Guiding human-computer music improvisation: introducing authoring and control with temporal scenarios

Jérôme Nika

Guiding human-computer music improvisation: introducing authoring and control with temporal scenarios

Jérôme NIKA

UNIVERSITÉ PIERRE ET MARIE CURIE

École doctorale Informatique, Télécommunications et Électronique (Paris)

Institut de Recherche et Coordination Acoustique/Musique UMR STMS 9912

Pour obtenir le grade de

DOCTEUR de l'UNIVERSITÉ PIERRE ET MARIE CURIE

Spécialité **Informatique**

soutenue le 16 mai 2016 devant le jury composé de :

Gérard ASSAYAG	Ircam
Gérard BERRY	Collège de France
Emmanuel CHAILLOUX	Université Pierre et Marie Curie
Shlomo DUBNOV	University of California San Diego
George LEWIS	Columbia University New York

Directeurs :	
Gérard ASSAYAG	Ircam
Marc CHEMILLIER	EHESS
Rapporteurs :	
Myriam DESAINTE-CATHERINE	Université de Bordeaux
Shlomo DUBNOV	University of California San Diego

Jérôme Nika: *Guiding human-computer music improvisation: introducing authoring and control with temporal scenarios*, March 2016.

SUPERVISORS:
Marc Chemillier (EHESS)
Gérard Assayag (IRCAM)

Abstract

This thesis focuses on the introduction of authoring and controls in human-computer music improvisation through the use of temporal scenarios to guide or compose interactive performances, and addresses the dialectic between planning and reactivity in interactive music systems dedicated to improvisation.

An interactive system dedicated to music improvisation generates music "on the fly", in relation to the musical context of a live performance. This work follows on researches on machine improvisation seen as the navigation through a musical memory: typically the music played by an "analog" musician co-improvising with the system during a performance or an offline corpus. These researches were mainly dedicated to free improvisation, and we focus here on pulsed and "idiomatic" music.

Within an idiomatic context, an improviser deals with issues of acceptability regarding the stylistic norms and aesthetic values implicitly carried by the musical idiom. This is also the case for an interactive music system that would like to play jazz, blues, or rock... without being limited to imperative rules that would not allow any kind of transgression or digression. Various repertoires of improvised music rely on a formalized and temporally structured object, for example a harmonic progression in jazz improvisation. The same way, the models and architecture we developed rely on a formal temporal structure. This structure does not carry the narrative dimension of the improvisation, that is its fundamentally aesthetic and non-explicit evolution, but is a sequence of formalized constraints for the machine improvisation. This thesis thus presents: a music generation model guided by a "scenario" introducing mechanisms of anticipation; a framework to compose improvised interactive performances at the "scenario" level; an architecture combining anticipatory behavior with reactivity using mixed static/dynamic scheduling techniques; an audio rendering module to perform live re-injection of captured material in synchrony with a non-metronomic beat; a study carried out with ten musicians through performances, work sessions, listening sessions and interviews.

First, we propose a music generation model guided by a formal structure. In this framework "improvising" means navigating through an indexed memory to collect some contiguous or disconnected sequences matching the successive parts of a "scenario" guiding the improvisation (for example a chord progression). The musical purpose of the scenario is to ensure the conformity of the improvisations generated by the machine to the idiom it carries, and to introduce an-

ticipation mechanisms in the generation process, by analogy with a musician anticipating the resolution of a harmonic progression.

Using the formal genericity of the couple “scenario / memory”, we sketch a protocol to compose improvisation sessions at the scenario level. Defining scenarios described using audio-musical descriptors or any user-defined alphabet can lead to approach others dimensions of guided interactive improvisation. In this framework, musicians for whom the definition of a musical alphabet and the design of scenarios for improvisation is part of the creative process can be involved upstream, in the “meta-level of composition” consisting in the design of the musical language of the machine.

This model can be used in a compositional workflow and is “offline” in the sense that one run produces a whole timed and structured musical gesture satisfying the designed scenario that will then be unfolded through time during performance. We present then a dynamic architecture embedding such generation processes with formal specifications in order to combine anticipation and reactivity in a context of guided improvisation. In this context, a reaction of the system to the external environment, such as control interfaces or live players input, cannot only be seen as a spontaneous instant response. Indeed, it has to take advantage of the knowledge of this temporal structure to benefit from anticipatory behavior. A reaction can be considered as a revision of mid-term anticipations, musical sequences previously generated by the system ahead of the time of the performance, in the light of new events or controls. To cope with the issue of combining long-term planning and reactivity, we therefore propose to model guided improvisation as dynamic calls to “compositional” processes, that it to say to embed intrinsically offline generation models in a reactive architecture. In order to be able to play with the musicians, and with the sound of the musicians, this architecture includes a novel audio rendering module that enables to improvise by re-injecting live audio material (processed and transformed online to match the scenario) in synchrony with a non-metronomic fluctuating pulse.

Finally, this work fully integrated the results of frequent interactions with expert musicians to the iterative design of the models and architectures. These latter are implemented in the interactive music system ImproteK, one of the offspring of the OMax system, that was used at various occasions during live performances with improvisers. During these collaborations, work sessions were associated to listening sessions and interviews to gather the evaluations of the musicians on the system in order to validate and refine the scientific and technological choices.

Résumé

Cette thèse propose l'introduction de scénarios temporels pour guider ou composer l'improvisation musicale homme-machine et étudie la dialectique entre planification et réactivité dans les systèmes interactifs dédiés à l'improvisation. On fait ici référence à des systèmes informatiques capables de produire de la musique en relation directe avec le contexte musical produit par une situation de concert. Ces travaux s'inscrivent dans la lignée des recherches sur la modélisation du style et l'improvisation automatique vue comme la navigation à travers une mémoire musicale provenant du jeu d'un musicien « analogique » improvisant aux côtés du système ou d'un corpus préalablement appris.

On cherche ici à appréhender l'improvisation pulsée et dite « idiomatique » (c'est-à-dire se référant à un idiome particulier) en opposition à l'improvisation « non idiomatique » à laquelle étaient dédiées les recherches mentionnées précédemment. Dans le cas idiomatique, l'improvisateur est confronté à des questions d'acceptabilité au regard de l'idiome. Ces questions se posent donc également à un système d'improvisation dédié à l'improvisation jazz, blues, rock... sans être pour autant limité à des règles impératives interdisant toute transgression et digression. En s'appuyant sur l'existence d'une structure formalisée antérieure à la performance dans de nombreux répertoires improvisés (une « grille d'accords » par exemple) ces travaux proposent : un modèle d'improvisation guidée par un « scénario » introduisant des mécanismes d'anticipation ; une architecture temporelle hybride combinant anticipation et réactivité ; et un cadre pour composer des sessions d'improvisation idiomatique ou non à l'échelle du scénario en exploitant la généricité des modèles.

On décrira donc tout d'abord un modèle pour l'improvisation musicale guidée par une structure formelle. Dans ce cadre, « improviser » signifie articuler une mémoire musicale et un « scénario » guidant l'improvisation, une « grille d'accords » dans l'improvisation jazz par exemple. Ce modèle permet d'assurer la conformité des improvisations de la machine au scénario, et utilise la connaissance a priori de la structure temporelle de l'improvisation pour introduire des mécanismes d'anticipation dans le processus de génération musicale, à la manière d'un musicien prévoyant la résolution d'une cadence.

Ce modèle peut être utilisé dans un processus compositionnel et est intrinsèquement « hors temps » puisqu'une de ses exécutions produit une séquence complète qui sera ensuite déroulée dans le temps. On présentera ensuite son intégration dans le cadre dynamique de l'improvisation guidée. Dans ce contexte, une « réaction » ne peut

pas être vue comme une réponse épidermique et instantanée mais doit tirer profit de la connaissance du scénario pour s'inscrire dans le temps. On considèrera donc une réaction comme une révision des anticipations à court-terme à la lumière de nouveaux évènements. La question de la conciliation entre planification long-terme et réactivité est abordée en modélisant l'improvisation guidée comme des appels dynamiques à des processus statiques, c'est-à-dire des appels « en temps » à un modèle compositionnel. Pour pouvoir jouer avec des musiciens et en utilisant le son de ces musiciens, cette architecture propose également un module de rendu audio permettant d'improviser en réinjectant le son des co-improvisateurs, traité et transformé en temps-réel pour satisfaire le scénario d'improvisation, tout en étant synchronisé avec le temps réel de la performance, mesuré par un tempo possiblement fluctuant.

Enfin, la généricité du couple « scénario / mémoire » et la possibilité de définir des scénarios dynamiques incitent à explorer d'autres directions que l'improvisation jazz. Des scénarios décrits avec un alphabet spécifique à un projet musical ou en termes de descripteurs audio-musicaux permettent d'aborder d'autres modes de guidage de l'improvisation musicale. De cette manière, les musiciens pour qui la définition d'un alphabet musical et la conception de scénarios d'improvisation font partie intégrante du processus créatif peuvent être impliqués en amont de la performance.

Ces recherches ont été menées en interaction constante avec des musiciens experts, en intégrant pleinement ces collaborations au processus itératif de conception des modèles et architectures. Ceux-ci ont été implémentés dans le système ImproteK, utilisé à de nombreuses reprises lors de performances avec des improvisateurs. Au cours de ces collaborations, les sessions d'expérimentations ont été associées à des entretiens et séances de réécoute afin de recueillir de nombreuses appréciations formulées par les musiciens pour valider et affiner les choix technologiques.

"

— ?

— Prendre des p'tits bouts d'trucs
et puis les assembler ensemble.

(Stupéflip, *L.E.C.R.O.U.*)

— Ici ! Oui !

(Valère Novarina, *L'opérette imaginaire.*)

— 'know what I'm sayin' ?

(Grand Puba, *Ya know how it goes.*)

— Oui.

(Érik Satie, *Mémoires d'un amnésique.*)

"

Acknowledgments

Merci à Marc et Gérard. Merci à Jean et Jean-Louis. Merci à Sylvie. Merci à Gérard Berry, Emmanuel Chailloux, Myriam Desainte-Catherine, Shlomo Dubnov, et George Lewis. Merci à Maxime Crochemore. Merci à Bernard Lubat, Rémi Fox, Jovino Santos Neto, Kilema, Charles Kely, Louis Mazetier, Michelle Agnès Magalhaes, Hervé Sellin, Georges Bloch, Benoît Delbecq, Jozef Dumoulin, Ashley Slater, et Gilbert Nouno. Merci à Fabrice Vieira. Merci à Arshia, Laurent, Benjamin, Philippe, Axel, et Carlos. Merci à Benjamin, Mehdi, Quentin, Geoffroy, Guillaume, Adrien, Sébastien. Merci à Mathieu, Victor, Jean-Gauthier. Merci à Jérémie et Louis. Merci à Aymeric et Pierre. Merci à Hélianthe et Dimitri, et au BAM. Merci à la famille. Merci à Etienne et aux Gascons. Merci aux tonneaux. Merci à Jules. Merci à José. Merci à Jack. Merci à Frank et Yaya-smine. Merci à Seu, Clément, Kevin Petyt, et Margot.

Contents

List of Figures

1 Introduction

1.1 Scope of the Thesis

This thesis focuses on the introduction of structures, authoring, and controls in human-computer music improvisation through the use of temporal scenarios to guide or compose interactive performances, and addresses the dialectic between planning and reactivity in interactive music systems dedicated to improvisation. This work follows on researches on machine improvisation seen as the navigation through a musical "memory" (see Chapter 2) which may consist of an offline corpus or of the continuous capture of the live music played by a human musician co-improvising with the system during a performance. These researches were mainly dedicated to free - generally non pulsed - improvisation: the work presented here will focus on idiomatic music - which generally respects a defined pulse - and extends to the general topic of composed improvisational frames, thus moving beyond the issue of practicing established idioms.

When an improviser is playing Be-bop or New Orleans for example, the notes, rhythms, phrasing... are not fully determined in a formalized way by written materials or rules, nevertheless the musician is playing "Be-bop" or "New Orleans". Bailey (1993) distinguishes *idiomatic* and *non-idiomatic* improvisation:

> Idiomatic improvisation "is mainly concerned with the expression of an idiom - such as jazz, flamenco or baroque - and takes its identity and motivation from that idiom. [...] Non-idiomatic improvisation has other concerns and is most usually found in so called 'free' improvisation and, while it can be highly stylized, is not usually tied to represent an idiomatic identity."

Within an idiomatic context, an improviser deals with issues of acceptability regarding the stylistic norms and aesthetic values implicitly carried by the musical idiom. This is also the case for an interactive music system that would like to play jazz, blues, or rock... without being limited to imperative rules that would not allow any kind of transgression or digression.

Various repertoires of improvised music rely on a formalized and temporally structured object, for example a harmonic progression in jazz improvisation. The same way, we aim at designing generation

models and architectures for human-computer improvisation relying on a generalized formal temporal structure that we call *scenario*.

1.2 Background and Motivation

This section presents a brief overview of the musical, ethnomusicological and philosophical background constituting the motivation and metaphorical inspirations for the design of the generation models and architectures proposed in this thesis. It focuses on the main topics addressed in the dissertation: formalized temporal structures in music improvisation and their articulations with a musical memory, anticipatory behavior, and reactivity.

1.2.1 Scenario

Jazz, blues, or rock improvisation generally relies on a chord progression defining a guideline for the performance. For ethnomusicologist Lortat-Jacob (2007), a jazz standard like the example in Figure 1.1 is anything but a standard, it is a rather a basis or an incipit that the musicians use to develop their improvisations.

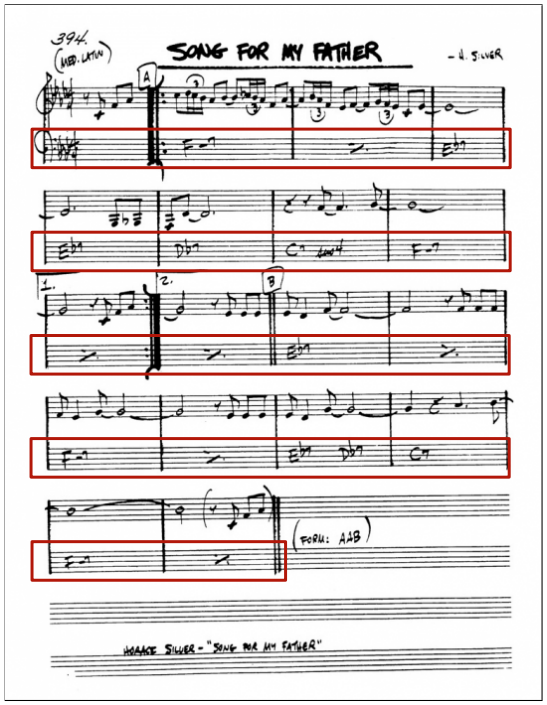


Figure 1.1: Chord progression of the jazz standard *Song for my father* (Horace Silver). From the *New Real Book* (Sher, 2005).

The baroque basso continuo leaves it to the interpret to realize the harmony by improvising with the right hand from the figured bass written for the left hand as in the example of Figure 1.2. Improvisation is guided on different levels in the indian raga: chaining of delimited parts with different lengths and specific identities at the higher level, evolution within each of these parts built from detailed descriptions in terms of melody, tempo, or register...

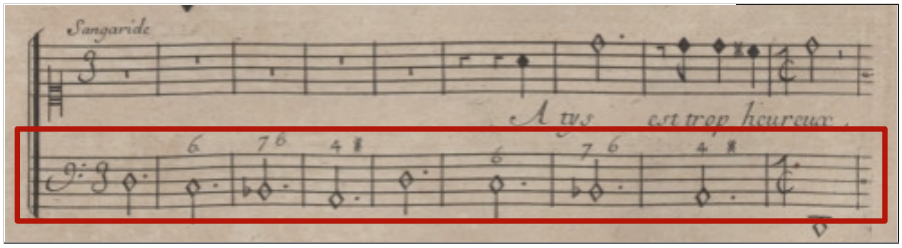


Figure 1.2: Figured bass extracted from *Atys* (Lully). Second edition by Henri de Baussen, 1708 (source: operacritiques.free.fr).

Such a formalized and temporally structured object - not necessarily described with a harmonic vocabulary - can be found in various repertoires of idiomatic improvisation, and also in contemporary music. In 1957, composer Edgard Varèse conducted improvisation workshops with jazzmen, including Art Farmer and Charles Mingus, in a series of Sunday afternoons organized by Earle Brown and in the presence of John Cage.

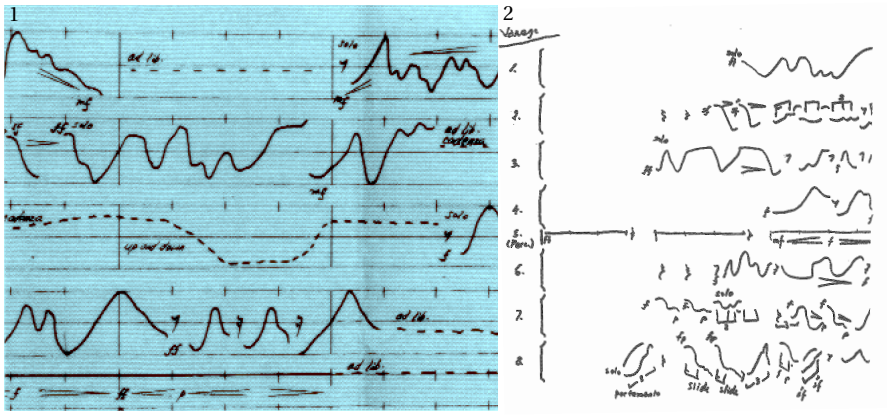


Figure 1.3: Edgard Varèse: 1. Extract of *Poème électronique*, 1958 (source: giorgiomagnanensi.com) ; 2. Portion of an untitled graphic score for improvisation, 1957 (from Johnson, 2012).

These experiments, described by Johnson (2012), relied on graphic sketches composed by Varèse which served as bases for improvisation (Varèse used then certain extracts of the workshops for his *Poème électronique*, see Figure 1.3 (1)). As illustrated in Figure 1.3 (2), these scores contained eight lines representing eight undetermined instru-

ments (excepted the percussions on line 5). Each line was an indication of general pitch contours, scattered rhythms, and precise indication of dynamic levels. Johnson reports that Cage, when he was asked how the piece sounded, replied "it sounded like Varèse". Indeed, he wrote later in 1958:

> "Recently, [Varèse] has found a notation for jazz improvisation of a form controlled by himself. Though the specific notes are not determined by him, the amplitudes are; they are characteristic of his imagination, and the improvisations, though somewhat indeterminate, sound like his other works." (Cage, 1973)

Graphic scores and other related forms such as textual scores and verbal scores played an important role in free improvisation (see Saladin, 2004), and, in particular, in the composerly avant-garde of the 1960s and 1970s (see Lewis, 2006). The main topic of this thesis being the design of generation models and architectures to perform idiomatic improvisation with a computer, we will not venture to draw any parallel between the chord progression of a jazz standard and a graphic score. Indeed, their relation to the actual music as well as their representation of time may be totally different: a chord progression involves a discretized time reference, while temporal graphic scores may define temporal evolutions without explicit absolute or relative time reference. Moreover, the notations and alphabets cannot be compared since a chord label represents a formal specification whose meaning is "universal", while the graphic notation elaborated by a composer can be very personal and describe the music through metaphorical and poetic associations.

Nevertheless, with the example of "organized sound" by Varèse and the comments by Cage, we simply underline the fact that these different sequences formalize musical organizations as *temporal* "virtual scores" (Manoury, 1990): the *nature* of the parameters taken into account is known as well as their sequentiality, but not their exact values, these latter being determined during the performance. These temporal structures can be seen as sequences of equivalence classes that the actual music will instantiate, the equivalence classes being defined by a chosen alphabet. Furthermore, contrary to the example above which defines a line by musician, a graphic, textual, or verbal score may be a common referent for all the performers, and, like a chord progression, become a shared temporal object "setting a collective reunion" (Saladin, 2004).

Outside the musical scope, this notion can be transposed to commedia dell'arte whose *canovaccio* outlines the sequence of events and situations of the plot that the representation will follow, or to the oral tradition of folktales or epic tales for which the improvised narration is based on an outline made of elementary units, sometimes

associated to a set of established formulas defining a grammar of the narration (Finley, 1954).

The idea of a preexisting temporal structure guiding the improvised performance is studied in the cognitive works dealing with improvisation which address the issue of planning associated with the need for reactivity characterizing improvisation. This temporal entity is broader than the formal scenario because it covers all the narrative strategies of an improviser. For example, the notion of *plan* for Sloboda (1982) or Shaffer (1980) is defined as an abstract object symbolizing the fundamental structure of the performance, letting the finer dimensions to be generated or organized in due time. Beyond the mere sequence of formalized constraints, Pressing (1984) introduces the *referent* as an underlying scheme guiding or aiding the production of the musical material. The referent extends to cognitive, perceptive or emotional dimensions, and its relation with the improvised behavior can be, among others, metaphoric, imitative, allegoric, or antagonistic.

Motivation: In order to design models to address *idiomatic improvisation* with interactive music systems, this background incites first to introduce a formalized temporal structure guiding the music generation process that we call *scenario*. In first approach, this scenario aims at ensuring the *conformity* of the machine improvisation: this object does not carry the narrative dimension of the improvisation, that is its fundamentally aesthetic and non-explicit evolution, but is a sequence of formalized equivalence classes for the machine improvisation.

Second, defining *generic* and extensible formal mechanisms independent of the alphabet on which the scenario is defined can widen the perspectives and enable to experiment *composed improvisation* by using scenarios to introduce *authoring* and *meta-composition* in machine improvisation in different musical scopes.

1.2.2 Scenario, Progression and Anticipatory Behavior

The first chapter of the book *Structural functions of harmony* by Schoenberg and Stein (1969) begins with the following considerations about *successions* and *progressions* of chords:

> "A *succession* is aimless; a *progression* aims for a definite goal. Whether such a goal may be reached depends on the continuation. It might promote this aim; it might counteract it. A *progression* has the function of establishing or contradicting a tonality. The combination of harmonies of which a progression consists depends on its purpose - whether it is establishment, modulation, transition, con-

> trast, or reaffirmation. A *succession* of chords may be *functionless*, neither expressing an unmistakable tonality nor requiring a definite continuation."

This fundamental distinction underlines the fact that a *progression* is oriented towards its future. Thus, it carries more than the step-by-step conformity of a *succession*. When envisaging an improvisation *scenario* as a progression (harmonic or not), its sequentiality has therefore to be exploited to introduce such motion or intention at each step of the generation.

In his work aiming at modeling musical anticipation, Cont (2008b) defines *anticipation* as "an action that a system takes as a result of *prediction*, based on current belief or *expectations*, including actions on its own internal state or belief". The notion of anticipation in this framework is not separate from that of *expectation*, and is based on the chain "*expectation* → *prediction* → *anticipation*" formalized by Huron (2006).

If we limit our scope for the moment to the relation between a musician and a scenario serving as a basis for improvisation, an explicit temporal *specification* takes the place of prediction and expectation in the definition given above. To illustrate this idea, we mention the example of *harmonic anticipation* in jazz improvisation which is defined in relation to a given chord progression.

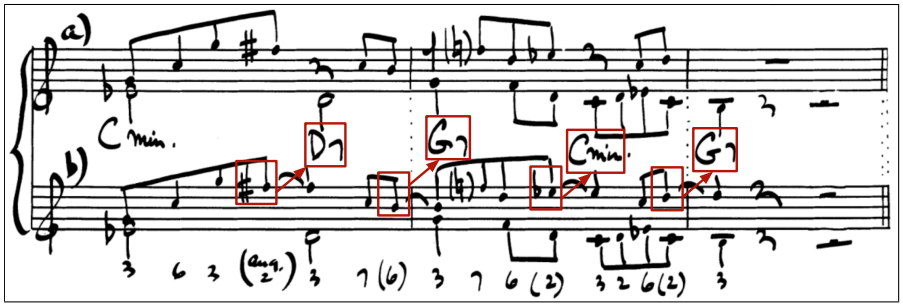


Figure 1.4: Examples of harmonic anticipations (from Russo, 1997).

As in the examples in Figure 1.4, a harmonic anticipation occurs when a note is played before the chord to which the note belongs, and then resolves when the anticipated chord is reached. Usually, this technique creates a sense of *forward motion*. Such a harmonic anticipation could be generated by an improvisation system only because it knew in advance the scenario, so anticipation in this case is the mere result of sharing the knowledge of the future (the scenario) between all the actors[1].

1 The example we give to illustrate this idea is very local. In general, the notion of *progression* that we want to address here involves a long-term direction, *e.g.* going from the tonic to the tonic in 4 or 8 measures in the first phrase of a chorale, going from the tonic to the dominant in 16 or 32 measures, etc.

Motivation: We want the scenario guiding the machine improvisation to be able to be a *succession* as well as a *progression*. In this last case, the aim is to take advantage of the prior knowledge of the structure to introduce anticipatory behavior in order to foster forward motions regarding harmony or phrasing for example. In the scope of this thesis, there is no expectation nor prediction involved but a formalized and explicit *temporal specification* provided by the scenario. The notion of *anticipation* is thus defined in relation to this specification. An *anticipatory behavior* will therefore refer to the fact that the future of the scenario is taken into account when generating each event of the improvisation. In other words, in an offline context, generating an event corresponding to the date T of the scenario is a response to a query asking to generate a musical sequence matching the scenario *from* the date T; in an online context, generating current date T of the improvisation means generating an *anticipation*: a sequence matching the scenario beginning at time T and ending ahead of the current performance time.

1.2.3 Scenario and Memory

In the 17^{th} century, Andrea Perrucci wrote in his treatise on commedia dell'arte (Perrucci et al., 2008):

> "It is not by stripping oneself entirely of scripted material that one should take up the challenge; rather, one should be armed with some general compositions that can be adapted to every kind of comedy."

These "general compositions" can be pre-written elements, lazzi, quotations... and in music correspond to "clichés", licks, programmed muscular routines (Sudnow, 1978), reminiscences of things previously heard, or more generally elements and mechanisms that come from the background, the training, the history and the experience of the improviser. According to philosopher Citton (2015), this quotation breaks any kind of exclusive opposition between "program" and "gesture", and thinking improvisation amounts to standing back in order to assess the naive notion of immanence.

A similar idea was expressed by jazzman Bernard Lubat during an interview we carried out (see Part IV, Chapter 17):

> "We are a music sheet that is not entirely wrapped up, never performed the same way, always the same but never alike. Improvising is triggering this secret score, gradually growing over the years, like a garden. You don't improvise from nothing, you improvise from an accumulation of data that we put I don't know where: in know-how, in

> muscles and nerves, in your mind, in love, in sickness, everywhere. Some people think improvising is a matter of doing something you never have done before, that there is sense of morality. People who criticize my improvisations say "but at times it is always the same", they're looking for ethic purity, for a divine apparition, it is almost mystical. That is a load of rubbish, it doesn't exist."

Motivation: Some creators draw on their "secret score" and thanks to it adapt to a chord progression, to the plot of a comedy... The improvisation process will therefore be formally modeled as the articulation between a *scenario* and a *memory*, and rely on re-injections, transformations and re-contextualizations of elements that have been eared or played in a different *context* defined with the same vocabulary. In this view, musicality lies in the anticipated or unexpected nature of these re-injections. Therefore, to avoid the rigidity of drawing patterns from a set of formula formally associated to given progressions (see Siron, 2007), we want to capture the possible associations between a scenario and a sequential memory using the information on the *context* of the elements in the memory rather than their *nature*.

Through the philosophy of Derrida, Chemillier (2009) discussed the apparent paradox he calls "unpredictable event and computational anticipation" when a machine supposed to take part in an "improvisation" relies on mechanisms of cloning and recombination of existing material. Indeed, according to Derrida (2004) an event which is planned has already occurred and is not an event anymore. Furthermore, transposing his reflexion on technology, he considers that a musical aesthetics such as that of saxophonist Ornette Coleman is intrinsically in opposition with the notions of computation, program, and cloning (Derrida, 1997). Nevertheless, Marc Chemillier concludes that the interaction between a human and a machine is a way to make space for the "incalculable" part of the event.

1.2.4 Scenario, Reaction, and Anticipation

The notion of event naturally leads to that of reaction. Jazzman Hervé Sellin (see Part IV, Section 18.2) continued the reflexion about improvisation as "triggering a secret score" quoted above with an explicit reference to reactivity:

> "One can think that true improvisation should be performed with an empty brain, reacting to what happens, here, right now, but we have lots of ready-made things which come up. In fact, improvisation comes from how quickly you use one material or another. If there is an art, or a talent, it is to be found at this level".

During the same interview, he distinguished *event-driven reaction*, a response a to salient element with a salient element, and *music-driven reaction*, a response to a longer musical discourse with a longer musical discourse. This is this particular view of reaction over time that we want to address here.

So far, we have situated this chapter in a context where a (human or computer) improviser did not have to take into account exogenous information in addition to the scenario. Roughly simplifying, it corresponded to the case of *solo* improvisation. A *collective* improvisation relying on a scenario requires to combine long-term planning with reactivity to the environment. This dialectic between planning and reaction has been approached from a wide range of disciplines, including psychology, cognitive science, and neuroscience. Among them, *action planning* (Prinz, 1997) studies the interaction between perceived events and planned actions when a gesture is prepared according to a given execution plan and has to be conciliated with new data from the environment. These interactions observed in a system of two agents (called *joint actions* in psychology) have been applied to music, and in particular regarding synchronization (Keller, 2008). The *forward models* or *efference copy models*, coming from the field of motor control and now widely used in the field of speech production (Hickok, 2012), are closer to our issue: an internal strategy produces an anticipation of the output of the system (*efference copy*) and is compared then to the actual output. A *reaction* is therefore seen as a *revision* of the strategy or of the output of the system.

In a collective improvisation, the previously introduced notions of conformity and forward motion regarding the scenario are preserved and exploited when facing "unpredictable" events. To illustrate this point, anthropologist Bonnerave (2010) gives the example of a bebop bass player who plays over an indicative chord progression in a different way every time, taking into account the rhythmic calls of the drummers, the accents of the pianist, etc. Studying the combination of planning and reactivity in music improvisation through the notion of "pré-voyance" introduced by Bourdieu (1972), Bonnerave considers that a soloist or accompanist improviser senses interaction in its immediacy in order to bring out possibilities of prolongations which are present in a latent state.

The notion of possible continuations for the future already existing in present time and selected depending on interaction underlines the link between reaction and anticipation in improvisation. This idea is closely related to that of Lortat-Jacob (2007), for whom improvisers are involved in a permanent process of anticipation (pointing out the fact that this analysis is not valid for some performances of contemporary free-jazz). Drawing insights from various theorists of music improvisation to study improvisation in the social and political life, philosopher Citton (2013) even considers that anticipation is an in-

tegral part of improvisation. Indeed, he explains that improvisation in our daily actions results from a certain mix of *on-the-spot improvisation* "thanks to which I can integrate the unavoidable novelty encountered in each situation", and *compositional foresight* "thanks to which I attempt to anticipate the future effects of my current behavior according to my rational understanding of the laws of nature".

> **Motivation:** We aim at designing models and architectures that can take part in collective improvisations. First, this requires to combine long-term planning provided by the scenario and reactivity to the environment, such as control interfaces or live players input. Is this context, reactivity has to take advantage of the prior knowledge of the scenario to benefit from a *compositional foresight* according to the internalization of the temporal structure. A *reaction* to an external event is therefore seen as a *revision* of previously generated anticipations matching the scenario. Furthermore, the integration of an interactive system to a collective improvisation in a context of idiomatic and pulsed music requires that it can be in turns *master or follower of a non-metronomic tempo*. Finally, this high level approach of reactivity incites to define generic and extensible reaction mechanisms that enable to *compose reactivity* as well as give musical controls to an *operator-musician*.

1.3 Outline of the Contributions Presented in the Thesis

This thesis presents a number of novel contributions to the field of computer music through the introduction of temporal scenarios to introduce authoring and control in interactive performance. It proposes new paradigms and models covering the entire chain of human-computer improvisation from generation to rendering. This work resulted in the design of generation models and architectures gathered in a novel interactive music system dedicated to human-computer improvisation in an idiomatic and pulsed context, ImproteK, that was used at numerous occasions during live performances with improvisers. The actual chronology of the work described in this thesis was a constant back and forth between music and science since it was developed in continuous interaction with experts musicians in order to validate and refine the scientific and technological choices. This incremental process simultaneously addressed different topics that are presented in this dissertation following a thematic outline: "Intentions", introducing temporal specifications in generation processes; "Anticipations", combining long-term planning and reactivity; "Playing", synchronization, rendering, and controls in a context of performance; "Practicing", performances, work sessions, and interviews with expert musicians.

Video A.1.1
ImproteK: compilation of video extracts

Hyperlink video *(or vimeo.com/jeromenika/improtek-compilation). See Appendix A: index of the videos cited in the thesis (descriptions and links).*

In the introduction chapter, we gave the musical background motivating the work presented in this dissertation. It is completed by the scientific and technical motivation presented through an overview of the related work in Chapter 2.

Part I, "Intentions": Composing Music Generation processes at the Scenario Level proposes a music generation model relying on a formal temporal structure.
Chapter 3 summarizes the contributions of Part I.
Chapter 4 introduces the principle of the scenario / memory generation model: ensuring the conformity to a predefined temporal structure and taking advantage of this prior knowledge to introduce anticipatory behavior.
Chapter 5 details the algorithms involved in the scenario / memory generation model, combining continuity with the future of the scenario and continuity with the past of the memory.
Chapter 6 focuses on the genericity of the model and introduces a protocol to define an alphabet, its properties, and associated transformations to go from conformity to an idiomatic structure to composition of improvisation sessions at the scenario level.

Part II, "Anticipations": Guided Improvisation as Dynamic Calls to an Offline Generation Model introduces the paradigm of guided improvisation modeled as a compositional process embedded in a reactive architecture in order to combine long-term planning and reactivity.
Chapter 7 summarizes the contributions of Part II.
Chapter 8 presents the general architecture of the ImproteK system, and how the scenario / memory generation model introduced in Part I is used in a real-time context to generate anticipations ahead of the performance time. The following chapters detail the agents constituting this architecture.
Chapter 9 proposes a model of reactive agent, the *Improvisation Handler*, handling dynamic calls to a generation model relying on a formal temporal specification to introduce a notion of reaction "over time". This agent reacts to external events by composing new mid-term anticipations matching the scenario ahead of performance time.
Chapter 10 models the interface between the environment and a dynamic generation process as a *Dynamic Score*: a reactive program driven by an external time source orchestrating the upstream processes (generation queries) as well as the downstream processes (rendering), and managing high-level temporal specifications.

Part III, "Playing" with the (Sound of the) Musicians focuses on adaptive rendering and synchronization of evolving musical sequences coming from dynamic generation processes using live external inputs, and presents some associated expressive musical controls.
Chapter 11 summarizes the contributions of Part III.
Chapter 12 introduces two architectures coping with dynamic musical sequences which are revised during the rendering.
Chapter 13 describes a performance-oriented architecture which offers adaptive rendering of dynamic multimedia sequences generated from live inputs. This autonomous architecture is designed to record and segment a live stream into beat-events that can immediately be played in synchrony with a non-metronomic pulse, according to a user-defined dynamic time mapping.
Chapter 14 focuses on how to use the models implemented in the ImproteK system as a software instrument offering declarative controls on the "intentions" impacting on generation, and temporal controls impacting on rendering.
Chapter 15 presents a rendering architecture dedicated to composition of guided musical processes using an offline memory.

Part IV, "Practicing": Let the Music(ians) (Pl/S)ay describes some collaborations with expert improvisers during performances and work sessions.
Chapter 16 summarizes the approach presented in Part IV: these interactions were an integral part of the iterative development of the models and of the ImproteK system. The public performances and work sessions were associated to listening sessions and interviews to gather numerous judgements expressed by the musicians in order to validate and refine the scientific and technological choices.
Chapter 17 focuses on the long-term collaboration with jazzman Bernard Lubat that led, through experimentation sessions and public performances, to the development of the first models and the first functional prototype of ImproteK.
Chapter 18 covers work carried out during this thesis with eight musicians to explore different idioms and types of interactions: Rémi Fox, Hervé Sellin, Jovino Santos Neto, Michelle Agnes Magalhaes, Louis Mazetier, Velonjoro, Kilema, and Charles Kely.

Part V concludes by summarizing the contributions and perspectives of the thesis.

Appendix A lists the videos referenced in the dissertation: performances and work sessions using ImproteK with expert musicians (A.1); demonstrations, early works, and experiments (A.2); extracts of work sessions with Bernard Lubat when designing the first models (A.3); interviews and listening sessions with musicians (A.4); archives of other collaborations briefly mentioned in the dissertation (A.5).

Appendix B gives extra material concerning the implementation of the generation models and architectures presented in the thesis.

Appendix C gives extra material concerning the listening sessions and interviews carried out with Hervé Sellin, one of the musicians who played with the system.

1.4 Publications

The work presented in this thesis led to several publications and submissions listed in this section. It was awarded the **Young Researcher Prize in Science and Music 2015** (attributed by AFIM, French Association of Computer Music; INRIA, French Institute for Research in Computer Science and Automation; IRISA, Institute for Research in IT and Random Systems; and Rennes University), and **AFIM Young Researcher Prize 2016**.

International Journals

Jérôme Nika, Marc Chemillier and Gérard Assayag. "ImproteK: introducing scenarios into human-computer music improvisation", *ACM Computers in Entertainment, Special issue on Musical Metacreation*, 2016. `[Accepted, forthcoming]`

Dimitri Bouche, Jérôme Nika, Alex Chechile and Jean Bresson. "Computer-aided composition of musical processes", *Journal of New Music Research*. `[Accepted, forthcoming]`

National Journals

Jérôme Nika and Marc Chemillier. "Improvisation musicale homme-machine guidée par un scénario temporel", *Technique et Science Informatique (TSI), Special issue on Computer Music*, vol. 33, n° 7-8, 2014.

Marc Chemillier and Jérôme Nika. ""Étrangement musical" : les jugements de goût de Bernard Lubat à propos du logiciel d'improvisation ImproteK", *Cahiers d'ethnomusicologie*, n° 28, 2015.

Marc Chemillier, Jean Pouchelon, Julien Andre and Jérôme Nika. "La contramétricité dans les musiques traditionnelles africaines et son rapport au jazz", *Anthropologie et société*, vol. 38, no. 1, 2014.

Proceedings of International Peer Reviewed Conferences

Jérôme Nika, Dimitri Bouche, Jean Bresson, Marc Chemillier and Gérard Assayag. "Guided improvisation as dynamic calls to an offline model", *Sound and Music Computing conference SMC*, Maynooth, Ireland, 2015.

Jérôme Nika, José Echeveste, Marc Chemillier and Jean-Louis Giavitto. "Planning Human-Computer Improvisation", *Proceedings of the International Computer Music Conference ICMC 2014*, Athens, Greece, 2014.

Jérôme Nika and Marc Chemillier. "ImproteK, integrating harmonic controls into improvisation in the filiation of OMax", *Proceedings of the International Computer Music Conference ICMC 2012*, Ljubljana, Slovenia 2012, pages 180-187.

Proceedings of National Peer Reviewed Conferences

Jérôme Nika, Marc Chemillier and Gérard Assayag. "Guider l'improvisation musicale homme-machine : une synthèse sur le système ImproteK", *Proceedings of Journées d'informatique musicale JIM 2016*, Albi.

Jérôme Nika and Marc Chemillier. "ImproteK : intégrer des contrôles harmoniques pour l'improvisation musicale dans la filiation d'OMax", *Proceedings of Journées d'informatique musicale JIM 2012*, Mons, Belgium, 2012, pages 147-155.

International Workshops

Musician and machine Workshop, Montreux Jazz festival, Switzerland, July 17, 2015.

Musical Rhythm Workshop, New York University Abu Dhabi (NYUAD), Abu Dhabi, October 12-15 2014. `[Travel grantee]`

Lisp for Music technology Workshop, ELS, 7th European Lisp Symposium, Ircam, Paris, May 5-6 2014.

2

Guiding Human-computer Music Improvisation

Our aim is to develop generation models and architecture models metaphorically inspired by the background provided in introduction (Chapter 1) in order to address idiomatic and pulsed improvisation with an interactive music system. This chapter gives an overview of the related work. Section 2.1 introduces some key notions relating to interactive music systems. Section 2.2 focuses on the different meanings that "guiding" can take in the field of computer music, and especially in human-computer improvisation. Section 2.3 gives some motivations regarding the design of an interactive music system dedicated to idiomatic or composed improvisation. Finally, Section 2.4 presents the research context of our work.

2.1 Using the Prior Knowledge of the Musical Context

2.1.1 Interactive Music Systems

Chadabe (1977) defines computer music as "music that is produced by a hardware and software system that importantly includes a computer, but may also include other analog or digital synthesis equipment". He formalized the general scheme of a system dedicated to *interactive composing*, depicted in Figure 2.1:

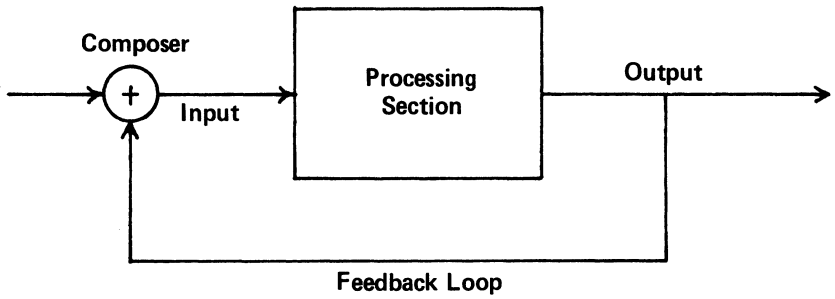


Figure 2.1: General scheme of a system dedicated to *interactive composing* (from Chadabe, 1977).

1. An input, by means of which a composer controls a system dedicated to interactive composition;
2. A processing section, which uses the input to produce the output;
3. An output, which is the music as sound;
4. A feedback loop, by means of which the actual output can be compared by the composer with what was expected.

As Chadabe underlines, in the non-realtime case, feedback has a cognitive value: the composer learns from the feedback, but to correct the output, the input must be changed and processed again from the beginning. We could add that it is also the advantage of non-real time where the composer has the convenience of time to correct her/his first assumptions. In the real-time case, feedback has a regulatory as well as a cognitive function, and depending upon the nature of the system - whether it is a memory or process automation system - either the performance or the composition is regulated.

In his attempt to classify interactive music systems as well as the different definitions proposed in the literature, Drummond (2009) distinguishes the approach chosen by Chadabe - where interactive composing is described as a performance process wherein a performer shares control of the music by interacting with a musical *instrument* - and the definition proposed by Rowe (1992). In this latter view, an interactive music system behaves just as a *trained human tmusician* would, listening to musical input and responding musically: "interactive music systems are those whose behaviour changes in response to musical input. Such responsiveness allows these systems to participate in live performances, of both notated and improvised music".

Our approach is closer to that of Jordà (2005), who adds that the previous definition implicitly restrains interactive music systems to systems which posses the ability to "listen", and that the input could be extended to *musical input* which is not only *music input*. Indeed, in the scope of idiomatic and pulsed improvisation, the musical context provides musical input taking the form of specification or prior knowledge, as it is developed in the following subsection.

2.1.2 Listening, Music generation, and Prior knowledge

In his classification system for interactive systems, Rowe (1992) proposes a combination of three dimensions: *score-driven* vs. *performance-driven* systems; *transformative*, *generative*, or *sequenced* response methods; and *instrument* vs. *player* paradigms. In this subsection, we introduce some (non-orthogonal) key notions that will be used in this chapter and that are particularly relevant in our scope

of guided human-computer music improvisation. They are illustrated by some related projects that will be described later on.

LISTENING When a system listens to the musical environment, it can be in order to *react* and / or to *learn*. "Listening" means here listening to a human musician co-improvising with the system, and does not concern controls that can be given to an operator-musician controlling the system.

Reaction: In this case, the playing of the musician is analyzed in real time and the result of this analysis can for example trigger some generative processes (*e.g.* Lewis, 2000), or be mapped to a corresponding event which is retrieved in a corpus (*e.g.* Moreira et al., 2013; Pachet et al., 2013; Bonnasse-Gahot, 2014).

Learning: In the *corpus-based* approach (see below), a system listens and learns by making its memory grow when the human co-improviser plays. Musical inputs can also be learnt to feed predefined generative models. For example, Band Out of a Box (Thom, 2001) is a computer accompanist with a fixed tempo in a "trading four" interaction scheme where a human improviser and a virtual partner repeatedly call and respond in four-bar chunks. Each bar of the human improvisation is analyzed and assigned to a cluster ("playing mode") and feeds the associated generation model. Then, the computer response is constituted by four bars belonging to the same sequence of modes using the generative models.

GENERATION The *corpus-based* interactive music systems create music from a musical memory constituted by offline corpora and / or live material. Sequences in this memory are searched, retrieved, transformed, and concatenated to generate the machine improvisation (*e.g.* Assayag et al., 2006b; Surges and Dubnov, 2013; François et al., 2013; Ghedini et al., 2016). With this approach, the resulting musical aesthetics strongly depends on the chosen musical memory. The system ImproteK implementing the models and architectures presented in this thesis belongs to this first category.

In the *rule-based* case, the musical structures are synthesized by rules or autonomous processes that can follow their internal logic (*e.g.* Blackwell, 2007) or interact with the environment. The pioneer system Voyager (Lewis, 2000), conceived and programmed by George Lewis since 1986, is a "player" program (using the classification proposed by Rowe (1992)) which is provided with "its own sound". It is designed as a virtual improvising orchestra of 64 asynchronous voices generating music with different sonic behaviors in real time. For each voice, the response to input goes from "complete communion" to "utter indifference". Voyager is defined by its author as a "kind of computer music-making embodying African-American cultural practice". Its design is indeed motivated by ethnographic and

cultural considerations, such as the concept of "multidominance" inspired by Douglas (1991) who formalized the notion of "multidominant elements" in musical and visual works of Africa and its diaspora.

PRIOR KNOWLEDGE OR SPECIFICATION When an interactive music system is not purely autonomous and takes the musical environment (in every sense) into account, it is not always only through listening. It can also be by using some upstream specifications or prior knowledge provided by a given idiom or the musical context. This criterion is the most relevant in our scope of "guided" human-computer music improvisation. Next subsection lists some cases when the musical context or the musical idiom provides temporal or logical prior knowledge or specification to the improviser, and how it can be used by a computer.

2.1.3 Prior knowledge of the musical context

In free jazz for example, historical, cultural, and social backgrounds play an important role in the way improvisation is approached and played (Lewis, 2008). In collective free improvisation, even in the absence of a shared referent (Pressing, 1984), musicians who have experience playing together come to share high-level knowledge which is not piece-specific but rather task-specific, *i.e.* an implicit mental model of what it is to improvise freely (Canonne and Aucouturier, 2015). Here, these non-formalized aspects are not addressed. In this subsection, we only focus on *explicit* and *formalized* prior knowledge or specification provided by the context.

2.1.3.1 *Precision of the knowledge*

Explicit and formalized prior knowledge or specification given by the musical context (when they exist) can be more or less specified. We focus here on the two ends of the spectrum before focusing on formal temporal specifications in 2.1.3.2. Planned inputs, as in performances based on a traditional score explicitly defining pitches, durations and dynamics, find computer music applications, for example, in the field of score following. On the other hand, planning can just describe a set of mechanisms, a temporal logic, or a group of events. In these latter cases, computer music applications can implement reactions to unordered events.

PLANNED INPUT A music performance may refer to predefined melodies, scores, audio materials or more broadly sequences of actions with their own temporality. The synchronization with a musician's performance of heterogeneous electronic actions (playing an audio file, triggering of a synthesis sound, or the execution of some analysis processes, *etc*) is a common problem of interactive music

systems. Many solutions have emerged to deal with this issue depending on musical purpose or available technologies, leading to the score following approach.The most elementary solution is to launch a predefined electronic sequence recorded on a fixed support (magnetic band, classical sequencer). In this case, the musician's performance is totally constrained by the time of the record. Score following is defined as the real-time alignment of an audio stream played by one or more musicians into a symbolic musical score (Schwarz et al., 2004; Cont, 2006). It offers the possibility to automatically synchronize an accompaniment (Dannenberg and Raphael, 2006), and thus can be used for the association of an electronic part to a predefined instrumental or in different creative ways in mixed music (Cont, 2011b), included improvised music contexts, for example when the theme of a jazz standard appears.

