (Bertschinger et al. 2008). Randomness is avoided if the Live Algorithm provides structured output, which may or may not incorporate past inputs or outputs. Although a Live Algorithm might produce randomness over one epoch $[t_1, t_2]$, an exploratory system would eventually produce structured output in order to avoid repeating epochs of randomness.[3] Exploratory behaviour, as noted above, prohibits prolonged repetitions of periods of randomness that might otherwise be counted as trivially novel.

Improvisation is not just about acting spontaneously, although this plays an important part. Output that lacks coherence and darts from idea to idea is a feature of

[3] We sidestep issues concerning the randomness, or not, of sequences of finite length.

inexperienced improvisors; more experienced improvisors exhibit some degree of coherence in their approach. Regularity, as captured by consistency and the ability to plan ahead, is measurable by the relationship of the current output to previous and future outputs.

The musical elements in question have not been defined, and a prescription of how to make the measurements has not been detailed. The process is dependent on the level of description, and it may be that several levels are needed. At the level of note events, for example, the elements are phrases, and comparisons can be made using a similarity measure based on a definition of the distance between two phrases (how closely the phrases shapes match each other, or the number of changes needed to bring the phrases into agreement, for example). Comparisons between streams are also possible using information-theoretic techniques such as mutual entropy; such methods might be important where an appropriate level cannot be defined.

6.3.5 Artificial Intelligence

Artificial Intelligence (AI) offers various schemes that may prove fertile for Live Algorithm research and strategies for developing functions f, as represented in general in the wiring diagrams above.

Reasoning can be based on a programmed rule set, or derived through training. In the former, the Live Algorithm designer has access to the vast experience of symbolic AI research that includes knowledge representation, problem solving, planning and expert systems. Within this AI framework, the problem domain is represented symbolically. The representation contains knowledge about the problem domain, and the symbols are syntactically manipulated until a solution is found. The main focus of the symbolic framework is on a suitable formal representation of the problem domain, the inclusion of domain specific knowledge and efficient algorithms.

Machine learning is another major AI framework. The learning algorithm can be based, for example, on a neural architecture or on Bayesian structures (e.g. Hidden Markov Modelling). Responses are learnt over a sequence of test cases. The focus is on the learning algorithm, the training set and the network architecture.

We can suppose that a human improviser can potentially refer to his/her own theoretic knowledge as well as her/his experiential knowledge of music making and of group improvisation in particular. It would be inhibitive to deny similar advantages to a Live Algorithm. Domain knowledge can be hard-wired into the Live Algorithm and trial performances offer ideal test cases for learning algorithms. Since the definition of the Live Algorithm only makes reference to inferred behaviour and not to any supposed mental states, the debate as to whether cognition is symbol manipulation (computationalism) or dependent on a neural architecture (connectionism), or indeed some other paradigm, is not relevant; rather, any technique can be requisitioned in order to further the overall goals of Live Algorithm research.

As an alternative approach to reasoning or learning, we mention here the *dynamical systems* framework which has already proven to be a rich source of ideas in Live Algorithm research.

Symbolic and connectionist approaches to mobile robotics have not been an unqualified success (Brooks 2009). The computational problem is the navigation of a dynamic, uncertain environment. An incredible amount of initial data would be needed in the closed system approach in order to account for all the possible inputs the robot might receive. In contrast, the open, dynamic framework has proven much more fruitful; the robot program is open, and modelled more closely on, say, how an ant might move through a forest. The similarity between the improvisational environment which will be very dynamic and uncertain, and the passage of an ant, or mobile robot, through an uncharted environment leads us to expect that the dynamical framework will be advantageous to Live Algorithm research too.

In a dynamical system, a state x evolves according to the application of a rule, $x_{t+1} = f(x_t, \alpha)$ where α stands for any rule parameterisation. The sequence $x_t, x_{t-1}, \ldots$ defines a trajectory in the space H of possible states. A closed dynamical system is one whose evolution depends only on a fixed parameter rule and on the initial state. These dynamical systems are non-interactive because any parameters are constant. The dynamical systems framework is quite comprehensive, encompassing ordinary differential equations, iterated maps, finite state machines, cellular automata and recurrent time neural networks.

Fully specified dynamical systems have a rich and well studied set of behaviours (Kaplan and Glass 1995 is an introductory text; Beer 2000 provides a very concise summary). In the long term state trajectories end on a limit set, which might be a single point or a limit cycle in which the state follows a closed loop. Stable limit sets, or attractors, have the property that nearby trajectories are drawn towards the limit set; states that are perturbed from the limit set will return. The set of all converging points is known as the basin of attraction of the attractor. In contrast, trajectories near to unstable limit sets will diverge away from the set. Infinite attracting sets with fractal structure are termed strange; trajectories that are drawn to a strange attractor will exhibit chaos. The global structure of a dynamical system consists of all limit sets and their basins of attraction and is known as a phase portrait. Phase portraits of families of dynamical systems differing only in the values of their parameters α, will not in general be identical.