LOGICAL PLANNING, SET OF LOGICAL MECHANISMS On the other hand, the prior knowledge provided by the context can take the form of an agreement on a set of logical mechanisms. It is for example the case of *soundpainting*, the method of "live composition" using physical gestures for the spontaneous creation of music invented by composer and saxophonist Thompson (2006). It is defined as a "universal live composing sign language for the performing and visual arts". To cope with this category of prior knowledge, the solution in the field of interactive music systems is a purely reactive approach, the "agreement" being a set of logical mechanisms associated to a reactive listening module. The online analysis of the playing of a musician can for example trigger predefined generative processes with complex behaviors (*e.g.* Lewis, 2000), or focus on a particular musical dimension. Among them, Sioros and Guedes (2011a,b) use a rhythmic analysis of the live inputs to steer generative models with a focus on syncopation. In the case of corpus-based systems, reactive listening triggers an instant response retrieving a matching element in a corpus according to predefined mappings (this category will be discussed in Section 2.2). With a more general approach, a dedicated programming language can be used to compose reactivity in the scope of a particular musical project by defining responses to complex events implying both musical events and logical conditions (*e.g.* Echeveste et al., 2013c).

2.1.3.2 *Formal temporal specification*

FORMAL TEMPORAL SPECIFICATION An intermediary between the most and the least temporally specified context is the formal temporal specification. When the prior knowledge on the structure of the improvisation is not as explicit as a classical score, a melody, or a theme, it may consist in a sequence of

formalized constraints or equivalence classes to satisfy. This is for example the case of a solo improvisation on a given chord progression or on a temporal structure as introduced in Section 1.2.1. This category will be discussed in Section 2.2: when the improvisation relies on a known temporal structure, a computer music system should take advantage of this knowledge to introduce anticipatory behavior (see 1.2.2) in the generation process and not follow a purely step by step process. To address this level of prior knowledge, we propose a "scenario / memory" generation model in (Part I).

FORMAL TEMPORAL SPECIFICATION AND REACTIVITY In the cases mixing long-term planning and reactivity, like the case of solo or collective improvisation with a machine on a given chord progression, we advocate for the computer a "scenario / memory" generation model embedded in a reactive architecture (Part II). The thesis focuses on idiomatic music relying on a formal temporal structure in a context of collective performance, for example a collective improvisation on a given chord progression. Since the improvisation is collective, the musical context is not only determined by the formal structure, but also by the other musicians. An interactive music system has therefore to cope with situations where it is not the master of the tempo, and with reactivity to external inputs. Furthermore, a *reaction* cannot only be seen as an instant response but has to take advantage of the prior knowledge of the structure to generate mid-term anticipations ahead of the performance time, refining or rewriting these anticipations over time in the light of new events.

2.1.4 "Self-Organization" and "Style Modeling" Paradigms

The last three paragraphs emphasized different approaches where the machine improvisation is *guided* by the environment or by a specification provided by the musical context. Before focusing on this notion of declarative *guidance* in Section 2.2, we present here some paradigms of machine improvisation that are not guided (in the sense that we give to this word in this dissertation) but steered by internal mechanisms of self-organization, or by the internal sequential logic of a corpus.

SELF-ORGANIZING SOUND A branch of generative audio systems (see 2.3) focus on *self-organization* (Blackwell and Bentley, 2002; Blackwell, 2007; Miranda, 2004). They are based on emergence of coherent patterns at a global level out of local interactions between the elements of a system. Self-organizing behaviors lead to a decrease

in entropy, while self-disorganizing behaviors lead to an increase of entropy. We invite the reader to see (Bown and Martin, 2012) for a discussion of the notion of entropy in this context, and the idea of autonomy in interactive music systems.

STYLE MODELING The interactive music systems focusing on style modeling are steered by the internal sequential logic of the musical material they learn. They aim at generating musical improvisations that reuse existing external material. The general idea is to build a model of the playing of a musician as it records it (or of an offline corpus) and to use this analysis to find new routes across this musical corpus. The machine improvisation consists in a navigation within this model that both follows the original paths (*i.e.* replays the original sequence) and, at times, ventures into those new passages, thus jumping to new location, and thereby providing a new version of the captured material. Concatenation is often based on the Markovian properties of the sequence itself: improvising thus amounts to recombine existing material in a way that is both coherent with the sequential logic of this material and so that it actually provides something different than a mere repetition of the original material while keeping with its statistical property.

The Continuator (Pachet, 2003) introduces a paradigm of *reflective interaction*. It uses variable-length Markov chains following on the work on statistical style modeling initiated by Dubnov et al. (1998) and Assayag et al. (1999) to generate new continuations from an input stream. The stream is parsed to build a tree structure, and as new inputs arrive, the tree is traversed to find continuations of the input. The system is therefore able to learn and generate music in any style without prior knowledge, either in standalone mode, as continuations of live inputs, or as interactive improvisation back up.

The real-time improvisation system Omax (Assayag et al., 2006b,a; Lévy et al., 2012) uses the Factor Oracle (Allauzen et al., 1999; Lefebvre et al., 2002), a deterministic finite automaton, to achieve style modeling (Assayag and Dubnov, 2004). The musical stream is first segmented into discrete units (a new "slice" for each new onset). Then, each slice is labeled using a chosen audio feature. Finally the resulting string of symbols is analyzed to find regularities in the musical material using the Factor Oracle. As we will see later on, this automaton is used in different ways within numerous research project addressing music generation and improvisation, and is also involved in the guided generation model that we propose in this thesis.

Figure 2.2 shows the resulting representation of the musical inputs with two different audio features: pitch and Mel-frequency cepstral coefficients (MFCCs). The analysis it provides serves as the basis of the generative process (see Section 5.3): by navigating this structure thanks to the Suffix Link Tree (Assayag and Bloch, 2007), one is able

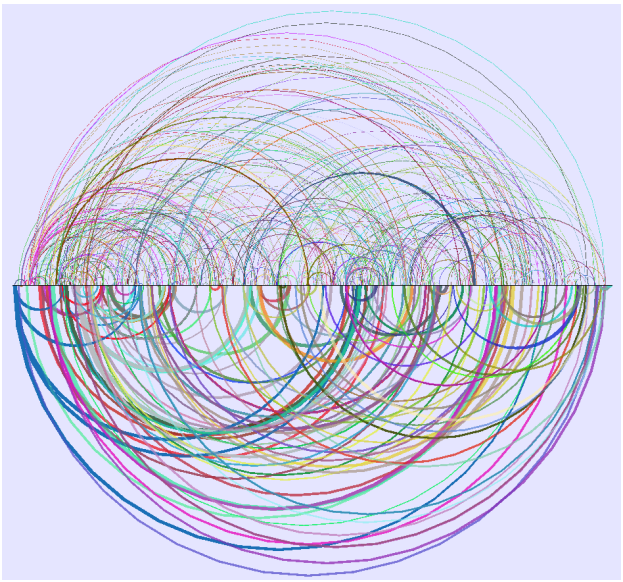


Figure 2.2: Omax: dual cartography (pitch and MFCCs) of a musical corpus (from Lévy, 2013).

to connect any location within the musical material of interest to any other location that has a common suffix, *i.e.* a common musical past (arches in Figure 2.2, corresponding to the suffix links provided by the Factor Oracle automaton). Reading this structure following non-linear paths generates a musical sequence that is both different from the original one, and coherent with its internal logic.

2.2 Guiding: "Follow my steps" / "Follow that way"

A number of existing improvisation systems drive music generation processes by involving a user steering their parameters. First, this user control can concern (low-level) system-specific parameters. This is for example the case of Omax (Assayag et al., 2006b; Lévy et al., 2012), which is controlled by an operator-musician steering the navigation through a model built in real time from the playing of a live musician, or Mimi4x (François et al., 2013) which involves a user in the construction of the performance by choosing the musical corpus and modifying the generation parameters, and displays the memory as wall as the machine improvisation as a piano roll.

> In this section, we refer to *guided improvisation* when the control on music generation (whether it is given to an operator or to automated processes) follows a more *declarative* approach, *i.e.* specifying targeted outputs or behaviors using an aesthetic, musical, or audio vocabulary independent of the system implemen-

tation, whether this control is short-term (2.2.1) or long-term (2.2.2).

2.2.1 "Guiding" Step by Step

On the one hand, *guiding* is seen as a purely reactive and step by step process. We focus here on the corpus-based approach.

Somax (Bonnasse-Gahot, 2014), for instance, extends the paradigm of automatic accompaniment using purely reactive mechanisms without prior knowledge. The system achieves a floating musical coordination with a human improviser driven by reactive listening. It uses a previously annotated corpus learnt in a simplified n-gram, and translates the musical stream coming from an improviser into activations of specific zones of this memory regarding different musical dimensions called "streamviews".

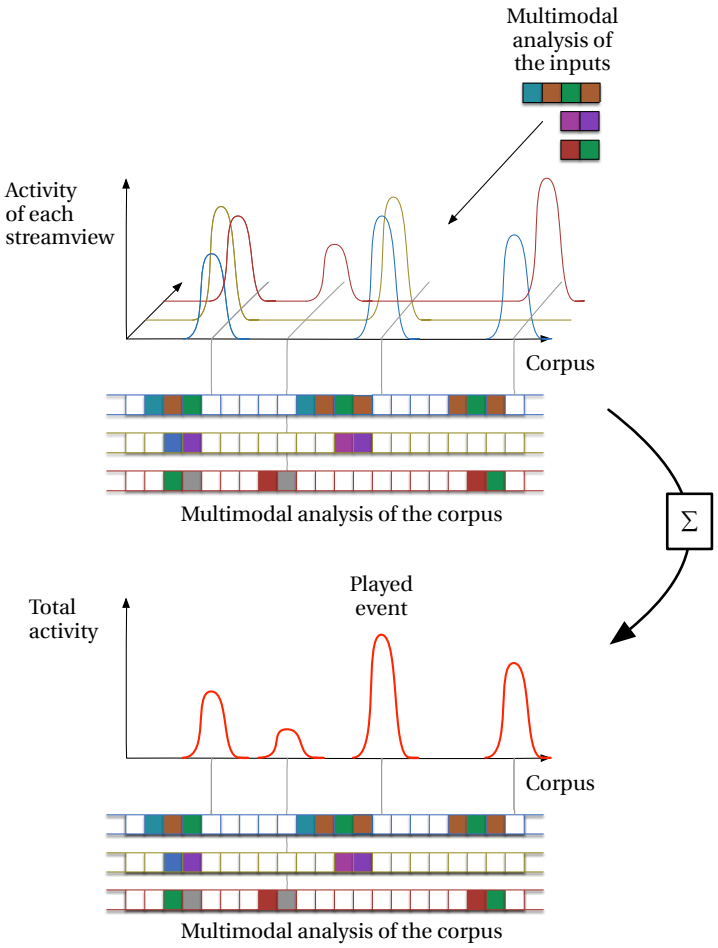


Figure 2.3: Somax: reactive listening activating different regions of a corpus. Adapted from Chemla-Romeu-Santos (2015) (see 19.2.1, B.1.3).

As illustrated in Figure 2.3, the multimodal analysis of the music played by the human co-improviser is compared to the annotations

of the corpus, and modifies the *activity* of each streamview (for example pitch, harmonic background, and self-listening). Finally, the system retrieves and plays an event in the memory presenting a maximal total activity. After being updated by the live inputs, the activity follows a continuous evolution and is propagated with a temporal decrease to introduce a *cognitive remanence*. This way, the system reacts to the present of the improvisation taking its recent past into consideration.

Several systems dedicated to pulsed music emphasize interaction and reactivity and extract multimodal observations from a musician's playing to retrieve musical segments in a memory in accordance to previously learnt associations. VirtualBand (Moreira et al., 2013), for example, is constituted by virtual agents representing the style of different musicians, and relies on *feature-based interaction*: interactions are modeled by connections between a master (human or virtual) agent and a slave virtual agent. In reaction to a feature value provided by the master agent (*e.g.* RMS, number of onsets, spectral centroid, or chroma), the connection specifies to the slave agent which audio chunk to play from its database. This mapping models the *intention* underlying a musical interaction between musicians, nevertheless, this view of intentionality is step by step and not oriented toward future. Furthermore, a chord sequence can be imposed as any other feature, and thus it is not used to introduce anticipatory behavior.

In the same line, Reflexive Looper (Pachet et al., 2013), explicitly dedicated to jazz improvisation, aims at enriching the experience that one can have when playing with a loop pedal, a digital sampler that playback audio played previously by a musician. Instead of simply playing back, the accompaniment played by the system reacts to what is currently being played by the musician by combining different playing modes: *bass line*, *chords*, and *solo*. Reflexive Looper is based on supervised classification and concatenative synthesis. A Support Vector Machine classifier is trained on a database using different audio features to be able to recognize the three different playing modes. Then, during the performance, if the musician plays solo for example, the looper plays bass and chords. In addition, the part played by the system uses the *feature-based interaction* paradigm introduced in VirtualBand to adapt its playing to the human player (the two listened dimensions are RMS and spectral centroid). Finally, a hard constraint imposed to the system is that each played-back audio segment should correspond to a correct chord in a chord progression provided a priori. Like VirtualBand, this chord progression is only used to take local decisions in a step by step process.

Closer to the algorithms we use, recent researches guide the generation process step by step: PyOracle (Surges and Dubnov, 2013) is a machine improvisation system using the Audio Oracle (Dubnov et al., 2007): a Factor Oracle automaton (introduced in 2.1.4) whose

construction is enriched by mechanisms suitable for the creation of an alphabet of audio features. It uses a measure called *Music Information Rate* which is defined and described in (Dubnov et al., 2011). PyOracle enables to learn musical structures from arbitrary features extracted from audio signals. These structures can then be used to generate new variations on the original input signal, and is guided by "hot spots" (single event targets). Following his previous works on the use of corpus-based concatenative synthesis for improvisation (Einbond et al., 2012, 2014), composer Aaron Einbond associates PyOracle and MuBu (Schnell et al., 2009), a generic container designed to store and process multimodal data (audio, motion tracking data, sound descriptors, markers, etc.) in CatOracle (Einbond, 2015).

2.2.2 "Guiding" with a Formal Temporal Structure or Description

The previous subsection gave an overview of works where guiding is considered as a step by step process. On the other hand, in the projects that we summarize here, *guiding* means defining upstream temporal structures or descriptions driving the generation process of a whole music sequence.

PRESERVING CHARACTERISTIC LOCAL STRUCTURES Some researches aim at generating new sequences favoring transitions or subsequences that are characteristic of a chosen corpus. In this view, Pachet and Roy (2011) propose to add constraints to Markovian processes to steer the generation of complete musical sequences. This *Markov Constraint Problem* can be guided by adding different types of constraints: for example constraints on the metric structure (Roy and Pachet, 2013), or constraints to generate Markov sequences with a maximum order to preserve characteristic sequences of the corpus while avoiding long replications (Papadopoulos et al., 2014).

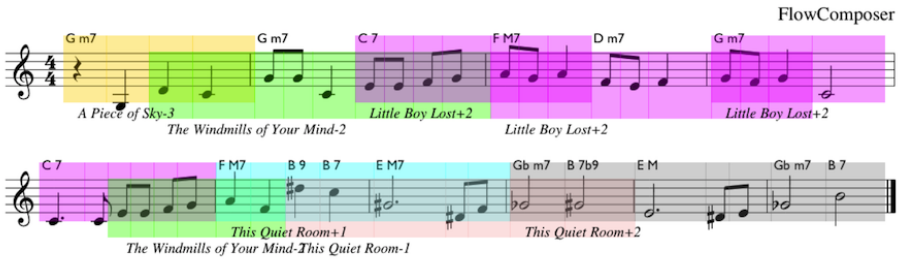


Figure 2.4: A leadsheet generated in the style of French composer Michel Legrand (from Papadopoulos et al., 2014).

Figure 2.4 shows a leadsheet generated using this last technique. The generation process takes as input a maximum order (6 in this ex-

ample) and a corpus of leadsheets by French composer Michel Legrand. A Maximum Order Automaton is built from this corpus by removing sequences of forbidden lengths from a previously built Markov automaton using Aho and Corasick (1975) string matching algorithm. Finally, a navigation through this automaton produces a new leadsheet constituted by tiled subsequences of the corpus whose lengths are smaller than the maximum order (colored rectangles).

With comparable motivations, Herremans et al. (2015) built a system to generate bagana music, a traditional lyre from Ethiopia, based on a first order Markov model. The authors propose a method that allows the conservation of structural patterns and repetitions such as cyclic patterns within the generated music. A long-term coherence is handled using first order Markov models within evaluation metrics. We can also mention the model for corpus-based generative electronic dance music proposed by Eigenfeldt and Pasquier (2013). It analyses a corpus to get the probabilities of different local structures considered as characteristic of this style (drum patterns, percussions, bass lines), and new pieces are generated from this Markovian model.

> To summarize, these techniques preserve some structural patterns of the musical memory, which cannot be achieved with a simple random walk through a Markovian model. Yet, they do not allow to provide a formal specification sequence defining the temporal structure that the generated sequence has to match.

GUIDING CORPUS-BASED AUDIO CONCATENATIVE SYNTHESIS

Corpus-Based Concatenative Synthesis makes it possible to explore and render high-quality synthesis of sound textures (see Schwarz, 2011). It is a data-driven approach to sound synthesis that relies on a database of annotated sounds (a corpus) containing a large number of audio segmented and their associated description. This technique can synthesize sound through exploration of a descriptor space (*e.g.* CataRT, Schwarz et al., 2006) or according to a target sequence of such descriptors, which is the approach we will focus on.

The domain of corpus-based audio concatenative synthesis generally involves descriptions on low-level audio features alphabets, which is musically different than using descriptions on idiomatic alphabets (see 2.3). Furthermore, when generating sound textures, it generally only addresses the issue of the mapping between chunks in a query and chunks in a corpus, without addressing a secondary elastic time mapping between the output and a third party fluctuating time reference (the “beat” in our case).

Nevertheless, it is a domain of interest in our study since guiding sound generation with a sequence provided as an audio input shares common issues with music generation guided by a temporal

idiomatic sequence. In this domain, generation has often to choose between causality and anticipatory behavior. The first category is closely related to the last examples given in 2.2.1. On the other hand, the approaches using Viterbi algorithm and dynamic programming (*e.g.* Schwarz, 2004), or constraints (such as Zils and Pachet (2001) and Aucouturier and Pachet (2006) using local search method for constraint solving of Adaptive Search (Codognet and Diaz, 2001)) often process the whole target input in one execution run.

Some related researches use the Factor Oracle automaton introduced previously. Among them, the Variable Markov Oracle Wang and Dubnov (2014a) extends the approach of PyOracle in an offline architecture using sequences instead of single events as query targets, and also finds application in 3D-gesture query matching (Wang and Dubnov, 2014b). This idea of local temporal queries shares common issues with the playing mode involving a dynamic scenario presented in this thesis. Nevertheless, the associated navigation algorithm makes step-by-step local decisions and does not implement the anticipatory behavior that we need in the case of idiomatic improvisation (to prepare the resolution of a cadence, for example).

Finally, we focus on Guidage (Cont et al., 2007), that shares some common motivations and tools with the generation model presented in this thesis. Indeed, it uses a predefined temporal query to guide the navigation through an audio memory learnt in a Factor Oracle automaton (more precisely an Audio Oracle (Dubnov et al., 2007)). The temporal query is not a dynamic idiomatic structure as in our case, but a fixed audio query. The search algorithm guides the resynthesis engine to parts of an audio memory pertaining to the given audio query, and allows reassembly of these factors to replicate the given query. For an audio query of length N, a forward pass returns N sets of tree structures determining variable lengths paths in the Audio Oracle starting at each index of the query. Then, a backward pass realizes a branching procedure using these trees and the Audio Oracle structure to find the best path.

Despite some similarities between this work and ours, both follow different objectives. Indeed, Guidage does not favor sequentiality in the retrieved sequences, and implements a matching threshold. Furthermore, it achieves partial reconstruction and matches variable length clips of the query input to new audio material without explicit temporal segmentation. Finally, the outline of the generation process (forward pass then backward pass) makes it inappropriate for our application case.

A major drawback of this first approach is that it does not take advantage of the prior knowledge provided by the query and follows a purely step by step approach. A revised version of Guidage (Cont, 2008b, Chapter 6) optimizes the temporal context of the generated sequence using Anticipatory learning. The idea behind this is to fur-

ther enhance the knowledge of the system by blending the ongoing immediate guides with their values in an infinite future horizon. It updates the selected states by favoring recurrent patterns but also appropriate contexts that lead to these states in the memory. This approach therefore introduces anticipatory behavior, but in a local way. Furthermore, it is offline and thus does not handle dynamic queries in a real-time reactive context. In this work, the integration of knowledge, interaction and composition was emphasized as promising perspectives.

ENFORCING A SEQUENTIAL STRUCTURE Finally, some projects aim at enforcing a formalized temporal structure in a music generation process. The general approach of the Flow Machines project (see Ghedini et al., 2016) is to apply a *style* (corpus) to a well-chosen *structure* (sequential content such as text or music) to generate creative objects. Within this project, for example, ReChord (Ramona et al., 2015) is an offline engine based on chord progressions generating new accompaniment tracks from a single recording of accompaniment, using concatenative synthesis at the chord scale.

The project of *Machine Improvisation with Formal Specifications* of Donzé et al. (2014) applies the concept of "control improvisation" (Fremont et al., 2014) to music in order to generate a monophonic solo similar to a given training melody over a given chord progression. This aim is close to that of the offline music generation we propose in Part I, but it does not use concatenative synthesis: following Conklin and Cleary (1988), it implements a flexible multiple viewpoint approach to music generation in which note aspects are predicted separately and then aggregated. A formal specification is enforced using three automata: a *plant* encoding the chord progression; a *generalization of the training melody* using the previously introduced Factor Oracle automaton; and a *specification* encoding additional specifications regarding pitch, rhythm, and the occurrence of particular licks on specific beats. The synchronous product of these three automata provides a general *improviser* structure. Finally, the generation process amounts to finding an accepting trace of this improviser satisfying some requirements of randomness and bounded divergence. A real-time prototype has been implemented in Ptolemy (Buck et al., 1994). It offers learning of the automata from live midi inputs, and takes a tune specification and a "creativity level" as interactive inputs (Donze et al., 2013).

2.2.3 Conclusion

Two conceptions of time and interactions are actually emphasized in the different approaches of "guidance" we presented. The purely step-by-step and reactive one offers rich interaction possibilities but cannot take advantage of the prior knowledge of a temporal structure, which is essential when addressing idiomatic music. On the other hand, the notion of structure can be local and limited to structural patterns which are characteristic of a given corpus. Besides, for different reasons including the wish to preserve causality, numerous projects using long-term structures to steer music generation do not use this prior knowledge to introduce anticipatory behavior (see 1.2.2). Finally, when it is achieved, the major drawback is that they lack responsiveness to changes in musical situations occurring during performance, such as a modification of the "scenario" itself or changes in interaction between players during improvisation.

This bi-partition in improvisation systems reflects the offline / online paradigmatic approaches in computer music systems regarding time management and planning/scheduling strategies. We aim at designing an architecture at an intermediate level between the reactive and offline approaches to combine anticipations relative to a predefined plan and dynamic controls. This frontier is studied in current works in computer music such as (Agostini and Ghisi, 2013; Echeveste et al., 2013a; Bresson and Giavitto, 2014; Bouche and Bresson, 2015a). On the one hand, "offline" corresponds to computer-aided composition systems (Assayag, 1998) where musical structures are computed following best effort strategies and where rendering involves static timed plans (comparable to timed priority queues (Kahrs, 1993)). In this case, scheduling only consists in traversing a precomputed plan and triggering function calls on time. On the other hand, "online" corresponds to performance-oriented systems (Dannenberg, 1989) where the computation time is part of the rendering, that is, computations are triggered by clocks and callbacks and produce rendered data in real-time (Maigret, 1992). In this case, only scheduling strategies matter and no future plan is computed.

2.3 Some Considerations about Software Architecture

Surges et al. (2015) define a *generative audio system* - distinguished from the more general *generative music system* - as a new kind of generative music system that generates both formal structure and

synthesized audio content from the same generative process. In this case, the synthesis and organizational processes are inseparable and operate at the sample level. This architecture is particularly relevant for various musical purposes, for example when working on *feedback* (Sanfilippo and Valle, 2013): transforming an input, the result of which becoming the output and appearing again at the input after a delay. The feedback systems include chains of complex and time-varying signal-processing blocks (*e.g.* filters, phase-shifters) and can be characterized by the polarity of their feedback, non-linearity, self-organization, and complexity (an exhaustive history and study of generative audio systems can be found in (Surges, 2015)).

> Our objective is to introduce temporal structures in order to address idiomatic and pulsed music. As we introduced in 2.2.2, contrary to the generative audio systems mentioned above, it requires to distinguish the generation of the musical phrases in the symbolic domain (using the alphabet chosen do write the scenario) and rendering (audio or not) in the time domain. In this section, we chose this aspect as a starting point to present our motivations regarding software architecture.

2.3.1 Behind the Symbol

Rowe (2009) studied the role of *symbolic* and *subsymbolic* levels in interactive music systems. We underline here the different musical meanings involved when working using a symbolic description defined on a audio alphabet or a symbolic description defined on an idiomatic alphabet.

To give a basic example: when a musical slice is labeled by a low-level symbolic label (*e.g.* when the slice is assigned to a given cluster regarding spectral centroid), it provides information on the *nature* of the *content* of the musical slice. On the contrary, in our scope, a chord label such as *G7* provides information on the *context* in which the musical slice is played. It does not have a causal and deterministic relation with the actual content of the signal. For example, when playing on a beat labeled by *G7* during a solo, a guitarist may play a chromatic sequence ending with a *B*, realize a chord substitution, hit the soundboard to get a percussive sound...

> When such a label is observed in the temporal context of a whole sequence, our aim is to capture implicitly the *functional* role of this idiomatic label without formalizing it. For example, this *G7* may have a particular role if it is preceded by a *Dm7* and followed by a *Cmaj7*. In this case, to a certain extent, this contextual information may be more important than the *G7* nature.

2.3.2 Meta-Composition

Rowe (1999) considers that interactive music systems:

> "become a ligature connecting improvisation to notated composition, as the same processes used to govern the notated music can be employed to generate new improvisations in real time. This possibility is an expansion of the domain of composition [...]. By delegating some of the creative responsibility to the performers and a computer program, the composer pushes composition up (to a meta-level captured in the processes executed by the computer) and out (to the human performers improvising within the logic of the work). An interesting effect of this delegation is that it requires a very detailed specification of the musical decisions needed to produce a computer program at the same time that the composer cedes a large measure of control over musical decision-making to the human improviser."

The introduction of a symbolic layer enables to develop generic formal mechanisms that can be used to explore different musical direction. Thanks to this genericity, we can address the “meta-level” of authoring/composition mentioned in the previous quotation by involving the musicians in the upstream process consisting in designing the musical language of the machine for each project (Chapter 6).

2.3.3 Beat and Synchronisation

Somax (Bonnasse-Gahot, 2014) segments its musical corpus into events delimited by onsets. Its pulsed mode uses a phase descriptor among the navigation constraints and retrieves an event tagged by the beat phase which is the closest to the current beat phase of the performance time. This approach achieves a floating synchronization between the musician and the machine whose flexibility enhances the creative possibilities. Yet, it is not dedicated to heavily rhythmic music such as a funk accompaniment for example.

Conklin and Witten (1995) also suggest to use the location of a note in a bar as a “viewpoint” among others. Pachet (2003) notes that this scheme forces to use quantization, which raises many issues which are intractable in an interactive real-time context. Instead, Pachet proposes to segment the input sequences according to a fixed metrical structure given by an external sequencer together with a fixed tempo, through Midi synchronization.

Most systems dedicated to pulsed improvisation require a fixed tempo. Among them we can quote BoB, (Thom, 2001), VirtualBand (Pachet et al., 2013), and Reflexive Looper (Moreira et al., 2013), mentioned previously, or GenJam (Biles, 2002). This last software provides

an accompaniment to support a musician's improvisation. Then, after a listening phase, it repeats some sequences modified through a genetic algorithm with a given tempo.

> Our aim is to synchronize the machine improvisation with a non-metronomic beat source to be able to integrate the system in a collective improvisation where it can successively follow or master the tempo. The generic non-metronomic beat input can come from an online beat tracking system, a time track, a tempo curve, etc. This is achieved thanks to the distinction between generation in the symbolic domain and rendering in the time domain. The generation process amounts to creating a symbolic mapping segmented into symbolic time units that are unfolded through time according to the elastic realization of this temporal reference in the time domain (Chapter 13).

2.3.4 A Chain of Self-Consistent Architectures

> Finally, the division of the whole guided improvisation process into well-defined tasks enables to propose self-consistent architectures addressing different issues when they are considered independently: an offline music generation model guided by a scenario structure; a dynamic agent embedding an offline generation model to generate an evolving musical sequence, combining anticipatory behavior and dynamic controls; a high-level framework to conciliate various improvisation strategies and scheduling generation as well as rendering; an architecture for adaptive rendering of multimedia sequences generated from live inputs according to dynamic user-defined time mappings and synchronized with a non-metronomic pulse.

2.4 Research Context

2.4.1 History of the project

ImproteK, implementing the work presented in this thesis, Omax, Somax, PyOracle and the associated projects belong to a family of related researches and implementations on machine improvisation. They share a sequential model, that we call here "memory", learnt from live or offline music streams, that is explored interactively at performance time with various types and degrees of constraints.

The generation model we propose follows on the work on statistical style modeling initiated in (Assayag et al., 1999; Dubnov et al., 1998) and developed in (Assayag and Dubnov, 2004) and its implementation in the real-time improvisation system Omax (Assayag et al.,

2006b,a; Lévy et al., 2012), dedicated to non-idiomatic and non-pulsed improvisation. Then, starting from a previous version of Omax including a fixed "beat" mode, Chemillier (2010) initiated this project to address idiomatic improvisation - in particular jazz - in the line of his works on the automatic generation of chord sequences (Chemillier, 2001, 2004, 2009).

With the first (midi) prototype of ImproteK (Nika and Chemillier, 2012), we introduced therefore a"beat", long-term constraints and a priori knowledge in music generation processes by means of a formalism conveying different musical notions depending on the applications, like meter as regards rhythm or chord notation as regards harmony. This first generation model ("conformity") and the associated sequential architecture will be briefly described in this dissertation (Nika and Chemillier, 2012, for further information, see). We focus here on the generalization of these long-term constraints with the "scenario" structure, the introduction of anticipatory behavior, the dialectic between anticipation and reaction, and the synchronization of audio rendering with fluctuating time references.

2.4.2 Vocabulary: "Scenario"

The word "scenario" has different meanings in the scope of interactive music systems. Even though they all carry a notion of planning or are related to the idea of temporal constraints, they address different issues. In the field of interactive multimedia sequencers such as I-score (Allombert et al., 2008) and its recent evolutions (Arias et al., 2014; Arias, 2015), a "scenario" is the organization of multimedia contents and controls structured in a spatial and temporal order according to users' requirements. These multimedia contents and controls interact with external actions and those of a performer (e.g., multimedia live performance arts, interactive museum installations, and video games). For Echeveste (2014), "scenarizing" means defining a set of logical or temporal mechanisms that can be written thanks to a dedicated programming language. These mechanisms handle the reactions to unordered complex events in a context of mixed music, going beyond the sequential aspect of traditional score following. These high-level *organizations* are related to the *improvisation plan* discussed in Chapter 10. In the scope of this thesis, the *scenario* is involved in *generation* and is a sequence of equivalence classes. It is formally defined as a symbolic sequence guiding the machine improvisation: a word defined on a relevant alphabet depending on the musical context, *e.g.* a chord progression, a metric structure, or a form defined on a abstract alphabet.

Part I

"INTENTIONS": COMPOSING MUSIC GENERATION PROCESSES AT THE SCENARIO LEVEL

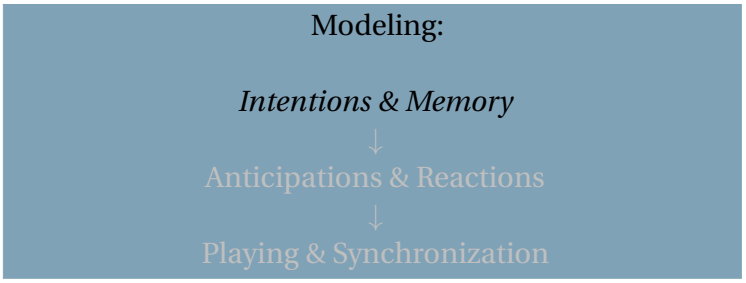
Part I proposes a music generation model relying on a formal temporal structure.

Chapter 3 summarizes the contributions of Part I.

Chapter 4 introduces the principle of the scenario / memory generation model: ensuring the conformity to a predefined temporal structure and taking advantage of this prior knowledge to introduce anticipatory behaviors.

Chapter 5 details the algorithms involved in the scenario / memory generation model, combining continuity with the future of the scenario and continuity with the past of the memory.

Chapter 6 focuses on the genericity of the model and introduces a protocol to define an alphabet, its properties, and associated transformations to go from conformity to an idiomatic structure to composition of improvisation sessions at the scenario level.



3

Summary and Contributions

3.1 Paradigm

Scenario and Memory: On the basis of the musical motivation introduced in Chapter 1, we model music generation relying on a formalized temporal specification (*i.e.* a chord progression) as the articulation between a *scenario* and a *memory*. In first approach, it ensures the conformity of the improvisation generated by the machine regarding the stylistic norms and aesthetic values implicitly carried by the idiom of the musical context. In second approach, the scenario gives access to a prior knowledge of the temporal structure of the improvisation which is exploited to introduce *anticipatory behaviors* in the generation process. This way, the future of the scenario is taken into account when generating the current time of the improvisation.This point is particularly relevant in the case of *hybrid* improvisation: when the improvisation is created using musical material exogenous to the scenario.

3.2 Algorithms

Overall algorithm: We propose a generation process taking advantage of the prior knowledge of the scenario as mentioned above, and of an analysis the musical memory to maintain the coherence of its musical discourse when digressing from the original material. The scenario and the memory are formally represented by words defined on the same alphabet.
The overall algorithm therefore addresses the general issue of indexing and selecting paths in a text (memory) matching successive factors of a word (the scenario), favoring sequentiality using the regularities in the pattern while being able to retrieve non-contiguous sequences using the regularities in the text.

Segmentation into generation phases: The generation process is divided into successive generation *phases*. Each *phase* is constrained by the *current scenario*, *i.e.* a suffix of the whole scenario. A *prefix of this suffix of the scenario*, that is to say *a part of what remains to be played* is chosen in the memory to be copied, or to provide a starting point to follow an equivalent non-linear path in the memory until a new phase is launched. First, thanks to this design, the model can be queried using temporal queries (portions of scenario) that enable to implement anticipatory behavior and to generate anticipations ahead of performance time when the model is used in a real-time context (see Part II). Second, it enables to optimize a generation phase using the results of the previous ones.

Future and past: In a generation phase, each step ensures both continuity with the future of the scenario and continuity with the past of the memory. Each phase consists in two successive steps:
1 - Anticipation: find an event in the memory sharing a common future with the scenario while ensuring continuity with the past of the memory (when it is possible);
2 - Copy or digression: retrieve the whole sequence or use the regularities in the memory to follow an equivalent non-linear path (and possibly extend it), and thus digress from the original material.
Continuity with the future of the scenario is handled with a dedicated algorithm indexing the prefixes of a pattern (the current scenario) in a text (the memory) using the regularities in the pattern.
Continuity with the past of the memory is provided by the automaton structure chosen to learn the musical memory: the Factor Oracle automaton.

3.3 Application and implementation

A meta-level of composition: The scenario / memory approach is generic, that is to say formally independent of the chosen alphabet. Taking advantage of this genericity, we developed a protocol to compose improvisation sessions. In this framework, musicians for whom the composition of scenarios is part of the creative process can be involved in a meta-level of composition, *i.e.* involved upstream to design the part of creativity which is delegated to the machine by designing an alphabet and its properties, equivalence classes, associated transformations of the contents, etc. Collaborations with musicians using scenarios defined on different alphabets are detailed in Part IV.

Implementation: The scenario / memory generation model has been implemented as a CommonLisp modular library within the OpenMusic environment (Bresson et al., 2011). Appendix B.1 presents some additional work relating to the scenario / memory generation model and its implementation:

- some examples of visual patches (B.1.1),
- an option for prefix indexing with k mismatches (B.1.3),
- an early model for harmonization and arrangement using a first version of the scenario / memory generation model (B.1.2),
- and a signal processing module to build scenarios or annotated memories from offline audio files (B.1.4).

4

Conformity, Anticipation, and Hybridization

We present in this part a music generation model relying on a formal temporal structure called "scenario". This object does not carry the narrative dimension of the improvisation, that is its fundamentally aesthetic and non-explicit evolution, but is a sequence of required equivalence classes for the machine improvisation. This chapter introduces the musical motivations and the principle of the scenario / memory generation model which is then detailed in Chapter 5.

4.1 "Scenario" and "Memory"

The generation model we propose articulates a *scenario* guiding the generation and a structured and indexed *memory* in which musical sequences or events are searched and retrieved to be transformed, rearranged and reordered to create new improvisations:

- the *scenario* is a symbolic sequence guiding the improvisation and defined on an appropriate alphabet (depending on the musical context),
- the *memory* is a sequence of events where each element is labeled by a symbol belonging to this same alphabet.

The scenario can be any sequence defined on a chosen alphabet suitable for the musical context, for example a harmonic progression in the case of jazz improvisation or a discrete profile describing the evolution of audio descriptors to control sound synthesis. For generation purposes, a symbol in the sequence is considered as an equivalence class.

The musical memory used during an improvisation session is a store of temporal musical sequences (MIDI, audio, parameters for sound synthesis...) represented as sequences of events. An *event* (Figure 4.1) has a duration and is indexed by its *position* (index $\in \mathbb{N}$) which will also be called *date* (both are equivalent since the time from the beginning of the sequence is stored within each event).

The memory is constituted by a *long-term memory*, using a stored database as training data, and a *short-term memory* using the same memory model (see 5.3) but only learns from the current piece as it

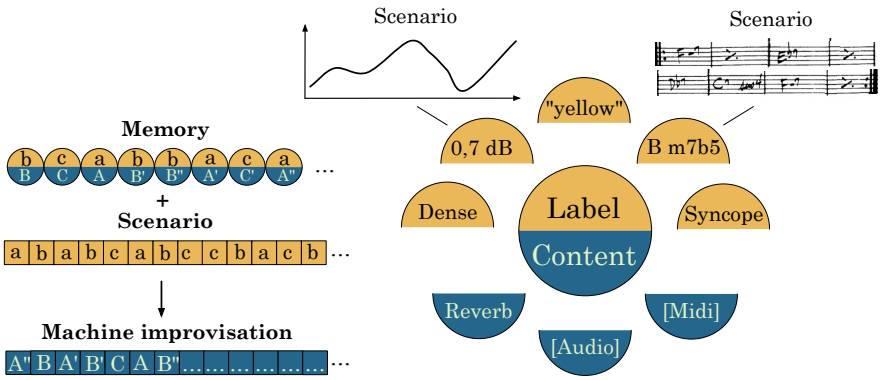


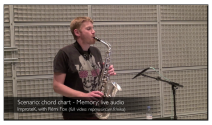
Figure 4.1: An *event*: elementary unit of the musical memory. It is constituted by a musical *content* annotated by a *label*. The scenario guides the concatenation of these contents to generate the machine improvisation.

unfolds. The memory can therefore be constituted online by recording the music played by the human co-improvisers during a live performance (the way musical inputs from the musicians are segmented into events, annotated, and learned in real time is described in chapter 8) and/or offline (from annotated material). All the sequences in the memory have to be segmented into events annotated using the alphabet chosen for the scenario (*e.g.* harmonic labels) but do not have to be created within the exact same source scenario (*e.g.* a set of recordings of solos on different jazz standards).

In this context, "improvising" means navigating through the memory in a creative way to collect some contiguous or disconnected blocks matching the successive parts of the scenario and concatenating them to create a musical phrase. We will first consider that an event in the memory *matches* a label of the scenario when the labels are equals. Yet, the events in the memory can also be transformed to virtually increase the size of the memory. The generic approach (equivalence classes on the labels associated with transformations of the contents) will be developed in Chapter 6, and a first example will be given in Section 4.3 with the case of transposition when the scenario is defined as a harmonic progression.

4.2 Conformity and Anticipation Regarding the Scenario, Coherence with the Memory

Video A.1.2



Hyperlink video *(or vimeo.com/jeromenika/improtek-fox-rentparty)*
Description: Appendix A.1.2.
Musical part IV: Section 18.1.

CONFORMITY In first approach, the scenario ensures the conformity of the machine improvisation regarding the stylistic norms and aesthetic values implicitly carried by the musical idioms. **Video A.1.2** gives a first example of how the scenario/memory model recontextualizes some subsequences of the memory to generate new musical sequences matching the scenario. In this example of jazz improvisation, the solo played by the musician is segmented in real time using beat markers and annotated with harmonic labels so that it can be immediately re-injected by the model to produce an improvisation matching the scenario which is here a simple chord progression.

ANTICIPATION Then, the scenario gives access to a prior knowledge of the temporal structure of the improvisation which is exploited to introduce *anticipation* in the generation process, that is to say to take into account the future of the scenario to generate the current time of the improvisation. This anticipation was first introduced to deal with issues of musical phrasing (see Part IV). Furthermore, this point is particularly relevant in the case of "hybrid" improvisation: when the scenario and the memory are different sequences defined on the same alphabet (see an illustration with the example of jazz improvisation in Section 4.3).

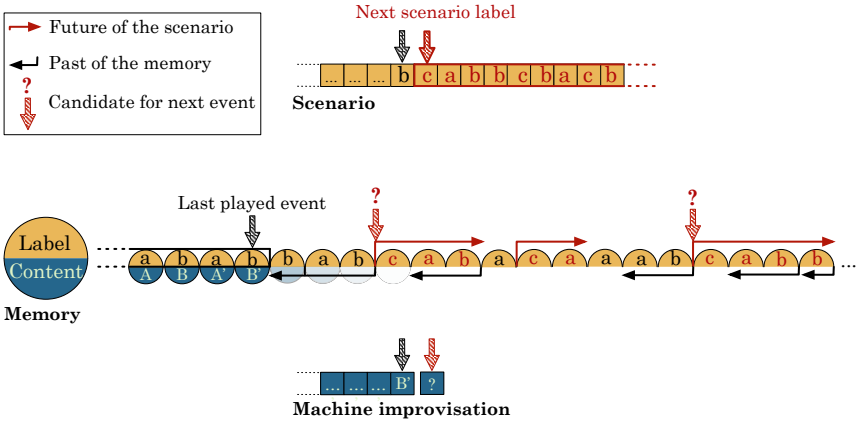


Figure 4.2: Using the scenario to introduce anticipation in the music generation process.

DIGRESSION The scenario/memory model searches for the continuity of the musical discourse by exploiting the similar patterns in the sequences, and the ability to create new contents that go beyond copy using the regularities in the memory (see the algorithms

in Chapter 5). This last condition maintains the coherence of the musical discourse when digressing from the original material, that is to say when non-contiguous subsequences of the memory are chained in the machine improvisation. Figure 4.2 represents a step in the improvisation process. It combines anticipation and coherence with the musical logic of the memory by searching for events ensuring both *continuity with the future of the scenario* (red arrows) and *continuity with the past of the memory* (black arrows).

COMPOSED IMPROVISATION Finally, we will show in Chapter 6 that the scenario approach can be used to go beyond simple conformity criteria and widens to the *composition of improvised performance*, in an idiomatic context or not.

4.3 "Hybridization": the Example of Jazz Improvisation

Basically, "hybridization" means here being able to improvise on *Blue in green* using some material which is exogenous to *Blue in green*. The example of jazz improvisation illustrates how anticipation can be used to create "hybrid" improvisations, and how a transformation such as transposition can be used to virtually increase the size of the memory. In the case of a scenario defined as the chord progression of a jazz standard and a memory recorded on different chord progressions, the simple idea is the following: if the scenario requires a *ii-V-I* progression, retrieving a *ii* located in a *ii-V-I* progression, then a *V* located in a *V-I* progression... is likely to produce a better result than the concatenation of a *ii*, a *V*, and a *I* independently retrieved in the memory.

Depending on the nature of the alphabet, adapted heuristics can be defined to complete the generic algorithm. In this example, we define the scenario and the memory on the alphabet constituted by the four notes chords deriving from the harmonization of the major scale commonly found in jazz harmony. Figure 4.3 presents the possible evolutions through the jazz standard *Blue in green* (scenario) only using the longest factors retrieved from the harmonic progression of *Autumn leaves* (memory). Here, we define equivalence on the labels modulo transposition, and an associated transformation of the contents when the slices are outputted in a transposed state (see 6.1.1 for the generic approach). When using a harmonic alphabet, more complex equivalences can be defined, for example transformations defined by chord substitution grammars (Chemillier, 2004).

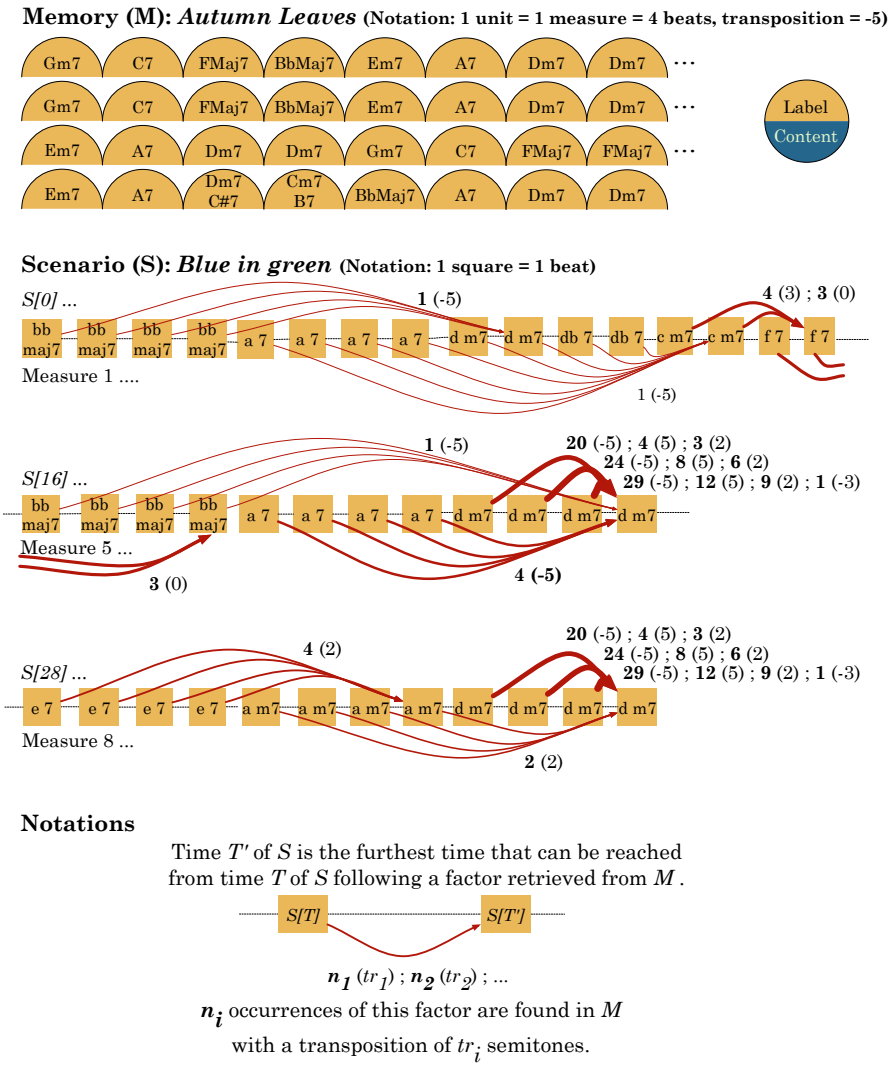


Figure 4.3: Example of improvisation using a harmonic alphabet: some ways to improvise on *Autumn Leaves* using an interpretation of *Blue in green* (simplified representation: only the longest factors).

To simplify the example in Figure 4.3, only the longest factors are represented: an arrow leaving a state of the scenario points on the furthest state that can be reached following a sequence extracted from the memory, in its original or transposed state. The number of factors matching this longest path is given for each of the relevant transpositions. In the case of jazz improvisation, transposition is an example of control on music generation: depending on the musical situation, one can sometimes prefer the longest paths whatever necessary transposition jumps (which may introduce discontinuities), sometimes chose the paths minimizing the transpositions even if some progressions or complete cadences that could be present in the memory with a different local tonality may be dismissed.

Video A.1.3

Hyperlink video *(or vimeo.com/jeromenika/improtek-lubat-early)*
Description: Appendix A.1.3.
Musical part IV: Chapter 17.

Video A.1.9

Hyperlink video *(or vimeo.com/jeromenika/improtek-sellin-themanilove1-finale)*
Description: Appendix A.1.9.
Musical part IV: Section 18.2.

Even though the formal processes at play are the same, using the model to generate music from live material or using an exogenous offline memory almost makes it two different instruments from a musical point of view. "Hybridization" will be discussed in Part IV presenting several collaborations with musicians (in particular in Chapter 17 and Section 18.2).

Video A.1.3 shows an example of such hybrid improvisations during a concert using an early MIDI version of the system ImproteK in which the generation model presented in this part is instantiated. The scenario is defined as a harmonic progression and the memory is a set of different jazz standards and ballads coming from previous improvisation sessions with different musicians (in particular Bernard Lubat and Jovino Santos Neto). The idea was to create a continuous musical discourse using an heterogeneous memory and combining short-term and long-term memory.

With another approach, **Video A.1.9** shows the finale of an improvisation by Hervé Sellin playing with the system. Its musical memory contains different recordings by Billie Holiday, Edith Piaf, and Elisabeth Schwartzkopf (singing Puccini, Mozart, and Mahler). The aim was here to create a quatuor with a live musician and a "virtual trio" with a patchwork aesthetics.

5

“Scenario / Memory” generation model

This chapter presents the algorithms involved in the scenario / memory generation model introduced in the previous chapter. First, Section 5.1 outlines the general algorithm. Then Section 5.2 focuses on the prefix indexing algorithm handling the continuity with the future of the scenario to introduce anticipatory behaviors, and Section 5.3 explains how the continuity with the past of the memory is obtained from the automaton structure chosen to learn the musical material. Finally, Section 5.4 deals with optimization of the algorithms.

5.1 The “Scenario / Memory” Algorithms

5.1.1 Objective

Our aim is to design a generation algorithm:

- navigating through a musical memory being guided by a scenario structure;
- providing exact matching, and all the solutions (not only the best);
- using the regularities in the memory to maintain coherence of the musical discourse when concatenating non-contiguous sequences;
- using the prior knowledge of the scenario to introduce anticipatory behaviors;
- segmented into generation phases to enable its integration in a dynamic and reactive architecture dedicated to real-time performance.

5.1.2 Notations and Definitions

The scenario and the sequence of labels describing the musical memory are represented as words on an alphabet A. Choosing an alphabet A for the labels of the scenario and the memory sets the equivalence classes labeling the musical contents in the memory (see Chapter 6.1 for some examples of alphabets).

Given a scenario S of length s, the letter at index T in S is denoted by $S[T]$. After defining a temporal unit for the segmentation, $S[T]$ is the required label for the time T of the improvisation. Given a memory M of length m, the letter at index k in M is denoted by $M[k]$. $M[k]$ is the equivalence class labeling the musical event corresponding to the date k in the memory M. In the following descriptions the memory will be assimilated to the word M. The labels and contents in the the memory will be distinguished when necessary using lower-case letters and upper-case letters respectively. For example, different musical contents B', B'', B''',... belonging to a same equivalence class b will be labeled by b.

Finally, the machine improvisation, that is to say the sequence of indexes of the events retrieved in M and concatenated to generate the improvisation, will be denoted by $\{i_T\}_{0\leq T<s}$.

Using the usual vocabulary, the zero letter sequence is called the *empty string* and is denoted by ϵ. A string x is a *factor* of a string y if there exist two strings u and v such that $y = uxv$. When $u = \epsilon$, x is a *prefix* of y; and when $v = \epsilon$, x is a *suffix* of y.

5.1.3 Current Scenario at Date T

The scenario gives access to a prior knowledge of the temporal structure of the improvisation to play. Anticipation can therefore be introduced by taking into account the required labels for the future dates to generate improvisation at current time T.

The *current scenario* at date T, denoted by S_T, corresponds to the suffix of the original scenario beginning at the letter at index T:

$$S_T \triangleq S[T]...S[s-1]$$

At each time T, the improvisation goes on from the last state i_{T-1} retrieved in the memory, searching to match the current suffix S_T of the scenario.

5.1.4 Anticipation and Digression: Definition of Index Sets

As introduced in Section 4.2, the model combines anticipation by ensuring continuity with the future of the current scenario S_T, and coherence with the musical logic of the memory M when digressing by maintaining continuity with the past of the memory. To achieve this, we define the following index sets of the memory M which are used in the scenario/memory generation algorithm (5.1.5):

FUTURE OF THE SCENARIO $\forall\, T \in [0, s[$,

$\underline{Future_S(T)}$

$=$ Indexes in M sharing a common future with S_T

$\triangleq \{k \in \mathbb{N} \mid \exists\, c_f \in \mathbb{N},\, M[k]...M[k+c_f-1] \in \text{Prefixes}(S_T)\}$

$k \in Future_S(T)$ is the left position of a factor of M equal to a prefix of the current scenario S_T. $M[k]$ shares a common future with $S[T]$ and provides *continuity with the future of the scenario* measured by the length c_f of the prefix.

The maximum length c_f associated to an index $k \in Future_S(T)$ is denoted by $c_f(k,T)$. See 5.2 for the algorithms to build $Future_S(T)$.

PAST OF THE MEMORY $\forall\, i \in [0, m[$,

$\underline{Past_M(i)}$

$=$ Positions in M sharing a common past with the event $M[i]$

$\triangleq \{k \in \mathbb{N} \mid \exists\, c_p \in [1, k],\, M[k-c_p+1]...M[k] \in \text{Suffixes}(M[0]...M[i])\}$

$k \in Past_M(i)$ is the right position of a factor of M equal to a suffix of $M[0]...M[i]$. $M[k]$ shares a common past with $M[i]$ and provides *continuity with the past of the memory* measured by the length c_p of the suffix. The maximum length c_p associated to an index $k \in Past_M(i)$ is denoted by $c_p(k,i)$. See 5.3 for the algorithms to build $Past_M(i)$.

> $c_f(k,T)$ and $c_p(k,i)$ previously introduced measure continuity regarding the future of the scenario and continuity regarding the past of the memory respectively.
>
> As we will see later on, these algorithmic parameters have a strong influence on the musical result, and imposing maximum values C_f and C_p for $c_f(k,T)$ and $c_p(k,i)$ respectively is among the provided real-time commands driving the system.

CHAINING SEQUENCES $\forall\, T \in [0, s[,\, i \in [0, m[$,

$\underline{Chain_{S,M}(T,i)}$

$=$ Positions in M starting a sequence chaining with $M[i]$ at time T

$=$ Positions in M sharing a common future with the current scenario S_T and preceded by a sequence sharing a common past with the event $M[i]$

$\triangleq \{k \in \mathbb{N}^* \mid k \in Future_S(T) \text{ and } k-1 \in Past_M(i)\}$

As illustrated in Figure 5.1, $k \in Chain_{S,M}(T,i)$ shares a common future with the current scenario S_T, and is preceded by a sequence sharing a common past with the event at index i. The indexes k in $Chain_{S,M}(T,i)$ are the left positions of sequences of length $c_f(k,T)$ constituting possible fragments of improvisation starting at time T.

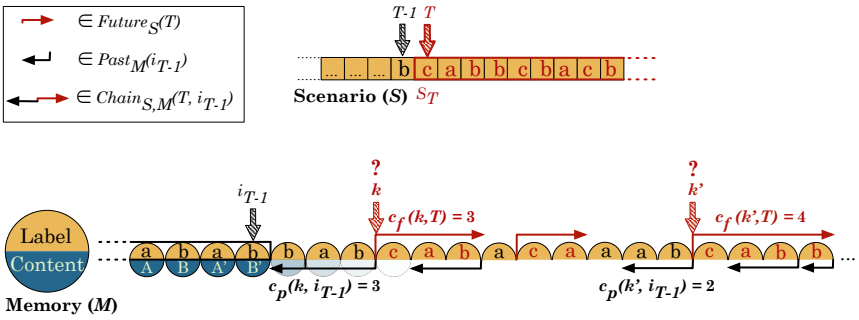


Figure 5.1: Construction of $Chain_{S,M}(T, i_{T-1}) = \{k, k'\}$: positions in M sharing a common future with S_T, and preceded by a sequence sharing a common past with the event $M[i_{T-1}]$.

Besides, they offer smooth transitions from $M[i]$ thanks to their common past of length $c_p(k, i)$, assuming that jumps between two segments sharing a common past preserve a certain musical homogeneity.

LOCAL CONTINUATIONS The sequences starting at positions $k \in Chain_{S,M}(T, i)$ can be simply copied[1]. To go beyond simple copy, $k \in Chain_{S,M}(T, i)$ can be used as a starting point to follow an equivalent non-linear path using regularities of the memory to jump to other zones sharing a common past with the previously retrieved slice. To cover both cases we define the set of possible *continuations* from the index $i \in [0, m[$ in M at date $T \in [0, s[$ as:

$\underline{Cont_{S,M}(T, i)}$

$= $ Possible continuations from $M[i]$ at time T

$= $ Positions in M matching the label $S[T]$ of the scenario and preceded by a sequence sharing a common past with the event $M[i]$

$\triangleq \{k \in \mathbb{N}^* \mid M[k] = S[T] \text{ and } k - 1 \in Past_M(i)\}$.

Finally, i_T is chosen in $Cont_{S,M}(T, i_{T-1})$.

5.1.5 Outline of the Generation Algorithm

SEGMENTATION INTO GENERATION PHASES The generation process is divided into successive generation *phases*. These successive navigation phases through the memory segment the algorithmic process, but they do not correspond in general to distinct musical phrases. Each *phase* is constrained by a suffix of the scenario. With ϕ_n the suffix of the scenario S constraining the generation phase n:

1 In this case, given $k \in Chain_{S,M}(T, i)$, $\forall l \in [0, c_f(k, T)[, i_{T+l} = k + l$.

$$\phi_n \triangleq \begin{cases} \phi_0 = S_0 \\ \phi_n = S_{l(0)+\ldots+l(n-1)} = S_{L(n-1)} \end{cases},$$

$l(n)$ being the length of the improvisation fragment generated in the generation phase associated to ϕ_n, and $L(n)$ being the total length of the improvisation generated with the successive generation phases associated to ϕ_0,... ϕ_n.

First, thanks to this design, the model can be queried using temporal queries (portions of scenario) that enable to implement anticipatory behavior and to generate anticipations ahead of performance time when the model is used in a real-time context (see Part II). Second, it enables to optimize a generation phase using the results of the previous ones (see 5.4.1).

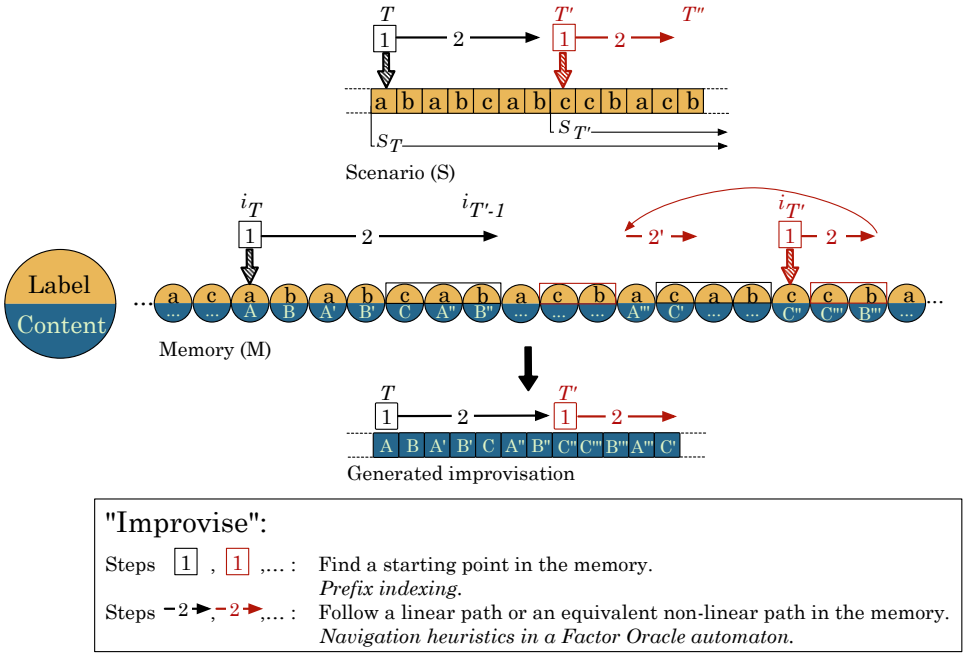


Figure 5.2: Scenario/memory generation model: example of two successive generation phases, $\phi_n = S_T$ (black) then $\phi_{n+1} = S_{T'}$ (red).

A GENERATION PHASE Figure 5.2 gives an example of two consecutive generation phases. The first one generates a fragment of improvisation starting from date T satisfying the current scenario S_T. At the end of this first phase, the prefix $S[T]...S[T'-1]$ of S_T has been processed. A new research phase over the suffix $S_{T'}$=$S[T']...S[s-1]$ of S is then launched to complete the improvisation up to $T''-1$. Among the possible candidates, $i_{T'}$ is chosen using $Past_M(i_{T'})$ (black rectangle). In the example of this second phase (red), the regularities in the memory indexed by the sets $Past_M(i_{T'}+2)$ are used to follow a non-linear path (red rectangle).

To summarize: each phase consists in two successive steps involving the previously defined index sets: *1-Anticipation*: find an event in the memory sharing a common future with the scenario while ensuring continuity with the past of the memory. *2-Copy or digression*: retrieve the whole sequence (example of the phase S_T, black) or use the regularities in the memory to follow an equivalent non-linear path (example of the phase $S_{T'}$, red) [2].

GUIDED GENERATION ALGORITHM Formally, the generation algorithm is summarized in Algorithm 1 and consists in:

ALGORITHM 1 . Guided generation algorithm

Inputs :
Scenario S of length s,
Memory M of length m,
Set of secondary generation parameters P (filtering, rules, etc.).
Output : $\{i_T\}_{0 \leq T < s}$, indexes of the slices retrieved in M and concatenated to generate the improvisation.

```
1  T = 0;
2  while T < s do
3      /* A generation phase:                                          */
4                                              /* Step 1: Starting point */
5      if T > 0 and Chain_{S,M}(T, i_{T-1}) ≠ ∅ then
6          i_T ← k ∈ Chain_{S,M}(T, i_{T-1}) satisfying P
7      else
8          i_T ← k ∈ Future_S(T) satisfying P (or alphabet-specific heuristics)
9      end
10     T++
11                                                  /* Step 2: Navigation */
12     while T < s and Cont_{S,M}(T, i_{T-1}) ≠ ∅ do
13         i_T ← k ∈ Cont_{S,M}(T, i_{T-1}) satisfying P
14         T++
15     end
16 end
```

1. *Anticipation: searching for a starting point* (l. 5-9 in Algorithm 1, steps 1 in Figure 5.2).
   The search first looks for events in M sharing a common future with the current scenario S_T and a common past with the last retrieved index i_{T-1} in M, i.e. $Chain_{S,M}(T, i_{T-1})$. When none of the events in M can provide both continuity with the future of the scenario and continuity with the past of the memory, only the first criterion is searched, *i.e* the continuity with the future of the current scenario. If no solution is found, alphabet-dependent transformations are used (see Chapter 6.1).

2 Note that depending on the chosen continuity parameters, this path can be shorter or longer than the prefix of the current scenario chosen in step 1 (see the `while` in Algorithm 1, line 12).

2. *Copy or digression: navigating through the memory* (l. 12-15 in Algorithm 1, steps 2 in Figure 5.2). After finding a factor of M matching a prefix of S_T, it can be copied or used as a starting point to follow an equivalent non-linear path in the memory using the continuations in $Cont_{S,M}(T, i_{T-1})$ until launching a new phase is necessary.

> At each of theses steps, the concerned index sets are built and the selection among the candidate positions ("$i_T \leftarrow k \in ...$" in Algorithm 1) is done in order to satisfy a set of *secondary generation parameters*. This set contains all the parameters driving the generation process which are independent from the scenario: parametrization of the generation model and content-based constraints to filter the set of possible results returned by the algorithm (see Section 6.2).

5.2 Continuity with the Future of the Scenario

$Future_S(T)$ (defined in 5.1.4) is the index set of M defined to deal with continuity with the future of the scenario. This section gives an overview of the proposed algorithm to build $Future_S(T)$ and then $Chain_{S,M}(T, i_{T-1})$. The construction of $Future_S(T)$ comes down to the general problem of indexing the prefixes of a pattern X=$X[0]...X[x-1]$ in a word Y=$Y[0]...Y[y-1]$.

5.2.1 Outline of the Algorithm

As illustrated in Figure 5.3, the algorithm for indexing the *prefixes* of X in Y follows the outline of the classic algorithms for indexing the *occurrences* of a pattern in a word: comparisons and calls to a failure function to shift a sliding comparison window in such a way that redundant comparisons are avoided.

The algorithm presented below uses the failure function f of the Morris-Pratt algorithm (Morris and Pratt, 1970) which indexes the occurrences of the pattern X in Y by describing the run of the deterministic finite automaton recognizing the language A^*X (where A is the alphabet on which X and Y are defined) on the word Y. The algorithm for indexing the prefixes of X in Y is divided into a preprocessing phase on the pattern X and a searching phase represented in Figure 5.3. The preprocessing phase provides the tools used in the searching phase: the failure function f and the function B defined below.

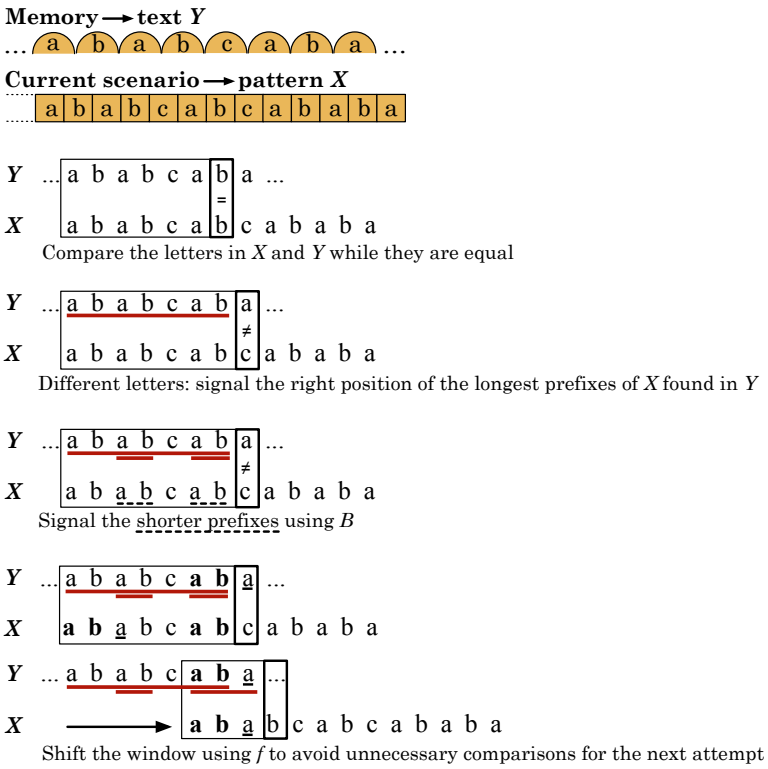


Figure 5.3: Indexing the prefixes of a pattern X in a text Y.

5.2.2 Preprocessing Phase

A *proper factor* of X is a factor of X different from X, and a *border* of a non-empty string X is a proper factor of X that is both a prefix and a suffix of X. The function f is defined as follows (see (Crochemore et al., 2007) for the construction of the borders table and f):

$$\forall i \in [1,x], f(i) \triangleq \begin{cases} f(0) = -1 \\ \text{length of the longest border of } X[0]...X[i-1] \end{cases},$$

and, when it exists, $\forall i \in [1,x], k \in \mathbb{N}\backslash\{0,1\}$,

$$f^k(i) = \text{length of the longest border of } X[0]...X[f^{k-1}(i)-1].$$

f is used as a *failure function* in the algorithm: f computed on the pattern X indexes some regularities in X which are used in the searching phase to shift the sliding window from the last index i to a relevant index $f^k(i)$ so that unnecessary comparisons are avoided (last step in Figure 5.3).

f is defined in such a way that when a prefix is found during the searching phase, it is the longest prefix of X at the concerned position of Y, and the indexes it covers will not be visited during the following attempts.

A shift of the sliding window has to be *valid, i.e.* it has to ensure that no prefixes are forgotten when sliding the window. We have therefore to signal the shorter prefixes that may be included in the previously found longest prefix before sliding the window (third step in Figure 5.3). This consists in reporting the relevant prefixes of X within X itself in this found prefix. In addition to its use as a failure function, we therefore use f to obtain the locations of the prefixes of X within X itself. To do so, we define the function B, with $B(i)$ the set of the lengths of all the borders of $X[0]...X[i]$:

$$\forall i \in [1, x-1], B(i) \triangleq \{\text{length of } b \mid b \text{ border of } X[0]...X[i]\}.$$

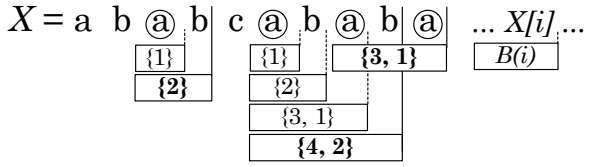


Figure 5.4: $B(i)$: sets of the lengths of the borders of $X[0]...X[i]$. The locations of the non-trivial occurrences of all the prefixes of the pattern X in X itself are then deduced from B (rectangles).

As illustrated in Figure 5.4, the locations of prefixes of X within X itself and their lengths are immediately deduced from B (rectangles in Figure 5.4), and B is directly obtained from f: indeed, by a simple recurrence (Proposition 1.5 in Crochemore et al., 2007):

$$\forall i \in]1, x[, B(i-1) = \{f(i), f^2(i), ..., f^n(i) > 0\}$$

5.2.3 Searching Phase: Indexing the Prefixes of the Current Scenario S_T in the Memory M

To summarize, the word X and the text Y are compared:

1. when a mismatch occurs after at least one match, the position is discarded in the Morris-Pratt algorithm, but in our case it is signaled because it is the end of a prefix,
2. before sliding the comparison window: the set B is used to signal the shorter prefixes that may be included in the longest prefix found during 1.

The failure function f is used twice: first, to slide the comparison window as in Morris-Pratt, second to build the set B used to index the prefixes of the word within itself.

Finally, with $X = S_T$ and $Y = M$, $Future_S(T)$ is built as follows:

1. Preprocessing phase: construction of f on S_T, hence B.
2. Searching phase, index the prefixes of S_T in M:
   Comparisons of the letters in S_T and M.
   When $S_T[i] \neq M[j]$ and $i > 0$:
   a) $(j-1)$ is the right position of a prefix of length i of S_T found in M: $(j-i) \in Future_S(T)$
   This is the longest prefix of S_T in M ending in $j-1$.
   b) Use B to signal the shorter prefixes of S_T in $M[j-i]...M[j-1]$.
   c) Go backward in S_T with f to avoid unnecessary comparisons for the next attempt.
3. Combine with $Past_M(i)$ (see 5.3) to get $Chain_{S,M}(T,i)$.

The positions and the lengths of the prefixes are recorded in a table. The lengths of the prefixes correspond to the parameters c_f introduced in 5.1.4.

5.2.4 Complexity and execution time

To summarize, the searching phase indexes the prefixes of S_T in M by signaling some locations dismissed in the Morris-Pratt algorithm, without proceeding to extra comparisons or moves backwards. We wanted a simple forward algorithm, this simple implementation therefore justifies the use of the failure function of the Morris-Pratt algorithm. Indeed, although f is not optimized (for example in comparison to that of (Knuth et al., 1977) or (Boyer and Moore, 1977)), it enables to move easily from the research of occurrences to the research of prefixes. Furthermore, the use of this failure function f from which we immediately get B enables to optimize a research phase using the results of the previous ones, as it will be detailed in Section 5.4.

Like the Morris-Pratt algorithm, the searching phase of the algorithm runs in time $\Theta(m)$ and does not exceed $2 * m - 1$ comparisons, and the preprocessing phase runs in time $\Theta(s - T)$. This execution time and this failure function proved to be suitable for our use case (this result is empirical: the improvisation system in which this model is implemented has been used many times during performances and work sessions with expert musicians). Indeed, the first fields of musical experimentation with the model were jazz chord progressions, and processed sequences contained therefore multiple regularities. Because of the harmonic rhythm (generally 2 or 4 beats), they are often of form $...x^4y^2z^2x^8...$

5.3 Continuity with the Past of the Memory

> **Remark:** The first generation model we set up is not described in this dissertation (see Nika and Chemillier, 2012). It is mentioned as "conformity model" since it *only* ensured the conformity of the machine improvisation to the scenario while digressing but did not implement any anticipatory behavior (see an example in **Video A.3.1**). It consisted in step by step filtering of possible paths when navigating through the automaton structure described in this section, which is used in a different way. The overall generation process presented in this chapter offers both conformity and anticipatory behavior.

Video A.3.1

Hyperlink video *(or vimeo.com/jeromenika/improtek-archive-recombine)*
Description: Appendix A.3.1.
Musical part IV: Chapter 17.

Continuity with the past of the memory is handled with $Past_M(i)$ introduced in 5.1.4. The sets $Past_M(i)$ are used both to filter the set of sequences sharing a common future with the current scenario to get the chaining sequences $Chain_{S,M}(T,i)$ (5.1.4), and to add non-linear paths to the set of possible continuations $Cont_{S,M}(T,i)$ when navigating through the memory (5.1.4). This section details how these sets are obtained from the automaton structure chosen to learn the musical memory: the Factor Oracle automaton (Allauzen et al., 1999; Lefebvre et al., 2002).

This automaton was introduced to remedy the difficulty to build a deterministic finite automaton recognizing the language constituted by the factors of a word. The Factor Oracle built on a word X is a deterministic finite automaton accepting at least the factors of X (for each of these words, there exists at least one path labeled by it in the automaton, leading to a final state). In the context of a musical application, this automaton presents the advantage of keeping the sequential aspect of the temporally structured musical memory, and does not aggregate in the same state all the contents labeled by the same equivalence class. Moreover, its construction algorithm (see (Allauzen et al., 1999)) is incremental and linear in time in space. It is therefore particularly relevant for real-time applications (see (Assayag et al., 2006b) for the application of this automaton to the issue of *stylistic reinjection* for human-computer music improvisation).

Like in most of the systems using the Factor Oracle for music improvisation, the automaton is not used here to proceed to proper pattern matching but in an indirect way: some construction links, the *suffix links*, carry the information to build the sets $Past_M(i)$. The Factor Oracle construction function, the *suffix link function* s, is used to index regularities in the memory.

The function s computed on a word X is defined as follows:

$s(i) \triangleq$ leftmost position where
a longest repeated suffix of $X[1]...X[i]$ is recognized.

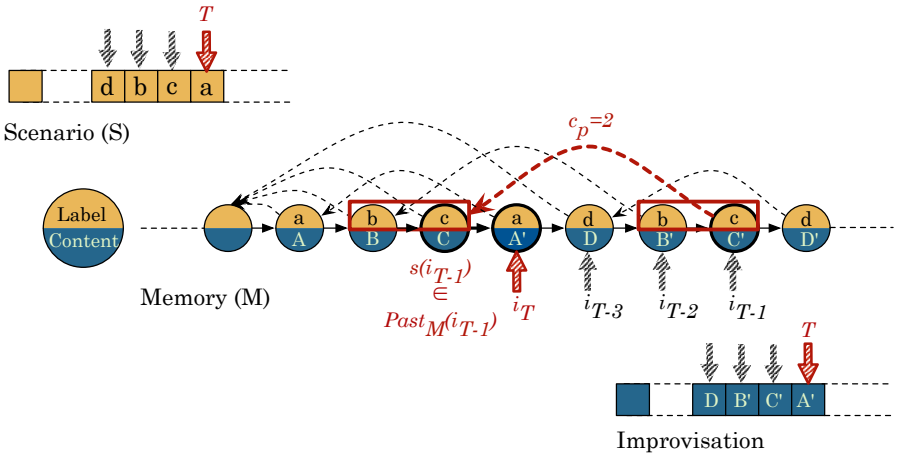


Figure 5.5: Using the regularities of the memory (s suffix link function of the Factor Oracle memory) to follow non-linear paths (continuations) or chain disconnected sequences while preserving musical coherence.

As illustrated in Figure 5.5, the suffix links index repeated patterns in the sequence and guarantee the existence of common suffixes between the elements that they link. These common suffixes are seen here as musical pasts shared by these elements: $s(i) \in Past_M(i)$. The main postulate of the musical models using the Factor Oracle is that following non-linear paths using these links creates musical phrases proposing new evolutions while preserving the continuity of the musical discourse, as studied in (Assayag and Dubnov, 2004).

To summarize, this section extends the heuristics for improvisation, harmonization and arrangement presented in a previous work (see Appendix B) based on the Factor Oracle navigation proposed in (Assayag and Bloch, 2007), adapted to a guided context. The navigation chains paths *matching the scenario* in the automaton, and alternates between linear progressions and jumps following suffix links. The length of the common *context* between two musical events in a sequence is computed in (Assayag and Dubnov, 2004) by embedding the method introduced in (Lefebvre and Lecroq, 2000) (linear in time and space) in the construction algorithm. The length of this context corresponds to the parameter c_p (5.1.4) which quantifies the expected “musical quality” of a jump. According to empirical observations made during performances and work sessions with musicians, the subset of $Past_M(i)$ reached by chaining calls to suffix links and reverse suffix links meets the requirements for chaining sequences and digressing from the original sequences.

5.4 Additional Information and Optimizations

5.4.1 From a Generation Phase to the Others

To find a starting point for a new generation phase at a given index T, $Future_S(T)$ is built by indexing the prefixes of the current scenario S_T in the memory M (see Algorithm 1, Section 5.1). The current scenario S_T=$S[T]...S[s-1]$ being by definition a suffix of the scenario S, repeating this operation for different indexes T, T', T''... amounts to indexing *some* factors of S in M. It is important to underline that we do not want to index *all* the factors of S in M, but only those who are necessary. Indeed, to generate the whole improvisation, we only need to index in M the prefixes of *some* suffixes ϕ_n of S (see 5.1.5):

$$\forall n < s, \phi_n \triangleq \begin{cases} \phi_0 = S_0 \\ \phi_n = S_{l(0)+...+l(n-1)} = S_{L(n-1)} \end{cases},$$

$l(n)$ being the length of the improvisation fragment generated in the generation phase ϕ_n, and $L(n)$ being the total length of the improvisation generated with the successive generation phases ϕ_0, ... , ϕ_n.

> We show here that the outline of the algorithm for prefix indexing in a generation phase (Section 5.2), and in particular the use of the failure function f, were also motivated by the fact that some information computed during the preprocessing phase, the sets B (5.2.2), can then be used to optimize the prefix indexing for a given phase ϕ_n using some results from the previous phases ϕ_m, $m < n$ when it is possible.

Indeed, the solution is the following (summary in Figure 5.6):

- When indexing the prefixes of ϕ_m=$S_{L(m-1)}$=S_T in M: store the result as well as the lengths

  $$\{B_T(i)\}_{i<s-T} = \{\text{length of } b \mid b \text{ border of } S[T]...S[T+i]\}_{i<s-T}$$

  which are computed during the preprocessing step of the research (see 5.2.2).

- Later, when indexing the prefixes of ϕ_n=$S_{T'}$ in M with $n > m$, $T' > T$, if ϕ_n=$S_{T'}$ and ϕ_m=S_T have a common prefix:
  - the length $LCP(m,n)$ of this common prefix can be immediately retrieved from $\{B_T(i)\}_{i<s-T}$ stored during the research associated to ϕ_m=S_T;
  - the prefixes of ϕ_n=$S_{T'}$ of length $\leq LCP(m,n)$ in M are immediately given using the result of the research for ϕ_m=S_T,

– the prefixes of ϕ_n=$S_{T'}$ of length $> LCP(m,n)$ in M can be obtained by concatenating the results stored for ϕ_m=S_T with the results of the research for a shorter suffix: the suffix $S_{T'+LCP(m,n)}$.

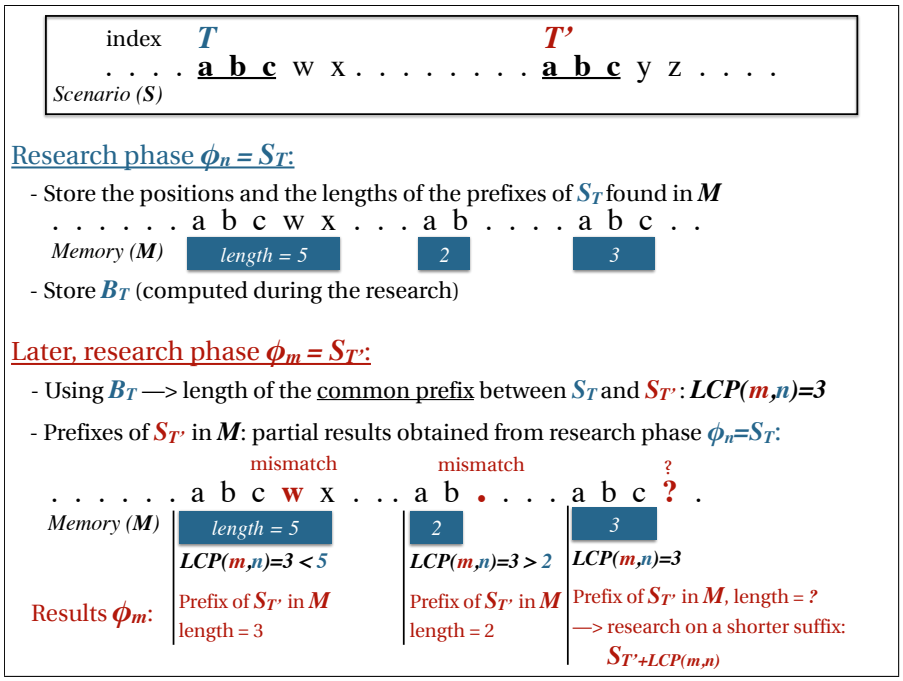


Figure 5.6: Optimization of a research phase using stored results.

5.4.2 Using Vectors of Descriptors as Labels

In the general case, the events in the memory can be labelled by vectors of descriptors and each descriptor can have a different role. This way, the generation process is able to provide continuity with the future of the scenario and continuity with the past of the memory regarding different descriptors. To simplify the figures in Chapter 4 and Chapter 5, and the notations in 5.1.4, we defined $Future_S(T)$ and $Past_M(i)$ in relation to the labels $M[i]$ of the memory M. In reality, in the general case, they are defined in relation to $M[i][j]$, j being the dimension of the descriptor chosen to build $Future_S(T)$ or $Past_M(i)$.

This way, for example, the scenario can be defined as a chord progression guiding the improvisation with harmonic labels, and the continuity with the past of the memory can be achieved on energy descriptors so that the improvisation avoids perceptive discontinuities when jumping between discontiguous regions of the memory.

In this case, the researches associated to the different continuities (future of the scenario: prefix indexing, Section 5.2; past of

> the memory: Factor Oracle automaton, Section 5.3) cannot benefit from each other since they are not performed on the same descriptor.

5.4.3 Optimizing a Generation Phase in the Case of Unidimensional Labelling of the Memory

The algorithm indexing the prefixes of a suffix S_T of the scenario in the memory provides all the required solutions and empirically proved to be fast enough for our application (see Section 5.2). However, in the case of unidimensional labelling of the memory, when the researches presented in Chapter 4 and Chapter 5 are realized on the same descriptor, it could be optimized from a theoretical point of view by using the regularities in the memory that are computed to ensure the continuity with the past of the memory.

In our case, these regularities in the memory are provided by the Factor Oracle automaton. It is built on the *text* (the memory) and not on the *pattern* (the current scenario) contrary to the classic algorithm of string matching to index occurrences of a pattern in a text using this automaton (Allauzen et al., 1999), and, as mentioned in the previous paragraph, it cannot be used in the general case to design the prefix indexing algorithm. Nevertheless, in this particular case, the metadata on the text that it provides could be used to prune the researches with additional heuristics.

However, the Factor Oracle only provides an approached index of the word it is built on. Maniatakos (2012) worked on the Factor Oracle to introduce the Multi-Factor Graph (MFG). The MFG is an automaton whose structure is close to that of the Factor Oracle, but which provides a lossless compact representation for continuation probabilities in a musical sequence. When working on the optimization of this particular case in future work, using this automaton to learn the memory could enable to have both the information we need to ensure continuity regarding the past of the memory (that are given by the suffix links in the Factor Oracle), and optimize the prefix indexing step using an exact index of the text (memory).

6

Scenarii, Scenarios... and "Meta-Composition"

In the video examples of the previous chapters, the musical purpose of the scenario was to ensure the conformity to the idiom it referred to, and to introduce anticipation in the generation process. Other musical directions than improvisation in an idiomatic context can be explored using the formal genericity of the couple scenario / memory and the possibility to define dynamic scenarios, that is to say scenarios modified during the performance (see 8.1.2 in Part II). Defining scenarios described with other idiomatic vocabularies, audio-musical descriptors, or any user-defined alphabet can lead to approach new dimensions of guided interactive improvisation.

Lewis (2000) underlines the fact that "musical computer programs, like any texts, are not objective or universal, but instead represent the particular ideas of their creators". As mentioned previously, Rowe (1999) outlines that the delegation of some of the creative responsibility to a computer and a performer when designing interactive musical systems pushes up musical composition "to a meta-level captured in the processes executed by the computer", and that "an interesting effect of this delegation is that it requires a very detailed specification of the musical decisions needed to produce a computer program at the same time that the composer cedes a large measure of control over musical decision-making to the human improviser". This chapter shows the genericity of the scenario / memory approach, and how it can be used to add a meta-level of authoring and composition in addition to that constituted by the design of the scenario itself. In this framework, musicians for whom the definition of a musical alphabet and the design of scenarios for improvisation is part of the creative process can be involved in this "meta-level" of composition, *i.e.* involved upstream to design a part of this "delegation".

6.1 From the Conformity to an Idiomatic Structure to Composed Improvisation Sessions

6.1.1 Defining an Alphabet

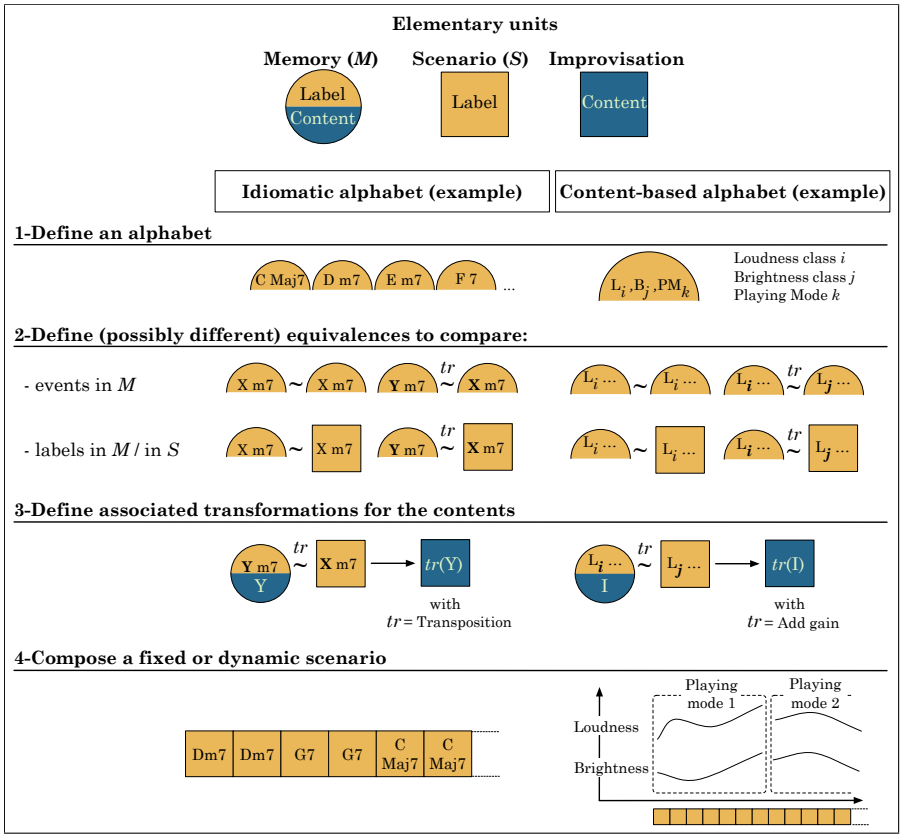


Figure 6.1: A protocol to compose improvised performances.

Figure 6.1 sketches a protocol to compose improvisation sessions. The generation model is implemented as a modular library extending this formal genericity in its implementation. It is designed to provide a protocol to compose musical sequences or improvised performances:

1. Define a segmentation unit and a musical alphabet for the labels.
2. Define the properties of this alphabet, *i.e.* equivalences and comparison methods between the labels. These equivalences can be different for the comparisons memory / memory (involved in learning) and the comparisons scenario / memory (involved in generation).
3. Define transformation rules between the musical contents belonging to the different equivalence classes.

4. Compose at the structure level (*i.e.* define a fixed or dynamic scenario).

Figure 6.1 gives two applications of this protocol with an idiomatic alphabet and a content-based alphabet. The content-based alphabet is illustrated with the example of a vector of chosen audio-descriptors. Transposition of the musical content or applying gain to the signal as in the examples in Figure 6.1 are intuitive transformations when the chosen alphabet is respectively harmonic or including a loudness descriptor. In the case of an arbitrary alphabet, this mechanism can be used in a creative way to define equivalences modulo user-defined transformations.

Drawing a distinction between these different alphabets is not only a technical question: running a generation model using regularities and common patterns in temporal structures leads to different musical results depending on whether these temporal structures describe an underlying formal evolution or the evolution of low-level signal features. Besides, these cases lead to differentiate the musical roles played by the scenario. When the scenario is defined over an idiomatic or arbitrary alphabet describing prior knowledge of an underlying structure, it represents a common referent for all the musicians and the machine. Therefore no analysis mechanisms are needed to label the live musical inputs since the machine shares a common plan with the musicians. On the contrary, in the case of a content-based alphabet, an online or offline analysis is required for learning the memory. The scenario may only describe the part of the machine which improvises without prior knowledge of its musical inputs. The typology of alphabets is thus strongly linked to the typology of scenarios which will be studied later on in Part II focusing on real time and reactivity.

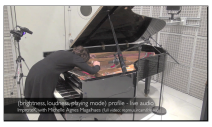
6.1.2 Some Examples of Scenarios Defined on Different Alphabets

The examples given in Chapter 4 used scenarios defined on harmonic alphabets. The following examples illustrate the application of the protocol presented in Figure 6.1 in three different contexts.

SCENARIO DEFINED ON A CONTENT-BASED AUDIO ALPHABET

Video A.1.5 shows an example of improvisation using a composed scenario (without pulse) using a content-based alphabet. It presents the first technical experiments with composer-improviser Michelle Agnes who works on structured improvisation. The chosen content-based alphabet is a 3−uple: loudness, brightness, playing mode on the prepared piano. This example illustrates the case where the scenario only describes the part of the machine improvisation. The system re-injects the live audio material matching the playing modes and descriptor profiles imposed by the scenario. This test initiated a

Video A.1.5



Hyperlink video *(or vimeo.com/jeromenika/improtek-agnes-composed)*
Description: Appendix A.1.5.
Musical part IV: Section 18.3.

future project using a scenario which will be composed so that the machine improvisation alternates between counterpoint and extension of the musical gesture of the musician.

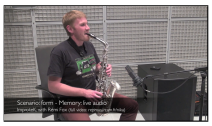
Video A.2.2

Hyperlink video *(or vimeo.com/jeromenika/improtek-starwospheres)*
Description: Appendix A.2.2.

USING THE ANALYSIS OF A TARGET AUDIO FILE AS SCENARIO **Video A.2.2** shows a short example of offline generation using the analysis of a target audio file as scenario. The content-based scenario is the profile of spectral centroid and roughness extracted from the soundtrack of a musicless movie scene (only sound effects) segmented into audio events. It is applied to a memory constituted by the piece *Atmospheres* (Ligeti) analyzed with the same couple of audio descriptors. The generated sequence replaces the original soundtrack.

COMPOSED SCENARIO USING AN ABSTRACT ALPHABET The idiomatic case is illustrated in Figure 4.3 with a harmonic alphabet. It should be noticed that this category can also cover specific metric structures, clave patterns, musical accents,... or any underlying structure materialized or not in the music itself. It generalizes to arbitrary user-defined alphabets so that a musician can define her/his own grammar associated to a specific online of offline musical material. **Video A.1.6** shows a structured improvisation during a work session with Rémi Fox. The software starts with an empty musical memory and improvises several voices by reinjecting the live audio material which is processed and transformed online to match the composed scenario while being reactive to external controls. The scenario defines two voices ("accompaniment" or "solo") and a form defined on an abstract alphabet (see the scenario in Appendix A.1.6, and the related section, Section 18.1).

Video A.1.6



Hyperlink video *(or vimeo.com/jeromenika/improtek-fox-composed)*
Description: Appendix A.1.6.
Musical part IV: Section 18.1.

These examples are detailed in Part IV focusing on the different musical collaborations carried out during the thesis.

6.2 Secondary Generation Parameters and Filtering

In order to introduce a more expressive and local dimension to the authoring that the generation model and the protocol in Figure 6.1 provide, we added some *secondary generation parameters* to drive the execution of the generation process. The scenario itself is indeed a sequence of formal constraints and, in most of the cases, does not carry the narrative aspect of improvisation.

> SECONDARY GENERATION PARAMETERS We call *secondary generation parameters* the parameters that are independent from the scenario.

- the *parametrization of the generation model*: contraints on the memory region, authorized transformations, maximal/minimal length of the sequences retrieved in the memory, measure of the linearity/non-linearity of the paths in the memory, *i.e.* constraints on the parameters c_f and c_p introduced in 5.1.4, etc.
- *user-defined content-based constraints* to filter the set of candidates matching the scenario: for example pitch interval, onset density, user-defined thresholds or rules, etc.

These secondary generation parameters can be used when generating a whole sequence, but this idea of "expressivity" is more relevant at a local level. Their use can therefore be scripted in an offline process, but they were mostly introduced to provide another level of real-time control when generating an improvisation matching a scenario. They are thus strongly linked to the notion of reaction, which will be the topic of Part II: the offline generation model presented in this part is embedded in a chain of dynamic agents to form an improvisation system that reacts to the online control of some *reactive inputs*, the scenario itself and these secondary generation parameters. The same way the scenario and the corresponding alphabet can be designed for a particular projet, the secondary generation parameters are defined by the user to compose reactivity (see Chapter 9).

Part II

"ANTICIPATIONS": GUIDED IMPROVISATION AS DYNAMIC CALLS TO AN OFFLINE GENERATION MODEL

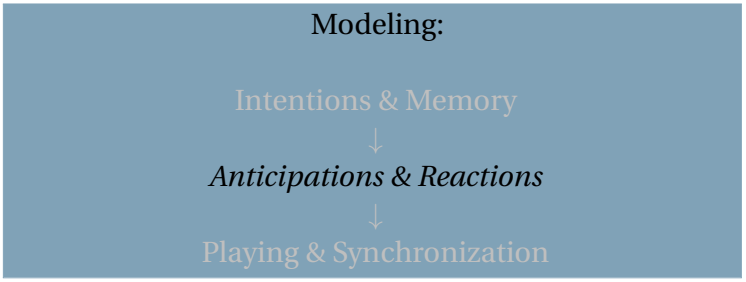
Part II introduces the paradigm of modeling guided improvisation as an offline process embedded in a reactive architecture to combine planning and reactivity.

Chapter 7 summarizes the contributions of Part II.

Chapter 8 presents the general architecture of ImproteK, and how the scenario / memory generation model introduced in Part I is used in a real-time context to generate anticipations ahead of the performance time. The following chapters detail the self-consistent agents constituting this architecture.

Chapter 9 proposes a model of reactive agent, the *Improvisation Handler*, handling dynamic calls to a generation model relying on a formal temporal specification to introduce a notion of reaction "over time". This agent reacts to external events by composing new mid-term anticipations matching the scenario ahead of performance time.

Chapter 10 models the interface between the environment and a dynamic generation process as a *Dynamic Score*: a reactive program driven by an external time source orchestrating the upstream processes (generation queries) as well as the downstream processes (rendering), and managing high-level temporal specifications.



7

Summary and Contributions

7.1 Paradigm

"Intentions" → "Anticipations", combining planning and reactivity: In this part, we introduce the paradigm of guided improvisation modeled as dynamic calls to offline models relying on a temporal scenario such as that presented in Part I. Intrinsically offline processes are embedded into a reactive framework, out of the static paradigm yet not using pure last moment computation strategies.
It results in a hybrid architecture dynamically revising previously generated data ahead of the time of the performance in reaction to the alteration of the scenario or of other reactive inputs. Anticipations matching the scenario are therefore represented by sequences outputted by the generation process when it is called in time during live performance. This way, reactions are not seen as instant responses but have consequences over time.
This is achieved by chaining two agents: an *Improvisation Handler*, a reactive agent embedding the scenario / memory generation model, and a *Dynamic Score*, handling high-level planning.