An open dynamical system has time dependent parameters and therefore many phase portraits. Since smooth variation of parameters can yield topological change at bifurcation points (a stable equilibrium point can bifurcate into two or more limit points, or even into a limit cycle), the global properties of open dynamical systems are highly context dependent and system behaviour can be very rich. In a live algorithmic setting, the open dynamical system parameters derive from the analysis parameters. If H is chosen to map directly to the control space of Q, system state can be directly interpreted as a set of synthesiser parameters. Inputs p could be mapped to attractors, with the advantage that trajectories moving close to p will resemble Ψ_{in} (participation). However x may not lie in the basin of attraction of p and the trajectory might diverge form p, potentially giving rise to novelty and leadership. Small changes in input might lead to bifurcations in the phase portrait, sending a trajectory into a distant region of H, giving unexpected outputs. The ability of an open dynamical system to adapt to an unknown input marks it out as a candidate for autonomy.

The flow of p into f is the virtual counterpart of the sonic interactions that are taking place between performers. In a dynamical system, p could become a state alongside x, with the difference that the dynamics of p are driven by the outside world, whereas the dynamics of x are enacted by a map. Interaction fits naturally within the dynamical systems approach, unlike reasoning and machine learning algorithms which are normally constructed as closed systems. The variety of possible inputs would have to be specified by the designer in a rule-based system, and in a learning system, response is dependent on the comprehensiveness of the test set. The dynamical systems approach offers a more robust alternative. Finally we note that an extremely large number of alternative outputs (the size of H) can be easily implemented in a dynamical system.

6.4 Live Algorithms in Context

This section considers aspects of Live Algorithms that cannot be directly programmed. Improvisers are characterised by individual behaviours which are the result of learning and playing music in a social and cultural context. We speculate that a Live Algorithm might also participate in these contexts. The section looks at some behaviours, and then discuses the social and cultural dimensions of improvisation.

6.4.1 Live Algorithm Behaviour

Young and Bown (2010) identify four distinct behaviours that might be exhibited by a Live Algorithm: shadowing, mirroring, coupling and negotiation. These behaviours give some indication of the capacities systems in Fig. 6.1E–G would need to demonstrate.

The behaviours are expected to be emergent, rather than directly programmed. In general it is better to set overall goals and let a system develop its own behaviours in order to that accomplish these goals. A top-down approach is rigid and relies on a complete analysis of the problem; a bottom-up approach is more robust. The performance goals for a Live Algorithm are not well understood; Sect. 6.3.4 advocates the study and codification of the observed function F of human improvisors.

6.4.1.1 Shadowing

Shadowing involves a synchronous following of what the performer is doing, mapped into a different domain. Systems E–H in Fig. 6.1 could produce this, although only E or F are necessary. A pitch shifter in both the audio or MIDI domain, or any synchronous audio or MIDI effect, are very simple examples. In such cases, shadowing reduces to the configuration shown in Fig. 6.1E, with or without a human

controller. The strength of shadowing lies in the fact that performer and Live Algorithm express a strong coherence, with tightly unified temporal patterning. In its simplest form, shadowing achieves strong but trivial participation, and little or no leadership, autonomy or novelty. However, even in this simple form, the appearance of coherence can have a strong effect for both performer and audience, and can contribute to the sense of autonomy of the system, and the generation of novelty through its interactive affordances. More complex forms of shadowing might involve more sophisticated musical responses such as counterpoint and harmony. A system based on rhythmic entrainment and temporal anticipation rather than an instantaneous response could achieve shadowing in a way that exhibited creativity, and the possibility to switch to a leadership role.

6.4.1.2 Mirroring

Mirroring involves some extraction of more abstract stylistic information or musical content from the performer, which is "reflected" back to the performer in novel ways. Pachet's *Continuator* system provides a highly sophisticated example of mirroring (Pachet 2004). System E in Fig. 6.1 would be the most apposite context.

In human performance mirroring is used as an explicit device or appears as a more implicit principle of building on a shared mood or theme. As with shadowing, the system predominantly takes the lead from the performer. This clearly demonstrates participation, and can contribute to a form of collaborative creativity through the opening up of new possibilities. As with shadowing, an appearance of autonomy comes with the sense that the musical output is coherent. By successfully achieving the local goal of mirroring with an unknown performer the system demonstrates a basic autonomous capacity, perhaps even implying that it "understands".

The mirroring approach is more immediately capable of leadership, but like shadowing must be enhanced by other behaviours to achieve this ends. Choices about how the mirroring is managed can lead to greater autonomy. In order to achieve leadership the mirroring must be capable of appropriate alteration to the style being reflected. Shadowing and mirroring preferably require an interaction scheme where the performer's output can be clearly distinguished from the environment, rather than where the state of the environment, consisting of the mixed output of both performer and Live Algorithm, is given as input. Naturally this can be achieved if the system is capable of distinguishing its own output from the mixed input, but this is challenging in practice.

Mirroring fits the fully fledged Live Algorithm scheme of Fig. 6.1H, where f involves the storage, analysis and retrieval of incoming feature data p.