Explore different musical directions using the same formal mechanisms: The articulation between the formal abstraction of scenario and reactivity enables to explore different musical directions with the same objects and mechanisms. In first approach, we differentiate two playing modes depending on the hierarchy between the musical dimension of the scenario and that of control. When scenario and control are performed on different features of the musical contents, the model combines long-term structure with local expressivity. When the scenario itself is dynamic, it deals with dynamic guidance and intentionality. The term "scenario" may be inappropriate in this second approach since it does not represent a fixed general plan for the whole improvisation session. Yet, whether the sequence guiding the generation is dynamic or static (*i.e.* whether the reaction impacts the guiding dimension or another one), both cases are formally managed using the same mechanisms.

7.2 Architectures

***Improvisation Handler*, guided improvisation as dynamic calls to an offline model:** When musicians play on a chord progression in a collective improvisation, they now this progression and they use it, so a computer has to do the same and its reactions have to take advantage of this prior knowledge. To achieve this, we propose the architecture of a reactive agent, the *Improvisation Handler*, reacting to dynamic controls by composing new mid-term anticipations ahead of performance time. This agent can embed a generation model relying on a formal temporal specification and handles the concurrent calls and accesses to shared data such as the musical memory and secondary generation parameters. The Improvisation Handler reacts to a modification of its reactive inputs by rewriting previously generated anticipations while maintaining coherence when overlaps occur.

***Dynamic Score*, interface between the environment of the performance and generation processes:** We model the decisions taken at the interface between the musical environment and dynamic guided generative processes (like the Improvisation Handler) as a reactive program called *Dynamic Score* scheduling dynamic calls to the models during the performance. The Dynamic Score involves a hierarchy of parallel processes listening and reacting to the environment and the elements generated by the models:

- it is involved simultaneously upstream and downstream to coordinate the generation queries and rendering of the associated outputs in due time,
- it synchronizes these processes with the musical inputs, in particular an external non-metronomic time source to adapt to the fluctuating tempo of the human co-improvisers.

7.3 Application and implementation

ImproteK: The improvisation system ImproteK uses the couple Improvisation Handler / Dynamic Score in association with a performance-oriented rendering module which will be described in Chapter 13 (Part III). Collaborations with musicians using this reactive architecture implemented in the system ImproteK are detailed in Part IV.

Genericity: In a context of performance, the Improvisation Handler offers the possibility to define reactive inputs that can be used to compose reactivity, provide controls to a human operator, or be linked with an external reactive listening module. Furthermore, the Improvisation Handler can be used autonomously in association with a composition-oriented renderer (see Chapter 12, Part III) within a compositional process.
The Dynamic Score is modular and extensible so that the proposed mechanisms can be coupled with other strategies such as score following or reactions to unordered complex events. Finally it enables to script a higher-level organization of the performance by writing improvisation *plans*.

Implementation: The Improvisation Handler has been implemented as a CommonLisp modular library within the OpenMusic environment (Bresson et al., 2011). Section B.2 in Appendix B gives examples of this reactive agent used in a reactive patching environment and communicating with a dynamic audio renderer.
The Dynamic Score has been implemented using the synchronization strategies of the score follower Antescofo (Cont, 2008a) and its associated programming language (Echeveste et al., 2013a,b). It is embedded in a user interface implemented in the graphical programming environment Max (Puckette, 1991).

8

Introduction

In the scope of music improvisation guided by a temporal specification, a *reaction* of the system to the external environment, such as control interfaces or live players input, cannot only be seen as a spontaneous instant response. The main interest of introducing a formal temporal structure is indeed to take advantage of this temporal structure to anticipate the music generation, that is to say to use the prior knowledge of what is expected for the future in order to better generate at the current time. In other words: when musicians play on a chord progression in a collective improvisation, they now this progression and they use it, so a computer has to be able to do the same. Whether a reaction is triggered by a user control, by hardcoded rules specific to a musical project, or by an analysis of the live inputs from a musician, it can be considered as a revision of the mid-term anticipations of the system in the light of new events or controls.

8.1 From Offline Guided Generation to Online Guided Improvisation

8.1.1 Combining Long-Term Planning and Reactivity

To deal with this temporality in the framework of a real-time interactive software, we consider guided improvisation as embedding an offline process into a reactive architecture. In this view, reacting amounts to composing a new structure in a specific timeframe ahead of the time of the performance, possibly rewriting previously generated material. Music generation models integrating a temporal specification are often static and offline in the sense that one run produces a whole timed and structured musical gesture satisfying the designed scenario which will then be unfolded through time during performance. It is in particular the case of the scenario / memory generation model presented in Part I that can be used to generate sequences satisfying given specifications in an offline compositional process. Yet, it was designed so that it is segmented into *anticipatory generation phases* to facilitate its use in a real-time context of performance (see 5.1.5).

8.1.2 Musical Typology of Reactive Inputs

The articulation between the formal abstraction of scenario and reactivity enables to explore different musical directions with the same objects and mechanisms, providing dynamic musical control over the improvisation being generated. In first approach, we differentiate two playing modes depending on the hierarchy between the musical dimension of the scenario and that of control. When scenario and control are performed on different features of the musical contents, the model combines long-term structure with local expressivity. When scenario and dynamic control act on the same musical feature, it deals with dynamic guidance and intentionality.

LONG-TERM STRUCTURE AND LOCAL EXPRESSIVITY We first consider the case where the specification of a scenario and the reaction concern different features, conferring them different musical roles (for example: defining the scenario as a harmonic progression and giving real-time controls on density, designing the scenario as an evolution in register and giving real-time controls on energy). In this case, a fixed scenario provides a global temporal structure on a given conduct dimension, and the reactive dimension enables to be sensitive to another musical parameter. The controlled dimension has a local impact, and deals with expressivity by acting at a secondary hierarchical level to filter dynamically the outputs resulting from the research on the fixed dimension (for example with instant constraints on timbre, density, register, syncopation etc.). This playing mode may be more relevant for idiomatic or composed improvisation with any arbitrary vocabulary, in the sense that a predefined and fixed scenario carries the notions of high-level temporal structure and formal conformity to a given specification anterior to the performance, as it is the case for example with a symbolic harmonic progression.

GUIDANCE AND INTENTIONALITY When specification and reaction act on the same musical dimension, the scenario becomes dynamic. A reaction does not consist in dynamic filtering of a set of solutions as in the previous playing mode, but in the modification of the scenario itself. In this case, the current state of a dynamic scenario at each time of the performance represents the short-term “intentionality" attributed to the system, which becomes a reactive tool to guide the machine improvisation by defining instant queries with varying time windows. The term “scenario" may be inappropriate in this second approach since it does not represent a fixed general plan for the whole improvisation session. Yet, whether the sequence guiding the generation is dynamic or static (*i.e.* whether the reaction im-

pacts the guiding dimension or another one), both cases are formally managed using the same mechanisms.

8.2 ImproteK: An Interactive System

This section introduces the general architecture of the ImproteK system implementing the models and architectures proposed in thesis (see Part IV describing the musical collaborations with expert improvisers who used the system). This architecture chains the different reactive and dynamic agents presented in the following chapters of this Part II.

During a performance, the system plays improvisations matching a scenario, and the music played by human co-improvisers is captured and added to its musical memory which can also contain offline material. The incoming stream is segmented using a chosen *external beat source* and is annotated by labels. If the scenario is a common referent for the musicians and the machine, the labels directly come from the scenario (Figure 8.1, left)[1]. If the scenario is seen as a score for the machine only, the labels come from a chosen analysis (Figure 8.1, right).

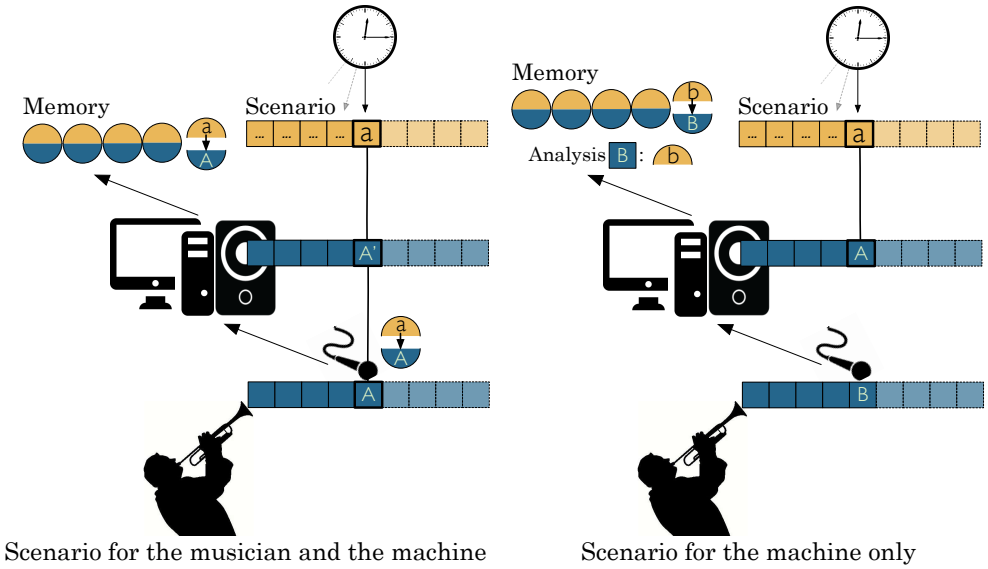


Figure 8.1: Possible interactions with the scenario during a performance.

1 This is the configuration we used in most of our musical collaborations.

The system reacts to the online control of *reactive inputs*:

REACTIVE INPUTS

- The scenario itself;
- The chosen/designed *secondary generation parameters* (introduced in Section 6.2).

The generic reactive mechanisms are described in Chapter 9 without focusing on where the controls come from: depending on the musical project, they can be given to an operator-musician controlling the system, launched by composed reactivity rules, defined in a higher-level improvisation plan (see 10.4), or plugged to an external listening module.

As introduced above, ImproteK uses an *external time source* which is used as a clock for the improvisation. This input is generic and can be plugged to a fixed metronome, an irregular time track, or a non-metronomic beat coming from a beat tracking system listening to the musician (the system includes a beat tracking module developed by Bonnasse-Gahot (2010))[2].

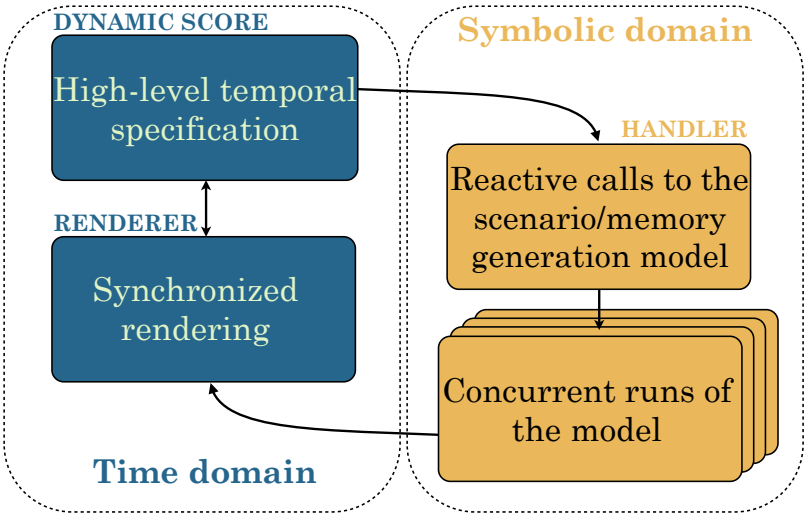


Figure 8.2: General architecture of the improvisation system.

Figure 8.2 schematizes the general architecture of the system. Basically, time domain is where listening, planning and rendering occur, while symbolic domain concerns the concurrent runs of the music generation model and the way they are dynamically handled. This architecture chains three main modules:

2 Next part (Part III) presents a dynamic rendering module able to deal with dynamically generated musical sequences and which synchronises the live audio reinjections with a non-metronomic beat.

GENERAL ARCHITECTURE

- an *Improvisation Handler* (Chapter 9), a *reactive agent* embedding the memory and the mechanisms handling music generation, which manages reaction and concurrency of the overlapping queries sent to the scenario/memory generation model to achieve dynamic and guided generation of evolving musical sequences;
- a *Dynamic Score* (Chapter 10), a *reactive program* driven by an external beat source orchestrating and synchronizing the upstream processes (generation queries) and downstream processes (communication with a renderer), and managing higher level temporal specifications;
- an *Improvisation Renderer / Sequencer* (Chapter 12, next Part III), recording the live inputs and synchronizing the rendering of the generated dynamic sequences with the environment (external beat source and controls).

These architectures have been designed so that they do not depend on each other in their conception. This way, they autonomously address different issues described in the following chapters.

9

Combining Planning and Reactivity: the Improvisation Handler

Human-computer improvisation systems generate music on the fly from a model and external inputs (typically the output of an "analog" musician's live improvisation). Introducing authoring and control in this process means combining the ability to react to external events with that of maintaining conformity to fixed or dynamic specifications. When improvisation is guided by a temporal specification such as the scenario introduced in this thesis (Part I), machine improvisation should take advantage of this prior knowledge to generate mid-term anticipations ahead of the performance time, and react to external events by refining or rewriting these anticipations over time. In addition, if the initial specification itself gets modified during the performance, the system may have to ensure continuity with the previously generated material at critical times.

To achieve this, we model guided improvisation as embedding an offline generation process relying on a formal temporal structure into a reactive agent handling concurrent calls to shared data (memory and generation parameters): the *Improvisation Handler*.

9.1 Guided Music Improvisation and Reactivity

With the Improvisation Handler, our objective is to devise an architecture at an intermediate level between the reactive and offline approaches for guided improvisation, combining dynamic controls and anticipations relative to a predefined plan. The architecture we propose for this agent has the following inputs and outputs:

- Inputs:
  - an offline music generation model with a formal temporal specification such as a "scenario",
  - an access to this specification and to secondary generation parameters,
  - time markers received from the environment (to be informed of the current position in the performance).
- Outputs: dynamic generation of an evolving musical sequence

- using the prior knowledge of the specification (fixed or dynamic),
- automatically rewriting previously generated material when modifying an input,
- handling internally the concurrent generation queries,
- maintaining continuity with previously generated material at tiling time.

The Improvisation Handler generates musical sequences on request. It translates dynamic controls into music generation processes and reacts to external events by generating an updated future matching the temporal specification of the music generation process it embeds, *i.e.* by composing a new sequence in a specific timeframe ahead of the time of the performance. As it is described in the following sections, this agent is autonomous. Within the ImproteK system, it is used in interaction with a *Dynamic Score* constituting an interface with the environment (see Chapter 10).

9.2 Improvisation Handler: Reactive Agent Embedding an Offline Model

9.2.1 Intentions and Anticipations

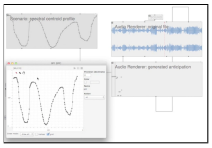
We describe here an Improvisation Handler embedding the scenario / memory generation model introduced in Part I. Thanks to the scenario, music is produced ahead of the performance time, buffered to be played at the right time or rewritten. For purposes of brevity:

- *intentions* is used to refer to the planned formal progression: the current state of the scenario and other generation parameters ahead of the performance time, (*e.g.* “from next beat, improvise over what remains of the current scenario: | EbMaj7 | AbMaj7 | D7 | G7 | Cm7 | C m7 || in a low register”).
- *anticipations* is used to refer to pending musical events: the current state of the already generated musical material ahead of the performance time, (*e.g.* a possible realization of : | EbMaj7 | AbMaj7 | in a low register).

These evolving anticipations of the machine result from successive or concurrent calls to the generation model. Introducing a reaction at a time when a musical sequence has already been produced amounts then to rewrite buffered anticipations.

The rewritings are triggered by modifications of the intentions regarding the scenario itself or other generation parameters (these two different cases correspond to the different musical directions discussed in 8.1.2). **Video A.2.3** shows the anticipations being rewritten in two simulations with different configurations regarding the scenario, the memory, and the chosen reactive inputs:

Video A.2.3



Hyperlink video *(or vimeo.com/jeromenika/improteksmc15)*
Description: *Appendix A.2.3*

- *Example 1:*
  - *chosen reactive inputs*: register and density,
  - scenario: harmonic progression (*Autumn leaves* with harmonic substitutions),
  - memory: heterogeneous MIDI corpus (captured solos on various blues or jazz standards).
- *Example 2:*
  - *chosen reactive inputs*: scenario and memory region,
  - scenario: spectral centroid profile,
  - memory: audio file (percussion solo by Trilok Gurtu).

9.2.2 Concurrent and Overlapping Runs of the Generation Model

To give control over these mechanisms, that is dynamically controlling improvisation generation, the *Improvisation Handler* agent (H) embeds the scenario / memory generation model and articulates it with:

- a dynamic scenario (S);
- a set of reactive generation parameters;
- current position in the improvisation t^p (*performance time*);
- the index of the last generated position t^g (*generation time*);
- a function f responsible for the output of generated fragments of improvisation (*output method*).

IMPROVISATION HANDLER This Improvisation Handler agent H links the real time of performance and the time of the generation model embedded in an *improviser* structure (see Figure 9.1). The improviser structure associates the generation model and the memory with a set of secondary generation parameters and an execution trace described below. The secondary generation parameters (Section 6.2) contains all the expressive parameters driving the generation process which are independent from the scenario: parametrization of the generation model (e.g. minimal / maximal length or region of the sub-sequences retrieved from the memory, measure of

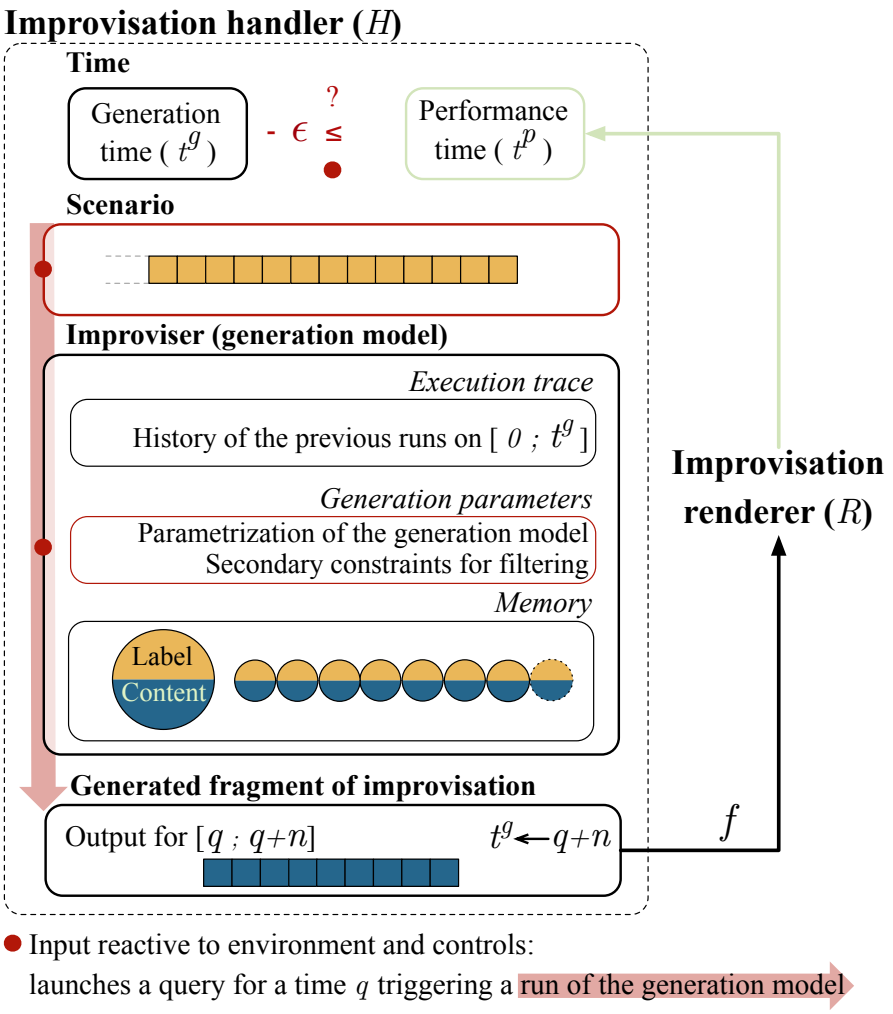


Figure 9.1: *Improvisation Handler* agent.

the linearity/non-linearity of the paths in the memory etc.) and content-based constraints to filter the set of possible results returned by the scenario matching step (e.g. user-defined thresholds, intervals, rules etc.).

EXECUTION TRACE The *execution trace* records history of paths in the memory and states of these generation parameters for the last runs of the generation model so that coherence between successive generations phases associated to overlapping queries is maintained. This way, the process can go back to an anterior state to insure continuity at the first position where the generation phases overlap.

FROM CONTROLS TO REACTION The interactions of the Improvisation Handler with the environment consist in translating dynamic controls on reactive inputs into reactive queries and redirecting the resulting generated fragments. We call *reactive inputs* the entities whose modifications lead to a reaction: the scenario and the set of

secondary generation parameters (see Section 8.2). In this framework, we call *reaction* an alteration of the *intentions* leading to a call to the generation model to produce a fragment of improvisation starting at a given position in the scenario.

QUERY We note Q a *query* to generate an improvisation fragment that will start at time q in the scenario[1]. Q triggers a run of the improviser, that is to say a generation phase of the generation model, to output a sub-sequence (or a concatenation of sub-sequences) of the memory which:

- matches the current state of the scenario from date q (i.e. a suffix S_q of the scenario, see Chapter 5),
- satisfies the current state of the set of generation parameters.

Formally, a *query* is a class whose main slots are:

- the date q it concerns (which is different from the date at which the query is received, *e.g.* when playing the event 4 of the improvisation, the system can receive and process a query concerning the index $q = 8$ in the scenario),
- the parameters it modifies and their new values.

Running a query means launching a thread executing a phase of the music generation model (5.1.5).

COMMUNICATION WITH A DYNAMIC RENDERER: SEND IMPROVISATION AND RECEIVE TIME The current time of performance is received form a dynamic improvisation renderer, ant the output method of the Improvisation Handler (f) is a settable attribute, so that generated improvisations can be redirected to any rendering framework. For instance, the Improvisation Handler can interface with Max (Puckette, 1991) via the Dynamic Score described in Chapter 10. In this case, f determines how resulting improvisation segments are sent back to the Dynamic Score where they are buffered or played in synchrony with the non-metronomic tempo of the improvisation session. Chapter 13 and Chapter 15 give two examples of performance-oriented and composition-oriented renderers respectively, and detail how the Improvisation Handler can be coupled with these modules depending on the musical project.

9.2.3 Triggering Queries for Rewriting Anticipations

We describe here the way control events are translated into generation queries triggered by the Improvisation Handler. This mecha-

1 q is the time at which this fragment will be played, it is independent from t^p and from the date at which the query is launched by the Improvisation Handler.

nism can be *time-triggered* or *event-triggered*, i.e. resulting respectively from depletion of previously generated material or from parameters modifications.

TIME-TRIGGERED GENERATION Rendering may lead to the exhaustion of generated improvisation. New generation queries have therefore to be launched to prevent the time of the generation t^g from being reached by the time of the performance t^p. To do so, we define ϵ as the maximum allowed margin between t^p and t^g. Consequently, a new query for time $q = t^g + 1$ is automatically triggered when the condition $t^g - t^p \leq \epsilon$ becomes true. This point will be developed later on in Section 10.2.2 describing the higher level temporal architecture (Dynamic Score).

EVENT-TRIGGERED GENERATION As introduced previously, the musical meanings of reactions to dynamic controls impacting the *scenario* itself or *an other musical dimension* are quite different (8.1.2). Yet, both cases of reaction can be formally managed using the same mechanisms of event-triggered generation. The reactive inputs (*i.e.* the scenario itself and the secondary generation parameters, see Section 8.2) are customizable so that any relevant slot of the Improvisation Handler can easily be turned into a reactive one. Modifying the scenario or one of these reactive slots launches a generation query for the time q affected by this modification. The triggering of a query by a reaction can indeed take effect at a specified time q independent of performance time t^p.

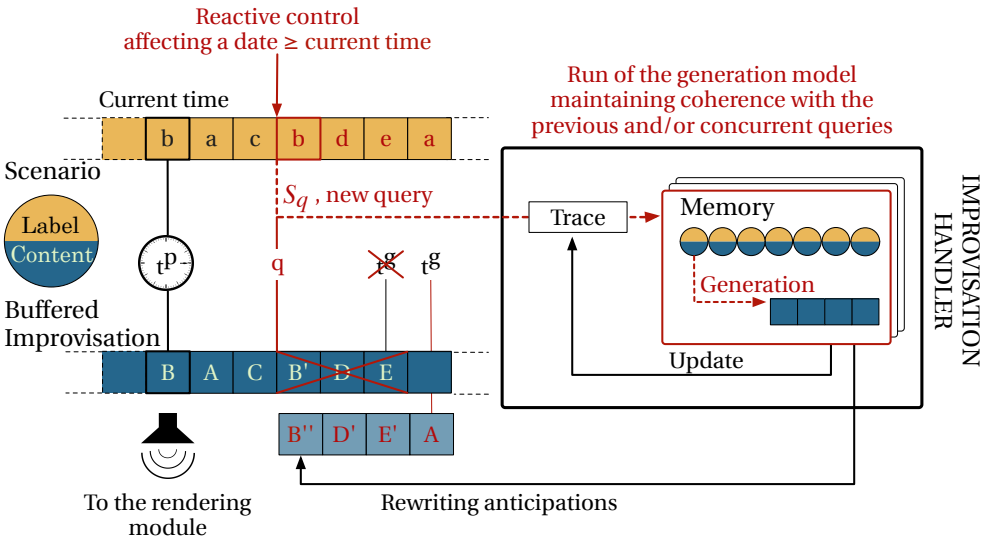


Figure 9.2: Reactive calls to the generation model.

As illustrated in Figure 9.2, the new improvisation fragment resulting from the generation is sent to the buffered improvisation while the improvisation is being played.The new fragments overwrites the

previously generated material on the overlapping time interval. The execution trace introduced in 9.2.2 enables to set mechanisms providing continuity at the tiling time q.

9.2.4 Rewriting Intentions: Concurrent Queries

Anticipation may be generated without ever being played because it may be rewritten before being reached by the time of the performance. Similarly, an intention may be defined but never materialized into anticipation if it is changed or enriched by a new event before being reached by a run of generation.

Indeed, if reactions are frequent or defined with delays, it would be irrelevant to translate them into as many independent queries leading to numerous overlapping generation phases. We then define an intermediate level to introduce evolving queries, using the same principle for dynamically rewriting intentions as that defined for anticipations.

This aspect is dealt with by handling concurrency of the queries and of accesses to shared data (memory, generation parameters, and execution trace) and working at the query level when the Improvisation Handler receives new queries while previous ones are still being processed by the generation module. Algorithm 2 describes how concurrency is handled, with (see Figure 9.3):

- Run(Q): start generation associated to Q. This function outputs generated data when it finishes,
- Kill(Q): stop run associated to Q and discard generated improvisation,
- Merge (Q_1, Q_2): create a new query [2] in which the list of impacted generation parameters and their associated new values correspond to the concatenation of that of Q_1 and Q_2,
- Relay(Q_1,Q_2,q): output the result of Q_1 for $[q_1; q[$, kill Q_1 and run Q_2 from q. The execution trace is read to maintain coherence at relay time q,
- WaitForRelay(Q_1,Q_2,q): Q_2 waits until Q_1 generates improvisation[3] at time q. Then Relay(Q_1,Q_2,q).

2 If two queries Q_1 and Q_2 concern the same date q_1=q_2 and do not impact the same parameters (*e.g.* Q_1*: "at date* q_1 *= index 10 of the scenario, replace the chord CMaj7 in the scenario by Am7"*, and Q_2*: "at date* q_2 *= index 10 of the scenario, only chose events containing less than 3 notes"*), a new query Q_3 concerning the index q_3=q_1=q_2=index 10 is created (Q_3*: "at date* q_3=q_1=q_2*=index 10, replace the chord CMaj7 in the scenario by Am7 and only chose events containing less than 3 notes"*).

3 More precisely, new generation phases are launched if needed until q is reached.

Note: "Run(Q)" in this Algorithm 2 runs a generation phase of Algorithm 1 (Section 5.1) with the suffix S_q of the scenario S as input.

ALGORITHM 2 . Concurrent runs and new incoming queries

```
1  Q_i, query for improvisation time q_i
2  RQ, set of currently running or waiting queries
3  CurPos(Q), current generation index of Run(Q)
4  Whenever RQ = {Q} and Q not running do
5  |  Run(Q)
6  Whenever new Q received do
7  |  for Q_i ∈ RQ do
8  |  |  if q = q_i then
9  |  |  |  if Q and Q_i from same inputs then
10 |  |  |  |  Kill(Q_i)
11 |  |  |  else
12 |  |  |  |  Merge Q and Q_i
13 |  |  |  end
14 |  |  else if q > q_i then
15 |  |  |  if q < CurPos(Q_i) then
16 |  |  |  |  Relay(Q_i,Q,q)
17 |  |  |  else
18 |  |  |  |  WaitForRelay(Q_i, Q, q)
19 |  |  |  end
20 |  |  else if q < q_i then
21 |  |  |  WaitForRelay(Q, Q_i, q_i)
22 |  |  end
23 |  end
```

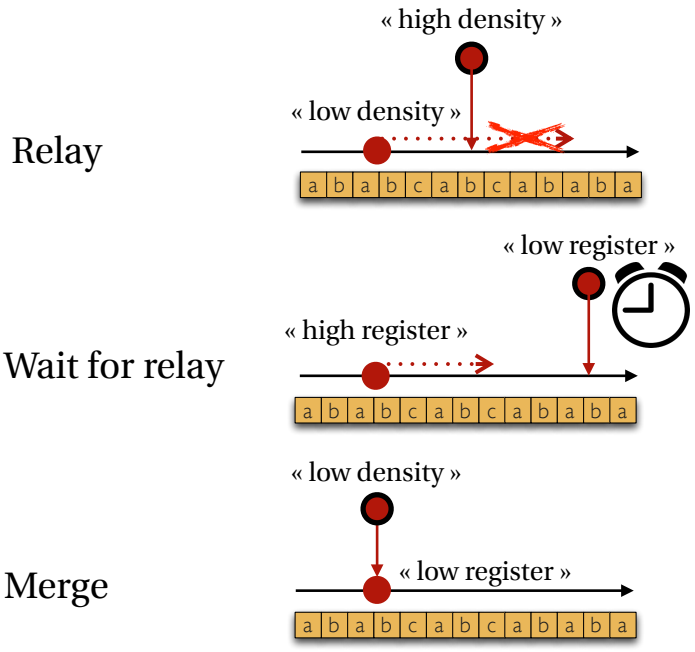


Figure 9.3: Improvisation Handler: concurrent queries.

This way, if closely spaced in time queries lead to concurrent processing, relaying their runs of the generation model at the right time using the execution trace enables to merge them into a dynamic query.

10

Planning Improvisation: the Dynamic Score

In this chapter, we propose a time-aware extensible architecture allowing the temporal coordination of the different process involved in machine improvisation (listening, learning, generating, waiting, playing) and of different improvisation strategies. It is designed to integrate generative strategies into a high-level structure that we call *Dynamic Score*, a reactive program that manages the high-level temporal specifications and synchronizes the generation and rendering processes with the inputs from the environment. The resulting framework handles the musical events, the triggering of generative processes at different time scales and the declarative specification of improvisation plans driven by the occurrence of complex events.

The architecture has been implemented in ImproteK using the Antescofo system and OSC messages to interact with the other modules. Chapter 9 presented how dynamic queries to a scenario / memory generation model could be handled (Improvisation Handler), this chapter describes how these queries are sent by the Dynamic Score. The Dynamic Score also makes the connection between the Improvisation Handler and an Improvisation Renderer (see Part III).

10.1 An Interface Between the Environment and Dynamic Music Generation Processes

With the Dynamic Score, we model the decisions taken at the interface between the environment and dynamic guided generative processes as a unified and extensible reactive program:

- involved simultaneously upstream and downstream to coordinate the calls to a guided dynamic generation process and the rendering of its outputs in due time,
- synchronizing these processes with the musical inputs and control inputs, in particular an external non-metronomic time source to adapt to the fluctuating tempo of the human co-improvisers.

In this chapter, the architecture model we propose is described through its use in the ImproteK system, that is to say using the Im-

provisation Handler agent introduced in Chapter 9 embedding the scenario / memory generation model (Part I) as an example of dynamic generation process with a formal temporal specification.

On the one hand, the Dynamic Score implements parallel processes listening to the musical inputs to segment and label them for learning; reacting to controls; and launching the queries to the improvisation handler. On the other hand, it receives the anticipated improvisations from the scenario / memory generation model embedded in the Improvisation Handler as portions of code which are executed in due time. The Dynamic Score acts as a dynamic sequencer: the anticipated improvisation fragments are received and buffered to be unfolded in the real time of the performance by sending control messages to a renderer in due time.

10.2 Scheduling the Reactions to the Environment

The Dynamic Score is generic, in the sense that it does not depend on the alphabet of the scenario or the nature of the contents in the musical memory. It listens to an external time source (8.2) providing the current date T (index), and so the associated label $S[T]$ in the scenario. Each update of the current position $S[T]$ in the scenario is used in different concurrent processes involved both upstream and downstream of generation. The four main tasks of this complex reactive program, illustrated in Figure 10.1, are the following:

DYNAMIC SCORE

1. Segment, label, and learn the musical material from the musicians playing with the system;
2. Send declarative controls (programmed automatic controls or controls from a user, see Chapter 14) that will be translated into generation queries by the Improvisation Handler (modifications of the *intentions*, Chapter 9),
3. Receive the *anticipations* from the Improvisation Handler and buffer them, waiting to refine them or play them,
4. Send control messages to a renderer to play the anticipations received from the dynamic generation process in due time and in synchrony with the non-metronomic time source.

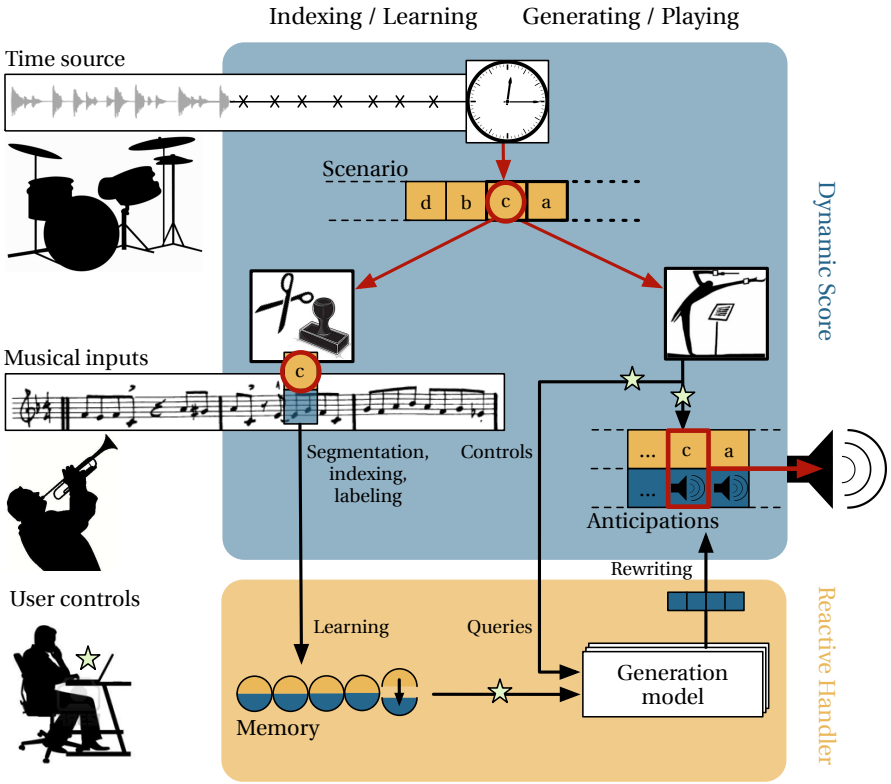


Figure 10.1: Orchestrating upstream and downstream processes.

10.2.1 The Processes in the Dynamic Score

This architecture involves a hierarchy of parallel processes listening and reacting to the environment, the elements produced by the model, and the instructions given by the operator or a higher scale improvisation plan.

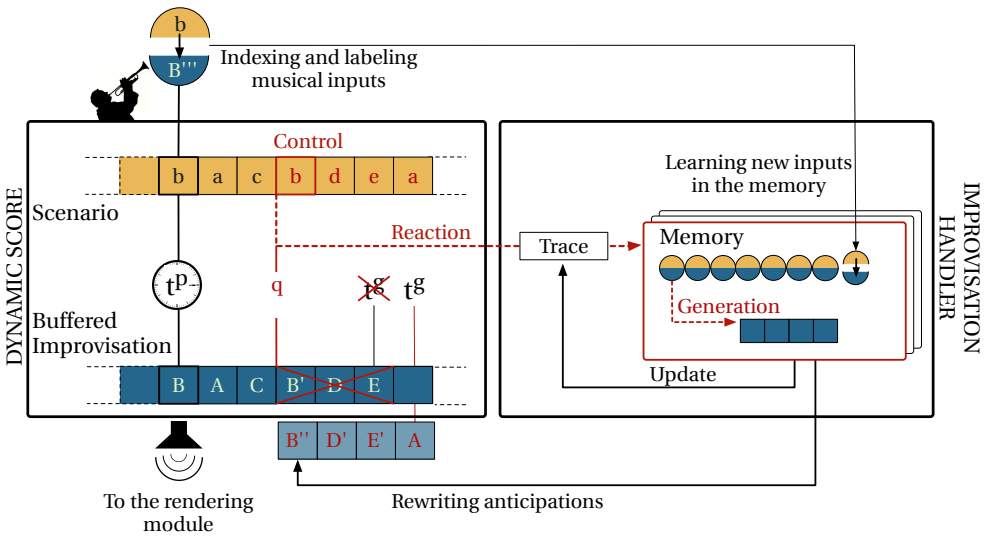


Figure 10.2: Launching queries and buffering anticipations.

Note: "Send query: generate(q, S_q)" in this Algorithm 3 creates a new Query Q processed then by Algorithm 2 (Section 9.2).

ALGORITHM 3 . Dynamic score (simplified)

Inputs :
T, current date / position in the scenario
S, original scenario, (S_T suffix of S beginning at index T)
RecvdEvent = (Idx,Content), received event generated by the model
Initial state :
Buffer (storing the generated musical anticipations) = $\emptyset$
E (index of the first empty position in the buffer) = 0
CurrentTimePlayed = false

```
1  Whenever T updated do
2      Learn inputs from [T − 1, T[ labeled by S[T − 1] in M
3      CurrentTimePlayed ← false
4      if Buffer[T] then
5          Play(Buffer[T])
6          CurrentTimePlayed ← true
7      end
8  || Whenever E − T < ε, minimal allowed delay do
9      q ← max(T, E)
10     Send query: generate(q, S_q)
11 || Whenever modif. of parameters or S affecting date q ≥ T do
12     Send query: generate(q, S_q)
13 || Whenever RecvdEvent = (Idx,Content) received do
14     if (Idx = T) & (¬ CurrentTimePlayed) then
15         Delay ← Date(update T) - Date(RecvdEvent)
16         Play(Content, Delay)
17         CurrentTimePlayed ← true
18     end
19     Buffer[Idx] ← Content
20     E ← max(Idx+1, E)
```

The different categories of processes correspond to the different parts in the simplified Dynamic Score are given in Algorithm 3 and illustrated in Figure 10.2[1]:

1. Listen to the updates of current date (variable T) to orchestrate the labeling and learning of the musical material, and the playing of the anticipated events stored in the buffer (learn inputs, play, lines 1-6 in Algorithm 3).
2. When it is required, send a query to the Improvisation Handler so that it generates a segment of improvisation starting at date $q \geq T$ associated to suffix a S_q of the scenario (send query, lines 8-12 in Algorithm 3).

1 We only describe here the planning instructions, but the Dynamic Score also implements user controls, the OSC (Wright et al., 1997) communication with the generation module...

3. Listen to the new elements generated by the model is received, to buffer them or immediately play managing potential delays (play, *buffer*, lines 13-20 in Algorithm 3).

The improvisations generated by the model are then played in synchrony with the musical environment, following the fluctuations of the tempo of the listened external time source (see Chapter 13). The synchronization strategies to manage the delays (lines 13-16 Algorithm 3) associated to anticipation are used to maintain musical coherence despite real-time modifications of generation parameters.

10.2.2 Communication with a Dynamic Guided Generation Model

We detail now the interactions between the Dynamic Score and the dynamic generation process generating and sending musical anticipations (the Improvisation Handler, Chapter 9).

SEGMENTING, LABELLING, AND LEARNING The live musical inputs are segmented, labeled, and indexed (see Chapter 13), and sent to the Improvisation Handler to be learnt in the memory (factor oracle automaton, see 5.3) in real time.

EVENT-TRIGGERED GENERATION User controls on the dynamic music generation are given through the interface of the Dynamic Score (see Chapter 14). These controls are sent to the dynamic generation process and translated into queries handling the coherence with previously generated material and with the scenario (Section 9.2). This corresponds to the event-triggered generation mechanism described previously in 9.2.3.

TIME-TRIGGERED GENERATION The Dynamic Score has to inform the dynamic generation processes of the current position in the performance to enable time-triggered mechanisms. Rendering may lead to the exhaustion of generated anticipations, new generation queries have therefore to be launched to prevent the time of the generation t^g from being reached by the time of the performance t^p (Figure 10.2). To do so, we define ϵ as the minimal allowed delay between t^p and t^g. Consequently, a new query for time $q = t^g + 1$ is automatically triggered when the condition $t^g - t^p \leq \epsilon$ becomes true.

Depletion of the previously generated improvisation generation occurs when generation over the whole scenario is not performed in a single run. Figure 10.3 illustrates two successive generation phases associated to queries Q_1 and Q_2 for time q_1 and q_2 respectively.

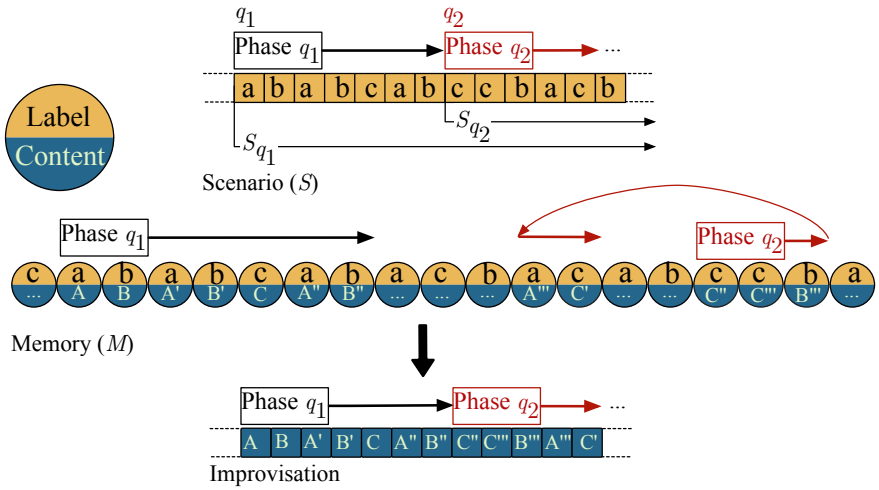


Figure 10.3: Phases of the guided generation process.

> A generation *phase* matches a scenario sub-sequence starting at a queried position q to a sub-sequence of the memory, i.e. the generation model searches for a prefix of the suffix S_q of the scenario S in the memory (phase q_1 in figure 10.3) or an equivalent non-linear path (phase q_2 in figure 10.3). The generation process waits then for the next query. Defining such phases enables to have mid-term anticipations generated ahead of the performance time while avoiding generating over the whole scenario if an event modifies the intentions (see the related algorithmic considerations in 5.1.5, Part I).

A generated fragment of improvisation resulting from a query Q for time q contains n slices where:

$$1 \leq n \leq length(S) - q, n \in \mathbb{N}$$

The search algorithm of the generation model runs a generation phase[2] to output a sub-sequence of the memory in time $\Theta(m)$ and does not exceed $2 * m - 1$ comparisons, where m is the length of the memory. In first approximation, the minimal delay ϵ is empirically initialized with a value depending on the initial length m. Then, in order to take into account the linear time complexity, ϵ increases proportionally to the evolution of m if the memory grows as the performance goes. Future works on this point will consist in informing the scheduling engine with the similarities between the scenario and the memory to optimize anticipation. Indeed, the number of calls to the model depends on the successive lengths n of the similar patterns between the scenario and the memory. For example, the shorter the

2 1) Index the prefixes of the suffix S_q of the scenario in the memory, 2) select one of these prefixes depending on the generation parameters, 3) output this prefix or an equivalent non-linear path in the memory.

common factors, the higher the number of queries necessary to cover the whole scenario.

10.2.3 Polyphonic Improvisations: *Voice* Processes

The other categories of processes scheduled by the Dynamic Score (listed in 10.2.1) will be detailed later on when defining an improvisation *voice*. The architecture described in this chapter is indeed simplified since it is monophonic. In reality, we want to be able to play polyphonic improvisations involving different voices playing different improvisations, using different memories, and following potentially different tempi. This *voice* process will be presented in Chapter 13 (Part III) which focuses on rendering of musical anticipations in synchrony with an external non-metronomic time source.

10.3 Writing a Dynamic Score and Improvisation Plans

The Dynamic Score has been implemented using the Antescofo (Cont, 2011a) system and the associated programming language (Echeveste, 2015), inspired by the synchronous programming language Esterel (Berry and Gonthier, 1992) dedicated to the development of complex reactive systems.

Antescofo is a real-time system for interactive music authoring and performing. It focuses on high-level musical interaction between live musicians and a computer, where the temporal development of musical processes depends on active listening and complex synchronization strategies (Cont, 2011a). Antescofo was chosen to support the high-level organization of the musician-computer interaction because it combines score following capacity with an expressive timed and reactive scripting language. This language provides the possibility to specify reactions to unordered complex events and musical synchronization expressed in relative time, *i.e.* relatively to the actual performance (see Chapter 13). In Max/MSP or PureData (Puckette, 1991) which are dataflow graphical languages and where control and signal processing are statically determined, it is easy to construct static behaviors, but much harder to organize and control changing behaviors according to a complex scenario.

Antescofo, compiled as a Max or PureData object, is used in these cases to interact with the external environment. Other dynamic languages encounter some success in the interactive music community such as SuperCollider (McCartney, 1996) or Chuck (Wang, 2009). These are textual languages facilitating the programming of audio and algorithmic processes. Their real-time properties make them ideal tools for “Live Coding” practices, often improvised, where the

time of composition (in the program) coincides with that of performance. However, the semantics of these languages does not allow the direct specification of the behavior of the external environment. Furthermore, their models of time is not directly linked with that of the performer. Compared to traditional sequencers such as LogicPro, ProTools or CuBase, Antescofo is dedicated to more dynamic and interactive situations. Ableton Live with Max4Live adds more possibilities of interaction compared to the sequencers cited above, but without providing a flexibility allowing to synchronize the electronic processes to the elastic time of the musician.

The Antescofo DSL enables the creation of an augmented score whose language integrates both programmed actions and expected musical events played by the human performer, allowing a unique and flexible temporal organization. The augmented score includes both the instrumental part to recognize and the electronic parts and the instructions for their coordination in real time during the performance. The syntax for writing the instrumental part allows the description (pitches and durations) of events such as notes, chords, trills, glissandi and improvisation boxes. Actions are divided into atomic actions, performing elementary computations, and compound actions. The atomic actions can be: messages sent to the external environment (for instance to drive a synthesis module), a variable assignment, or another specific internal command.

10.4 From Scheduling to Logical Planning

When improvising, different temporal structures exist concurrently at different time scale, for example:

- at short-term, the synchronization of the notes in a generated sequence with the current tempo;
- generation of musical sequences satisfying global constraints in mid-term;
- and at a higher level, switching from an improvisation part to another, defined by different sets of prior knowledge, memory, mechanisms and rules (such as switching from lead to follow, from free to idiomatic, etc.).

The coordination of these temporal sequences and reactions constitute complex high-level improvisation plans (such as the improvisation plans presented in Section 18.3 Part IV, or in Appendix C.2.2.1 and C.2.2.2) that can be defined in the extensible Dynamic Score.

Beyond scheduling aspects detailed in the previous sections, this section presents Antescofo features allowing the high-level temporal or logical specification of different kind of processes that can be

involved in an improvised interactive music performance. Recent developments of the language integrate handling of dynamic duration, complex events specification and dynamic processes. This generalizes the notion of score following beyond triggering of an action or recognition of causal events. The score is no longer subject to linear rigidity of classical scores. It can be seen as an interactive system where events, actions, durations, tempi and all the temporal structures can change dynamically.

10.4.1 Some Features of the Antescofo Language

We sketch three features of the language making possible to react to a logical event, to define complex events involving duration, and to create new parallel threads of computations.

WHENEVER The `whenever` statement launches actions conditionally on the occurrence of a signal. Actions triggered by a `whenever` statement are not statically associated to an event of the performer but to the dynamic satisfaction of a predicate. They can be triggered by the result of a complex calculation, launched by external events, or by any combination of both. Each time variable referred by the predicate (any arbitrary expression) are updated, the expression is re-evaluated. The `whenever` statement is a way to reduce and simplify the specification of the score, particularly when actions have to be executed each time an event is detected. It escapes the sequential nature of traditional scores.

PATTERN The `whenever` structure is relevant when the user wants to define a reaction conditionally to the occurrence of an event. A logical event is specified thanks to a combination of variables. Complex events corresponding to a combination of atomic events with particular temporal constraints are however tedious to specify. Antescofo patterns make the definition of this kind of events concise and easy. A pattern is made of atomic events (*event*) and of events with durational constraints (*state*).

PROCESS Processes are groups of actions dynamically instantiated. Unlike the other actions, the runtime structure associated to a process is not created during the loading of the score but at the time of the call, in accordance with its definition. Then, all the expressions involved in the process (durations, command names, attributes, etc.) may depend on parameters of the performance. Processes are first-class values: for example, a process can be passed as an argument to a function or to another process. It can be recursively defined and various instances of the same process can be executed in parallel. Processes are quite adapted to the context of improvised music, and

can be used for example as a library of parametrized musical phrases that are instantiated following the musical context.

10.4.2 Writing Improvisation Plans

The set of tools presented in this section enables to write *improvisation plans* defining different kinds of interactions. The schematic example in Figure 10.4 shows a generic example of such a plan. In this context, the score of the musician is not completely defined and the inputs of the reactive module are not necessarily extracted from Antescofo listening machine but can also be provided by external modules.

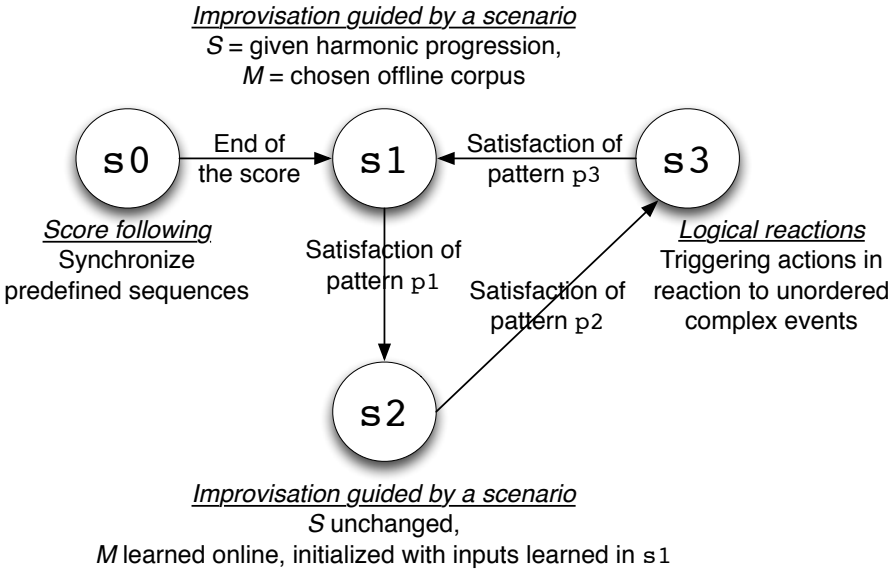


Figure 10.4: Schematic example of an improvisation plan.

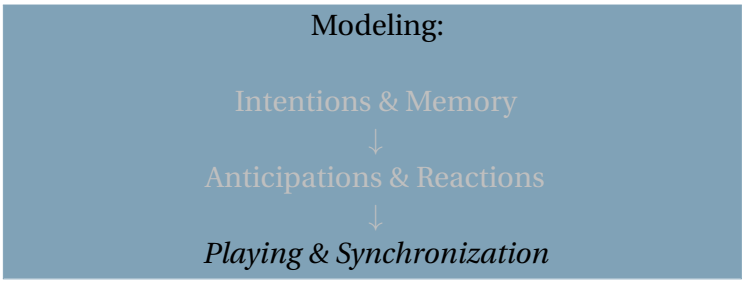
Each state corresponds to an interaction mode between the performer and the system. Satisfaction of temporal patterns `p1`, `p2` or `p3` allows to switch between the different states `s0`, `s1`, `s2` and `s3`. These patterns can for example be defined as temporal evolutions of some audio descriptors. `s0` is associated to a classical phase of interaction of a score following system with electronic actions adapting to a sequence of predefined events. Reaching the end of the sequence leads to the beginning of the next part (`s1`) where the musician improvises with the generation model guided by a scenario chosen as a given harmonic progression, and a musical memory initialized with a chosen corpus. The part corresponding to `s2` continues with the same scenario using the memory learned from the musician's performance during `s1`. Finally, `s3` is a combination of predefined interactive mechanisms associating electronic reactions to unordered events.

As illustrated by this example, improvisation strategies with different degrees of indeterminism can be employed within the same plan. This way, generative processes such as described in Part I can be coupled with score following techniques or purely reactive processes in a unified improvisation plan. Besides, the high-level temporal organization of the performance and the modifications of memory of scenario can themselves be scripted.

Part III

“PLAYING" WITH THE (SOUND OF THE) MUSICIANS

Part III focuses on adaptive rendering and synchronization of evolving musical sequences coming from dynamic generation processes using live external inputs, and presents some associated expressive musical controls.
Chapter 11 summarizes the contributions of Part III.
Chapter 12 introduces two architectures coping with dynamic musical sequences which are revised during the rendering.
Chapter 13 describes a performance-oriented architecture which offers adaptive rendering of dynamic multimedia sequences generated from live inputs. This autonomous architecture is designed to record and segment a live stream into beat-events that can immediately be played in synchrony with a non-metronomic pulse, according to a user-defined dynamic time mapping.
Chapter 14 focuses on how to use the models implemented in the ImproteK system as a software instrument offering declarative controls impacting on generation, and temporal controls impacting on rendering.
Chapter 15 presents a rendering architecture dedicated to composition of guided musical processes using an offline memory.



11

Summary and Contributions

"Anticipations" → "Playing": In this part, we focus on rendering evolving musical sequences coming from dynamic generation processes, and present some associated expressive musical controls. We propose two architectures coping with dynamic musical sequences which are revised during the rendering. The first architecture, dedicated to performance and used in the system ImproteK, proceeds to the elastic temporal mapping between a symbolic improvisation and the real time of performance and offers downstream musical controls. The second architecture is an autonomous renderer conceived to be integrated in a framework of composition of musical processes using an offline memory and driven by the internal time of the musical material.

11.1 Beat, Synchronization, and Dynamic Time Mappings

Performance-oriented rendering for guided improvisation: We present a rendering architecture dedicated to performance that offers adaptive rendering of dynamic multimedia sequences generated from live inputs. It is conceived to record and segment live streams into beat-events that can immediately be played in synchrony with a non-metronomic pulse, according to a user-defined dynamic time mapping. This architecture handles perceptive continuity by ensuring synchronization with the environment and realizing crossfades when the events to render are not contiguous in the memory.

Synchronization with an external beat source: The proposed architecture interleaves event-triggered and adaptive time-triggered mechanisms parametrized by the tempo of external beat sources. These latter are listened to trigger tempo estimations modifying the periods of periodic synchronization processes so that the system can be in turns master or follower of the tempo.

Playing with time references: Different voices can be defined with different musical memories and different time references. Therefore, complex time relations between different voices and between inputs and outputs can be defined. For example, the fluctuating tempo of a given voice (*e.g.* an accompaniment) can be defined as a time reference, and local accelerations relative to this fluctuating reference can be performed with another voice (*e.g.* a solo). This way, the tempo of a voice can wander around the tempo of another and synchronize with it again when desired, which provides local expressive controls during performance.

11.2 Application and implementation

ImproteK: An autonomous use of the performance-oriented renderer can first be seen as an enrichment of what can be done with loop pedals. In addition to the synchronization it offers, the time mapping can be entirely composed before the performance with a finer grain and be modified in real-time. The dynamic time mapping can also be fed by dynamic generation processes sending sequences of symbolic improvisation, as in the case of the system ImproteK. These anticipations are buffered, waiting to be rewritten or rendered in due time.

Upstream and downstream musical controls: We show that the generation models and the reactive architecture can run autonomously, but also be used as a software instrument. We give an overview of different types of commands to take direct action on the music, and distinguish upstream and downstream controls. *Upstream controls* give declarative controls on the "intentions" querying the generation model and introduce the idea of DJing dynamic music processes. *Downstream controls* happen between generation and rendering and consist in the live alteration of what the computer plays (for example introducing online temporal transformations such as agogic accents or creating figures with loops and accelerations) while keeping the time reference provided by an external beat source.

Scientific collaborations and implementation, Antescofo: The performance-oriented renderer used in ImproteK, presented in Chapter 14, is implemented in the graphical programming environment Max (Puckette, 1991) using the score follower Antescofo (Cont, 2008a) and its associated programming language (Echeveste et al., 2013a,b). The work presented in this thesis was used as a motivation and application case to conceive new features of the language: temporal variables.

Scientific collaborations and implementation, OpenMusic: Our work was also used as an application case to design the new scheduling engine of the OpenMusic environment (Bresson et al., 2011; Bouche et al., 2016). This lead to the composition-oriented renderer presented in Chapter 12, using an offline memory and driven by the internal time of the musical material. This renderer enables to use the previously introduced generation models to explore musical ideas and create offline material in an interactive way, and to make them available in a framework aiming at composing music generation processes in a “meta-score” containing heterogeneous objects (the OpenMusic maquette).

12

Rendering, Synchronization, and Controls

The scenario / memory generation model introduced in Part I proceeds to a symbolic mapping between the units of the scenario and that of the memory, and it provides continuity at the scale of the form, *i.e.* the global *structure*. The reactive architecture constituted by the chain of agents presented in Part II preserves this structure and provides continuity at a *local* scale when rewriting previously generated anticipations. The last step detailed in this part is therefore to elaborate a mapping between this symbolic dynamic improvisation and the real time of the performance, and to provide continuity *in the time domain* when rendering and playing. Depending on the situation, it requires to deal with the hybrid time of the composition of musical processes, or to proceed to the elastic temporal mapping between the symbolic improvisation fragments and the real time of performance.

The architecture for dynamic and guided music generation presented in Part II receives from the environment:

1. the live inputs (in the case of online learning),
2. the control events,
3. the current performance time.

In return, it sends improvisation fragments back to the same environments (4). Live inputs and interaction (1,2) are managed autonomously by the reactive architecture. The next step is therefore to design a dynamic rendering architecture able to cope with dynamically generated material. The connection with the real performance time (3,4) is managed in an unified process through the continuous interaction with an *Improvisation Renderer*.

As illustrated in Figure 12.1, this architecture can be deployed in different context depending on the musical use and how we interleave scheduling and rendering. Chapter 13 presents the architecture model of an adaptive sequencer / renderer dedicated to performance where time markers are sent by the environment (an external non-metronomic beat source), and Chapter 14 gives an overview of the expressive musical controls that it offers. Chapter 15 presents an autonomous rendering architecture dedicated to dynamic composition of musical processes using an offline memory and driven by the internal time of the musical material.

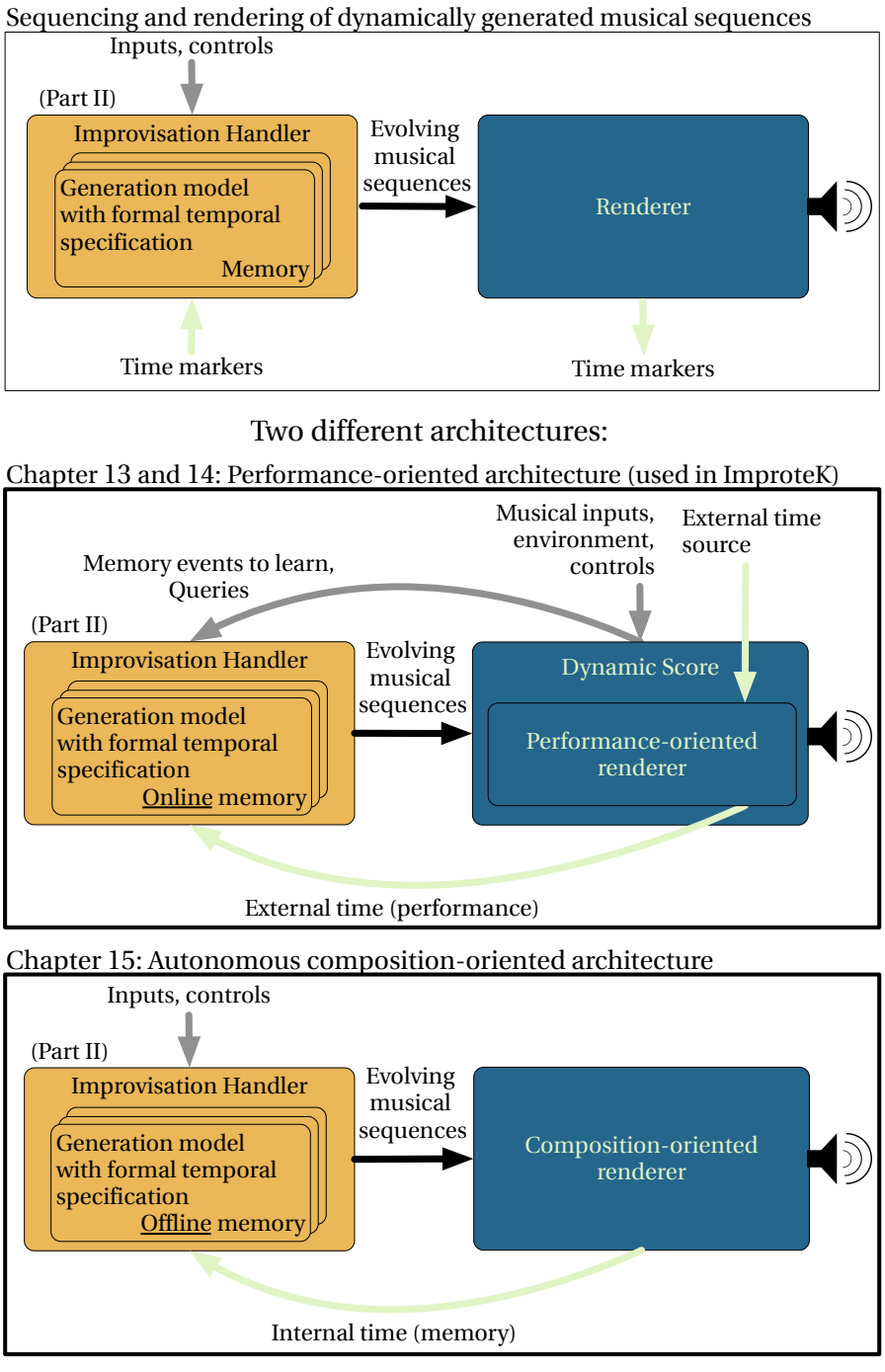


Figure 12.1: Performance-oriented and composition-oriented renderers.

13

An adaptive Performance-Oriented Sequencer

The work described in this chapter was realized in collaboration with José Echeveste and the Mutant team (Ircam-INRIA-CNRS) in the framework of the design of the programming language associated to the system Antescofo (Echeveste, 2015).

This chapter presents an architecture for adaptive rendering of dynamic multimedia sequences generated from recombination of live inputs[1]. This module dedicated to performance is *self-consistent* (and is used in association to the modules presented in Part II in the improvisation system ImproteK).

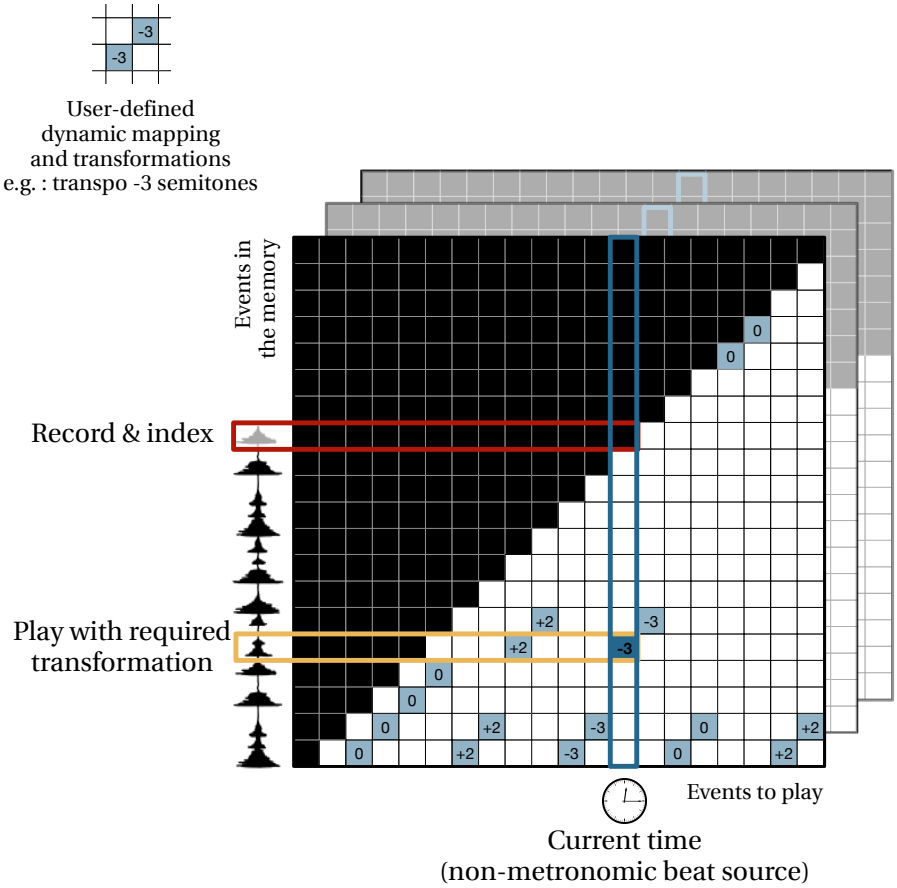


Figure 13.1: Record, segment, index, map, sequence, render, and synchronise beat-events coming from a live audio stream.

This architecture is designed to record and segment a live audio stream into beat-events that can immediately be played in synchrony with a non-metronomic pulse, according to a user-defined dynamic

1 Section 14.2 in Chapter 14 will give an example of video rendering, but we focus here on audio.

time mapping at event i in the performance, play event j in the memory with transformation (i, j). For instance, in Figure 13.1, the audio slice in the memory mapped to the current time is played with the transformation *transposition of 3 semitones*.

> The aim is here to design an architecture handling different voices with the following inputs and outputs:
>
> - Inputs:
> - a live audio stream (or offline musical sequences),
> - user-defined dynamic time mappings and transformations,
> - an external beat source.
> - Outputs: Live audio rendering
> - satisfying the current state of the time mapping,
> - synchronized with the external (non-metronomic) beat source,
> - handling the discontinuities when playing non contiguous sequences in the memory.

The proposed model is generic, as well as its implementation, and is independent of the way the evolving time mappings are given. An autonomous use of this architecture can be seen as an enrichment of what can be done during performances with loop pedals. [2] Here the time mapping can be entirely composed before the performance with a finer grain, and can be modified in real-time.

> **Remark:** This architecture enables to perform dynamic and synchronized re-injections of live material, as in the examples **Video A.1.4** and **Video A.1.2**. Of course, the same processes apply to offline memories, as in **Video A.1.9**. This is why, in some figures of this chapter, some dynamic time mappings linking the events of the performance and the events of the memory can be located above the "$y = x$" line in the time-time diagrams.

2 Indeed, loop pedals offer commands such as "from now, repeat what was recorded from 'now-4' beats to 'now-1' beat", and the possibility to store these loops in order to use them later.

13.1 Live Audio Re-Injection for Guided Improvisation

> The specified symbolic time mapping can evolve through time and be set at any time through an interface, or be received from outside via OSC protocol. In this chapter, we describe the proposed architecture through its use in the improvisation system ImproteK: the dynamic time mapping schematized in Figure 13.1 is fed by the dynamic generation processes presented in Part II which sends sequences of symbolic improvisation at each reaction.

The scenario / memory generation model introduced in Part I proceeds to a symbolic mapping between the units of the scenario and that of the memory, and it provides continuity at the scale of the structure. The reactive architecture constituted by the chain of agents presented in Part II preserves this structure and provides continuity at a local scale when rewriting previously generated anticipations. The audio renderer presented in this chapter proceeds then to the elastic temporal mapping between the evolving symbolic improvisation and the real time of performance, provides continuity in the time domain when rendering and playing.

LIVE AUDIO RE-INJECTION The musical memory is recorded in an audio buffer, and an index of the time markers corresponding to the events segmented by the external beat source is built online in the dynamic score (Chapter 10). This way, each unit of the symbolic sequences returned by the generation model is associated to the corresponding dates to read in the buffer. The system can therefore improvise by re-injecting live audio material, which is processed and transformed online to match the scenario, in synchrony with a fluctuating pulse (Figure 13.2).

Different voices constituting the machine improvisation are defined as different instances of a same generic process running in parallel. This process continuously sends the positions to read in the buffer via a phase vocoder, SuperVP (Depalle and Poirot, 1991), whose resynthesis of sound files can be modified by transformations such as time stretching, pitch shifting, filtering, etc. When two successive positions of the scenario are mapped to discontiguous slices in the buffer, a crossfade effect is used between a new voice instance and the previous one which is killed only after this relay.

SYNCHRONIZATION The synchronization with the environment and a fluctuating pulse is achieved by combining the synchronization strategies of the dynamic score and the phase vocoder which enables time-stretching with preservation of the transient signal components (Röbel, 2003).

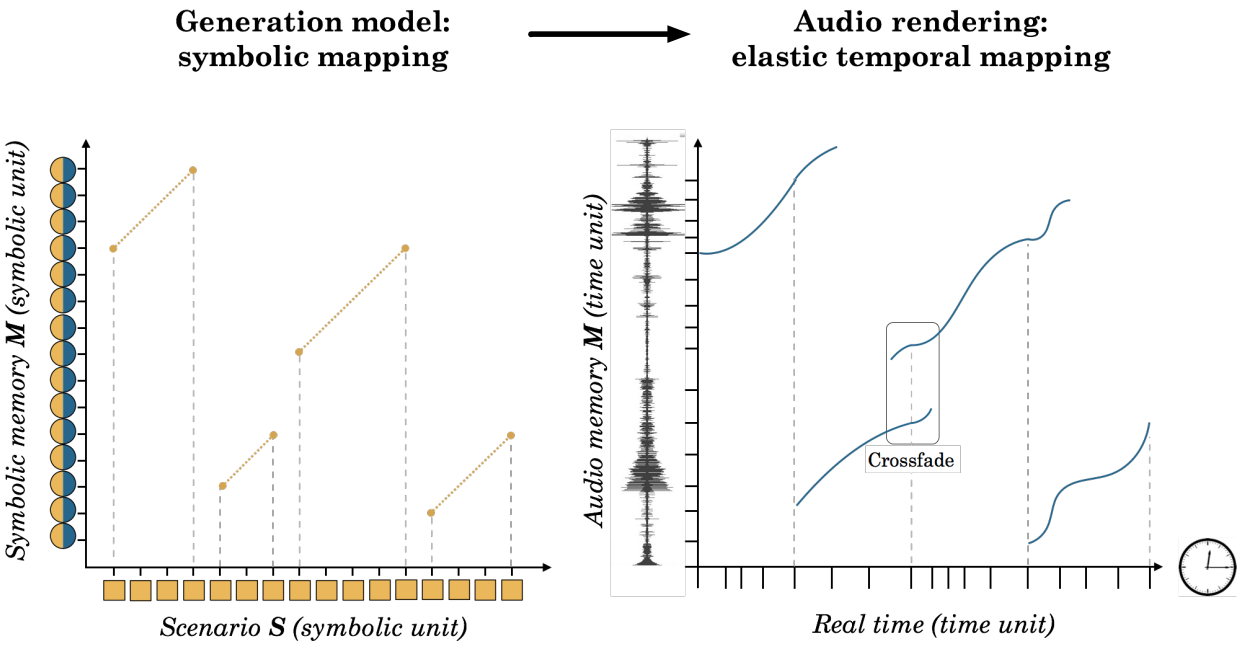


Figure 13.2: Generation model: symbolic mapping / Renderer: elastic time mapping.

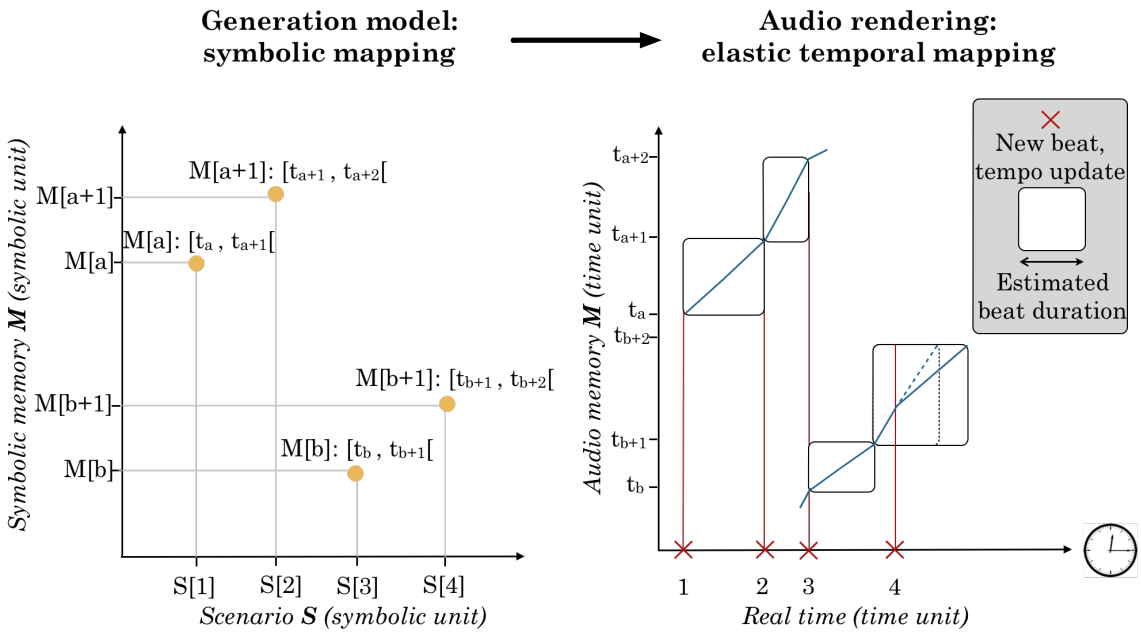


Figure 13.3: Tempo estimation, synchronization of the audio rendering with a non-metronomic beat.

We use specific *temporal variables* (see 13.4.1) whose updates are listened to like the musical events coming from a musician in a performance of mixed music using score following. This way, the system can synchronize with the updates of these variables the same way it follows the tempo of a performer. When such a variable is de-

clared with a prior knowledge on the periodicity of its updates, a new tempo estimation using the algorithms introduced in (Large, 2001; Large and Jones, 1999) is performed every time the variable is updated (see Section 13.4). The dynamic adaptation of the speed for reading the buffer at the pace of the performance is thus done by binding a synchronization variable to the external non-metronomic beat (Figure 13.3). In this chapter, the temporal variable associated to the updates of the external beat source will be denoted by T.

***Video* A.1.4**

Hyperlink video *(or vimeo.com/jeromenika/improtek-lubat-scat)*
Description: Appendix A.1.4.
Musical part IV: Chapter 17.

Video A.1.4 illustrates the processes managing synchronized live audio re-injections with examples of scat co-improvisations between a musician and the system. For all these improvisation sessions, the software starts with an empty musical memory and improvises by re-injecting the live audio material which is processed and transformed online to match the scenario while being reactive to external controls.

> We propose an architecture interleaving event-triggered and time-triggered mechanisms where the adaptive time-triggered mechanisms are parametrized by the event-triggered mechanisms which continuously modify the period of periodic synchronisation processes. This architecture relies on a hierarchy of nested periodic processes parametrized by the estimated tempo of the external beat source. This processes are detailed in the following sections.

13.2 Level 1 : the *Voice* Process

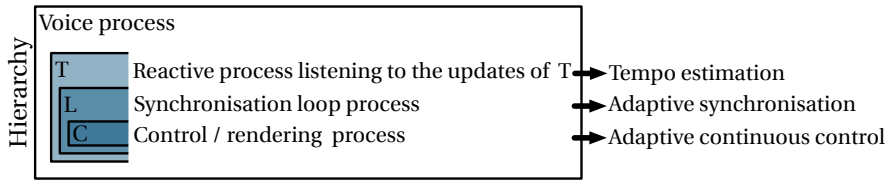


Figure 13.4: Hierarchy of processes in a "voice". T: temporal variable listening to the updates of the external beat source.

This section details the generic reactive process handling the generation, synchronisation, and rendering of a voice, parametrized by the estimation of the current tempo. Each voice is instantiated in the dynamic score (Chapter 10) to play polyphonic improvisations whose different tracks may have different memories and different tempo references. The hierarchy of processes nested in a *voice* process is represented in Figure 13.4: a reactive process listening to the updates of T (external beat source), a *synchronisation loop* process in

charge of adaptive synchronisation, and a *control / rendering* process sending continuous controls to the audio rendering unit.

Figure 13.5 schematizes the role of a voice process. The dynamic symbolic mappings between events in the scenario and events in the memory are built by the dynamic generation model. These sequences of time mappings and transformations (for example transpositions) are stored in a buffer, waiting to be rewritten or played in due time. When T is updated, the last slice of musical inputs is segmented, labeled, and sent to be learnt in the memory (Section 5.3). Simultaneously, a new tempo estimation is performed to update the period of the synchronization loop process described in the next section.

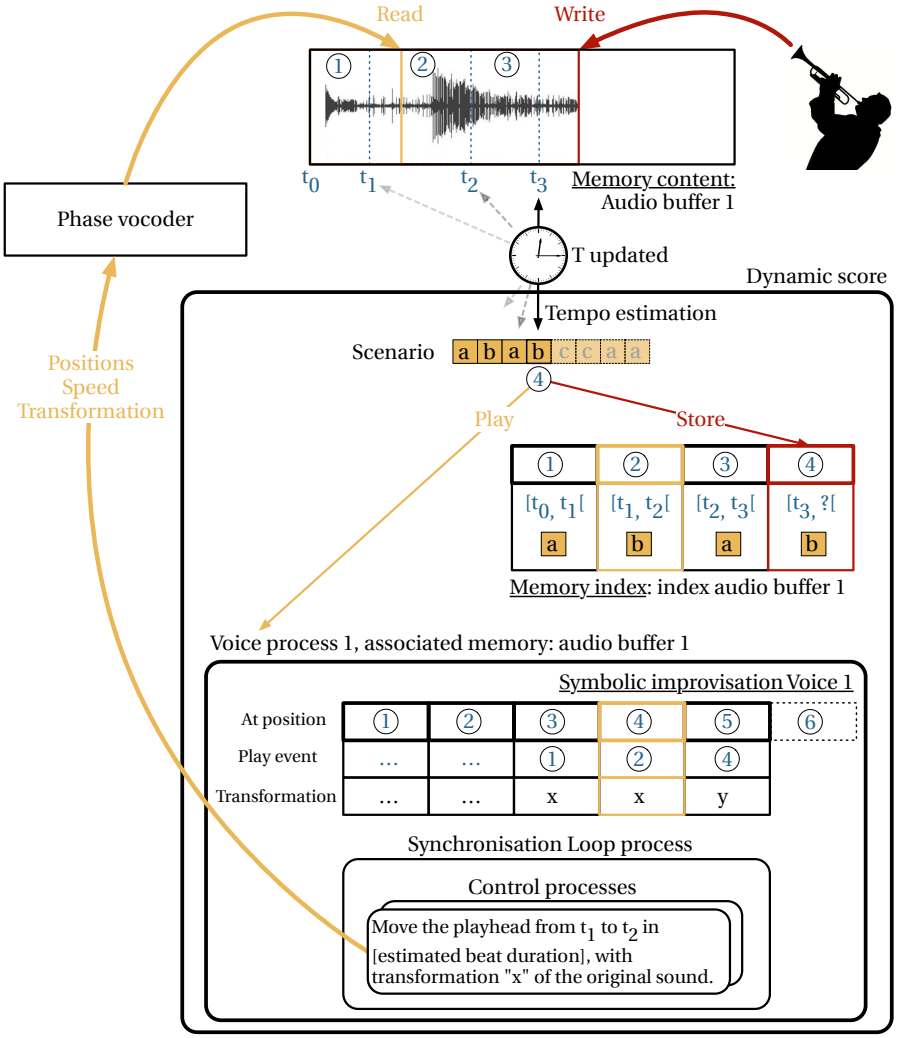


Figure 13.5: Concurrent processes writing and reading in an audio memory.

13.3 Level 2: the Adaptive *Synchronisation Loop* Process

The *synchronisation loop* is a periodic process whose period is updated each time a new beat is received from the beat source. It is in charge of launching rendering processes to send the position of the playhead in the buffer with an adaptive speed depending on the estimated tempo. A synchronization loop process runs as long as the events to read in the audio memory are contiguous and it manages the articulation between event-triggered and time-triggered runs of the rendering processes.

A discontinuity occurs when the next event to play in the memory is not contiguous to the previous one, or when the transformation applied to the audio contents changes. To describe the role of the synchronization loop process, we distinguish different cases:

CONTINUITY This case is illustrated by Figure 13.6:

- *If a new beat arrives before the predicted date:* the loop process aborts the running rendering process (C2), updates its periodicity according to the new tempo, and launches a new rendering process (C3) to read the next improvisation event with an adapted speed.
- *If a new beat arrives after the predicted date:* the loop has already launched a rendering process (C4) to read the next improvisation event from the anticipated date. When the new beat arrives, the loop updates its periodicity according to the new tempo, and modifies the speed of the already running rendering process (C4).

DISCONTINUITY In this case, the synchronization loop process handles the overlap of simultaneous rendering processes to enable crossfades. This case is illustrated by Figure 13.7:

- *If a new beat arrives before the predicted date:* the running loop process (L1) is aborted, but its child, the currently running rendering process (C1), still lives for a certain time. When the new beat arrives, a new loop process (L2) is launched with a periodicity deduced from the estimated tempo, and it launches a new rendering process (C2) to read the next non-contiguous event in the audio memory with an adapted speed. This way, the rendering processes C1 and C2 overlap during an interval $\delta 1$, and a crossfade (at constant total audio level) between the two associated players is realized updates its periodicity according to the new tempo, and launches a new rendering process to read the next improvisation event with an adapted speed.

- *If a new beat arrives after the predicted date:* we follow the same relay process with overlap between two rendering process C2 and C3, excepted that the currently running rendering process C2 had already been launched by the loop L1 running at that time.

> This architecture has been conceived so that the machine can successively follow the tempo of the external beat source or "take the lead on the tempo": in this last case, the computer can stop listening to the external time source and, for example, use the last tempo estimation to set the periodicity of the loop processes, that can also be modified dynamically through an interface (see 13.4.2).

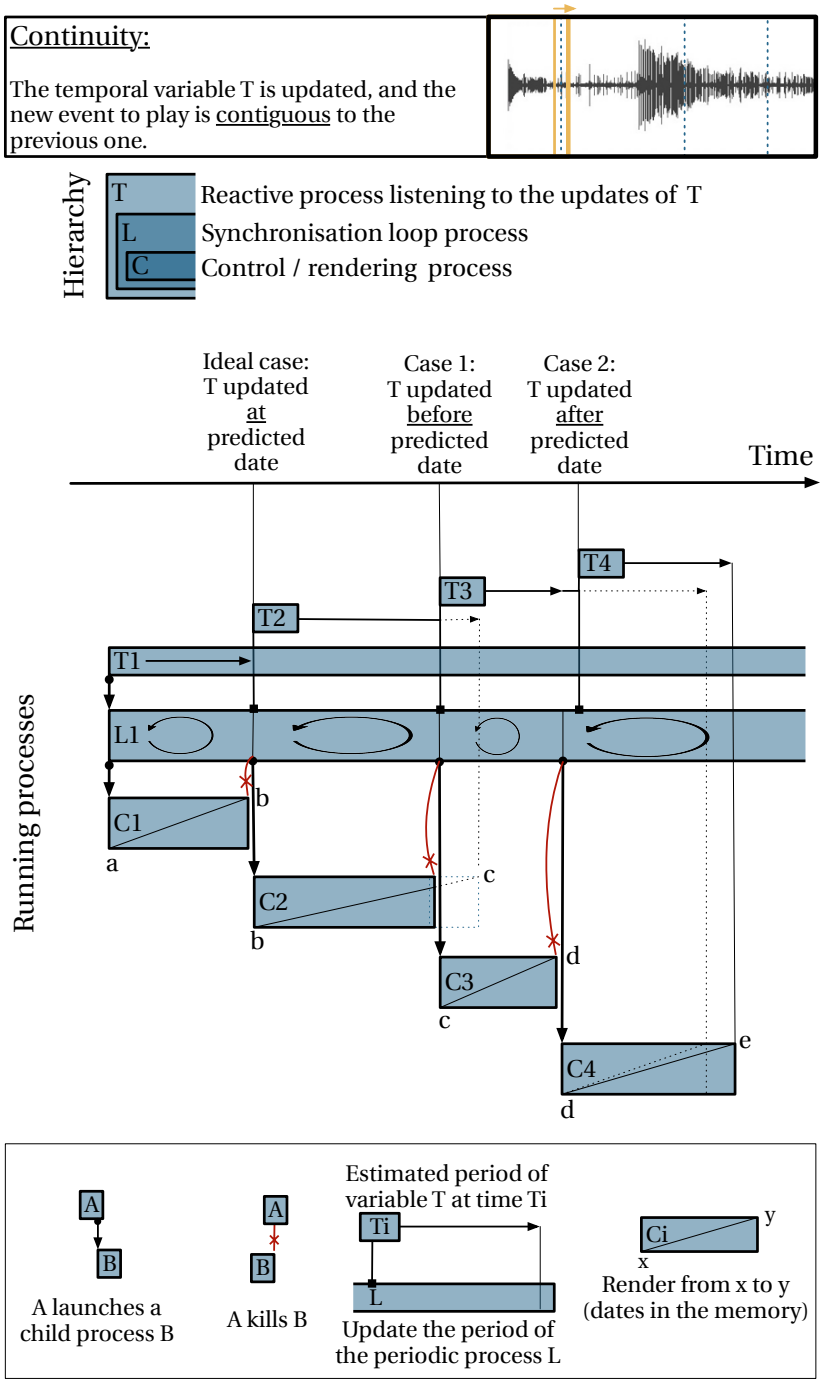


Figure 13.6: Adaptive control of rendering: continuous case

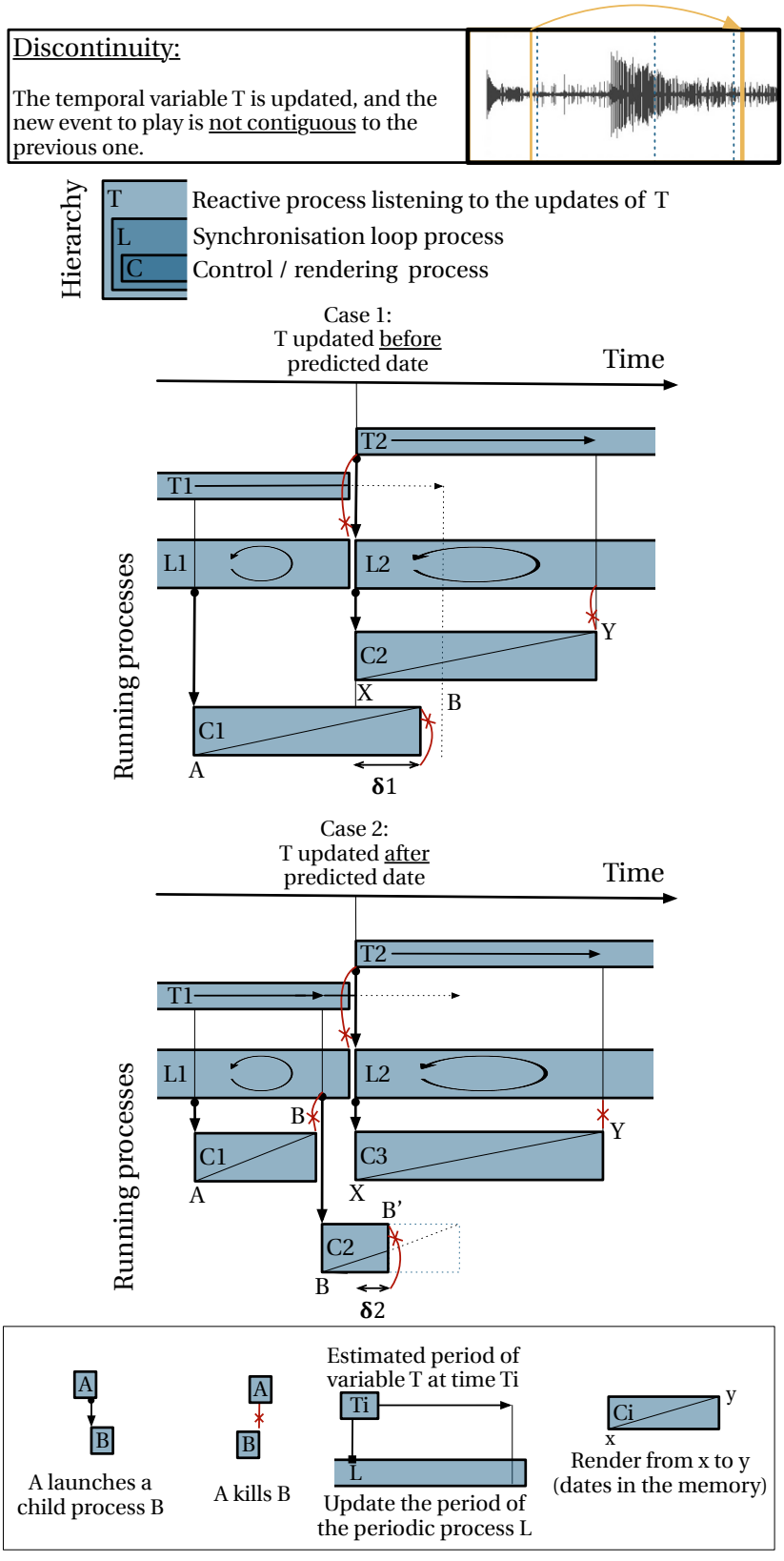


Figure 13.7: Adaptive control of rendering: discontinuous case

13.4 Tempo Estimation: Listening to Temporal Variables

13.4.1 Listening to Temporal Variables

PROGRAMMING MUSICAL TIME As introduced in Section 10.4, the dynamic score (Chapter 10) is written using the language associated to the system Antescofo. In the case of performance-oriented rendering, we decided to locate the controls sent to the rendering unit (audio player, midi player, video player,... depending on the nature of the contents in the memory) within the dynamic score to benefit from its tempo estimation features. During performance, the runtime system evaluates the dynamic score and controls processes synchronously with the musical environment, thanks to data received from the machine listening.

Antescofo offers the possibility of dating the events and the actions relatively to the tempo, as in a classical score. Within the augmented score language, the user can thus decide to associate actions to certain events with delays, to group actions together, to define timing behaviors, to structure groups hierarchically and to allow groups act in parallel. Delays and durations are arbitrary expressions and can be expressed in relative time (in beats) or in physical time (in seconds).

Antescofo provides a predefined dynamic tempo variable. During a performance of mixed music, this variable is extracted from the audio stream by a listening machine, relying on a cognitive model of the behavior of a musician (Cont, 2010). Besides, local tempos can be attributed to groups of actions using a dedicated attribute. All the temporal expressions used in the actions within this group are then computed depending on this frame of reference. As for other attributes, a local tempo is inherited if groups are nested.

In our application case, we needed to be able to define a specific time framework: a global tempo constituting a reference, and the possibility to define a proper tempo for each improvisation voice in order to apply local expressive tempo changes while being aware of the common reference. Furthemore, in such a context of guided improvisation, there is no predefined score but a scenario and an external beat source to follow (Chapter 8).

> TEMPORAL VARIABLE Originally, tempo estimation in Antescofo was based on the comparison between the expected positions of events in a predefined score, and their real positions played by a musician during a performance. In a first version of the sequencer/renderer presented in this chapter, we used a fictive score with phantom events so that a jump from an event to another triggered tempo estimation. Temporal variables were then introduced in the Antescofo language and used for the first

> time in the context of the work presented in this thesis (it is now used in various other musical projects).

(From Giavitto et al., 2015)

The synchronization mechanisms of a sequence of actions with the stream of musical events (specified in the score) has been generalized to make possible the synchronization of a sequence of actions with the updates of an ordinary variable. Antescofo provides a specific variable capable of tracking its own tempo and enabling synchronisation of processes with such variables. The updates of the variable act as the events in a score but the variable must be declared using the @tempovar declaration. The "tempo of the variable" is computed using the same algorithm used by the listening machine to track the tempo of the musician. A temporal variable $v stores in particular the following internal information that can be accessed at any time:

- $v.tempo: the internal tempo of $v
- $v. position: the internal position (in beats) of $v

Using temporal variables, it is possible to define synchronisation strategies of processes based on their progression using the @sync attribute to specify the synchronization of a sequence of action with the update of a temporal variable.

13.4.2 Implementation of the *Voice* Process

Pseudo code 13.1 presents the skeleton of the voice process described in Section 13.2.

```
@proc_def ::voice(...)
{
        @local $T
        whenever($T)
        {
                /* Retrieve the improvisation event at index
                $T in the buffered anticipations */
                (...)
                if(discontinuity)
                {
                        /* Launch a synchronisation-loop process
                        synchronized on the updates of $T */
                        if (! @is_undef($L) ) {abort $L @NOREC}
                        $L := synchronisation-loop  @sync $T(...)
                }
        }
}
```

Pseudo code 13.1: Definition of a voice process

It is driven by the updates of a local variable `$T`, which is visible by its descendants, in first place the synchronization loop process. In case of discontinuity, the running loop process is aborted *but not its descendants* (thanks to the `@NOREC` attribute) to have on overlap between the relaying rendering processes that makes possible to realize a crossfade.

Pseudo code 13.2 shows how a voice is instantiated in the dynamic score and linked to a chosen temporal variable.

```
/* Define a @temporal variable which will be
modified from outside by the environment, a priori every 1 beat*/
@tempovar $T($initial-tempo,1)

/* Launch a voice process synchronized on
this temporal variable*/
$Voice1 := ::voice(...) @tempovar $T
whenever($T)
{
        $Voice1.$T := $T
}
```

Pseudo code 13.2: Create a voice synchronized on a variable `$T` modified by the environment.

DOWNSTREAM EXPRESSIVE CONTROLS Two voices can be defined with different musical memories.

> Another advantage of this design is that it enables the definition of different voices with different time references. This can be used to give controls on local expressive modifications of tempo restricted to one voice in a polyphonic improvisation (see the application in 14.1.2). For example, the fluctuating tempo of one voice can be defined as a time reference (*e.g.* the accompaniment), and local agogic accelerations relative to this fluctuating reference can be performed with another voice (*e.g.* a solist).

To achieve this, the voices are linked as shown in Pseudo code 13.3. This way, the tempo of a voice can wander around the tempo of another and synchronize with it again when desired.

```
@tempovar $Tref($initial-tempo,1)
@tempovar $T1($initial-tempo,1)

/* Local modification of the tempo of a voice by the environment
    */
whenever ($coeff-mult-tempo)
{
        let $T1.tempo := $coeff-mult-tempo * ($Tref.tempo)
}
```

Pseudo code 13.3: Make a voice wander around the tempo of another.

These controls are “downstream controls", that is to say, controls on the music played by the machine located *after* the generation of symbolic improvisations (see Chapter 14).

13.5 Level 3: *Control / Rendering* Process

Finally, we go back to the description of the hierarchy of processes constituting the rendering architecture with the *control / rendering* process, sending continuous controls to rendering units with an adaptive speed interpolated from the tempo interpolation. This process is based on a the `curve` structure provided by the Antescofo language, inheriting the synchronization on a temporal variable from his ascendants. Curves are defined by breakpoint functions sampled by a variable which is used inside the `@Action` attribute to determine what the curve must do at each interpolation point.

```
$grain-start:= $start-memory
curve C @sync $T @target[k*duration-beat] @conservative
@Grain := ($step)ms
/* Define the actions to do for a grain $x*/
@Action := [...Actions($x)...]
{
        /* Render from $start-memory to $end-memory in 1.0 beat,
        then a little more for the crossfade */
        $x {
        {($start-memory + $step_mem)}
        1.0 {($end-memory+$step_mem)}
        ($Xfade-dur)ms {($end-memory+$Xfade-dur_mem+$step_mem)}
        }
}
```

Pseudo code 13.4: Synchronized control of a rendering unit.

In Pseudo code 13.4, `[$start-mem,$end-mem]` is the slice of memory that has to be rendered in one beat (specified in relative time: `1.0`), then a little more is rendered (additional length `$Xfade-dur_mem` in the memory) within a specified interval (specified in absolute time: `$Xfade-dur`) to enable a crossfade with another rendering process in case of discontinuity[3].