6.4.1.3 Coupling

Coupling refers to a system's behaviour that is largely driven by its own internal generative routines, which are perturbed in various ways by information coming from

the performer. This is a particular application of system G and H in Fig. 6.1. Designers can place the greatest emphasis on the design and behaviour of f, exploring the musical behaviour of diverse computational systems, possible with a flexible approach to the form of P and Q. Through such mutual influence, the performer and Live Algorithm can be seen as a coupled dynamical system, where both participants are capable of acting independently. Coupling does not prescribe a specific behaviour, and may involve aspects of mirroring and shadowing (in the latter case the coupling would be tighter), but tends to refer to a situation in which the system can clearly be left to lead (by acting more independently of the performer), possibly to the detriment of the sense of participation (in which case we can think of the algorithm as "stubborn" or unresponsive). However, a sense of participation depends on the attitude of the performer. A performer may deride shadowing and mirroring as a failure to truly participate, that is, to bring something original to the collective performance. A successful coupling-based system would demonstrate autonomy and creativity, and in doing so achieve participation.

Coupling is a practical behaviour because it is essentially trivial to achieve; it evades strict requirements about the kind of interactive behaviour the system exhibits, as long as the performer is demonstrably exerting some kind of influence over the system. This offers creatively fertile ground for what McLean and Wiggins (2010) refer to as the *bricoleur programmer*, building a Live Algorithm by musical experimentation and tinkering. It also relates to an aesthetic approach to computer music performance that favours autonomy, leadership and the potential for surprising variation (novelty and thus creativity) over participation. It allows for the introduction of various generative art behaviours into a performance context.

6.4.1.4 Negotiation

Negotiation is defined as a more sophisticated behaviour that is related to coupling but is based on aspects of human cognition. Only system H in Fig. 6.1 allows for this behaviour. A system that negotiates constructs an expectation of the collective musical output and attempts to achieve this global target by modifying its output. Since the collective musical output depends on the performer as well, negotiation, as the name suggests, may involve attempts to manipulate the behaviour of the performer, or equally, to adjust one's expectations in light of the direction of the music. As with coupling, with negotiation the Live Algorithm is understood as interacting directly with a piece of music, rather than with other individuals. More sophisticated Live Algorithms could perform acoustic source separation and use a "theory of mind" to infer individual behaviour from the environment.

Negotiation can be seen as a framework for the design of a Live Algorithm (see Young and Bown 2010), involving the interaction between an expectation and a behaviour (which contributes to the musical environment), either of which can be modified to create a better fit between them. This is harder to achieve than the other behaviours, so is less pragmatic. The challenge is to obtain a system that achieves

strong collaboration through negotiation: that has distinctive expectations (leadership) and finds a way to satisfy those expectations whilst also accommodating the behaviour and expectations of the performer (participation). Achieving this balance can be seen as a well-formed creative challenge, one that requires autonomy.

Systems that are capable of successful negotiation are a goal for Live Algorithms research. A relevant question is how minimal a system can be whilst achieving a sense of negotiation in performance. Coupling can be seen as a weaker form of negotiation. Negotiation can be seen as fulfilling the traits autonomy, participation and leadership most fully. Novelty (leading to creativity) can be introduced into the expectation of the system.

Shadowing and mirroring can be seen as behaviours that attempt to offer the semblance of participation through acknowledgement of the performer's output, demonstrating the ability of the system to produce meaningful responses. Coupling and negotiation, on the other hand, can be seen as behaviours that attempt to create a sense of mutualism between performer and Live Algorithm (thus autonomy and leadership on behalf of the Live Algorithm), by imposing the reciprocal demand on the performer to satisfy some expectation in the Live Algorithm itself.

6.4.2 Agency and Live Algorithms

Music involves temporal dynamics on a number of time scales, from the waves and particles of microsound, through the patterning of rhythm, meter and melody, through movements, concerts, compilations and mixtapes, and out towards larger periods of time, those of genres, subcultures, individual lives and eras (see Chap. 7 for a similar discussion). Musical agency, the influence someone or something has on a body of music, which can be thought of in terms of the four categories presented in Sect. 6.2.2, actually applies at all of these time scales.

For sensible reasons, Live Algorithms focus on the kind of agency that is concentrated in a single performance, defined by Bown et al. (2009) as *performative agency*. But a great deal is lost about the musical process if we strictly narrow our focus to this time scale: in short, what a performer brings to a performance. For a free improvisor, we can think of the information stored in their bodily memory. For a DJ, we can include the material carried in their record box. In all cases, performing individuals bring with them a tradition, embodied in the development of a style, whether through practice, through social interaction or through the construction and configuration of their musical tools and resources (including instruments and bits of data).

It is hard to categorise exactly what is going on in terms of performative agency when you hear the remote influence of performer A in the behaviour of performer B, but it is necessary to consider this process in its broadest sense in order to correctly approach the agency of Live Algorithms. There are many channels through which the work of one individual becomes involved in the work of another, through the imitation of playing or singing styles, cover versions and remixes, the copying of

instruments, effects, orchestration, and more recently through the shared use of software and audio samples.