Finally, this generic architecture can easily be used to control different kinds of units (rendering, audio effects,...) by defining the action to realize upon an interpolation point (`[...Actions($x)...]` in Pseudo code 13.4). When driving a phase vocoder in case of audio rendering, the simple set of messages to send is defined in Pseudo

3 Thanks to another curve structure, an automation is sent to the players associated to the running rendering processes to realize the crossfades between these players within the specified time interval.

code 13.5 (an example of communication with another rendering unit is given in Section 14.2).

```
group {send-to-phase-vocoder $player /transfo $transfo
send-to-phase-vocoder $player /read-from ($grain-start+$latency)
send-to-phase-vocoder $player /read-to-in ($x+$latency) $step
$grain-start:= $x}
```

Pseudo code 13.5: Example: elementary group of control actions defined for audio rendering.

14

Interface and Controls: Toward an Instrument

In Chapter 9 and Chapter 10 (Part II), we underlined the fact the designed reactive mechanisms are generic and do not focus on where the controls come from: depending on the musical project, they can be launched by composed reactivity rules, defined in a higher-level improvisation plan, or plugged to an external listening module... that is to say without human intervention. These commands can also be left to an operator-musician. In this chapter, we concentrate on the latter perspective, regarding how to use the models and the system as a software instrument. This way of using the system was valued by several musicians we worked with (see in particular Section 18.1 in Part IV).

14.1 Upstream and Downstream Controls

With this in mind, we put different types of commands into place to be able to take direct action in the "upstream" or "downstream" musical process (Figure 14.1), that is to say by providing declarative controls on the "intentions" before generation, and other happening between the generation of anticipations and rendering. Theses commands - presented in their algorithmic and computerized aspects in previous chapters - were all made available via a graphical interface and physical interfaces [1] to allow better responsiveness, going towards an interpretive dimension (see Appendix B.3).

This way, the system allows an operator-musician to improvise from musical memories stored with their phrasing and articulation to then be recombined and transformed to match the scenario. It is important to underline that this overall concept has an impact in the playing aspect of "instrumentality" since, when the system is used as a software instrument, its user plays with phrases without producing the instrumental gesture necessary to the performance. His attention is entirely focused on more abstract issues towards improvisation: the choice of phrases played along with their melodic, harmonic, rhythmic features and responses to other musicians within the collective improvisation.

1 Using simple controllers such as Ableton Launchpad. Currently, a tablet version is being developed.

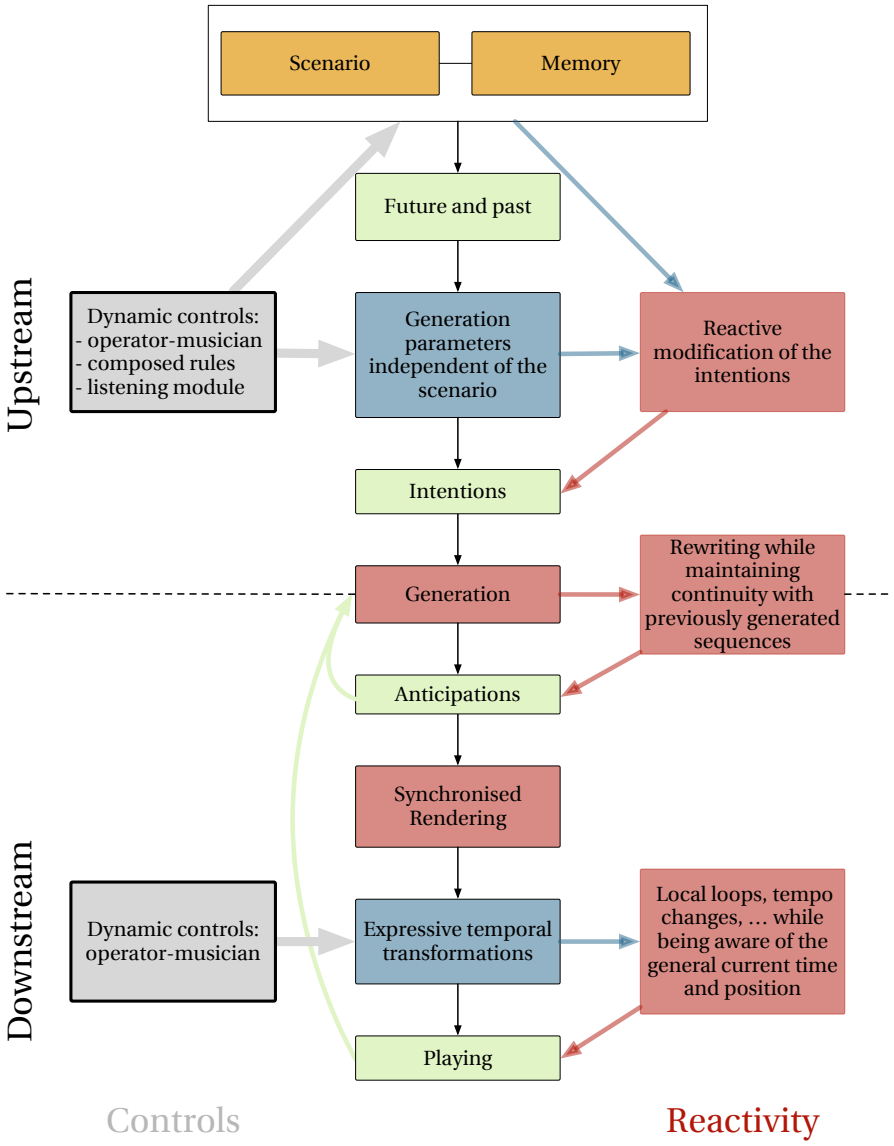


Figure 14.1: Upstream and downstream controls and reactivity.

The focal point of this skills transfer between man and machine, where the computer is entrusted with musical performance tasks, is to allow the user to explore "unprecedented" things and more specifically, things impossible to perform on an instrument (see the discussion on "machinic virtuosity" with Bernard Lubat in 17.4.3).

14.1.1 Upstream Controls, "Upstream Reactivity"

The interfaces enable the user to define and play with controls located before music generation, that is to say to send manually the dynamic controls exploiting the reactive mechanisms described in Part II (in particular Chapter 9). Among the upstream "reactions" im-

pacting generation, for instance, it is possible to reset the musical memory, change content, apply filters on secondary settings that do not depend on the scenario, etc. as mentioned with the basic examples in **Video A.2.3**. In all these cases, the user's reactions change the "intentions" consequently changing the logical "anticipation" sequence produced, which is not to be neglected from an aesthetic point of view.

This idea of "recalling a memory" according to a scenario that serves as a guide for improvisation is pretty close to what musicians actually do when learning to master certain sequences with his fingers in order to replay them in alternative situations. Other commands given to the user are standard modifications of model settings, for example those relating to the recombination stability of the fragments retrieved in the memory. The interfaces also enable to chose in real time the sequences or areas of memory among those that have just been captured or among loaded sequences (older sessions or annotated music). This way, the operator can choose to dig into another recording to produce the next seconds of the scenario like a musician quoting a piece of another theme in the middle of an improvisation. This command goes beyond the idea of quotation (which will be detectable only if the musical memory has long subsequences in common with the current portion of the scenario).

> By choosing sequences recorded in the context of different scenarios other than the scenario in progress, the user performs "hybridization" (Section 4.3). We go from DJing samples to DJing a dynamic process: for a DJ, choosing the memory is like triggering a sample playback, meaning an "inert" recording. Here, it means initiating a generative procedure that will rummage through memory, for instance by calculating a solo on *Autumn Leaves* from solos on *All the things you are*, or by mixing solos played by different musicians.

14.1.2 Downstream Controls, "Downstream Reactivity"

Responsiveness can also result in the live transformation of what the computer plays, and be therefore located after generation within the processing chain. In this view, the user is given "downstream" controls through the physical interface. This is made possible by the architecture of the improvisation audio renderer presented in Chapter 13: different "voice" processes coexist within it, and can have a different memory, but also different local tempos and different positions in the scenario while maintaining the common time reference so they can readjust to it (13.4.2).

So, it is possible to loop an arbitrary number of beats or play voices n times faster or slower, and these actions can be combined to create

figures. When analyzing jazz solos, one can see that these transformations are omnipresent in some musicians practice. Looping what was just played by the system within a given number of beats (two, four or eight usually) is the equivalent to what is called a riff. Accelerating the flow of a sequence is like splitting the tempo in two or more locally to an agogic acceleration (passaging from an eighth note to sixteenth note performance).

The different reactivity modes are complementary and go one after the other within the improvisation. Calculating sequences provides greater autonomy to the machine, but less reactivity to the user. The loopback and acceleration are simpler technical operations, but they allow to act immediately by giving a different color to the material that has just been played even though it has already happened (later we will see the importance of this immediacy when handling what Bernard Lubat called “errors”, see Chapter 17).

14.2 Network Architecture and Video Rendering

14.2.1 Network Architecture

We implemented a network architecture (Figure 14.2) to be able to synchronize different instances of ImproteK on a same beat source. One computer is considered as a master and broadcasts the pulse (imposed by an accompaniment played by the machine or coming from a beat tracking system) to the other instances of the system. All the computers of the network can share a same scenario and modify the position within it, broadcasting this modification to the others computers. This is useful if, for example, the band actually decides to play one more chorus on “theme A” instead of going to “theme B” as it was first planned.

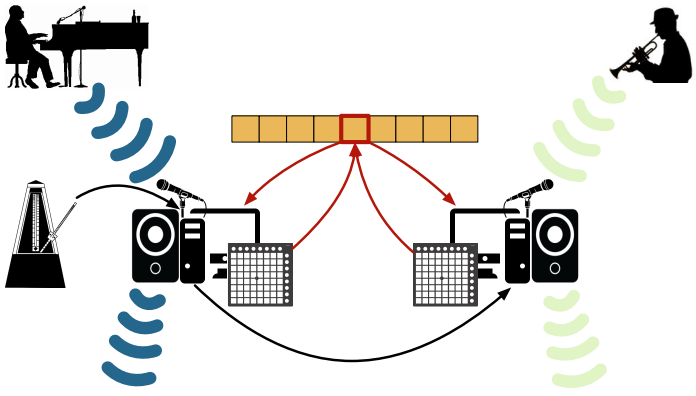


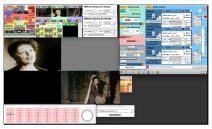
Figure 14.2: Network: several instances of ImproteK synchronized with a same pulse and scenario.

14.2.2 Video Rendering

Thanks to the modularity of the rendering architecture (Chapter 12) and the genericity of the models and their implementation (Chapter 6), the work documented here can easily be transposed to any alphabet for the improvisation scenarios, but also to different types of contents. This dissertation is focused on the generation of music and sound, but different experiments were carried out, notably to generate text and video from offline and/or online memory.

The adaptive performance-oriented renderer presented in Chapter 14 has been generalized to become a control unit to pilot multimedia rendering of sequences generated from offline or online inputs. For instance, Georges Bloch linked the Omax video player (Bloch et al., 2008) to the system: in this case, the control processes (13.5) within each voice of the Dynamic Score (Chapter 13) do not only send control messages to audio modules to pilot audio rendering, but also to video units to pilot video rendering through simple messages: indexes of frames and speed coefficients. This way, polyphonic video improvisations following a given scenario can be created using pre-loaded video clips or online recorded video as memories, while being reactive to the same controls than that described for audio or midi (see **Video A.2.4**).

Video A.2.4



Hyperlink video
(or vimeo.com/jeromenika/improtek-impro-video)
Description: Appendix A.2.4.

15

A Composition-Oriented Renderer

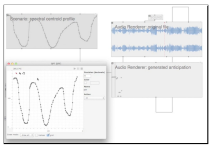
The work described in this chapter was realized in collaboration with Dimitri Bouche and Jean Bresson in the framework of the design of the new scheduling engine of OpenMusic.

Chapter 13 and Chapter 14 described the adaptive sequencer / renderer of the ImproteK system used during performances with musicians. In this chapter, we present *an other* rendering architecture allowing an autonomous use of the Improvisation Handler (Chapter 9) within a compositional process. This composition-oriented renderer is implemented in the OpenMusic environment (Bresson et al., 2011) to use the new scheduling engine of this environment (Bouche and Bresson, 2015b). It is designed as a scheduling module which runs in parallel to the Improvisation Handler all along the performance.

15.1 Composition of Music Generation Processes

15.1.1 Motivations

Video A.2.3



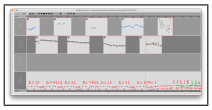
Hyperlink video *(or vimeo.com/jeromenika/improteksmc15)*
Description:
Appendix A.2.3

The reason for implementing a dynamic composition-oriented renderer is to provide the possibility to use the models to explore musical ideas and create offline material in an interactive way, as in the examples of **Video A.2.3** (Chapter 9).

We also want to make the models available in a framework aiming at composing music generation processes in a "meta-score" containing heterogeneous objects (musical sequences and dynamic musical processes), and to chain them with other compositional processes. In this case, the composition phase do not aim at producing a list of static events to render but to establish a set of rules and constraints from which events will unfold in real time. This approach can be related to the notion of processes in I-score (Desainte-Catherine et al., 2013) used in conjunction with temporal constraints. In addition, we aim here at interleaving control and generation. It results in changing dynamically the content of the "meta-score", thus the equivalent set of rules.

15.1.2 Example

Video A.2.5



Hyperlink video
Description:
Appendix A.2.5

The work presented in this thesis was used as an application case to design the new scheduling engine of the OpenMusic environment (Bouche et al., 2016). We describe here a simple example which served as a basis to develop this research (see **Video A.2.5**).

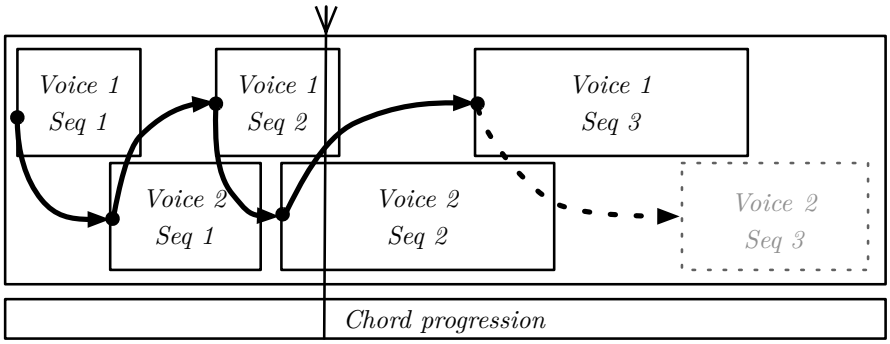


Figure 15.1: Integrating dynamic generation processes in a meta-score (from Bouche et al., 2016).

Figure 15.1 depicts the integration of Improvisation Handler agents (Chapter 9) embedding the scenario / memory generation model (Part I) in a "meta-score". The objective is to embed musical agents generating musical material in high-level, formal while interactive time structures. We consider two dynamic musical processes (voice 1 / voice 2), playing short sequences in turn (question/answer alike) on a given chord progression. These solos can overlap and have heterogeneous durations. The aim is that each one is determined by the specified scenario but also by some contextual parameters depending on the recent past (for instance the output produced by the other agent), and by real-time user controls over some parameters such as the frequency of the trade between voices. Every sequence played by an agent should determine and dynamically schedule the next sequence played by the other agent. The two agents use two distinct instances of the ImproteK music generation engine with different generation parameters.

We implemented the musical agents as visual programs (patches) in OpenMusic, integrated in an OpenMusic "maquette" (meta-score containing heterogeneous musical objects). Each agent embeds a reactive Improvisation Handler (Chapter 9) and is able to produce such musical sequences on demand. The program is computed by the engine when an agent is reached by the score playhead. It performs two operations:

1. It generates some data (a MIDI sequence) according to the scenario (chord progression) and other generation parameters. The scheduling of this data is automatically handled by the system.

2. It launches a new computation generating another agent in the other voice (user-defined behavior).

Figure 15.2 shows the main score interface. The preliminary evaluation of a control patch builds the two instances of the dynamic musical processes, includes them within the two interconnected agents,

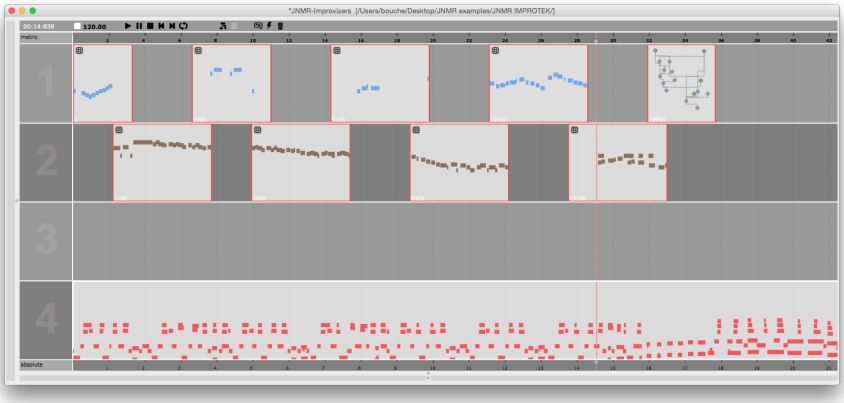


Figure 15.2: OpenMusic maquette performing the example.

and adds the first agent on the first track to start the sequence. The rest of the process unfolds automatically at rendering time, computing the sequences in alternation from the two agents.

15.2 Scheduling Strategy

To describe the scheduling architecture we need to introduce a number of additional concepts: an *action* is a structure to be executed (specified by a function and some data); a *plan* is an ordered list of timed actions; the *scheduler* is an entity in charge of extracting plans from musical structures; and the *dispatcher* is the entity in charge of rendering plans.

A hierarchical model is used to represent musical data (for example, a chord as a set of notes, a note as a set of MIDI events...) and to synchronize datasets rendering. To prepare a musical structure rendering, the scheduler converts the object into a list of timed actions. Triggering the rendering of a "parent object" synchronizes the rendering of its "children"[1]. Then, the dispatcher renders the plan, i.e. triggers the actions on time (Bouche and Bresson, 2015a).

The scheduler and dispatcher cannot operate concurrently on a same plan, but they can cooperate. Scheduling is said *dynamic* when the dispatcher is likely to query the scheduler for short-term plans on the fly, and/or when the scheduler is likely to update plans being rendered by the dispatcher (Desjardins et al., 1999; Vidal and Nareyek, 2011). Our strategy is based on a *bounded-term lookahead* scheduler: instead of planning a list of actions representing the whole content of a musical object, the scheduler is called on-time by the dispatcher and outputs plans applicable in a specified time window.

1 As introduced in (Rodet et al., 1983). In terms of scheduling, the hierarchical representations also eases the development of optimized strategies (Firby, 1987).

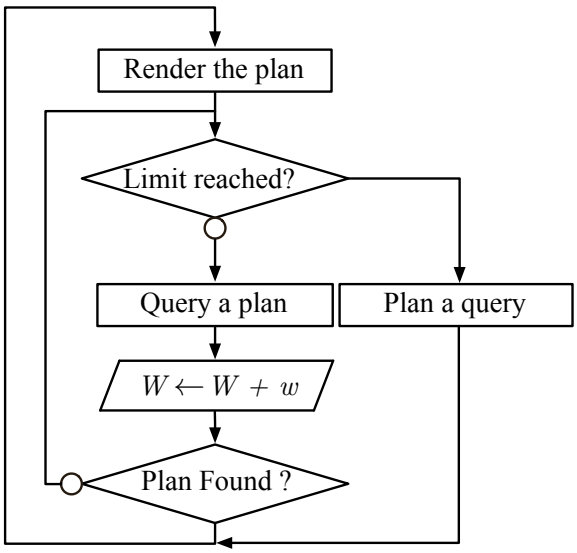


Figure 15.3: Short-term plan extraction flowchart.

The flowchart on Figure 15.3 summarizes the plan extraction algorithm used by the system to render musical objects. Typically, the dispatcher calls the scheduler for a plan applicable in a time window W of duration w, then the dispatcher can render this short-term plan on time and query the scheduler for the next one. The lower w, the most reactive the system is, at a cost of more computations (w can be tweaked accordingly). If the scheduler returns no plan (i.e. there is nothing to render in the queried time interval), the dispatcher can query again for the next time window until a plan is returned. Therefore, the time window W can be far ahead of the actual rendering time of the structure, and might not be the same across concurrently rendered objects. Plan queries themselves can also be planned as actions to execute in the future. For instance, a limit of successive plan queries can be set to avoid overload (e.g. if there is nothing else to play): in this case sparse planning queries can be planned at the end of each time windows.

15.3 Interactions with the Improvisation Handler

The *Improvisation Renderer* (R) connected to the Improvisation Handler (Chapter 9):

- receives and renders the produced fragments,
- communicates the current performance time t^p.

With regard to the scheduling architecture, R is a structure containing two children objects, the mutable priority queues:

- RC (*render action container*) containing actions to render, extracted from improvisation fragments.
- HC (*handler action container*) containing time marker actions to send back to the handler H.

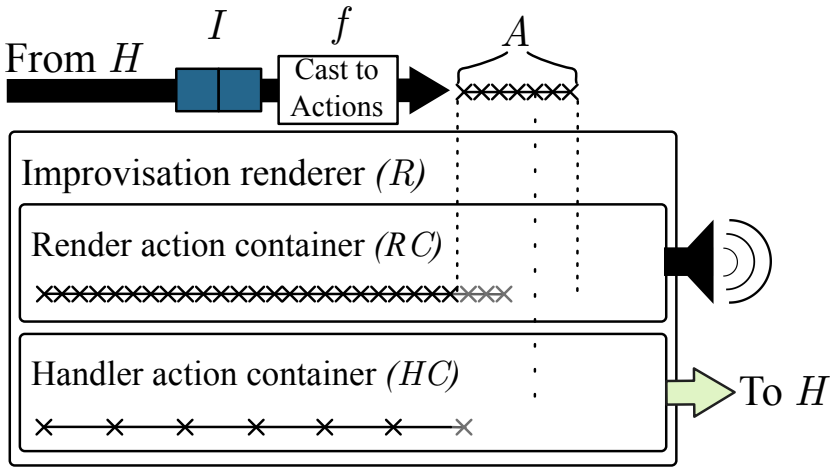


Figure 15.4: The *Improvisation Renderer*.

Figure 15.4 depicts the Improvisation Renderer and its communication with the Improvisation Handler. An improvisation fragment I is outputted from the Improvisation Handler, and this fragment is casted into a list of actions A integrated into the render action container RC. This translation can be defined depending on the type of improvised data. For instance, if the improvisation slices contain MIDI, actions will consist in calls to *midi-on* and *midi-off*. If the list of actions is overlapping with existing content (i.e. with previously generated fragments of improvisation already stored as actions in RC), the new actions substitute the overlap and add the last generated improvisation to RC. At the same time, information about slices timing of I is extracted to feed the handler action container HC with time markers that will be sent back on time to the Improvisation Handler.

To perform the previous operations, we define the following functions:

- $Cast(I)$: cast an improvisation fragment I into a list of timed actions A,
- $Timing(I)$: extract a list of actions from an improvisation fragment I, corresponding to the slices' respective times,
- $Tile(C, A)$: integrate an action list A in the action container C, overwriting the overlapping actions.

In order to connect the Improvisation Handler (section 9.2.2) to the Improvisation Renderer, the output method f of the Improvisa-

tion Handler shall therefore be the function of an improvisation fragment I and the Improvisation Renderer R defined as:

$$f(I,R) = \begin{cases} Tile(RC, Cast(I)) \\ Tile(HC, Timing(I)) \end{cases}$$

R can then be planned and rendered as a standard musical object, although this object will be constantly growing and changing according to performer's inputs or user controls. The short-term planning strategy will allow for changes not to affect the scheduled plans if they concern data at a time ahead of the performance time by at least w. In the contrary case (if data is modified inside the current time window) a new short-term plan extraction query is immediately triggered to perform a re-scheduling operation.

Part IV

"PRACTICING": LET THE MUSIC(IANS) (PL/S)AY

Part IV describes some collaborations with expert improvisers during performances and work sessions.

Chapter 16 summarizes the approach presented in Part IV: these interactions were an integral part of the iterative development of the models and of the ImproteK system. The public performances and work sessions were associated to listening sessions and interviews to gather numerous judgements expressed by the musicians in order to validate and refine the scientific and technological choices.

Chapter 17 focuses on the long-term collaboration with jazzman Bernard Lubat that led, through experimentation sessions and public performances, to the development of the first models and the first functional prototype.

Chapter 18 covers work carried out with eight other musicians during this thesis, exploring different idioms and types of interactions.

16

Summary and Contributions

The work presented in the previous parts of this thesis was realized in close and continuous interaction with expert musicians. These interactions were an integral part of the iterative development of the models and of the ImproteK system. In this part, we describe some creative uses of the system with improvisers on stage or during work sessions and residencies. More than 15 demonstrations, public performances, and concerts using the successive versions of the ImproteK system were performed between 2012 and 2016. Among them: Novart Festival, Bordeaux, France (2013); "Mons Capitale de la culture" Festival, Belgium (2015); Montreux Jazz Festival, Switzerland (2015); Uzeste Festival, France (2015); "Pietre che cantano" Festival, L'Aquila, Italy (2015).

One of the usual forms of ethnomusicology "participant observation" is for the researcher to learn to play the instruments he studies. Hood (1960) emphasized how this approach requires the development of specific skills that he grouped under the term of "musicality": ears, eyes, hands, voice all at work to acquire the necessary fluency not only for performance but also the understanding of the studied music. In a manner of speaking, Part IV describes a case of ethnomusicology participant observation except that the one striving to learn to play music on the field is not an ethnomusicologist equipped with his musical instrument, but a computer program (and its designers looking to move it forward). We can say that this study establishes an element of "simulation" in participant observation.

When defining "idiomatic" improvisation, what Bailey refers to as "special language" goes back to the cultural conditioning that results from a particular historical evolution which led to the creation of a universe of predetermined sounds. This can be linked to particular music styles and to associated institutions where the music is performed. With this in mind, model development cannot be separated from fieldwork relating to these musical genres and their specific institutions. When the system operates certain transformations applied to sequences captured in the performance of a musician, the result can be "inappropriate", if not to say absurd in a particular idiom context. So, since ImproteK is used in an idiomatic improvisation context, the assessment of musicians specialized in the idiom at hand - we will refer to them as "experts" - on results generated by the software takes on a considerable place.

In the following chapters, we present the validations and discussions of the different musical dimensions we studied from the judgments of these experts. Beyond validation and discussion, these collaborations were intended to move from “simulation” to “stimulation”. We exploited the successive versions of the models, seeking then to divert them and perpetuate these diversions in order to participate in the creative processes of the musicians. These collaborations led to hours of filmed and documented music sessions, listening sessions, and interviews. The discussions about the models and the successive prototypes of ImproteK led to general discussions about music improvisation in general, and the analysis of the interactions with the system served as a starting point for some musicians to develop some thoughts about their own approach of music improvisation.

Musical collaborations (in chronological order):

Bernard Lubat and "La Compagnie Lubat":

- "Jazzed-up song", jazz, scat, and free improvisation.
- *Focuses:* long-term collaboration to design the first models and prototypes: recombining and phrasing; downstream controls; reduction and multiplication; "hybridization"; rhythm, synchronization, groove.

Jovino Santos Neto:

- Brazilian music and jazz.
- *Focuses:* improvisation using an online musical memory; harmonization, arrangement; "hybridization"; rhythmic phrasing.

Kilema, Velonjoro, and Charles Kely:

- Marovany zither and jazz.
- *Focuses:* contrametricity; rhythmic articulation.

Louis Mazetier:

- Stride piano.
- *Focuses:* mixing offline and online memory; scenario defined on an alphabet describing harmony and macrostructure; secondary generation parameters.

Michelle Agnès Magalhaes:

- Composed contemporary improvisation.
- *Focuses:* improvisation using an online musical memory; non-idiomatic composed improvisation; content-based alphabet; scenario: discretized multimodal profile of audio descriptors.

Rémi Fox:

- Funk, jazz, and structured "generative improvisation".
- *Focuses:* improvisation using an online musical memory; interface and controls: duet between a musician and an operator; rhythm, synchronization, groove; definition of an alphabet, a scenario and constraints.

Hervé Sellin and Georges Bloch:

- Jazz and "deconstruction of the idiom".
- *Focuses:* "Hybridization"; mixing offline and online memory; improvisation plans; "music-driven" vs. "event-driven" interactions; video improvisation.

17

Bernard Lubat: Design of the First Prototype

This chapter gives an account on a study led with jazzman Bernard Lubat for the development of the first models and prototypes of ImproteK. Through close and constant interaction, we gathered many of his assessments on the results produced by the successive versions of the prototypes. We will call these assessments "judgments of taste" although this is not without difficulty since Bernard Lubat openly states that he "does not trust his taste" (Lubat, 2010). In fact, to him, "taste" still reflects cultural standards that blindfold the personal contribution of the artist involved in his creation and opens his unconscious to those standards. We describe the method used during this work and outline the chronology of the different performance modes designed during the many close interactions with Bernard Lubat. We will analyse certain stages of the study by focusing on the phrasing of his "tastes" and aesthetic stakes, reflected both in universal values, but also in some strict restrictions imposed by specific musical idioms. The outline of this chapter follows the chronological design of the first versions of the generation models and of the first prototype of ImproteK.

17.1 Study with a Jazzman: Bernard Lubat

17.1.1 Bernard Lubat

The first viable prototype was developed with the participation of a specific jazzman, Bernard Lubat. While numerous other musicians participated in the study (see Chapter 18), the study described in this chapter is essentially monographic. While this approach loses in generality, it gains in depth with the analysis of the musician's reactions, since we conducted dozens of experimental sessions and interviews with him (over ten on-site visits of several days each time, in his village, Uzeste in Gironde, spread over three years). The methodology for this study has three aspects. The first aspect is, generally speaking, semi-structured interviews with Bernard Lubat on topics dealing with the use of improvisation technology. The second aspect relates more specifically to interaction experiments between him and the software, which was aimed at improving certain features. These

experiments were filmed and the music recorded. The third aspect was to carry out listening sessions with the musician of music played together during the experiments. In addition to live listening, playback enables favourable distance for comments and criticism. The study period, 2011-2013, coincides with the ImproteK software development getting to a point of maturity sufficiency to be used in concert, which was the case November 16th 2013 at the Novart festival in Bordeaux. After the period covered in this chapter, collaborations continued through numerous communications, workshops, and concerts.

Why Bernard Lubat? The first reason being that Bernard Lubat is a tireless experimenter and that kind of inquisitive mind was needed to devote that amount of time to the babbling encounter of an improvisation software apprentice and his set of unusual experiments. The second reason is that Bernard Lubat's reflection on music is built on an interest in philosophy and political commitment, which gives his perception unusual depth, as we will see later on. The third reason is that Bernard Lubat is at the crossroads of influences that range from contemporary creation and free collective improvisation to the deepest roots of African-American jazz tradition. This last part is clearly from his biography. A percussionist, pianist, accordionist and singer, Bernard Lubat was born July 12th 1945 in the village of Uzeste in Gironde. From the age of three, he played the drums alongside his father, Alban, who played the accordion for the Estaminet balls, the village dance bar run by his parents. After studying piano, he turns to learning percussion at the Conservatoire de Bordeaux. At the age of eighteen, Bernard Lubat moves to Paris to attend courses at the National Music Conservatory. Going to jazz clubs, he meets one of the bebop masters, drummer Kenny Clarke, which he replaces in pianist Bud Powell's trio. At age twenty, he plays in the opening act for John Coltrane as a vibraphonist for the Jef Gilson orchestra. At twenty-five, he spends two years touring as a drummer with Stan Getz, Eddy Louiss on the organ and René Thomas on guitar. So, from the early years of his career, Bernard Lubat's imagination was captivated through direct contact by some of the most important personalities in jazz history such as Kenny Clarke, Bud Powell, John Coltrane or Stan Getz.

In addition to the traditional ball of his Gascon origins and jazz discovered in Paris in the 1960s, Bernard Lubat practices various musical genres: contemporary creation, free collective improvisation, songs. But his career was forged in response to certain aesthetic and social standards, primarily those of written music taught at the conservatory. The traditional Gascon ball and jazz taught him a different approach to music based on oral practice, what Béthune (2004) calls "second orality". Bernard Lubat is interested in music itself, but also in its place in society and its social role. In 1975, he creates the

"Compagnie Lubat" which sets up in the Rue Mouffetard Theater to explore different paths of radical deconstruction of the artist/spectator relationship.

1978 marks a turning point in his career: he puts his parisian multigenre musician career aside to return to Uzeste and create a festival originally inspired by the Festival of Humanity combining music, politics, theater, philosophy, dance and pyrotechnics. The festival still exists over three decades later. Ignored by most of the media from the very beginning, over the years, he has won the loyalty of countless musicians (Archie Shepp, Martial Solal, Eddy Louiss, Michel Portal and others who started with the Company such as Andre Minvielle or Louis Sclavis), singers (Claude Nougaro, Jacques Higelin), actors (Richard Bohringer, André Benedetto, Philippe Caubère).

Musically speaking, over the past thirty years, Bernard Lubat has been developing what he calls his "jazzed-up songs" gathering several musical idioms which reflect the abundance of his musical influences: bebop, biguine, blues, funk, musette French folk music. Certain versions that he undertakes sometimes radically deconstruct the original idiom. Others explore the idioms by altering them through somewhat visionary reconstruction. Bernard Lubat himself explains his approach halfway between loyalty to traditional idioms and their radical deconstruction "My trips between Eddy's aquatic, deep, organic swing which came from Kenny and deconstruction are compatible". With the help of an experimenter musician like Bernard Lubat, IT development was able to explore new ways of musical improvisation while relying on expert knowledge of the studied musical idioms.

17.1.2 Incremental Design of the First Models and Playing Modes During Working Sessions

In idiomatic cases, the improviser is confronted with "acceptability": what is acceptable in terms of the idiom and what is not? This project is precisely in that context, trying to clear a sort of "idiom extension" that explores, within a given idiom, areas non-affected by current musical practice, that is to say "manual", not computerized. The musical idioms that are of interest here (the "jazzed-up songs", bebop, biguine, funk, blues, French folk musette) have something in common: a beat and a harmonic scale that serves as a reference for improvisations. During the first experiments with the system, the musician played following a tempo and a scale provided by the computer (playing an accompaniment for example), the software recorded the positioning of the beat and harmonies of phrases played by Bernard Lubat allowing to reuse them by adjusting them to a different tempo and new progressions. From the early "acceptability" experiments there were problems regarding what the computer produced in terms

of stylistic standards and aesthetic values implicitly associated with the idioms used. The purpose of the study was to better understand the formal and aesthetic features of these idioms: validate and clarify the scope of the various features in the making ("recombination", "conformity", "anticipation", "hybridization"... see Chapter 4, Part I) and controls left to the musician-operator playing with the system (Chapter 14, Part IV). This section contains a number of assessments outlined by Bernard Lubat during the study. His wording is very colorful and rich in suggestive onomatopoeia, and the purpose is not here to do a text study. The aim is to outline his most recurrent themes when talking about the experiments with the system: phrasing, improvisation conduct, and rhythm. The collaborations with Bernard Lubat led to hours of recorded music and interviews which are impossible to reproduce here. The remarks quoted in this chapter are chosen among the most representative ones about the concerned issues.

(More transcriptions of interviews in French in Chemillier and Nika, 2015)

The following sections summarize the conception of the different playing modes designed in collaboration with Bernard Lubat between 2011 and 2013.

> This process being incremental, each of the "playing modes" described here benefits from the possibilities introduced by the previous ones. While, the examples given in the previous chapters are the results of combinations of all these features, we present here their incremental design to show their isolated impacts on the musical result through examples of work sessions using successive versions of the early prototypes.

17.2 Recombining and Phrasing

17.2.1 "Recombine": Digression and Conformity to the Scenario

> The elementary musical feature of the system is to "recombine" its musical memory. Each musical sequence produced by the model is a concatenation of discontiguous events or fragments retrieved in its musical memory. Before introducing richer generation processes and interaction possibilities, the very first experiments with Bernard Lubat consisted then in a sequential process: learn material from Bernard Lubat on a given chord progression and then generate new musical material from it, varying the parametrization of the model.

These first work sessions followed a sequential outline:

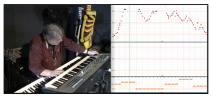
1. Bernard Lubat plays an accompaniment or a bass line on a given chord progression, often based on one of his own songs.
2. The system takes over playing an accompaniment generated from this material, often choosing high values for the continuity parameters in the generation model (see 5.1.4) to get an accompaniment track very close to what was played.
3. Bernard Lubat plays a theme and/or develops a chorus covering several occurrences of the chord progression.
4. The system takes over playing a new chorus generated from this material, often authorizing transposition and choosing low values for the continuity parameters in the generation model (see 5.1.4) to get a solo both conform to the chord progression and digressing from the original material.
5. These last two steps are repeated several times so that the musician and the machine trade choruses. Each time Bernard Lubat plays, the musical memory of the system grows.
6. Interview with Bernard Lubat after playing, and watching/listening to the recording of the session.

Video A.3.1

Hyperlink video *(or vimeo.com/jeromenika/improtek-archive-recombine)*
Description: *Appendix A.3.1.*

Video A.3.1 is a typical example of these early work sessions. At that time, continuity regarding the future of the scenario (see 4.2 and 5.2) had not been introduced yet, and the generation model was a first version of that presented in Part I. Among the different parameters of the model, the only continuity parameter was therefore the continuity regarding the past of the memory (see 4.2 and 5.3), which was enough to test co-improvisation on a given scenario using a memory learnt in the context of this same scenario. In this example, the accompaniment track is generated by the system from a memory learnt on Bernard Lubat's style, and we focus on the generation of a chorus. The maximum of the continuity parameter it set with a medium-high value (maximum = around a bar) so that fragments from the theme and from the choruses can be identified. The goal was indeed to play a sort of enriched "theme and variations". A piano roll represents the different parts: the first one is the solo played by Bernard Lubat, the second one is the chorus by the machine, and the third part is a representation of the pulse (dots: positions of the beats) and of the chord progression (dashes: roots of the chord labels).

Video A.3.2



Hyperlink video *(or vimeo.com/jeromenika/improtek-archive-recombine2)*
Description: Appendix A.3.2.

Video A.3.2 is an extract from another work session experimenting this very first feature. Bernard Lubat wanted to test the "reaction of the machine if [he fed] it with fast lines, to see it how it [could] dislocate it", using low values for the continuity parameter.

17.2.2 The Issue of Phrasing

The playing modes described in this subsection and the following are then incrementally added on the basis of the elementary feature of "recombination" presented above. Bernard Lubat validated the improvisations by the machine presented in 17.2.1, nevertheless he pointed out issues of phrasing in some other experiments.

Jazz phrasing often consists in introducing a kind of elasticity to the notes output in order to contrast with the regularity of the beat. This contrast is generally referred to as "swing" and is highly appreciated from an aesthetic point of view. It results in an uneven split of the beat in "long-short" referred to as "ternary", in which the connection between both parts is fluctuant. However, by construction, the software creates discontinuities in the sequence of notes originally played by the musician. These discontinuities may affect the fluidity and elasticity of the musical phrase. Here are Bernard Lubat critics from November 2011 referring to an improvisation on *All the things you are*, and illustrating this idea:

> "Strictly speaking, this isn't a rhythm problem, but the connection between the notes is not present. It goes tic-tac instead of deeee-ap [Bernard gives an example on the keyboard]. The upstrokes and downstrokes don't fall into place. The upstrokes and downstrokes start acting up, as soon as it starts gathering notes within the chord. It is not precisely "not in place", but more the associations that are too rigid."

We therefore had to move from the naive conception of the first "conformity" model (see Nika and Chemillier, 2012, and the remark in Section 5.3) to a second version including anticipation (Section 5.1 and 4.2), providing more consistency to these discontinuities.

The introduction of anticipation in the generation process brought a critical improvement of the musical result regarding phrasing, even before starting to work on "hybridization" (see 4.3 and Section 17.5 in this chapter). Indeed, it enabled to retrieve consistent sequences without recopying the initial material by using research (modulo transposition) instead of step-by-step filtering. This way, the system could for example take advantage of the recurrence of some harmonic sequences in different local tonalities in the memory to create consistent (while new) phrases. The phrasing problems were then partly

resolved, nevertheless they brought up the question of processing the “errors”.

17.3 Downstream Controls

17.3.1 Errors and “Trombination”

Often when the computer phrasing was faulty, the musician questioned his own phrasing reporting “errors” in his performance, meaning poorly articulated notes during the capture phase. In such cases, an improviser reacts to recover the error to his advantage by influencing the musical discourse so that the error, which was originally an unwanted default, appears as a desired effect. Later on we will see that Bernard Lubat values the presence of errors as an attempt to go beyond limits, including technical ones:

> “Instead of recombining, we should “trombine” [= distort, Lubat is playing with the word “combine” to describe the calculation process applied to sequences played by the musician]... As we do when our fingers miss or play an appoggiatura that we hadn’t plan and we make it work. It happens in every chorus, except for the bad ones which are perfect. That is the challenge, for the machine to really perform. It is not interesting when it comes to naive errors, whereas when the improviser makes errors, he “subjectivises”, he doesn’t apologize. Now the errors are doing my head in, it is just bad music.”

It then appeared essential that an operator needed to be able to act on sequences immediately in order to change things when the sequences calculated by the computer contained errors, “trombining” machine data as Bernard called it. Implementing basic commands such as intentional loops with a certain flexibility relating to the beat (see 13.4.2 and 14.1.2) helped set up a successful flow in terms of swing approved with Bernard’s input: “Now, the machine is in the groove “. However, not all the loops work that well and the procedure requires a certain understanding. While watching what Bernard Lubat wanted to do with the system, it became obvious that implementing commands enabling a musician-operator to steer the instrument was important.

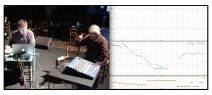
“Trombine”: the original neologism in french was “trombiner”.

17.3.2 "Downstream Controls": Interaction and Temporal Transformations

> The second step addressed what would become the end of the chain in the musical creation process of the system. That is to say what happens to the musical improvisations generated by the system between the moment when they are generated and the moment when they are played.

The direction was to introduce reactivity "downstream of the generation" to offer the possibility to use the system as a software instrument providing first basic controls on the "narration" at the expressive level to an operator-musician. We saw that an operator could control the generation parameters or the scenario itself (see 14.1.1), but we isolate here the *downstream* controls. As introduced previously (14.1.2), these basic manipulations consist for example in launching loops and accelerations in real time. They allow to create local riffs and slow/fast phrasing by applying online temporal transformations on the music generated by the system which is able to "land on its feet" when the control is released by keeping counting and being aware of the current performance time (see 13.4.2).

Video A.3.3



Hyperlink video *(or vimeo.com/jeromenika/improtek-archive-downstreamcontrols)*
Description: Appendix A.3.3.

In **Video A.3.3** for example, the accompaniment track is generated by the system from a memory learnt on Bernard Lubat's style, and the listening session focuses on the local temporal transformations. The learnt material is a single chorus played by Bernard Lubat on the standard *Autumn Leaves*. New material is generated with very high continuity to get a result close to a simple copy in order to submit to Bernard Lubat the results obtained through local transformations only. The transformations can be visualized in the lower part of the piano roll representing the virtual pulse (dots: positions of the beats) and chord progression (dashes: roots of the chord labels) of the voice being played, which are sometimes in advance, late, or shifted regarding the actual pulse and chord progression, depending on the transformations.

17.4 Reduction, Multiplication and Limits

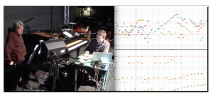
17.4.1 "Polyphonic Expansion", One to Many

> The very first generation model was oriented toward variation on a homogeneous memory, using a quite similar scenario. The downstream controls and the basic interface allowing the first transformations illustrated in the previous paragraph gave the

possibility to play with different voices getting in and out during the performance. Then, we experimented a playing mode of "polyphonic expansion" to create polyphonic improvisations from a single learning material. The musical aim was to be able to create a polyphonic texture from a single chorus.

Video A.3.4 shows a listening session focusing on "polyphonic expansion" following an improvisation with Bernard Lubat. His song *J'aime pour la vie*, whose chord progression is the scenario in this example, is a good field to experiment this playing mode since it is modal, is partly based on a pedal, and presents therefore numerous regularities in the harmonic progression (the global structure is *AB* with A: repeated chords, and B: based on harmonic walks).

Video A.3.4



Hyperlink video *(or vimeo.com/jeromenika/improtek-archive-poly-expansion)*
Description: Appendix A.3.4.

The lower part of the piano roll represents the accompaniment generated by the system from an accompaniment played by Bernard Lubat, and the upper part shows the polyphonic improvisation where all the voices (different colors) are generated from the same chorus. In this example, the maximum values for the continuity parameters of the different voices are set with low values (2 beats) to counterbalance the extreme regularity of the chord progression and to generate different voices with local echo effects.

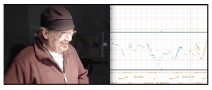
Bernard Lubat was particularly interested in this effect of multiplication that he wanted to use at multiple occasions in others work sessions and performances of "scat" co-improvisations with the audio version of the system (see 17.6.3).

17.4.2 "Monophonic reduction", Many to One

The previous examples had in common to use a material coming from one improvisation session, recorded within a given scenario, and to generate improvisations on this same scenario. Following the experimentations on the different ways a scenario and a memory could be articulated, the symmetrical playing mode to that of the previous paragraph is to create a monophonic improvisation using material coming from different memories.

This playing mode was the first basic achievement of the initial aim of the project, that is to improvising on a given chord progression by combining instant memory (musical material captured during the current improvisation session) and a deeper memory (musical material eared during anterior improvisation sessions). This first step towards hybridization introduces the idea of the musical memory seen as a store of different musical sequences. In this first case these sequences can be very different but "endogenous" to the scenario guiding the current improvisation session.

Video A.3.5



Hyperlink video *(or vimeo.com/jeromenika/improtek-archive-mono-reduction)* *Description: Appendix A.3.5.*

In the work session presented in **Video A.3.5**, Bernard Lubat plays a chorus on his song *D'ici d'en bas*, then the system successively generates three solos on the same scenario using a memory constituted by the chorus just played and a set of different solos from previous improvisation sessions. As is in the previous examples, the generated improvisations can be followed on a piano roll. The color of the machine improvisation track changes each time the solo used as memory for the generation model is changed.

The example shows three successive solos generated by the machine from a mixed online/offline memory (the live solo and a preloaded corpus). The musical intentions behind them are different: for the first one, the aim was to create a continuous melody using the different sequences in the memory. The following ones use transformations as that of the example in 17.3.2. Indeed Bernard Lubat asked us to deform the original material and make it "burst" or "go crazy" to get a non-human very fast phrasing in the counter-melody.

17.4.3 "Machinic Virtuosity" to Explore the Limits

Bernard Lubat looked for ways in which the machine would question the relation between the "machinic virtuosity" and the exploration of limits by playing phrases that could not be played by a human musician. Hence the first attempts toward this presented in **Video A.3.5** inserting counterpoints obtained by accelerating elements drawn from different recordings between theme phrases. Bernard's enthusiastic input became a long commentary in which he addressed the question of "pushing back limits".

> "It is great! Basically, the first improvisation [his own chorus] needs to have dramatic tempo and harmonies, relating to the metre and the harmonies of each measure. And there you have it, it improvises wonderfully, stays in the tempo, in the harmonies, and wanders throughout the limits. This is a first, getting to what the computer is playing. When I'm in a good mood, I like to do that, but it is very difficult: dismantling everything, all the while staying within the limits of phrasing framework, while pushing them at the same time... I've never heard that before, the dialectic between knowledge and lack there of, between limits and pushing to their extreme. It is dialectic, constant questioning, that is what I find most exciting with jazz, struggling against the framework, "the obstacle being a passageway" [expression used by poet and leading figure in theater A. Benedetto, friend of Bernard Lubat]."

The limits to which Bernard Lubat is referring to are those of the scale and beat within a musical idiom. Musicians often brush idiom

acceptability limits when they play, meaning they step away from the tempo and the grid chords (referred to as "in" and "out") making up for it with more or less agility, falling back on the beat and good harmonies. But the idea of exceeding limits also involves other aspects of improvisation. For Bernard Lubat, improvisation is very much an exploration of awareness limits in itself. This concept was already laid out regarding errors as indicated above. An improviser is confronted with the unexpected (that of his fellow musicians within the collective performance, but also his own mistakes) and must accept that he has not fully running the show of his performance. Part of what he plays eludes him and reveals the subconscious.

He came back to the issue of speed in February 2013. Not only does it open a door to the unconscious and the imaginary, the "self-doubting self" as Bernard Lubat calls it, but it also reflects a representation of individual time to a given culture and era. Bernard Lubat urged us to expand the possibilities of phrase transformation calculated by the computer to "annoy" the memory data, in order to "work on what we can disrupt" in the recombined phrases.

> "It raises questions seeing music play faster and faster as does the machine. We're overcoming boundaries. Before, we didn't go fast at that speed because it was not aesthetic, taste wise. While nowadays, guys "jump off of cliffs"... and finally manage to "land", it is astonishing. [...] When you go fast, you don't have time to think, it is going too fast. And all the while, you have to put something together. You have no time to think about what you are doing, a door is left open for the unconscious to slip through... Improvising means self invention, giving confidence to the self-doubting self."

17.5 "Hybridization"

In all the previous examples, the memory of the system was always chosen among previous sessions recorded within the same scenario than the current one or using live material. We did a lot of experiments using exogenous memory with the first version of the generation model, but its naive step-by-step process did not provide satisfactory results. As introduced previously (4.3), the introduction of anticipation mechanisms enabled to get more consistent "hybrid" improvisations. Even though the formal processes at play are the same, using the system to play with live material or using an exogenous offline memory almost makes it two different instruments from a strictly musical point

> of view. The musical directions that can be explored are therefore, depending on the project and on the chosen corpus, to make the musical memory fit a new context, or on the contrary make the scenario "go out of itself to go towards the aesthetics of the memory" (see also Section 18.2 with Hervé Sellin).

An interesting indicator was that, for the first time after many work and rehearsal sessions with the system, Bernard expressed the desire to deepen his understanding of the algorithmic mechanisms at the heart of the models during the first "hybridization" experiments. Although he was familiar with the musical directions and playing modes provided by the models in development, it is a solo created by the machine based on a solo that he had played on another harmonic progression that appealed to his scientific curiosity. Here is his take on a hybridization improvised by the computer on *All the things you are* from a solo recorded on *D'ici d'en bas* (**Video A.3.6**). The notes are not "strangers" to the harmony as with "playing outside": Bernard Lubat describes them as "strange" because he cannot find the logic of his own arrangements. Bernard Lubat highly appreciated this "strangeness", saying he was more excited by this result than those previously obtained using an endogenous musical memory.

Video A.3.6

Hyperlink video *(or vimeo.com/jeromenika/improtek-lubat-hybrid)*
Description: Appendix A.3.6.

> "That sounds great, and even if there are two or three strange areas, the notes aren't "strangers", they're strangers to the harmony, but great. Here, it is excellent. Is it using *D'ici d'en bas* [the chosen musical memory] ? Here it is very skillful, the chorus is outstanding. It is the stranger... the strange stranger ["l'étrange étranger, l'égaré qui est mal garé", B. Lubat plays on words, impossible to translate]... With something else, it is better than with the same... I will tell my mum! I don't understand, musically it did a superb chorus. It is not outside harmony, it is "weirdly musical". It is much better than what you come up with on the chorus I just did [previous transformation tests on "All the things you are" only using downstream controls]. There's nothing to add."

Video A.1.3

Hyperlink video *(or vimeo.com/jeromenika/improtek-lubat-early)*
Description: Appendix A.1.3.

Video A.1.3 shows an example of "hybridization" during the first concert performed with ImproteK at Novart Festival in Bordeaux (2013). The scenario is defined as a harmonic progression and the memory is a heterogeneous sets of different jazz standards and ballads coming from previous improvisation sessions with different musicians (in particular Bernard Lubat and Jovino Santos Neto).

17.6 Transversal Issues

17.6.1 Narration: Harmonically Free Experiments

One of the fundamental questions raised by improvisation is the conduct of musical discourse, meaning musical narrative. On this subject, Bernard Lubat approach comes down to the equation "narrative = composition", the difference being that unlike the composer, the improviser is not able to erase. We discussed the narrative issue in detail, carrying out non-idiomatic improvisation experiments because, according to Bernard Lubat, it was necessary to advance in this area and work the narrative by breaking free from idiomatic constraints in situations where "a basic model no longer exists, released from any ground. It is pure energy, within a infernal dialog". So we set up metric scenarios, then "only" pulsed scenarios to go in that direction.

> "I'm leading you but I'm led by what happens. It is like when you are on a raft, on a river. You are guiding the raft, but the river is guiding you. [...] You are drawn into waves. You don't need to overcome the waves because you use them, you sail above and within. There's a secret music sheet which is passing time. There's cultural time, which is domesticated, but there's another unknown time, that of the unconscious, time with no age [...] A narrative means that something happens in time. You move forward, but there are obstacles, encounters, bypasses, clashes, rests, absences. [...] One minute you are on the river, and the next, a rock. You need to have a comeback."

According to him, it is the paradox of the improviser: facing the unknown while looking into the future. He needs to push ahead even if he knows nothing about what "ahead" has in store for him:

> "The idea is that first, the projectile, the project, is put into motion. Improvisation is in motion, and you need to follow what is in motion, whatever the impact. That is why going back to the past with material we've been through is very difficult. Or it needs to be used in another way. But then it is too late, you can no longer use it, you are no longer in the same world, in the same time, in the same obstacle. [...] So you have to go "towards". And all this at the speed of light, so you have to cling to passing time."

It is interesting to mention the metaphorical consistency between these reviews and the remark made in a previous section: in fact,

during the experiments focused on hybridization, Bernard Lubat distinctly approved the machine's improvisations for which the ratio between continuity with respect to the future of the scenario / continuity with respect to past memory was in favor of the future. While the statements above primarily concern non-idiomatic music, we are catching a glimpse of their relevance regarding jazz improvisation based on a chord progression since the very concepts of "pace" and "resolution" call for a logic oriented towards future.

17.6.2 Scenario and Memory, Future and Past

One of the impacts that Bernard draws on the narrative plan is to never go back to material already treated. From a technical point of view, this meant offering the possibility for the generation process to browse through different areas of the memory and disable or not some areas already used.

> "It is paramount, it mustn't come back as a memory. We went elsewhere. You have no memory of the turmoil before, the rock before, because you have one before you."

However, the improviser does not start from nothing, he draws from his own past. Hence the (metaphorical) necessity to be able to call upon instant memory as well as an earlier and potentially heterogeneous memory.

> "We are a music sheet that is not entirely wrapped up, never performed the same way, always the same but never alike. Improvising is triggering this secret score, gradually growing over the years, like a garden. You don't improvise from nothing, you improvise from an accumulation of data that we put I don't know where: in know-how, in muscles and nerves, in your mind, in love, in sickness, everywhere. Some people think improvising is a matter of doing something you never have done before, that there is a moral. People who criticize my improvisations say "but at times it is always the same", they're looking for ethic purity, for a divine apparition, it is almost mystical. That is a load of rubbish, it doesn't exist."

17.6.3 Synchronization

One of the major difficulties regarding rhythm was synchronizing with the beat and preserving the "groove". We tested several state of the art beat detection (real-time) methods and used a specially developed module for the system so that the computer can synchronize its improvisations (see Bonnasse-Gahot, 2010; Nika, 2012).

According to these experiments, the overall result is good when the beat is detected in really "straightforward" music or when the phrases by the machine to be synchronized are quite "floating" regarding the beat. However, sometimes it is inadequate when trying to detect a beat on jazz drums with swing effects for example, or when the machine improvisations are very rhythmic (*e.g.* accompaniments) because each beat variation gives it an unbearable wobbly aspect. Moreover, these observations conceal technical issues and deeper ethnomusicology issues (Chemillier et al., 2014). The "beat" relies on the sensitivity of each individual. Within a group, it is drawn from a compromise, an adjustment between several individual subjective viewpoints.

The chosen solution was the modular architecture previously mentioned (Chapter 8), orchestrated by the Dynamic Score (Chapter 10) which synchronises the calls to the generation model and rendering processes with any external beat source, according to the musical project. Generation queries as well as audio rendering control messages are continuously accelerated or slowed down. These speeds are interpolated using the continuous estimation of tempo of the external source performed in the Dynamic Score.

During the collaborations with Bernard Lubat, the configuration we mostly used was the following: the initial tempo is given by the musician when he begins to play, and the computer locks onto this pulse before taking over.

> "How come it is so synchronous? [...] I had the feeling that everything was going around in a circle: what was happening, tempo, and what there had been before. I didn't even know who was doing what, what were the relations with the 'I': 'I' can do that, 'I' had nothing to do with this. While you do that [with the computer], I can do this: blow bubbles, I blow bubbles [he picks up a bubble pipe]. [...] I have no idea where that relay came from, it was flawless, I had the impression it was me playing."

Video A.1.4

Hyperlink video *(or vimeo.com/jeromenika/improtek-lubat-scat)*
Description: Appendix A.1.4.

Bernard accompanied the transition from midi rendering to audio rendering (Chapter 13). This way, after the period detailed in this chapter, we could transpose the work on phrasing and rhythmic articulation using another material: the sound of his voice in scat improvisation sessions such as that of **Video A.1.4**.

17.7 Conclusion

In addition to the validation of the successive versions of the models and playing modes, Bernard Lubat's set of "judgements of taste"

brought together substantial and creative material enabling an overall questioning of the concept of "beauty" in music under oral traditions. Although Bernard Lubat does not refer to "beauty" as such, his assessments outline what would be his aesthetic ideal, which can be considered "beauty" upon agreement.

Some assessments relate to objectively measurable characteristics such as tempo handover between the musician and the computer (*e.g.* there must be an objective agreement between the two tempos). But objectivity does not prevent from being bound by standards specific to an idiom. That was made clear concerning jazz phrasing. As soon as a specific idiom is referred to, highly restrictive acceptability standards that rely on certain aesthetic values associated with that idiom come into play. Going beyond a specific idiom, the desired features regarding the "beauty" idea could depend on a given culture or era, or stretch out universally. On the other hand, some aspects are tied to the fundamental nature of improvisation as a projection into the unknown, which gives them a universal scope. It is the same with recycling elements of improvisation from one to another because one cannot escape his own experience and memory.

The assessment that Bernard Lubat describes as "moral" in which the improvisation is to forget the past and start from nothing is only a "myth" to him. He also adds to the intrinsic nature of improvisation the fact that a musician should include his own mistakes in his discourse. But the essential concept of "error" for Bernard Lubat opens the door to a more subjective vision of beauty. As a matter of fact, the idea of beauty cannot remove irreducible subjectivity. Bernard Lubat emphasized that certain choices made by the artist were not reduced to idiomatic standards, or aesthetic value but pure subjectivity. According to him the performance speed of certain jazzmen for example, to the extent that they exceed the musician's monitoring capabilities, favors these unconscious telltale choices.

18

Collaborations with Expert Musicians

This chapter gives an overview of some other musical collaborations with experts musicians that were initiated in the framework of the thesis. As described in the previous chapter, the work sessions and performances were associated to listening sessions and interviews to collect some judgements about different aspects of the models and of the system.

The first two sections detail two long-term collaborations which lead to performances and concerts (a performance at Montreux Jazz Festival 2015, and two concerts in Italy and Paris in 2015 and 2016 respectively). Section 18.1 presents the work carried out with saxophonist Rémi Fox in two different situations: "traditional" jazz improvisation and "generative improvisation". Section 18.2 focuses on the project by Hervé Sellin and Georges Bloch using ImproteK to create a quartet constituted by Hervé Sellin (piano) and a virtual trio: Billie Holiday, Edith Piaf, and Elisabeth Schwarzkopf. Then, Section 18.3 presents the work initiated with Michelle Agnes Magalhaes in a context of structured contemporary improvisation. Finally, we briefly present earlier collaborations exploring different idioms ant types of interactions: Jovino Santos Neto (jazz and Brazilian music) in Section 18.4, Louis Mazetier (stride piano) in Section 18.5, Velonjoro, Kilema, and Charles Kely (Marovany zither) in Section 18.6, and the "Ateliers Inattendus" organized by philosopher Bernard Stiegler and IRI in Section 18.7.

18.1

Rémi Fox

FOCUSES

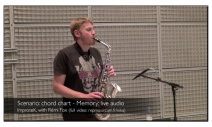
- Funk, jazz, and "generative improvisation".
- Improvisation using an online musical memory.
- Interface and controls: duet between a musician and an operator-musician.
- Rhythm, synchronisation, groove.
- Definition of an alphabet, a scenario and constraints.

SHORT BIOGRAPHY Rémi Fox is a saxophonist, improviser, and composer focusing on jazz and contemporary music. At the age of 25, he graduated in jazz and improvised music from the "Conservatoire National Supérieur de Musique et de Danse de Paris" (CNSMP). During his studies, he entered the department of "Generative Improvisation" directed by Vincent Lê Quang and Alexandros Markeas where he began to work on sound textures. Rémi always seeks to confront his music to other forms of art such as dance, video or visual arts, and frequently collaborates with IRCAM, CNC (National Centre for Cinema and the Moving Image) or theatres. The "nOx project" (different formations: nOx.3, nOx.6, and nOx.8) that Rémi started in 2013 offers a mix between jazz, free improvisation, and contemporary electronic music. In July 2015, "nOx.3" won the Tremplin Rezzo Focal of the "Jazz à Vienne" festival.

COLLABORATION In this section we focus on a performance at Montreux Jazz festival 2015 with Rémi Fox, and more particularly on the associated work sessions and repetitions. We worked with Rémi Fox to prepare two improvisations exploring two different directions: an improvisation over a very simple chord progression where the musician and the machine trade choruses on the top of a pre-recorded accompaniment track (18.1.1), and a "generative improvisation" using a form defined on a abstract alphabet as a scenario (18.1.2). In both improvisations, conceived as duets between the musician and an operator-musician playing with ImproteK, the software starts with an empty musical memory and improvises from the music recorded and learnt during the performance.

18.1.1 Dialog on "Rent Party" (Booker T. Jones)

Video A.1.2



Hyperlink video *(or vimeo.com/jeromenika/improtek-fox-rentparty)*
Description: Appendix A.1.2.

18.1.1.1 *Work Sessions and Performances*

The first part of the project is based on a very simple chord progression and played on the top of a playback accompaniment played by the system. The software starts with an empty musical memory and improvises by re-injecting the live audio from the saxophonist which is recorded, processed, and transformed online to match the scenario while being reactive to external downstream controls given to the operator. The scenario is the chord progression of *Rent party* (Booker T. Jones) segmented into beats[1]:

||: Cm7 Bb7 | AbMaj7 Bb7 :||.

Video A.1.2 shows a compilation of different sessions of this first improvisation. We tried different configurations: for the first tests, Rémi played alone at the beginning to let the system learn some material

1 |...| = a bar = 4 beats.

in order to be able to develop long phrases when it comes in. Then Rémi decided to make the system play from the very beginning to see its evolution during the performance: from basic repetitions to the development of a new musical discourse.

18.1.1.2 *Listening Sessions and Interviews*

Video A.4.1

Hyperlink video *(or vimeo.com/jeromenika/interview-fox-1)*
Description: Appendix A.4.1.

In addition to his validations and remarks at the time of the collaboration, we realized listening sessions of video recordings with Rémi Fox several months after the performance to analyze the repetitions taking a step back. Some extracts of the interviews are presented in **Video A.4.1**.

Rémi compared the interaction with the system to "a dialog": sometimes "a question / answer between [him] and [him]", sometimes a "fight". During the repetitions and listening to the recordings, he repeated several times that he "really had the impression that there were two saxophonists playing together." He added that he did not have the feeling that he was playing with a machine and that the experiment was "impressive [bluffant]". Furthermore, he found this interaction "stimulating" from a musical point of view but also because he wanted to see "how far the system could go". According to him, the next step should be to integrate the system in a band and use it within more collective dynamics. He emphasized the fact that he particularly enjoyed to play in duet with a musician-operator controlling the system.

At a particular moment of the session Rémi underlined a problem in the narrative aspect of the improvisation:

> "A real saxophonist would never do that, insisting that much on the first phrase [...] it is too heavy !"

Indeed, on such a repetitive scenario, a human improviser would have developed a long-term discourse and would have built an evolutive narration developed along the successive occurrences of the short chord progression. The advantage of having so many regularities in the memory and in the scenario (a repetition of the same four chords) is that the generation model has a lot of possibilities of recombinations and have a convincing phrasing. Nevertheless, it does not autonomously develop a horizontal musical discourse on four or five successive occurrences of the short chord progression. Even if this issue was isolated, we asked Rémi Fox his opinion on the future work to do in order to tackle this kind of issue regarding the narrative aspect: linking the reactive inputs of the system to reactive listening modules reacting to his playing, or providing finer controls to an operator playing with the system ?

Rémi was not really excited by the idea of "a clone" or of a "virtual improviser", but said on the contrary that he saw the system as a powerful "instrument" providing rich possibilities and encouraged us to go in this direction in future work. He repeated that he was really interested in the system used as instrument or an "effect unit" fully controlled by an operator-musician integrated in a band.

Rémi concluded on this first part of the project:

> "Regarding playing, we are almost there! The system really understands what I play, and does not play mechanically".

He added that working on the sound itself by adding audio effects (manually or integrating them in the generation processes) would be the last step to make it "more human".

Working on timbre is not the focus of this thesis, yet, this last remark is interesting since it reflects some expectations of the musician regarding the system, or human-computer improvisation in general:

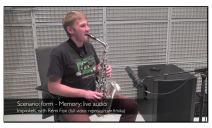
> "... make it more 'human', because I think this is what you want to do. Or at least I hope."

This remark expresses a point of view that differs, for example, from that of Bernard Lubat who was also interested in "machinic virtuosity" (17.4.3).

18.1.2 "Generative Improvisation" on an Abstract Form

18.1.2.1 *Work Sessions and Performances*

Video A.1.6



Hyperlink video *(or vimeo.com/jeromenika/improtek-fox-generative1)*
Description: Appendix A.1.6.

In the second part of the project, we designed the scenario as an abstract structure segmented into beats: ||: A_1 B_1 B_2 A_1 B_2 :|| with:

$$A_1 = || \,\mathrm{X}\,|\,\mathrm{X}^{+5}\,|\,\mathrm{X}^{-2}\,|\,\mathrm{X}^{+3}\,||$$
$$B_1 = || \,\mathrm{Y\,Z}\,|\,\mathrm{Z}^{+5}\,\mathrm{X}^{+3}\,|\,\mathrm{Y\,X}^{+5}\,|\,\mathrm{Z}^{+5}\,\mathrm{X}^{+3}\,|\,\mathrm{Y\,X}^{-4}\,|\,\mathrm{Y}^{+3}\,|\,\mathrm{Z}^{-5}\,\mathrm{Z}\,|\,\mathrm{Z}^{+5}\,\mathrm{X}^{+3}\,||$$
$$A_2 = || \,\mathrm{X}\,|\,\mathrm{X}\,|\,\mathrm{X}^{+5}\,|\,\mathrm{X}^{+5}\,|\,\mathrm{X}^{-2}\,|\,\mathrm{X}^{-2}\,|\,\mathrm{X}^{+3}\,|\,\mathrm{X}^{+3}\,||$$
$$B_2 = || \,\mathrm{Y\,Z}\,|\,\mathrm{Z}^{+5}\,\mathrm{X}^{+3}\,|\,\mathrm{Y\,X}^{+5}\,|\,\mathrm{Z}^{+5}\,\mathrm{X}^{+3}\,|\,\mathrm{Y\,X}^{-4}\,|\,\mathrm{Y}^{+3}\,|\,\mathrm{Z}^{-5}\,\mathrm{Z}\,|\mathrm{Z}^{+5}/\mathrm{X}^{+3}\mathrm{Y}||^{2},$$

where X, Y, Z are abstract equivalence classes and the exponents represent transpositions in semitones.

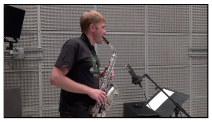
Video A.1.6 shows a first example of improvisation session based on this scenario. The idea was to play different voices with the system: a minimal and repetitive "accompaniment" on the top of which several "solos" got mixed up with that of the saxophonist. All these voices were generated from the live music recorded during the session.

2 |...| = a bar = 4 beats.

During the work sessions, Rémi enjoyed the fact that, with such an abstract scenario, it was possible to perform different improvisations over the same structure. Indeed, the aesthetics of the improvisation depended on what he "decide[d] to give to the machine at the beginning", defining "on the fly" what X, Y and Z were when they first occurred. After several tests, Rémi elaborated an improvisation plan:

1. Rémi plays alone on A_1 and the first measures of B_1,
2. a repetitive "accompaniment" voice is generated by the system from these first measures,
    - *Remark*: this voice is not a simple loop since it uses what Rémi played on A_1 and the first measures of B_1 to generate an accompaniment on the whole scenario (using transposition).
3. Rémi starts playing on this accompaniment,
4. several "solo" voices generated by the system come in after a moment using a memory which keeps growing as the saxophonist plays,
5. the tempo increases continuously during the performance (using a script in the Dynamic Score presented in Chapter 10 and the adaptive audio renderer presented in Chapter 13).

Video A.1.7

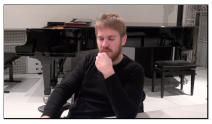


Hyperlink video *(or vimeo.com/jeromenika/improtek-fox-generative2)*
Description: Appendix A.1.7.

This process is illustrated in **Video A.1.7** by a second extract of improvisation session using the abstract scenario introduced above.

18.1.2.2 *Listening Sessions and Interviews*

Video A.4.2



Hyperlink video *(or vimeo.com/jeromenika/interview-fox-2)*
Description: Appendix A.4.2.

Some extracts of the listening sessions associated to this second part of the project are presented in **Video A.4.2**. Once again, Rémi judged that the musical result was "impressive [bluffant]": "if we close our eyes, we do not know who is playing, me or the machine".

Even more than in the first part of this project (18.1.1), Rémi said that he took the full measure of the "the interest of providing a structure to the machine [...] since its reactions are done in accordance with this structure". Rémi added that, paradoxically, he felt really free in this structured improvisation because he was not the only one in charge of the performance as it could be the case during other experiments he carried out with interactive music systems.

Since the system "knows the structure", records what he plays to transform it, and reorganizes it so that it matches the scenario, Rémi said that, from a certain moment, he could improvise "freely", relying on the system and reacting to what he heard, "without thinking things like 'there is a modulation in 4 measures' ".

Video A.1.8

Hyperlink video *(or vimeo.com/jeromenika/improtek-fox-generative3)*
Description: Appendix A.1.8.

Rémi was particularly enthusiastic when listening to the session presented in **Video A.1.8**: "it works, from a musical [and not only technical] point of view". This interview opened to a more general discussion about the role of structures and forms in his approach of improvisation, in particular with his band nOx, (see **Video A.4.2**), and led to a new project using ImproteK. This project will be decomposed in two phases: during a first period of experimentations with Rémi, we will test and prepare alphabets and compose scenarios. Then, the second phase will consist in repetitions with the band to prepare a set of performances.

18.2 Hervé Sellin

FOCUSES

- "Hybridization" (Section 4.3).
- Mixing offline and online memory.
- Improvisation plans.
- Defining properties of the alphabet (Section 6.1.1).
- "Music-driven" vs. "event-driven" interactions.
- Idiomatic music: jazz and deconstruction of the idiom.

Adapted from inter-jazz.com.

SHORT BIOGRAPHY Hervé Sellin was born in 1957 in Paris. He started playing trumpet, then trombone and did classical piano studies at the Conservatoire National Supérieur de Paris with Aldo Ciccolini. He obtained, in 1980, a double prize of piano and chamber music. During the same period his father, great French trumpet player Pierre Sellin, introduced him into Jazz. So he started playing with soloists such as Sonny Grey, Guy Lafitte, Gérard Badini, Eddie "Lockjaw" Davis, Joe Newman, Eddie "Cleanhead" Vinson, Harry "Sweets" Edison, Art Farmer, Barney Wilen, Clifford Jordan, James Moody, Chet Baker... In 1990 Sellin obtained the Django Reinhardt award from The French Jazz Academy for his activities as pianist, composer and arranger. In 1991 he met Branford Marsalis and recorded with him "Hervé Sellin Sextet featuring Branford Marsalis". From 1995 to 2000 Hervé toured with French drummer/composer Bertrand Renaudin playing on concerts and tours and recording three albums with him. In October 2003 Hervé was invited by Wynston Marsalis to play two concerts at The Lincoln Center of New-York with his tentet. In 2008 he released the album Marciac-New-York Express, by The Hervé Sellin

Tentet, and got the award for Best French Jazz Album of the Year given by The French Jazz Academy. Hervé works also full-time as a teacher at the Jazz and Improvised Music department of the “Conservatoire National Supérieur de Musique de Paris”.

18.2.1 Description of the Project

Hervé Sellin and Georges Bloch (using ImproteK) intensively worked with the system for 6 months, from the beginning of their project to the first concert in August 2015. This collaboration led to two concerts (August 21st 2015, Festival Internazionale “Pietre che cantano", L’Aquila, Italy; February 6th 2016, Conservatoire du sixième arrondissement, Paris, France) and a studio session to record the pieces.

18.2.1.1 *Memories*

The idea of their project “Three Ladies” was to create two improvised pieces played by Hervé Sellin and a virtual trio: Edith Piaf, Billie Holiday, and Elizabeth Schwarzkopf. In addition to the live music played by Hervé Sellin, the musical memory of the system was constituted by the following recordings:

- Billie Holiday: *The Man I Love*; *The End of a Love Affair*; *I’m a Fool to Want You* (*); *Saint Louis Blues*.
- Edith Piaf: *Mon Dieu*; *La Vie en Rose*; *Milord*; *Les amants d’un jour*.
- Elisabeth Schwarzkopf:
  - Mahler: Symphony No. 4, fourth movement, *Das himmlische Leben*; *Des Knaben Wunderhorn*, *Lob des hohen Verstands*; *Des Knaben Wunderhorn*, *Das irdische Leben*.
  - Mozart: *Don Giovanni*, *Mi tradi quell’alma ingrata*; *Le nozze di Figaro*, *Porgi Amor* (*).
  - Puccini: *Turandot*, *Tu che del gel sei cinta*.

All these musical memories were used during repetitions and will be exploited in future work. For the two above mentioned concerts, the pieces marked with “(*)” were not used.

18.2.1.2 *Scenarios*

The two different scenarios used in this project were the chord progressions of *Autumn Leaves* and *The Man I Love*. Hervé Sellin and Georges Bloch annotated the recordings mentioned above with harmonic labels, and used the possibility to define an alphabet and its properties (Chapter 6) so that the generation process could compare these labels with the jazz chord labels used to define the scenarios.

Video A.1.9

Hyperlink video *(or vimeo.com/jeromenika/improtek-sellin-themanilove1)* *Description: Appendix A.1.9.*

Hervé Sellin summarized the different approaches for the two pieces (see the statement of intent in Appendix C.2):

> "Somehow, *Autumn Leaves* is more of a chord sequence whereas *The Man I Love* is more of a story. That's for sure, even from an historical point of view. First the history is not the same, for *The Man I Love* there is a context, it has been written for a reason. *Autumn Leaves* went through all the colors of the rainbow. Of course, at first, it was a song, but it is not approached like that anymore, and the chord sequence we use is not exactly that of the original song. It is the chord sequence that the jazzmen seized. *The Man I Love* is different, it is another canvas which ties harmony together in a different way. So it generates other elements at every level."

Video A.1.10

Hyperlink video *(or vimeo.com/jeromenika/improtek-sellin-autumnleaves2)* *Description: Appendix A.1.10.*

Therefore, the chosen direction was to approach the improvised piece based on *The Man I Love* as a "horizontal story", and the improvised piece based on *Autumn Leaves* as a "vertical patchwork" (the improvisation plans designed for the pieces can be found in Appendix C.2.2). **Video A.1.9** and **Video A.1.10** show recordings of improvisation sessions respectively based on *The Man I Love* and *Autumn Leaves*.

18.2.2 Listening Sessions and Interviews

We present here some listening sessions we realized after the recording of the pieces in studio in October 2015. Hervé Sellin analyzed different takes of each piece to compare the different versions and discuss the integration of the system in the creative process. This method enabled to have a finer analysis and to see the evolution of the discourse of the musician. Indeed, the analysis of the recordings of the second piece (*The Man I Love*) refined his opinion on the first piece (*Autumn Leaves*). These interviews were realized using successive recording sessions from the least to the most "successful" according to Hervé Sellin.

Videos A.4.3 to A.4.8

Some extracts of these discussions are presented in **Videos A.4.3** to **A.4.8**, and transcriptions (translated into english) can be found in Appendix C.1. We briefly list below the comments that Hervé Sellin made about each session before presenting general conclusions in the following subsections (18.2.3 to 18.2.5).

Hyperlink videos *Description: Appendix A.4.3 to A.4.8. Transcriptions: Appendix C.1.*

AUTUMN LEAVES The first session was "not so bad" because "there was nothing out" but Hervé played in his "normal way" and did not have a lot of interactions with the system (see C.1.1.1). In this first session, the system played a bass line in addition to the "virtual singers". Hervé said it tends to "pulled him" to the actual standard and that the process of hybridization in his own playing was reduced because

of this accompaniment. The initial goal was to set a reference to play "in and out", but for the following sessions he decided therefore to play without this bass to go further into hybridization (see C.1.1.2). The second session, without the bass line, was "much more interesting" since Hervé saw the "development of a connection" between him and the system, "at least at the event level": "we can see a draft of the contacts that are being established" (C.1.2.1). Finally, the third session was "good" and the interactions between the musician and the machine produced "beautiful events" (see C.1.3), that made him "develop his own playing" instead of "reacting at the 'event level' ".

THE MAN I LOVE Hervé considered that the machine "did really better" in the first recording session on the piece based on *The Man I Love*, and that the improvisation was "much more into horizontality" (see C.1.5.1): "This is great. [...] Somehow, I feel much more comfortable here, and the elements fit together in a very nice way". He added that this session was "really beautiful and very subtle. In *Autumn Leaves* it was extremely boring, but here I find it very subtle" (see C.1.5.3). Finally, his conclusion about the second session was (C.1.6.4): "I am not playing an accompaniment. [...] Here, we propose something. Because, you see, this has a meaning for me. It could be another pianist playing [talking about the machine] and it has a meaning, it makes sense."

18.2.3 Discussion About the Role of the System

GUIDING THREAD AND ANTICIPATORY BEHAVIOR Hervé summarized his expectations regarding the system at the beginning of the project (see C.1.4.3):

> "It should become 'intelligent', structure itself within chord progressions, and not chose an element for a chord, then another for the following chord... The chord progression in *Autumn Leaves* is the famous *ii-V-I*. In *The Man I Love*, it is more complicated because it develops on eight bars in a row with a guiding thread which is an inner counter-melody. The machine should be able to handle it. [...] Playing on a theme such as *The Man I Love*, it is much more complex because the progressions are not standard."

Regarding the piece based on *The Man I Love*, the system met his expectations (see C.1.5.2):

> "I have the impression that I am playing with an orchestra: I react according to what is being played by the second pianist of the orchestra, then the singer comes back, the rhythm section comes back..."

The introduction of anticipatory behaviors (see Section 4.2 and Chapter 5) in the generation process led to "a nice organization in the musical space" (see C.1.5.4), in particular in this same piece:

> "It is indeed more long-term, in particular when the three ladies sing together (see C.1.5.5). And the tempo has to do with it too, necessarily a bar lasts longer. Regarding the elements retrieved from Piaf, Holiday, and Schwarzkopf, we have the time to ear 3 notes, 4 notes, a phrase... It fills you somehow."

INTERACTION At some point of the listening sessions, Hervé Sellin emphasized the fact that he was stimulated by the interactions with the system that triggered his "own mechanisms" at unexpected moments, and made him develop his "own playing" in a different way (see C.1.3.2):

> "It is funny because here I used a pattern which belongs to what I usually do, but what I heard from the machine made me develop it longer. Here I thought 'OK, well done !'. [...] I tried to uncover what belongs to my own playing, but I would not have done it without the stimulus of the machine. [...] Here, we are in the good story. [...] This is a good sequence, something that happened spontaneously. I played something which is not usual for me, and it created a beautiful event with what happened in the machine."

He introduced a distinction between "music-driven" reaction ("réaction à l'élément musical") and "event-driven" reaction ("réaction à l'élément évènementiel") to define his attitude regarding *The Man I Love* and *Autumn Leaves* respectively, and valued the "music-driven" reactions that the system triggered in his own playing in the case of *The Man I Love* (see C.1.5.3):

> "Here, my reactions are 'music-driven', and not really 'event-driven'. [...] A 'musical element', here, is for example when a piano plays something and I play something which is complementary, and then a singer arrives and she sings to me at least 8 bars so I know where to place myself..."

On the contrary, in the first session of *Autumn Leaves*, Hervé was not satisfied because he had the feeling that he was only "doing his job" and that it was "it sense of craftsmanship" that made him play more than a real interaction with the system (see C.1.1.4). Indeed, he was torn between the "permanent call to order" regarding the "invasive standard" imposed by the bass line played by the system (see C.1.1.2), and the "free patchwork" of the virtual singers. The "patchwork" was due to the fact that the chosen memory was quite small in

the case of the piece on *Autumn Leaves*. This underlines the fact, if proof were needed, that the choice of the memory in a corpus-based process is critical even if mechanisms dedicated to "hybridization" are designed. This aesthetics was intentional, nevertheless Hervé Sellin judged that the first sessions of the piece based on *Autumn Leaves* "sat on the fence".

HYBRIDIZATIONS The "hybridization" process in the case of *The Man I Love* was considered as "a success". As introduced previously, taking a step back, Hervé Sellin said about the first sessions on *Autumn Leaves* (see C.1.5.6 and C.1.2.2):

> "We approached [it] in the same way, because it is a song all the same. We often tried to bring the song out of it, and this may be the mistake."

> "I have mixed feelings here, because we play a quite known and famous standard which is a song with a form, and from a certain point, the standard may become invasive. [...] This is a version that we did not really do, that is to say, never playing the chords or the theme nor going into the form of the tune, but only reacting to sound and events."

Hervé discovered in the last sessions of the piece on *Autumn Leaves* that the system could be used to achieve hybridization "in the other sense", that is to say, making the scenario enter the universe of the memory, contrary to the approach of the piece based on *The Man I Love*. According to him, the interesting way to push on the experiment would be to play with the material itself, and the "strange harmonies" produced by the hybridization: "We should forget *Autumn Leaves* and say at the end 'It was *Autumn Leaves*' ! ". His conclusion on this piece was the following (see C.1.6.2):

> "I think it would be worthwhile to go through with this intention, including the 'patchwork' aspect. [...] We have to remove everything that does not help to go to an optimal result on the pretext of providing a false security [*e.g. the bass line*]. It is now obvious to me."

18.2.4 General Discussion About Improvisation

According to Hervé Sellin, the analysis of the improvisations with the system revealed some aspects of his own relationship with music improvisation in general (see C.1.4.1 and C.1.2.3):

> "It is good for me to watch this again. I see what works, what does not. I can see when I totally screwed up in my own habits, my own contradictions, my fears... "

> "Here we meet with something else, my history, my training, the way I work. I am quite pragmatic, and quite traditional in my modus operandi: I like the themes, I like when it is straight, I like when it is precise, I like nice harmony... I am not mad about 'discovering' or wild improvisation, even if I know how to do it. [...] I come from a very strict and 'well-defined' background. I took me years to be able to do this [showing the video]."

He added that these experiments underlined his natural tendency to search for "complementarity" in improvisation that he attributes to his "craftsman / Good Samaritan" side (see C.1.3.1). Going further, he added (see C.1.2.4):

> "I thought about that: I know why it works a little with me. It is because my approach of music is that I never listen to myself. I listen to the others, and then it generates reactions within me. These reactions are consistent thanks to my knowledge, but I absolutely never ask myself things like 'I heard that, so the best thing to play is E natural...' [...] In a jazz orchestra, whatever it is, my first preoccupation is to listen to the others and to play according to what I ear. There is not a single moment when I decide alone what I am going to do."

According to him, these experiments with the system were a way to go beyond this complementarity (see C.1.5.4):

> "There is still work to do, to tend to something a little more original, given the fact that the machine sends unexpected information. [...] When something happens either you think "I am the guard, I have to ensure safety" or you play the game and you think "what do I suggest ?" "

Hervé concluded that he wanted to go further in the experiments to surprise himself (see C.1.6.1 and C.1.6.5):

> "I also want to surprise myself and to meet the level of demand of such a project. I know what I am able to do, but what is exciting is to look for something else."

> "If it is nice, successful, interesting and flawless in the end, then it is great. But here the goal won't be reached once it has been nice or flawless."

18.2.5 Perspectives and Future Collaborations

The current dynamics is to capitalize on this work to follow on this project integrating live video improvisations (see Section 14.2). In addition, Hervé suggested to work on modal improvisation with the system (see C.1.2.5 and C.1.6.3):

> "In this case the system would discover much more possibilities, and everything would be multiplied by time. We should work on both aspects: [...] see what the machine can do and what the musician can do, then analyze the difference, it must be easy to do."

> "Trying to work on a more continuous material while being in a very vertical chord progression and a fast tempo may be interesting too as a research experiment regarding the machine. The same way, on a modal scenario we would have to see what happens if we try to take the opposing view. We have to try all directions, and then have various possibilities."

Finally, Hervé Sellin concluded that:

> "The interest [with the machine] is to play: I could write 10 pages of music and it would be monstrous because I would exactly know what to do regarding the harmony, the melody... This is not the interest, the interest is to find in real time." (see C.1.6.6)

18.3 Michelle Agnes Magalhaes

FOCUSES

- Improvisation using an online musical memory.
- Non-idiomatic composed improvisation.
- Content-based alphabet.
- Scenario: discretized multimodal profile of audio descriptors.

SHORT BIOGRAPHY Michelle Agnes Magalhaes holds a Doctorate in music from the University of Sao Paulo (USP, Brazil), and a Master degree from the University of Campinas (UNICAMP, Brazil). Her musical studies and activities include composition, free improvisation, piano and musicology. Michelle Agnes' musical formation began on the piano. Since then, this instrument played a key role in her development as a musician, improviser and composer. She started studying composition in 1994 and continued her studies at the University of Campinas. She worked as composer in residence at IMEB (Institut International de Musique Electroacoustique de Bourges, France), and between 2009 and 2011, she played in many concerts of free improvisation music with the Abaetetuba Collective and began to play

Adapted from michel-leagnes.net.

in duo with the bassist Celio Barros. She moved to France, in 2013, to join the team "Analysis of musical practice" at IRCAM. Michelle Agnes worked with a variety of international ensembles, and her music has been performed in many festivals[3]. She is currently working with the Musical Representations team at IRCAM.

COLLABORATION We carried out experiments with composer and improviser Michelle Agnes Magalhaes who works on structured improvisation. The basis of these experiments were her previous piece *Mobile* for prepared piano [4], a tribute to John Cage, inspired by his *Sonata IV for prepared piano*.

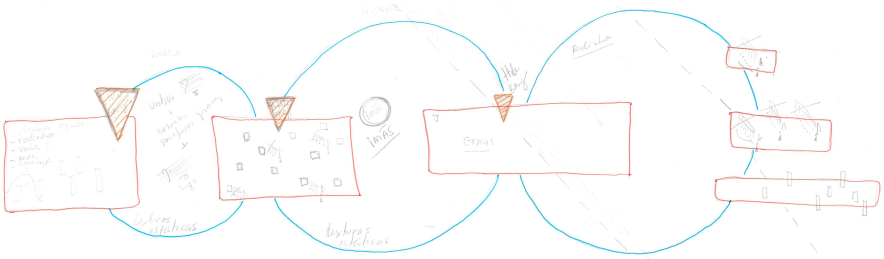
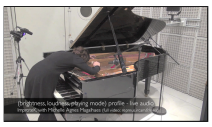


Figure 18.1: Improvisation plan, *Mobile for prepared piano*, Michelle Agnes.

Michelle considers *Mobile* as a "piece" as well as an "improvisation plan". She conceived it using a general improvisation plan (18.1) as those described in Section 10.4.2. This plan is divided into three parts with different scenarios describing precise temporal evolutions of energy, register and timbre, and playing modes. We translated these scenarios using a content-based alphabet: 3−uples describing the loudness, the brightness, and the playing mode. This work illustrates the case where the scenario only describes the part of the machine improvisation (see 6.1.1).

Video A.1.5



Hyperlink video *(or vimeo.com/jeromenika/improtek-agnes-composed)*
Description: Appendix A.1.5.

As in the example of **Video A.1.5**, the system re-injects the live audio material matching the playing modes and descriptor profiles imposed by the scenario. The first technical tests we carried out initiate a future project using a scenario which will be composed in such a way that the machine alternates between counterpoint and extension of the musical gesture of the musician.

The main limitation of the current version of the system we will have to cope with to pursue this work is a simple matter of implementation. Indeed, for the moment, the events in the memory are considered as beats since we mainly focused on pulsed music. We had therefore to "hack" the system to realize these experiments, and we will have to introduce properly a representation of events with different relative durations.

3 See www.michelleagnes.net.
4 https://soundcloud.com/michelle-agnes/michelle-agnes-mobile-for-solo

18.4 Jovino Santos Neto

FOCUSES

- Jazz and Brazilian music.
- Improvisation using an online musical memory.
- Harmonization and arrangement (Section B.1.2).
- "Hybridization" (Section 4.3).
- Rhythmic phrasing.

SHORT BIOGRAPHY Jovino Santos Neto is a Brazilian pianist, composer and arranger. Currently based in Seattle, Washington, he has throughout his career been closely affiliated with the Brazilian master Hermeto Pascoal. He was an integral part of Pascoal's group from 1977 to 1992, performing around the world and co-producing several records. Currently, Jovino leads his Seattle-based Quinteto and teaches piano and composition at Cornish College of the Arts. He can also be heard as a piano soloist, working with symphony orchestras, jazz big bands, chamber music groups, and in collaboration with musicians such as his mentor Hermeto Pascoal, Bill Frisell, Airto Moreira, Claudio Roditi, David Sanchez, Joe Locke, Marco Granados and many more. Since moving to the US from his native Rio de Janeiro in 1993, Jovino Santos Neto has continued to tour the world and to record prolifically (Canto do Rio, 2004; Roda Carioca 2006; Alma do Nordest, 2007; Piano duo with Weber Iago, 2008; Veja o Som, 2010; Corrente, 2011; Piano Master's Series Vol. 4...). His compositions have been performed by the Seattle Symphony, NDR Big Band in Hamburg and by numerous chamber music groups. Jovino gives lectures, clinics and master classes worldwide on the music of Brazil.

Adapted from jovisan.net.

COLLABORATION On the one hand, we first worked with Jovino Santos Neto in May 2013 to experiment co-improvisations with the first MIDI prototype of the system in an idiomatic context, in particular jazz and Brazilian jazz ballads as shown in the short excerpt in **Video A.5.1**. As described in the first experiments with Bernard Lubat (Chapter 17) the outline of each improvisation session was the following:

1. Jovino Santos Neto played an accompaniment on a given chord progression.
2. The system took over playing an accompaniment generated from this material.

Video A.5.1

Hyperlink video *(or vimeo.com/jeromenika/improtek-jovino-balaio)*
Description: Appendix A.1.9.

3. The musician played a theme and/or developed a chorus covering several occurrences of the chord progression.
4. The system took over playing a new chorus generated from this material, often authorizing transposition and choosing low values for the continuity parameters in the generation model (see 5.1.4) to get a solo both conform to the chord progression and digressing from the original material. In addition, we used the "downstream controls" introduced in Section 14.
5. These last two steps were repeated several times so that the musician and the machine trade choruses.

On the other hand, Jovino validated the model of automatic harmonization and arrangement mentioned in Section B.1.2. We submitted to him different accompaniments of jazz solos in the style of Hermeto Pascoal - using the *Calendário do som* (Pascoal, 2000) as corpus for learning - that he judged "plausible".

In May 2015, we carried out new experiments focusing on rhythmic phrasing in the case of "hybrid" improvisations (see Section 4.3, Part I). The idea was to use corpora of different recordings with a same metric structure, but presenting different phrasing styles (*e.g.* some had a "ternary" phrasing and others had a "bossa" phrasing). This study revealed some difficulties when mixing these different styles in terms of rhythmic articulation. Indeed, it underlined the fact that, when the musical goal is not to use "hybridization" to go out of the idiom (see 18.2) but to create a continuous discourse from exogenous material, the design of the algorithms cannot prevent from taking care of the choice of the corpus when the chosen alphabet is only constituted by harmonic labels. Annotations regarding the type of rhythmic articulation should have been used as filtering constraints, (Section 6.2, Part I) or included to the alphabet used to label the memory and define the scenario (Section 6.1, Part I).

18.5 Louis Mazetier

FOCUSES

- Stride piano.
- Mixing offline and online memory.
- "Hybridization" (Section 4.3).
- Scenario defined on an alphabet describing harmony and macrostructure (Section 6.1.1).
- Secondary generation parameters (Section 6.2).

Marc Chemillier carried out some experiments with Louis Mazetier[5], who is an internationally renowned composer, pianist, and improviser, specialist of pre-be bop piano, and in particular stride piano.

Roughly summarizing, the musical forms associated to this style relate to that of early ragtime and are generally based on nested themes more than on the classical "32 measures" outline. Therefore, the scenarios were described using harmonic labels but also other structural labels such as "introduction", "theme 1", "transition 1", "theme 2", "conclusion", "coda", etc. Furthermore, stride piano involves very contrasted and figurative playing modes (contrary to the linear post-bop style of Bernard Lubat for example). This style alternates between phrases played with full chords, fast motives repeated in low or high registers, ascents or descents of the keyboard, etc. Additional labelling of the memory and secondary generation parameters (Section 6.2) were thus used to avoid "patchwork" recombination and to be able to chose the playing mode during the performance.

Video A.5.2

Hyperlink video
Description:
Appendix A.5.2.

Video A.5.2 shows a demonstration by Louis Mazetier (piano) and Marc Chemillier using the MIDI version of ImproteK. The musical memory of the system is a set of transcriptions from the "stride" repertoire. The scenario is the chord progression of *Handful of keys* (Fats Waller) using an alphabet constituted by chord labels and labels describing the macrostructure (such as "introduction", "theme 1", "conclusion..."), and secondary generation parameters to define the successive playing modes.

18.6 Velonjoro, Kilema, and Charles Kely

FOCUSES

- Marovany zither, contrametricity.
- Rhythmic articulation.

Marc Chemillier studied the use of the system in the context of the music played with the Marovany zither, a traditional instrument of Madagascar. From a humanistic perspective, the motivation for studying this particular instrument was its cultural importance due to its association with a possession ritual called *tromba*. This project involved three musicians, experts of this instrument: Velonjoro, Kilema[6], and Charles Kely[7]. This work used the first MIDI prototype of the system in collaboration with the LAM laboratory (Lutherie, Acoustics, Music, UMR 7190 CNRS, Paris) which conceived an original optical-based retrieval system dedicated to the Marovany zither (Cazau

5 See http://www.francemusique.fr/personne/louis-mazetier.
6 See www.aido.fr/kilema.htm
7 See www.charleskely.com

et al., 2013). The aim of these collaborations was to get a validation to assess the credibility of the improvisations generated by the system in this musical context characterized, among other things, by a very fast tempo and a strong contrametricity (Chemillier et al., 2014).

Video A.5.3

Hyperlink video
Description:
Appendix A.5.3.

Video A.5.3 shows the first experiments realized by Marc Chemillier with Velonjoro in Madagascar in July 2014. The validation by the musicians, in particular regarding the rhythmic articulations, led to several public demonstrations, and in particular a trio performance in June 2015 at the International Workshop on Folk Music Analysis by Charles Kely playing Marovany zither, Kilema playing Katsa (percussion instrument made of a tin), and Marc Chemillier using ImproteK.

18.7 "Ateliers Inatendus"

The approach exposed in the previous sections underlined an important aspect of the work presented in this thesis: attempting to catch some aspects of music improvisation through computer modeling. In this view, we took part in the festival "Rencontres inattendues 'musique et philosophie' de Tournai" organized by philosopher Bernard Stiegler and IRI (Institut de Recherche et d'Innovation)[8] in the framework of the project "Mons capitale de la culture 2015". This project addressed the questions of improvisation and annotation / categorization in music and philosophy.

ImproteK played an important role in the "Ateliers Inatendus" ("Unexpected Workshops"): from October 2014 to August 2015, a "traveling school" was organized between Belgium and France with workshops in Mons, Lille, and Tournai, attended by numerous amateur and professional musicians. The workshops associated sessions of collective musical practice orchestrated by Bernard Lubat, Fabrice Vieira and guests (such as Michel Portal), and sessions of collective discussions with interventions of philosophers and musicologists and ethnomusicologists (such as Bernard Stiegler, Jean During, Pierre Sauvanet and Yves Citton). This project ended with a summer school in Tournai during Tournai Jazz Festival 2015.

The scenography of the successive workshops was centered on a visual representation of ImproteK inputs and outputs displayed on a screen in the middle of the stage (see Figure 18.2) destined both to the musicians and to the audience. The system was integrated in a network of devices introducing new interactive practices for the musicians and enabling the audience to take part in an active listening process through annotations of the improvisations (see Figure 18.3).