As well as offering immediate presence on stage, then, Live Algorithms can also involve forms of musical presence occurring at this cultural time scale. The *OMax* system of Assayag et al. (2006), for example, can load style data, and can be used to generate such data through analysis. Here is a powerful new form of culturally transmissible data—style, encoded for use by a generative system—which can spread, evolve, and potentially accumulate complexity through distributed cultural interaction. In this way a system such as *OMax* offers a potential mechanism for bringing a less immediate kind of agency to Live Algorithm performance, reducing the burden of proof through mirroring, although not necessarily achieving the cognitive sophistication of human musical negotiation. In general, a medium term goal for Live Algorithms research may be to find formats in which behaviour can be abstracted and encapsulated in transferrable and modifiable forms, such as file formats that encode styles and behaviours.

Bown et al. (2009) categorise this interaction as memetic agency, an agency that applies outside of an individual performance, which complements performative agency and makes sense of it by accounting for the musical innovation that did not happen there and then on stage. Memetic agency adds an additional temporal layer to the taxonomy of systems presented in Sect. 6.3, which are focused on the timescale of performative agency, by requiring terms for the dynamical change of the elements P, Q and f, the initial conditions of each system, and the configuration of interacting elements, from one system to the next.

The term "memetic" refers loosely to numerous forms of cultural transmission. By strictly focusing on the performative agency of Live Algorithms, all aspects of memetic agency would appear to be left to the algorithm's designer or user: a human. And yet this agency can be critical to understanding a performance. At the extreme, pop singers who mime are almost completely unengaged from the act of musical performance, and yet memetic agency allows us to make sense of such performances. In musical styles such as jazz, much structure is already mapped out and can easily be hard-wired into a Live Algorithm's behaviour, and yet the musical style itself is emergent, not coming from a single human originator, but through repeated listening, copying and mutation. Software is rapidly become a part of this emergent social process. Today, hard-wiring is inevitable at some level in Live Algorithm design, and Live Algorithms designers, as creative practitioners themselves, can gauge the relevance of such factors in specific musical styles and performance contexts. There is nothing wrong with hard-wiring style into a system, and expecting it still to be creative.

However, as the origin of the term implies, memetic agency encompasses a notion of cultural change in which individual humans are not the only agents. Dawkins' original use of the term *meme* referred to a fundamental unit of cultural reproduction, comparable to the biological unit of the gene (Dawkins 1976). As contemporary evolutionary theory emphasises, human agency is predicated on the service of genetic success, and is not an intentionality in and of itself. Memes are just an equivalent hypothesised cause in service of which human behaviour can be manipulated. Individuals may aspire to achieve a more idealised intentionality in the

tradition of the enlightenment, and this is a common way to view musical performance, but whether an individual had achieved such a feat would be hard to prove and to explain.

Between this memetic view of objects as passive participants, secondary agents in the terminology of Gell (1998), and their potential, through AI, to act as creative, human-like, active participants (primary agents). Live Algorithm design seeks a spectrum of agency and influence, rather than a distinct split between the human-like and the artefact-like. We expect to see this agency emerging not just on the stage but off it as well.

6.4.3 Live Algorithms as Musicians

Given that musicians are accustomed to negotiation as a form of improvised musical practice, a Live Algorithm ought to allow musicians to be themselves. There is no necessity to make direct contact through a control interface with the machine, as represented by the unattended systems F and H in Fig. 6.1; such contact might undermine the relationship, both in the eyes of observers, and in fact in any claim to machine autonomy. So, just as in the human world of performance practice, the use of additional tools, novel instruments and experimental interfaces is a matter of aesthetic choice, not practical necessity. Contact ought to be of a more profound, conceptual, nature.

Live Algorithms allow human-machine interactions that preserve the "basic agency relationships" (Godlovitch 1998) we expect in performance. These relationships are developed and expressed in the musical sound itself. Linkage between agent and result need not be absent or vestigial as can easily be the case in the complex, technical world of computer performance. Rather, sonic events operate at different semantic levels, generating both musical affect and effecting communication between players.

Performers are customarily valued for their capacity to demonstrate skill under constraints. This is true even of free improvisation, where the chief constraints relate to the higher-level aspects of group interaction already noted. Whatever is true of human performers must be also be true of Live Algorithms, at least in the imagination of other participants and observers. Arguably, a concert audience attributes value to a performance empathetically, in accordance with Husserl's transcendental concept of "intersubjectivity" (Husserl 1999). Performers are recognised as subjects who are themselves experiencing the world, so intentions and abilities can be attributed to them by the observer, and a critical experience of musicianship and technical accomplishment is experienced in proxy, that is, through empathy with them. Even if the observer cannot play a violin they can develop an empathetic reaction in observing it done, and this is arguably the foundation of the live music experience.

Collective, participatory performance should be considered as a social medium for participants and their audience alike, along with how we can expect Live Algorithms to be regarded as social beings is a matter for imaginative speculation.

In group performance we may see evidence of social "intimacy" (Reis and Shaver 2008) in the extent of evident mutual engagement, i.e. the close—albeit staged—interpersonal relations that occur between players. Intimacy in social psychology is characterised as a reciprocal, "interactional process" that develops between individuals; this is as true of music-making as any imaginable praxis. Intimacy develops when revelatory self-disclosure from one subject in turn finds validation through another's response. This is subsequently interpreted by the subject as evidence of an emergent and binding understanding with the other participant. Intimacies are evidence of psychological proximity, cohesiveness and trust (Prager 1995); trust that a partner can offer what is wanted (or if not, that they can offer what will provide benefit rather than harm). The development of trust occurs in situations that require interdependence, as when experience is shared, and activity and aims co-ordinated ('agentic' cohesiveness), or when there is an apparent need for a reciprocal exchange of information, for mutual control and a state of *quid pro quo* in order to achieve something desirable. All these are significant facets of participatory music performance.

If intimacy is learned over time, through a series of transactions and negotiations, it cannot be designed for in advance. Freely improvised music rests upon this premise as well. To situate a computer in this setting could be a grossly simplistic and anthropomorphising endeavour. But there are instances in which trust is fostered without direct social contact. On-line or computer-mediated intimacy has been studied by Parks and Floyd (1996) showing how trust develops free of non-verbal cues or immediate trust situations. Human-computer musical intimacy might occur in a similarly shared but restricted environment; i.e. the music itself, even though the respective understandings of that environment would differ entirely (Young 2010).

6.5 Prototypes

Many systems exist that aim to satisfy the goal of achieving some or all of the features listed in Sect. 6.2.2 (in general expressing "performative agency" as discussed in Sect. 6.4.2), validating their performative efficacy there and then in a performance context. The fellow performers and audience must be convinced of the autonomy, creativity, participation and leadership of the system through what it does on the stage. For this reason, a successful behaviour for a Live Algorithm is mirroring, performing in deference to the human improvising partner by deriving performance information from it.

A clear example of a mirroring algorithm is François Pachet's *Continuator* system (Pachet 2004). The *Continuator*, from which the term mirroring is borrowed, is explicitly designed to develop improvised responses to a solo performer in the style of that performer, using a Markovian analysis of the performer's input (see also Chap. 5 in this volume). The continuator works in a MIDI domain and performs on a MIDI instrument such as a synthesised piano. Pachet describes this as a tool to achieve creative flow, in which the performer has aspects of their playing style

revealed to them in novel ways, as with a mirror. It is clearly participatory, can lead to novelty through interaction, and is autonomous in its capability to independently infer and reproduce style. The *OMax* system of Assayag et al. (2006) uses a similar framework of behavioural modelling, but is more geared towards the construction of improvising behaviours beyond that gathered by a performer in real-time. As such it can also exhibit leadership.

In terms of our PQf wiring diagrams, such systems are complete Live Algorithms (Fig. 6.1H) typically operating in a MIDI or other music symbolic domain: the f system operates directly on such symbolic data, in tandem with some kind of stored representation of a responsive behavioural strategy, such as a Markov model. Note that here as in other cases, the symbolic form of data flows p and q mean that f can easily be simulated in simpler virtual environments. This can be practical for training purposes.

A number of related systems provide frameworks that straddle the range of behaviours from shadowing to negotiation. Research into granular audio analysis and resynthesis offers a lower-level alternative to MIDI and introduces timbral information to an agent's perceptual world. Casey (2005) proposes a method for dissecting sequences of audio into acoustic lexemes, strings of short timbral/tonal categories. Based on this principle, Casey's *Soundspotter* system (Casey 2009) can be used to match incoming audio from one source with pre-analysed audio from another, offering rich creative potential. Schwarz's *CataRT* system uses a similar mechanism, providing a scatter plot interface to a corpus of pre-analysed audio data (Schwarz et al. 2006).

In its raw form, *Soundspotter* offers a powerful new kind of shadowing (more powerful than the MIDI domain given the kinds of timbral transformations and within-note control it allows), and can be considered more as a novel timbral effect or a creative tool than a Live Algorithm. This fits with the scheme of Fig. 6.1E. The *Soundspotter* framework, however, provides a firm foundation for more generative and interactive use, as demonstrated in *Frank* developed by Plans Casal and Morelli (2007), which introduces a generative process based on a coevolutionary algorithm, effectively introducing a novel f operating on feature data. As with MIDI data, here the data flows p and q take the form of (lower level) symbolic data (lexical, in Casey's terms, Casey 2005), meaning that there is a convenient model for embedding different f's in a stable musical context. Although *Frank* does not directly map input to output, it is able to take advantage of the shadowing nature of the *Soundspotter* system, for example by giving the impression of echoes of musical activity from the audio input. Britton's experiments with chains of feedback in *CataRT* have likewise explored the generative capabilities inherent in Schwarz's concatenative synthesis framework (Schwarz et al. 2006).

Thus whilst MIDI is a well established domain based on musical notation in the Western music tradition, timbral analysis and acoustic lexemes indicate new ways for music to be transformed into a conceptual space and then retransformed into sound. These principles of transformation are key to the formulation of a Live Algorithm, central to which is the identification and isolation of an abstract nested behavioural module, f, which enjoys some degree of transferability between contexts.

In these cases (and elsewhere, as in Fourier transform-based analysis and resynthesis) a compelling feature is that the channels of analysis (p) and synthesis (q) are equivalent, such that if f were simply bypassed $f(x, p) = p$ then the input data would be recovered in some shadowed form. In the extreme case, $Q = P^{-1}$, the input sound comes back unaltered.

The *Swarm Music* and *Swarm Granulator* systems of Blackwell (2001) and Blackwell and Young (2004) explore both MIDI and timbral domains using swarm dynamics. Incoming analysis parameters are mapped onto attractors in an internal space. A swarm of musical events explores this space, on occasion becoming drawn to attractors. The positions of swarm members are mapped directly onto synthesis parameters so that if the individuals were sitting directly on the attractors, the output would precisely mirror the input. Novelty arises from the exploration of phase space, and participation is coded in the tendency to move towards any discovered attractor. *Swarm Music/Granulator* is a direct realisation of the dynamical systems programme advocated in Sect. 6.3.5.

The behaviour of the above prototypes is not always human-like, and indicates how machine improvisers are already super-human in certain respects. Humans have an evolved capacity for vocal imitation, extended to imitation on musical instruments, but the mechanisms for perception and action are far from equivalent, the latter using learnt motor movement to make sound. An individual learns the relationship between action and perception through practice. The capacity for shadowing cannot be taken as given for humans, therefore, not only because our response times are too slow, but because we do not have inherent mechanisms to generate sound in equivalent ways to how we perceive sound. That said, the way we hear sound is deeply influenced by the salience of spoken language to us, a fact which matters in modelling human music perception, and the cognitive methods used by human to achieve this mapping may also turn out be useful to further Live Algorithms research.

A number of Live Algorithm systems also demonstrate a lack of equivalence between input and output interfaces by combining standard hard-wired audio analysis tools (P), domain-general AI techniques or dynamical systems such as neural networks, particle swarms or generative grammars (f), and bespoke hand-programmed generative systems (Q). This modular approach provides the opportunity to integrate human and machine decision-making processes by breaking down the behaviour of Live Algorithms into a set of problems that can be solved in different ways.

Lewis' celebrated *Voyager* system is an example of a system that is hand-coded with complex rule-based behaviour (Lewis 2000). The system uses standard audio analysis tools in order to render incoming audio in a MIDI-based form. *Voyager* is designed in a modular way according to Lewis' introspective detailing of improvisation behaviours. It encodes musical knowledge, acting as a proxy for Lewis' creative agency, and achieves each of our four goals through the success of this knowledge encapsulation. It achieves novelty using simple combinatoric processes.

Young's *Piano_prosthesis* and related prosthesis projects (Young 2008) demonstrate a more hybrid approach using standard analysis tools, such as IRCAM's *yin* ~

object for Max/MSP, to establish an internal representation of the musical environment, a composed generative music system, and a process of machine-learning establishing a connection between the two. A generative music system designed by Young acts as a flexible, parametrically controllable improvisation system with composed elements. A feedforward neural network is then used to learn a set of mappings from the musical environment to the parameters of the generative system in real-time as the performance is taking place. Young's systems exhibit elements of mirroring and shadowing in their generative rules, and come close to a notion of negotiation, as the system continually updates an internal model of what is happening in the music (without predetermined rules governing what is learned or when), which it can try to manipulate.

Similarly, Bown and Lexer (2006) have explored the use of recurrent neural networks that exhibit simple low-level dynamical behaviours such as repetition with variation and coordinated activity with an input. These networks can be embedded in a Live Algorithm by hand-coding connections between standard audio analysis tools and the recurrent neural network at one end, and between the recurrent neural network and a stochastic generative music system at the other end. In an extreme case, the recurrent neural network updates at sample rate, receiving the input audio signal directly, and generating the output audio signal directly from the activation of a single output node.

6.6 Further Considerations

This concluding section offers some directions for future Live Algorithm research. The list cannot be comprehensive since it is impossible to predict which route(s) will further the ultimate aims of the Live Algorithm agenda, but it is expected that these topics will play some part in the process.

6.6.1 Embodiment

Brooks (2009) and other researchers in embodied robotics have argued against the symbolic, representational AI approach to cognition, favouring instead a physically grounded framework in which robots are situated in the world (they deal with the world by perception and immediate behaviour, rather than by abstract representations and symbolic manipulation), and are embodied in the world in the sense that their actions have immediate feedback on their own sensations. The complexity of the environment is a key issue; rather than building fragile approximate models of the world, the embodied approach utilises the world itself in order to pursue a goal. The complexity of the world is (hopefully) tamed by working in and with the world, rather than by attempting to imagine and represent the world. A consequence of this is that embodied and situated systems can themselves have complex, emergent behaviour.

One concomitant implication for a Live Algorithm would be an embodiment that involves a means to play a physical instrument by movement rather than a synthetic production of sound by electronics. The task of learning to play a physical device would involve the development of a better potential to listen to the type of sounds the instrument could make. Hence sensor and actuator components would develop together in a feedback loop. Ultimately the expectation would be that the Live Algorithm would have a greater ability to make and hear complex sounds, an important aspect of human improvisation. Robots move in real space, and have some goal, even if only not to fall over. The analogy for our purposes would be movement in a sonic field. It remains to be seen what goals would be pertinent; perhaps to navigate between two timbres, or to find an interpolative rhythm.

6.6.2 Learning

It is without doubt a feature of improvisation that practitioners improve with time. It would be unreasonable to deny our Live Algorithm the chance to reflect on its own performance and find ways to improve. Consequently, the algorithm must be able to make mistakes. The definition of what a mistake might be for a Live Algorithm raises many fundamental issues.

There are many machine learning techniques that can be imported into the field, but they would all require the existence of some kind of performance metric, as discussed in Sect. 6.3.4. Some kind of objective evaluation of an improvised performance is needed. Such a measure could be developed by the analysis of human group performance. Unfortunately results in this area are lacking. An information-theoretic approach might be fruitful: organisation (and dis-organisation) can be computed over various timescales using entropy and complexity measures. The analysis would have to be then checked against human evaluation.

Ultimately we would require that the Live Algorithm becomes its own critic; should the algorithm feel shame as well as satisfaction?

6.6.3 Anticipated Criticisms

In human-machine dialogue, the human input to the machine is already a source of considerable organisation and information. Many algorithm designers exploit this information either intentionally or by accident. The algorithm ultimately feeds on the inherent musical organisation of the input stream.

In order to guard against this, tests could be set up involving groups of Live Algorithms (i.e. without human performers). If such a group could spontaneously generate new structures there would be more confidence in the ability of the algorithm to create its own patterns within the context of a machine-human dialogue. Interestingly, Miranda (2008) has demonstrated the emergence of songs from interacting robots; giving Live Algorithms the chance to interact with other artificial musicians might provoke growth in unexpected directions.

6.6.4 Cultural Embeddedness

Given the great importance of memetic agency to performance, a grand challenge for Live Algorithms is to expand into this realm of activity, through the implementation of long-term social-musical behaviour: the development of a style over numerous performances and practice, the appropriate absorption of influence, including appropriate copying of resources such as software instruments, raw musical data and audio samples. Social interaction can also be achieved through the development of a reputation and authority amongst a niche of interested individuals, an aspect of memetic agency not considered above since it involves long-term interaction with individuals that are not themselves musical producers. Only by addressing this time scale can we encapsulate the gamut of traits that we associate with human creativity. The emergence of software-mediated social networks makes this possibility more tractable, as the medium through which social musical interaction takes place becomes increasingly amenable to software-sourced agency. Software agents that trawl the pages of on-line musical networks such as MySpace (www.myspace.com) or Last.fm (www.last.fm), and distribute new musical output through their own pages, are already at work.[4]

6.6.5 A Final Note

The field is very active and many approaches are currently being followed. It is hard to guess which direction (whether on our list or not) will ultimately provide the biggest insights. Perhaps progress will be made with large hybrid systems that incorporate self-organising dynamical systems, machine learning, physicality and machine culture.

It should be stressed that the overall objective is not to imitate the practice of human improvisation. We do not need surrogate human performers. The aim is very different from an artificial accompaniment machine, a replacement bass player for example (although such devices may spin-off from Live Algorithm research), since such systems would not be capable of leadership, a fundamental property of Live Algorithms. Rather we seek artificial improvisers that can play alongside humans in a way that enhances our musical experience. We expect that Live Algorithms will give us access to an alien world of computational precision and algorithmic patterning, made accessible through the interface of real-time interaction. We also hope that the study of artificial improvisation will provide insights on the human activity.

Live Algorithms already enjoy an active musical life. The Live Algorithms for Music network[5] provides a nexus for musicians, engineers and cognitive scientists.

[4] For example, the *Cybraphon*: http://www.wired.com/gadgetlab/2009/07/cybraphon/.

[5] See http://www.doc.gold.ac.uk/~mas01tb/LiveAlgorithms/livealgorithms.html.

A workshop in 2009 at Goldsmiths, University of London, with a concert at the alternative London venue Café OTO, attracted a spectrum of systems which took part in a series of duets with internationally renowned improvisers.

The future of computer music is surely an exploitation of the creative potential that intelligent machines may offer, rather than the mundane speeding up of routine tasks or in menu-driven tools. Ideas that lie broadly under the umbrella of Artificial Intelligence and Artificial Life will become increasingly adopted by computer musicians and engineers.

Live Algorithms—performing near you, soon.

Acknowledgements Our thanks to all those who contributed to the Live Algorithms for Music concerts and symposia, the UK Engineering and Physical Sciences Research Council for initial network funding (grant GR/T21479/0) and the Goldsmiths Electronic Music Studios for hosting LAM concerts. Oliver Bown's research was funded by the Australian Research Council under Discovery Project grant DP0877320. We dedicate this chapter to the memory of the late Andrew Gartland-Jones, in acknowledgement of his encouragement and vision during the early days of the LAM network.

References

Assayag, G., Block, G., & Chemillier, M. (2006). Omax-ofon. In *Proceedings of sound and music computing (SMC) 2006*.

Bailey, D. (1993). *Improvisation*. Da Capo Press.

Beer, R. (2000). Dynamical approaches to cognitive science. *Trends in Cognitive Sciences*, *4*(3), 91–99.

Beer, R. D. (1995). On the dynamics of small continuous recurrent neural networks. *Adaptive Behavior*, *3*(4), 469–509.

Bertschinger, N., Olbrich, E., Ay, N., & Jost, J. (2008). Autonomy: an information theoretic perspective. *Biosystems*, *91*(2), 331–345. Modelling Autonomy.

Blackwell, T. M. (2001). *Making music with swarms*. Master's thesis, University College London.

Blackwell, T. M. (2007). Swarming and music. In E. Miranda & A. Biles (Eds.), *Evolutionary computer music*. Berlin: Springer.

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

Blackwell, T., & Young, M. (2004). Self-organised music. *Organised Sound*, *9*(2), 137–150.

Boden, M. (2004). *The creative mind: myths and mechanisms* (2nd ed.). London: Routledge.

Bonabeau, E., Dorigo, M., & Theraulaz, G. (1999). *Swarm intelligence*. London: Oxford University Press.

Bown, O., Eldridge, A., & McCormack, J. (2009). Understanding interaction in contemporary digital music: from instruments to behavioural objects. *Organised Sound*, *14*(02), 188–196.

Bown, O., & Lexer, S. (2006). Continuous-time recurrent neural networks for generative and interactive musical performance. In F. Rothlauf & J. Branke (Eds.), *Applications of evolutionary computing, EvoWorkshops 2006 proceedings*.

Brooks, R. A. (2009). New approaches to robotics. *Science*, *253*, 1227–1232.

Casey, M. (2005). Acoustic lexemes for organizing internet audio. *Contemporary Music Review*, *24*(6), 489–508.

Casey, M. (2009). Soundspotting: a new kind of process? In R. T. Dean (Ed.), *The Oxford handbook of computer music and digital sound culture*. London: Oxford University Press.

Cope, D. (1992). Computer modelling of musical intelligence in EMI. *Computer Music Journal*, *16*(2), 69–83.

Dawkins, R. (1976). *The selfish gene*. London: Oxford University Press.
Flake, G. W. (1998). *The computational beauty of nature: computer explorations of fractals, chaos, complex systems, and adaptation*. Cambridge: MIT Press.
Gell, A. (1998). *Art and agency: an anthropological theory*. Oxford: Clarendon Press.
Godlovitch, S. (1998). *Musical performance: a philoshophical study*. London: Routledge.
Husserl, E. (1999). *Cartesian meditations: an introduction to phenomenology*. Norwell: Kluwer Academic.
Kaplan, D., & Glass, L. (1995). *Understanding non-linear dynamics*. New York: Springer.
Lewis, G. E. (2000). Too many notes: computers, complexity and culture in voyager. *Leonardo Music Journal*, *10*, 33–39.
McCormack, J., Eldridge, A. C., Dorin, A., & McIlwain, P. (2009). Generative algorithms for making music: emergence, evolution, and ecosystems. In R. T. Dean (Ed.), *The Oxford handbook of computer music* (pp. 354–379). New York: Oxford University Press.
McLean, A., & Wiggins, G. A. (2010). Bricolage programming in the creative arts. In *22nd annual psychology of programming interest group*.
Miranda, E. R. (2008). Emergent songs by social robots. *Journal of Experimental and Theoretical Artificial Intelligence*, *20*(4), 319–334.
Pachet, F. (2004). Beyond the cybernetic jam fantasy: the continuator. *IEEE Computer Graphics and Applications*, *24*(1), 31–35.
Parks, M., & Floyd, K. (1996). Making friends in cyberspace. *Journal of Communication*, *46*(1), 80–97.
Plans Casal, D., & Morelli, D. (2007). Remembering the future: an overview of co-evolution in musical improvisation. In *Proceedings of the 2007 international computer music conference*, Copenhagen, Denmark.
Prager, K. (1995). *The psychology of intimacy*. New York: Guildford.
Reis, H., & Shaver, P. (2008). Intimacy as an interpersonal process. In S. Duck et al. (Eds.), *Handbook of personal relationships: theory, research and interventions* (pp. 367–389). Oxford: Wiley.
Russel, S., & Norvig, P. (2003). *Artificial intelligence: a modern approach*. New York: Prentice Hall.
Schwarz, D., Beller, G., Verbrug, B., & Britton, S. (2006). Real-time corpus-based concatenative synthesis with CataRT. In *Proceedings of 9th international conference on digital audio effects*.
Xenakis, I. (2001). *Formalized music: thought and mathematics in music*. Hillsdale: Pendragon Press.
Young, M. (2008). NN music: improvising with a 'living' computer'. In R. Kronland-Martinet et al. (Eds.), *Computer music modelling and retrieval: sense of sounds*. Berlin: Springer.
Young, M. (2010). Identity and intimacy in human-computer improvisation. *Leonardo Music Journal*, *20*, 97.
Young, M., & Bown, O. (2010). Clap-along: a negotiation strategy for creative musical interaction with computational systems. In *Proceedings of the first international conference on computational creativity*.