8 See http://penserimproviser.org/wp/.

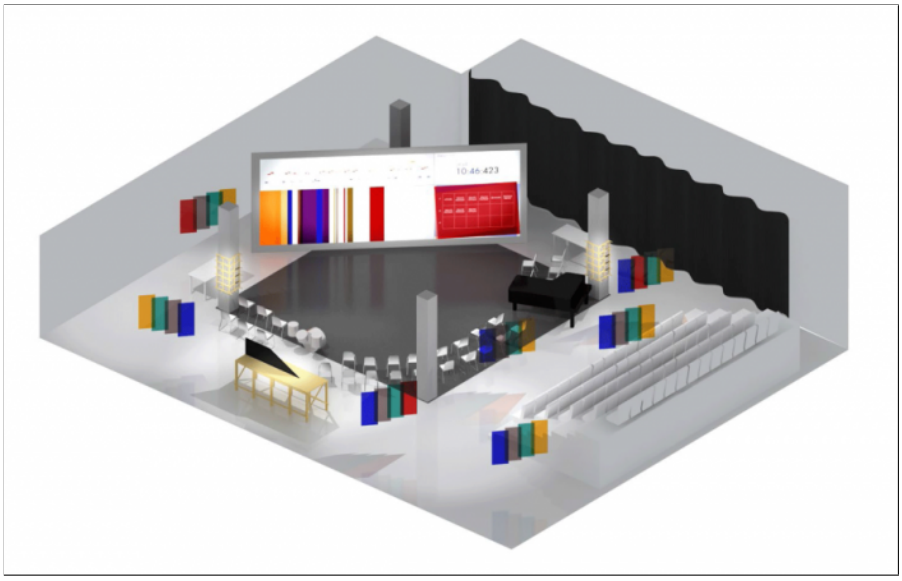


Figure 18.2: Scenography “Ateliers Inatendus” by Gaëtan Robillard and Isabelle Daëron (Source: penserimproviser.org).

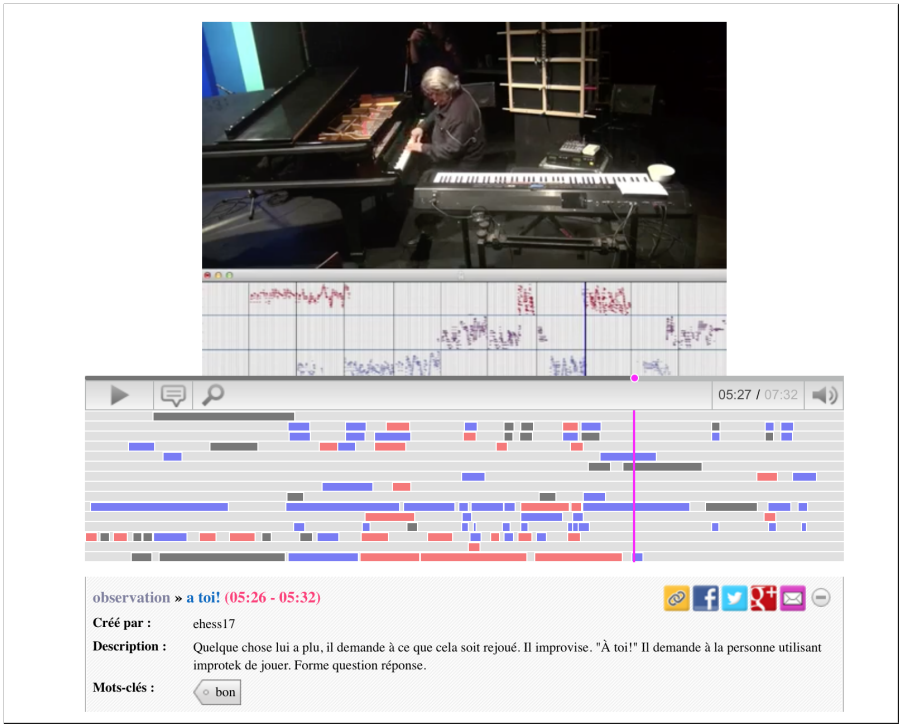


Figure 18.3: Example of annotations of an improvised performance (Source: penserimproviser.org).

Part V

CONCLUSION

19

Conclusion

19.1 Summary and Contributions

What makes our approach of human-computer music improvisation particular is its temporal specification articulated between a *scenario* and a *memory*. The *scenario* is a symbolic sequence guiding the improvisation and defined on an appropriate alphabet depending on the musical context (*e.g.* a chord progression). The *memory* is a sequence of musical events where each element is labeled by a symbol belonging to this same alphabet (*e.g.* solos played by a human co-improviser).

19.1.1 Guided Music Generation Processes

Part I

In first approach, the consistency between scenario and memory ensures the conformity of the improvisation generated by the machine regarding the stylistic norms and aesthetic values implicitly carried by the idiom of the musical context. In second approach, the scenario gives access to a prior knowledge of the temporal structure of the improvisation which is exploited to introduce anticipatory behavior in the generation process. This way, the future of the scenario is taken into account when generating the current time of the improvisation.

The generation process we propose is divided into successive generation phases constrained by nonconsecutive suffixes of the whole scenario (*i.e.* what remains to be played once the previous phase is ended). Thanks to this design, the model can be queried using temporal queries (portions of scenario) that enable to generate anticipations ahead of performance time when the model is used in a real-time context. In a generation phase, each step ensures both continuity with the future of the scenario and continuity with the past of the memory.

The scenario / memory approach is generic, that is to say formally independent of the chosen musical alphabet. We developed a protocol to compose improvisation sessions. In this framework, musicians for whom the definition of scenarios is part of the creative process can be involved in a meta-level of composition by designing an alphabet and its properties, equivalence classes, associated transformations of the contents, etc.

19.1.2 Combining Planning and Reactivity

Part II

We introduced the paradigm of modeling guided improvisation as dynamic calls to offline models relying on a temporal scenario. Intrinsically offline processes are embedded into a reactive framework, out of the static paradigm yet not using pure last moment computation strategies. It results in a hybrid architecture dynamically revising previously generated data ahead of the time of the performance in reaction to the alteration of the scenario or of other reactive inputs. Anticipations matching the scenario are therefore represented by sequences outputted by the generation process when it is called in time during a live performance. This way, reactions are not only seen as instant responses but have consequences over time. This is achieved by chaining two agents: an Improvisation Handler, a reactive agent embedding the scenario / memory generation model, and a Dynamic Score, a reactive program modeling the decisions taken at the interface between the musical environment and dynamic guided generative processes.

The Improvisation Handler reacts to dynamic controls by composing new midterm anticipations ahead of performance time. It reacts to a modification of its reactive inputs by rewriting previously generated anticipations while maintaining coherence when overlaps occur. The Dynamic Score implements a hierarchy of parallel processes listening and reacting to the environment and the elements generated by the models. It is involved simultaneously upstream and downstream to coordinate the generation queries and rendering of the associated outputs in due time.

19.1.3 Playing on Time and with Time

Part III

We proposed two rendering architectures coping with dynamic musical sequences which are revised during the rendering, and presented associated expressive musical controls. The architecture described in Chapter 15 is an autonomous renderer conceived to be integrated in a framework of composition of musical processes using an offline memory and driven by the internal time of the musical material. The architecture described in Chapter 13 and Chapter 14 is dedicated to performance and used in the system ImproteK. It proceeds to the elastic temporal mapping between a symbolic improvisation and the real time of performance, and offers downstream musical controls.

This performance-oriented architecture achieves adaptive rendering of dynamic multimedia sequences generated from live inputs. It is conceived to record and segment a live stream into beat-events that can immediately be played in synchrony with a non-metronomic pulse, according to a user-defined dynamic time mapping. It interleaves event-triggered and adaptive time-triggered mechanisms para-

metrized by the tempo of an external beat source. Different voices can be defined with different musical memories and different time references. Therefore, complex time relations between different voices and between inputs and outputs can be defined.

We showed that the generation models and the reactive architecture could run autonomously, but also be used as a software instrument. We gave an overview of different types of commands to take direct action on the music during a performance. Upstream controls give declarative controls on the intentions querying the generation model and introduce the idea of DJing dynamic music processes. Downstream controls happen between generation and rendering and consist in the live alteration of what the computer plays (*e.g.* online temporal transformations such as agogic accents, loops and accelerations) while keeping the time reference provided by the external beat source.

19.1.4 Computer Music Tools and Scientific Collaborations

Appendix B

This work led to the conception of self-consistent architectures addressing different issues in human-computer improvisation.

We implemented as CommonLisp libraries in the OpenMusic environment (Bresson et al., 2011): an offline music generation model guided by a scenario structure; a dynamic agent embedding an offline generation model to generate an evolving musical sequence, combining anticipatory behavior and dynamic controls; and an autonomous renderer dedicated to compositional processes.

We implemented in the graphical programming environment Max (Puckette, 1991) using the score follower Antescofo (Cont, 2008a) and its associated programming language (Echeveste et al., 2013a,b): a high-level framework to conciliate various improvisation strategies and scheduling generation as well as rendering; and an architecture for adaptive rendering of multimedia sequences generated from live inputs according to dynamic user-defined time mappings and synchronized with a non-metronomic pulse.

The work presented in this thesis was used as a motivation and an application case to conceive new features of the Antescofo programming language such as temporal variables (Echeveste, 2015), and to design the new scheduling engine of the OpenMusic environment (Bouche et al., 2016).

19.1.5 Continuous Interactions with Experts Musicians

Part IV
Appendix A

The proof of concept simply is the numerous concerts we have given and the enthusiasm generated among musicians. The system ImproteK, implementing the models and architectures presented in the thesis was used at various occasions during live performances with

expert improvisers (more than 15 performances between 2012 and 2015, among them: Novart Festival, Bordeaux, France, 2013; “Mons Capitale de la culture” Festival, Belgium, 2015; Montreux Jazz Festival, Switzerland, 2015; Uzeste Festival, France, 2015; “Pietre che cantano” Festival, L’Aquila, Italy, 2015.)

This thesis was a constant back and forth between music and science and was developed in continuous interaction with experts musicians in order to validate and refine the scientific and technological choices through performances, residencies, work sessions, listening sessions, and interviews. Through numerous collaborations, we were able to experiment different idioms and types of interactions: Bernard Lubat and “La Compagnie Lubat” (“jazzed-up songs”, jazz, and free improvisation), Jovino Santos Neto (jazz and brazilian music), Kilema (marovany zither), Velonjoro (marovany zither), Charles Kely (jazz and marovany zither), Louis Mazetier (stride piano), Michelle Agnès Magalhaes (contemporary music), Rémi Fox (jazz, funk, and generative improvisation), Hervé Sellin and Georges Bloch (jazz and contemporary music).

We presented in Part IV the validations and discussions of the musical dimensions we studied from the judgments of these experts. Beyond validation and discussion, these interactions were intended to move from “simulation” to “stimulation”. We exploited the successive versions of the models, seeking then to divert them and perpetuate these diversions in order to participate in the creative processes of the musicians. These collaborations led to hours of filmed and documented music sessions, listening sessions, and interviews. The discussions about the models and the successive prototypes of ImproteK led to broader discussions about music improvisation in general, and the analysis of the interactions with the system served as a starting point for some musicians to develop some thoughts about their own approach of music improvisation.

19.2 Perspectives

19.2.1 Scenario and Reactive Listening

The architecture model we presented to combine scenario and reactivity proposes an answer to the question “how to react?”, but does not address the question “when to react and with what musical intention?”. Indeed, the model defines the different types of reactions that have to be handled and how it can be achieved. It choses to offer genericity so that reactions can be launched by an operator using customized parameters (this approach was valued by several musi-

cians we worked with), or by a composed reactivity defining rules specific to a particular musical project.

Combining our work with reactive listening could enable to have reactions launched from the analysis of live musical inputs, and therefore fully combine the guiding "follow that way" and guiding "step by step" paradigms. The Master's thesis of Axel Chemla-Romeu-Santos, that I supervised at Ircam with Gérard Assayag, is a first step towards this direction (Chemla-Romeu-Santos, 2015). This work sketched an architecture guiding the machine improvisation along two different musical dimensions by associating an anticipation module based on the models and architectures presented in this thesis, and a reactive listening module inspired by Somax (see 2.2.1). In this framework, for example, a scenario can be defined as a chord progression, and reactive listening can be performed on energy descriptors. Then, during the performance, different processes associated to the different modules (anticipation regarding the scenario, reactive listening, self-listening) continuously modify the activity of the musical memory to retrieve and play the most relevant events on the fly.

19.2.2 Inference of Short-Term Scenarios for the Future

In the same line, another promising direction is that of the inference of short-term scenarios. The models we proposed are queried by successive "current scenarios", that is to say subsequences of a scenario *defined* before the performance and that can be dynamically modified. These temporal queries could as well come from an analysis of the playing of a musician. Reactive listening would not only trigger an instant response, but also discover underlying formal structures in what the musician plays. Then, these underlying structures would be used to infer short-term scenarios for the future. Finally, the models we propose could be used to generate anticipations from these *predicted* structures.

Such a process could enable to take advantage of the anticipatory behavior we propose and introduce directionality in the music generated by the machine without informing it of any prior knowledge regarding the temporal structure of the improvisation. This way, the successive temporal specifications given to the system coming from predictions, our work on "*specification* $\rightarrow$ *anticipation*" could take part in the chain "*expectation* $\rightarrow$ *prediction* $\rightarrow$ *anticipation*" mentioned in introduction.

19.2.3 Creative Perspectives

COMPOSITION OF COMPLEX TIME RELATIONS In Chapter 13, we actually presented a particular use of the performance-oriented rendering architecture in which we used a common time reference

for the musical inputs and outputs, using the "beat" as a common subdivision. In this case, the processes handling the inputs and those handling the outputs listen to a same temporal variable: an external beat source. If inputs and outputs are linked to different time references, this architecture offers a simple and declarative way to define complex time relations between musical voices generated in real time from live inputs. For example, real-time tempo canons such as that of Trapani and Echeveste (2014) inspired by Nancarrow's canons (Gann, 2006; Thomas, 2000) could be easily realized by defining the time reference of the outputs as a tempo curve applied to the time reference of the inputs.

CHAINS OF IMPROVISING AGENTS Considering the genericity of the alphabets used by the scenario / memory generation model and that of the reactive architecture, future works will focus on chaining improvising agents. A preliminary study of such chaining will use an ascending query system through the tree of plugged units to avoid data depletion, and message passing scheduling between multiple agents to ensure synchronization. A first application case could be to work on a realtime version of the early work on harmonization and arrangement mentioned in Appendix B.1.2, or to cope with other types of musical vertical associations using chains of improvisers working on different alphabets.

ADAPTIVE SOUNDSCAPES Other perspectives suggest to make use of such guided and reactive music generation to produce evolving and adaptive soundscapes, embedding it in environments generating changing parameters while including a notion of plot. In interactive installations for example, a composed dimension could be defined as a scenario in association with an interactive dimension using the reactive inputs of the architecture.

MUSICAL PERSPECTIVES As mentioned in Part IV, new projects using our work were launched by the musicians we worked with, and will see the day in the near future. Other musical collaborations will be initiated to experiment guided improvisation in various musical contexts, from improvisation on jazz standards to composed or controlled improvisation situations where defining an alphabet and a grammar is part of the creative process[1]. Among them, the results of our researches will be used in an ambitious creative project in association with Montreux Jazz Festival around Montreux Digital Heritage archive: 5000 hours of concert in audio and video, listed as UNESCO World Heritage.

1 *e.g.* the project "Secret Heroes" with Benoît Delbecq, Jozef Dumoulin, Doctor Bone a.k.a Ashley Slater, and Gilbert Nouno (Premiere on June 22nd 2016, Grande Salle, Pompidou Center, Paris) where three instances of ImproteK are used with scenarios defined as text by beat generation writers.

APPENDIX

A

Videos referenced in the thesis: links and descriptions

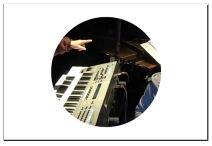
All the videos listed in the thesis can be found online in a dedicated video channel[1].

A.1 Performances and Work Sessions Using ImproteK

The videos listed in this section can be found online in a dedicated playlist[2] (or at vimeo.com/channels/improtek).

A.1.1 Compilation of Different Interaction Situations

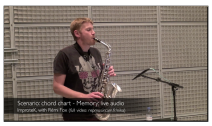
- Cited in Section 1.3.
- Hyperlink to the video,
- or vimeo.com/jeromenika/improtek-compilation.



Compilation of short video extracts of music improvisation sessions cited in the dissertation.

A.1.2 Conformity to an Idiomatic Structure, Improvisation on a Simple Chord Progression - Rémi Fox

- Cited in 4.2. Related section: 18.1.1.
- Hyperlink to the video,
- or vimeo.com/jeromenika/improtek-fox-rentparty.



Sax improvisations on a simple chord progression using the music improvisation software ImproteK. Work session with Rémi Fox, rehearsal for a performance at Montreux jazz festival. The software starts with an empty musical memory and improvises by reinjecting the live audio material which is processed and transformed online

1 Click on the **hypertext link**
or go to the youtube channel "Jérôme Nika":
www.youtube.com/channel/UCAKZIW0mMWCrX80yS96ZxAw.

2 Click on the **hypertext link**
or go to www.youtube.com/playlist?list=PL-C_JLZNFAGfGwtMPrRz9gOD3LnAMnHkO.

to match the scenario while being reactive to external controls. The scenario is the chord progression of *Rent party* (Booker T. Jones) segmented into beats:

||: Cm7 Bb7 | AbMaj7 Bb7 :||[3].

A.1.3 Example of "Hybridization" with an Early MIDI Version of the System

- Cited in 4.3. Related chapter: 17.
- Hyperlink to the video,
- or vimeo.com/jeromenika/improtek-lubat-early.

Excerpt of a concert with Bernard Lubat. Co-improvisation using an early MIDI version of the system playing theme / variations and a chorus. The musical memory used by the system is constituted by the captured live midi material and a very heterogeneous offline corpus (recordings of more than 10 jazz standards or ballads by different interprets). The scenario is the chord progression of *D'ici d'en bas* (Bernard Lubat) segmented into beats:

||: (Fm7 | G7 | Cm7 | Cm7 | F7 | G7 | Cm7 | Cm7)*2
| Fm7 | Bb7 | EbMaj7 | AbMaj7 | D7 | G7 | Cm7 | C7 |
| Fm7 | Bb7 | EbMaj7 | AbMaj7 | D7 | G7 | Cm7 | C m7 :||

A.1.4 "Scat" Co-improvisations, Synchronized Audio Rendering

- Cited in 13.1. Related chapter: 17.
- Hyperlink to the video,
- or vimeo.com/jeromenika/improtek-lubat-scat.

Compilation of "scat" co-improvisations with Bernard Lubat and Louis Lubat. For all these improvisation sessions, the system starts with an empty musical memory and improvises by re-injecting the live audio material which is processed and transformed online to match different idiomatic scenarios while being reactive to external controls and synchronized with a non-metronomic beat.

The scenarios used in the different examples are metric structures and/or harmonic progressions. In particular, the scenario of the last session presented in the video (1'36) is the chord progression of *J'aime pour la vie* (Bernard Lubat) segmented into beats:

||: (D7 | D7 | D7 | D7)*4 | (G7 Ab7 | G7 F7 | G7 Ab7 | G7 F7)*2 :||

3 In this chord progression and in the following: |...| = a bar = 4 beats.

A.1.5 Interactive Improvisation with a Composed Scenario Using a Content-based alphabet - Michelle Agnes Magalhaes

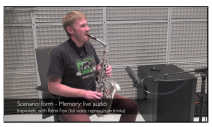
- Cited in 6.1.1. Related section: 18.3.
- Hyperlink to the video

- or vimeo.com/jeromenika/improtek-agnes-composed.

First technical experiments with composer-improviser Michelle Agnes Magalhaes who works on structured improvisation. The chosen content-based alphabet is a $3-$uple: loudness, brightness, playing mode. This example illustrates the case where the scenario only describes the part of the machine improvisation (see 6.1.1). The system re-injects the live audio material matching the playing modes and descriptor profiles imposed by the scenario. This test initiates a future project using a scenario which will be composed in such a way that the machine improvisation alternates between counterpoint and extension of the musical gesture of the musician.

A.1.6 Interactive Improvisation with a Composed Scenario Using an Abstract Alphabet - Rémi Fox

- Cited in 6.1.1. Related section: 18.1.2.
- Hyperlink to the video,



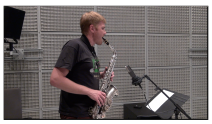
- or vimeo.com/jeromenika/improtek-fox-generative1.

Structured improvisation, work session with Rémi Fox, rehearsal for a performance at Montreux jazz festival. The software starts with an empty musical memory and improvises several voices by re-injecting the live audio material which is processed and transformed online to match the composed scenario while being reactive to external controls. The scenario defines two voices (“accompaniment” or “solo") and an abstract structure segmented into beats:

$$||: A_1\ B_1\ B_2\ A_1\ B_2 :||$$

with:
$A_1 = ||\,\mathrm{X}\,|\,\mathrm{X}^{+5}\,|\,\mathrm{X}^{-2}\,|\,\mathrm{X}^{+3}\,||$
$B_1 = ||\,\mathrm{Y\,Z}\,|\,\mathrm{Z}^{+5}\,\mathrm{X}^{+3}\,|\,\mathrm{Y\,X}^{+5}\,|\,\mathrm{Z}^{+5}\,\mathrm{X}^{+3}\,|\,\mathrm{Y\,X}^{-4}\,|\,\mathrm{Y}^{+3}\,|\,\mathrm{Z}^{-5}\,\mathrm{Z}\,|\,\mathrm{Z}^{+5}\,\mathrm{X}^{+3}\,||$
$A_2 = ||\,\mathrm{X}\,|\,\mathrm{X}\,|\,\mathrm{X}^{+5}\,|\,\mathrm{X}^{+5}\,|\,\mathrm{X}^{-2}\,|\,\mathrm{X}^{-2}\,|\,\mathrm{X}^{+3}\,|\,\mathrm{X}^{+3}\,||$
$B_2 = ||\,\mathrm{Y\,Z}\,|\,\mathrm{Z}^{+5}\,\mathrm{X}^{+3}\,|\,\mathrm{Y\,X}^{+5}\,|\,\mathrm{Z}^{+5}\,\mathrm{X}^{+3}\,|\,\mathrm{Y\,X}^{-4}\,|\,\mathrm{Y}^{+3}\,|\,\mathrm{Z}^{-5}\,\mathrm{Z}\,|\mathrm{Z}^{+5}/\mathrm{X}^{+3}\mathrm{Y}||$
where X, Y, Z are abstract equivalence classes and the exponents represent transpositions in semitones. A constraint is added to the “accompaniment” voice to get a repetitive structure: its memory is restricted to A_1 and the first measures of B_1.

A.1.7 Interactive Improvisation with a Composed Scenario Using an Abstract Alphabet #2 - Rémi Fox



- Related section: 18.1.2.
- Hyperlink to the video,
- or vimeo.com/jeromenika/improtek-fox-generative2.

This video presents a second improvisation session on the abstract scenario described in **Video A.1.6**.

A.1.8 Interactive Improvisation with a Composed Scenario Using an Abstract Alphabet #3 - Rémi Fox

- Related section: 18.1.2.
- Hyperlink to the video,
- or vimeo.com/jeromenika/improtek-fox-generative3.

This video presents a third improvisation session on the abstract scenario described in **Video A.1.6**.

A.1.9 Hervé Sellin Playing *The Man I Love* with Billie Holiday, Edith Piaf & Elizabeth Schwarzkopf

- Cited in 4.3. Related section: 18.2.
- Hyperlink to the video,
- or vimeo.com/jeromenika/improtek-sellin-themanilove1-finale.

The Man I Love #1 improvisation by Hervé Sellin (piano) and Georges Bloch (using ImproteK).The scenario provided to the system is the chord progression of the song, and its musical memory is:

- Hervé Sellin playing *The Man I Love*,
- Billie Holiday singing *The Man I Love*,
- Edith Piaf singing *Mon dieu* and *Milord*,
- Elisabeth Schwarzkopf singing *Mi tradì quell'alma ingrata*(Mozart, Don Giovanni), and *Tu che del gel sei cinta*(Puccini, Turandot).

A.1.10 *Autumn Leaves* by Sellin, Bloch, Mahler, Mozart, Puccini

- Related section: 18.2.
- Hyperlink to the video,
- or vimeo.com/jeromenika/improtek-sellin-autumnleaves2.

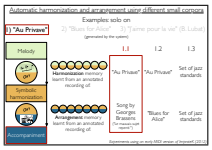
Autumn Leaves #2: improvisation by Hervé Sellin (piano) and Georges Bloch (using ImproteK). The scenario provided to the system is the chord progression of the song, and its musical memory is constituted by recordings of Elisabeth Schwarzkopf singing Mahler, Puccini and Mozart.

A.2 Extra Material: Demos, Early Works, and Experiments

The videos listed in this section can be found online in a dedicated playlist[4] (or at vimeo.com/channels/improtek2).

A.2.1 Early Experiments of Automatic Melody Harmonization and Arrangement

- Cited in B.1.2.
- Hyperlink to the video,
- or vimeo.com/jeromenika/improtek-early-harmo.

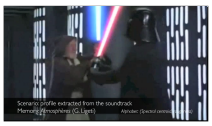


Early experiments to create accompaniments for solos generated by the system. They were realized using an early version of the generation model (the "conformity" model, see remark 5.3) and the first midi rendering module. The accompaniment is generated by chaining two instances of the generation model working on different alphabets:

- symbolic harmonization: a first instance uses the melody (segmented into beats) as a scenario and outputs a sequence of harmonic labels,
- arrangement: a second uses this sequence of harmonic labels as a scenario and outputs an accompaniment track.

4 Click on the **hypertext link** or go to www.youtube.com/playlist?list=PL-C_JLZNFAGehl0BOpMCWNGab63wTtei-.

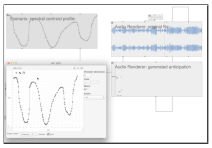
A.2.2 Using the Analysis of a Target Audio File as Scenario



- Cited in 6.1.1.
- Hyperlink to the video,
- or vimeo.com/jeromenika/improtek-starwospheres.

A short offline example using the analysis of a target audio file as scenario. The content-based scenario is the profile of spectral centroid and roughness extracted from the soundtrack of a musicless movie scene (only sound effects) segmented into audio events. It is applied to a memory constituted by the piece *Atmospheres* (Ligeti) analyzed with the same couple of audio descriptors. The generated sequence replaces the original soundtrack.

A.2.3 "Reaction Over Time" as Rewriting Anticipations



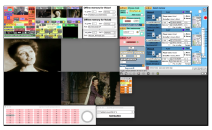
- Cited in 6.2 and 15.1. Related chapter: 9.
- Hyperlink to the video,
- or vimeo.com/jeromenika/improteksmc15.

Simulations "behind the interface": focus on the Improvisation Handler agent embedding the offline scenario/memory generation model in a reactive framework. This video shows the anticipations being rewritten when the chosen reactive inputs are modified.

- *Example 1:*
  - *chosen reactive inputs*: register and density,
  - scenario: harmonic progression (*Autumn leaves* with harmonic substitutions),
  - memory: heterogeneous MIDI corpus (captured solos on various blues or jazz standards).
- *Example 2:*
  - *chosen reactive inputs*: scenario and memory region,
  - scenario: spectral centroid profile,
  - memory: audio file (percussion solo by Trilok Gurtu).

A musical segment (one color) played by the system is not a musical chunk retrieved as it is in the memory, but a subsequence generated by the scenario / memory model.

A.2.4 Example of Video Improvisation

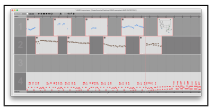


- Cited in 14.2.
- Hyperlink to the video,
- or vimeo.com/jeromenika/improtek-impro-video.

Example of video improvisation realized by Georges Bloch. The system can be chained to a video player and improvise by re-injecting offline or online video, transformed and reorganized to match a given scenario. In this example the scenario provided to the system is the harmonic progression of *The Man I Love*, and its memory is constituted by several videos:

- Lisa della Casa singing *Mi tradi quell'alma ingrata*, Don Giovanni, Mozart,
- Edith Piaf singing *Mon dieu*,
- Billie Holiday singing *The Man I Love*,
- Hervé Sellin playing *The Man I Love*.

A.2.5 Composition-oriented Rendering: Example of Dynamic Generation Processes Integrated in a Meta-score



- Related subsection: 15.1.
- Hyperlink to the video,
- or youtube.com/watch?v=GmDVoiisnDM.

Example of implementation of the models using a composition-oriented renderer: integrating two dynamic musical processes in a "meta-score" (OpenMusic maquette).

A.3 Bernard Lubat: Design of the First Prototype

The videos listed in this section can be found online in
a dedicated playlist[5]
(or at vimeo.com/channels/improtekarchiveslubat).

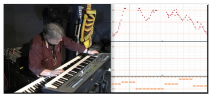
A.3.1 "Recombining", Conformity and Digression, Medium Continuity Regarding the Past of the Memory

- Cited in 5.3. Related section: 17.2.
- Hyperlink to the video,
- or vimeo.com/jeromenika/improtek-archive-recombine.

ImproteK early (midi) prototype: design of the first models and playing modes in collaboration with Bernard Lubat (2011-2013).
1) "Recombining" the musical memory: conformity and digression,
a) Learning continuous phrasing and generating with medium continuity.

This video gives an example of the first feature implemented in the early prototype of ImproteK during the incremental design of the playing modes with Bernard Lubat: "recombine" the memory while matching the scenario. Bernard Lubat plays a theme and/or develops a chorus covering several occurrences of the chord progression. Then, the system takes over playing a new chorus generated from this material, often authorizing transposition and choosing low values for the continuity regarding the past of the memory in the generation model (see 5.1.4) to get a solo both conform to the chord progression and digressing from the original material. Here, the continuity parameters are set so that fragments from the theme and from the choruses can be identified, in order to play a sort enriched "theme and variations". As in the other videos, a piano roll represents the different parts: the first one is the solo played by Bernard Lubat, the second one is the chorus by the machine, and the third part is a representation of the pulse (dots) and of the chord progression (dashes).

A.3.2 "Recombining", Conformity and Digression, Low Continuity Regarding the Past of the Memory



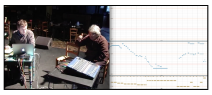
- Related section: 17.2.
- Hyperlink to the video,
- or vimeo.com/jeromenika/improtek-archive-recombine2.

5 Click on the **hypertext link**
or go to www.youtube.com/playlist?list=PL-C_JLZNFAGe69yRv3TudUQEz_4zEMxwZ.

ImproteK early (midi) prototype: design of the first models and playing modes in collaboration with Bernard Lubat (2011-2013).
1) “Recombining” the musical memory: conformity and digression,
b) Learning virtuoso phrasing and generating with low continuity.

A.3.3 "Downstream Controls”, Online Temporal Transformations and Interaction

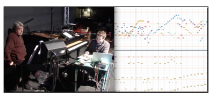
- Related section: 17.3.
- Hyperlink to the video,
- or vimeo.com/jeromenika/improtek-archive-downstreamcontrols.



ImproteK early (midi) prototype: design of the first models and playing modes in collaboration with Bernard Lubat (2011-2013).
2) “Downstream controls” online temporal transformations of the generated machine improvisation.

A.3.4 "Polyphonic Expansion”, Creating a Polyphonic Improvisation from a Single Solo

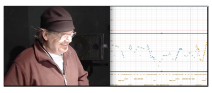
- Related section: 17.4.
- Hyperlink to the video,
- or vimeo.com/jeromenika/improtek-archive-poly-expansion.



ImproteK early (midi) prototype: design of the first models and playing modes in collaboration with Bernard Lubat (2011-2013).
3) “Polyphonic expansion": polyphonic improvisation from a musical memory restricted to one solo.

A.3.5 "Monophonic Reduction”, Creating a Monophonic Improvisation from Different Memories Annotated by the Same Scenario

- Related section: 17.4.
- Hyperlink to the video,
- or vimeo.com/jeromenika/improtek-archive-mono-reduction.



ImproteK early (midi) prototype: design of the first models and playing modes in collaboration with Bernard Lubat (2011-2013).
4) “Monophonic reduction": generating a chorus from different recordings of the same song. 1st chorus by the machine: theme and counterpoints; 2nd/3rd chorus: using downstream controls to “disturb” the music. It is important to remember here that a musical segment (one

color) played by the system is not a musical chunk retrieved as it is in the memory, but a subsequence generated by the scenario/memory model following the elementary process of "recombination" illustrated in 17.2.1.

A.3.6 "Hybridization": Generating a Chorus Using an Exogenous Memory

- Related section: 17.5.
- Hyperlink to the video,
- or vimeo.com/jeromenika/improtek-lubat-hybrid.

ImproteK early (midi) prototype: design of the first models and playing modes in collaboration with Bernard Lubat (2011-2013).
5) "Hybridization": generating a chorus using an "exogenous memory". Bernard Lubat's analysis (in French) when listening again to a recording of the previous improvisation session when the system improvised a chorus on *All the things you are* using a solo he played on his song *D'ici d'en bas* as musical memory.

A.4 Some Listening Sessions and Interview with Musicians

The videos listed in this section can be found online in a dedicated playlist[6] (or at vimeo.com/channels/nikainterviews).

A.4.1 Rémi Fox - About the First Improvisation of the Performance at Montreux Jazz Festival ("Rent Party")

- Related section: 18.1.1.
- Hyperlink to the video,
- or vimeo.com/jeromenika/interview-fox-1.

This video shows some extracts of an interview carried out with Rémi Fox during a listening session. This listening section focused on studio recordings of dialog on *Rent Party* (Booker T. Jones) during repetitions for a performance at Montreux Jazz Festival 2015 (see Section 18.1.1).

6 Click on the **hypertext link** or go to www.youtube.com/playlist?list=PL-C_JLZNFAGcyjSWATaNRFEjf2tFp1-Nw.

A.4.2 Rémi Fox - About the Second Improvisation of the Performance at Montreux Jazz Festival ("Generative improvisation")

- Related section: 18.1.2.
- Hyperlink to the video,
- or vimeo.com/jeromenika/interview-fox-2.

This video shows some extracts of an interview carried out with Rémi Fox during a listening session. This listening section focused on studio recordings of "generative improvisations" during repetitions for a performance at Montreux Jazz Festival 2015 (see Section 18.1.2).

A.4.3 Hervé Sellin - About The Man I Love #1

- Related section: 18.2
- Hyperlink to the video,
- or vimeo.com/jeromenika/interview-sellin-5.

This video shows some extracts of an interview carried out with Hervé Sellin during a listening session. This listening section focused on the first studio recording of an improvised piece based on *The Man I Love* of the "Three ladies" project described in Section 18.2. Transcriptions of this interview translated into english: Appendix C.1.5.

A.4.4 Hervé Sellin - About The Man I Love #2

- Related section: 18.2
- Hyperlink to the video,
- or vimeo.com/jeromenika/interview-sellin-6.

This video shows some extracts of an interview carried out with Hervé Sellin during a listening session. This listening section focused on the second studio recording of an improvised piece based on *The Man I Love* of the "Three ladies" project described in Section 18.2. Transcriptions of this interview translated into english: Appendix C.1.6.

A.4.5 Hervé Sellin - About Autumn Leaves #3

- Related section: 18.2
- Hyperlink to the video,
- or vimeo.com/jeromenika/interview-sellin-3.

This video shows some extracts of an interview carried out with Hervé Sellin during a listening session. This listening section focused on the third studio recording of an improvised piece based on *Autumn Leaves* of the "Three ladies" project described in Section 18.2. Transcriptions of this interview translated into english: Appendix C.1.3.

A.4.6 Hervé Sellin - About Autumn Leaves #2

- Related section: 18.2
- Hyperlink to the video,
- or vimeo.com/jeromenika/interview-sellin-2.

This video shows some extracts of an interview carried out with Hervé Sellin during a listening session. This listening section focused on the second studio recording of an improvised piece based on *Autumn Leaves* of the "Three ladies" project described in Section 18.2. Transcriptions of this interview translated into english: Appendix C.1.2.

A.4.7 Hervé Sellin - About Autumn Leaves #1

- Related section: 18.2
- Hyperlink to the video,
- or vimeo.com/jeromenika/interview-sellin-1.

This video shows some extracts of an interview carried out with Hervé Sellin during a listening session. This listening section focused on the first studio recording of an improvised piece based on *Autumn Leaves* of the "Three ladies" project described in Section 18.2. Transcriptions of this interview translated into english: Appendix C.1.1.

A.4.8 Hervé Sellin - Comparison of the Pieces

- Related section: 18.2
- Hyperlink to the video,
- or vimeo.com/jeromenika/interview-sellin-4.

This video shows some extracts of an interview carried out with Hervé Sellin during a listening session. This interviewed focused on the comparison of the two improvised pieces (*Autumn Leaves* and *The Man I Love*) of the "Three ladies" project described in Section 18.2. Transcriptions of this interview translated into english: Appendix C.1.4.

A.5 Archives: Other Collaborations

A.5.1 Brazilian ballad - Jovino Santos Neto

- Related section: 18.4.
- Hyperlink to the video,
- or vimeo.com/jeromenika/improtek-jovino-balaio.

Work session with Jovino Santos Neto, using an early midi prototype of the music improvisation system ImproteK (2013). Improvisation on “Balaio” by Hermeto Pascoal (the scenario is the chord progression of the ballad).

A.5.2 Stride piano - Louis Mazetier

- Related section: 18.5.
- Hyperlink to the video,

Demonstration by Louis Mazetier (Piano) and Marc Chemillier using the midi version of the music improvisation system ImproteK. The musical memory of the system is a set of transcriptions from the “stride” repertoire. The scenario is the chord progression of “Handful of keys” (Fats Waller) using an alphabet constituted by chord labels and macro-structure labels (such as “introduction”, “theme 1”, “conclusion...").

A.5.3 First Experiment in Madagascar, Marovany Zither - Velonjoro

- Related section: 18.6.
- Hyperlink to the video,

“Sojerina”, first experiment in Madagascar by Marc Chemillier using the midi version of the system with musician Velonjoro expert of the marovany zither (July 2014).

B

Implementation

B.1 A Library for Guided Generation of Musical Sequences

This section presents some additional work related to the scenario / memory generation model (Part I) and its implementation.

B.1.1 Scenario / Memory Generation Model

The generation model is implemented as a CommonLisp modular library, and within the OpenMusic environment (Bresson et al., 2011). This library will be integrated to a next release of the OpenMusic visual programming language and its current version will be available at http://repmus.ircam.fr/nika/code. It includes the definition of harmonic alphabets, alphabets of classes of audio descriptors, and abstract alphabets, and is conceived in a modular way so that one only has to follow the three steps listed in Chapter 6 to design new alphabets to define the scenarios. This implementation can be used offline as a Lisp library or in a patching environment to compose music sequences at the scenario level, as illustrated in Figure B.1 and Figure B.2.

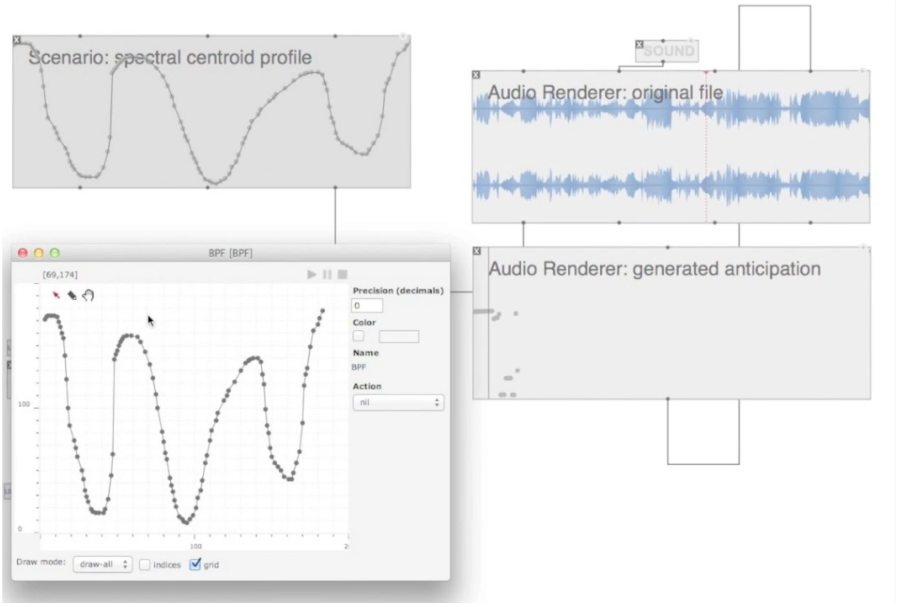


Figure B.1: Patch example in OpenMusic 7 (beta). Memory: annotated audio, scenario: audio descriptor profile.

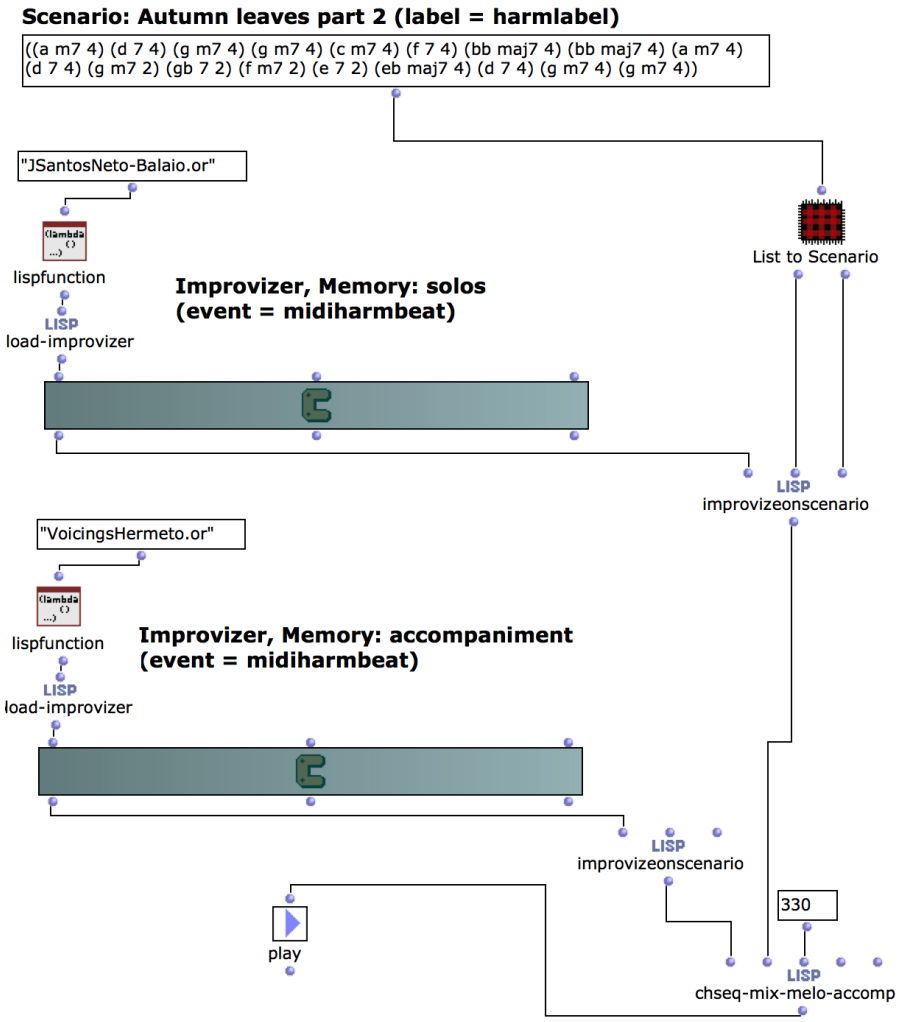
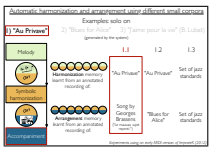


Figure B.2: Patch example in OpenMusic 6. Two instances of the generation model with a same scenario.

B.1.2 Harmonization and Arrangement

Video A.2.1



Hyperlink video *(or vimeo.com/jeromenika/improtek-early-harmo)*
Description: Appendix A.2.1.

At the beginning of this project, experiments were carried out to create accompaniments for solos generated by the system. They were realized using an early version of the generation model (the "conformity" model, see remark 5.3) and the first midi rendering module. The accompaniment is generated by chaining two instances of the generation model working on different alphabets (Figure B.3):

- symbolic harmonization: a first instance uses the melody (segmented into beats) as a scenario and outputs a sequence of harmonic labels,
- arrangement: a second uses this sequence of harmonic labels as a scenario and outputs an accompaniment track.

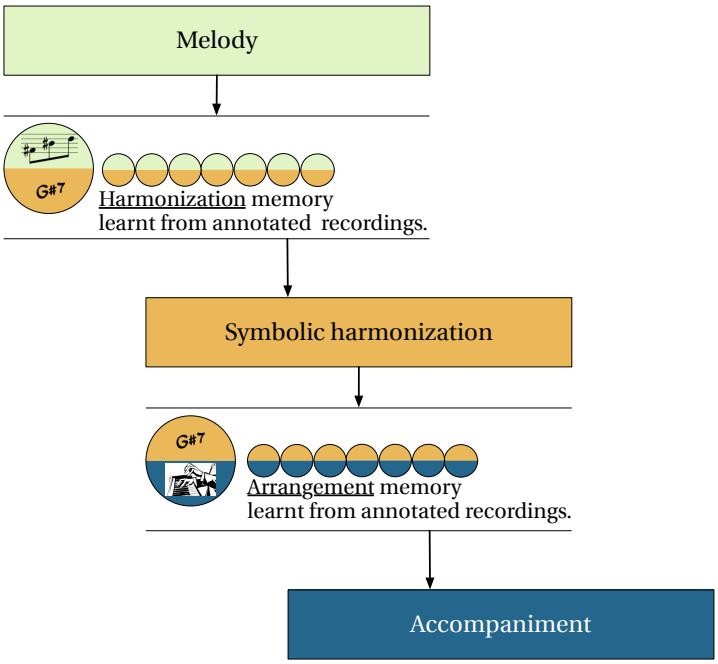


Figure B.3: Automatic harmonization and arrangement by chaining two generation processes using different alphabets.

The outline of this model enables to multiply the possibilities: a phrase can indeed be harmonized using a memory learnt on a given corpus, and then be arranged using a memory learnt on a completely different corpus (see examples in **Video A.2.1**). Furthermore, the terms "harmonization" and "arrangement" come from the fact that at that time, ImproteK had only been used in tonal jazz sessions. In other musical contexts, its genericity enables an understanding of other types of vertical associations that can be indexed in an agnostic way using other alphabets. We will go back to these tasks in future work. Indeed, harmonization and arrangement could benefit from the anticipation mechanisms provided by the last version of the generation model (Chapter 5).

B.1.3 Prefix Indexing with k Mismatches

This aspect was developed by Chemla-Romeu-Santos (2015) during his Master's Thesis (supervisors: Jérôme Nika and Gérard Assayag).

The modularity of the library enabled to implement prefix indexing (see Section 5.2) with k mismatches only by working at the local level, by overloading the comparison methods between labels of the scenario and events in the memory. This way the research follows on in case of mismatch until the maximum k of errors is reached, and handles all the different particular cases as illustrated in Figure B.4.

The aim was to introduce more flexibility than exact alignement, that could dismiss excellent solutions because of a single mismatch. This mode was mostly dedicated to scenarios defined on content-

based alphabets. In the case of idiomatic alphabet, it enables to add second options such as prefix indexing with k chord substitutions.

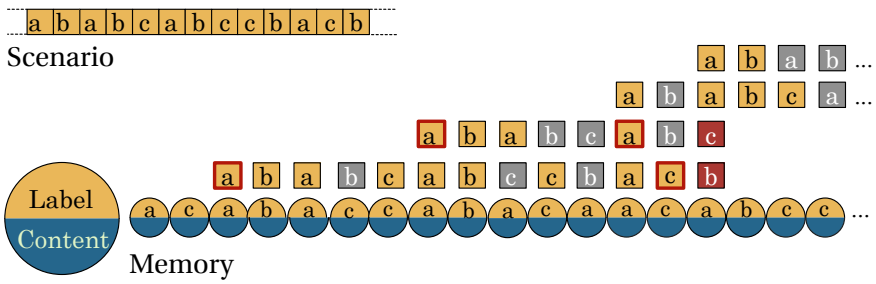


Figure B.4: Example of 3-mismatch prefix indexing (adapted from Chemla-Romeu-Santos, 2015).

B.1.4 Building Symbolic Sequences When Using a Content-Based Alphabet

OFFLINE We developed a signal processing module to build symbolic sequences from audio files with chosen audio descriptors, in the case of scenarios defined on content-based alphabets. These sequences can both serve as labelling sequences for audio memories or as target scenarios. This Matlab library uses the MIRToolBox (Lartillot and Toiviainen, 2007) and is inspired by a module developed by Bonnasse-Gahot (2014) for segmenting and analyzing offline audio corpora, and was then generalized by Chemla-Romeu-Santos (2015).

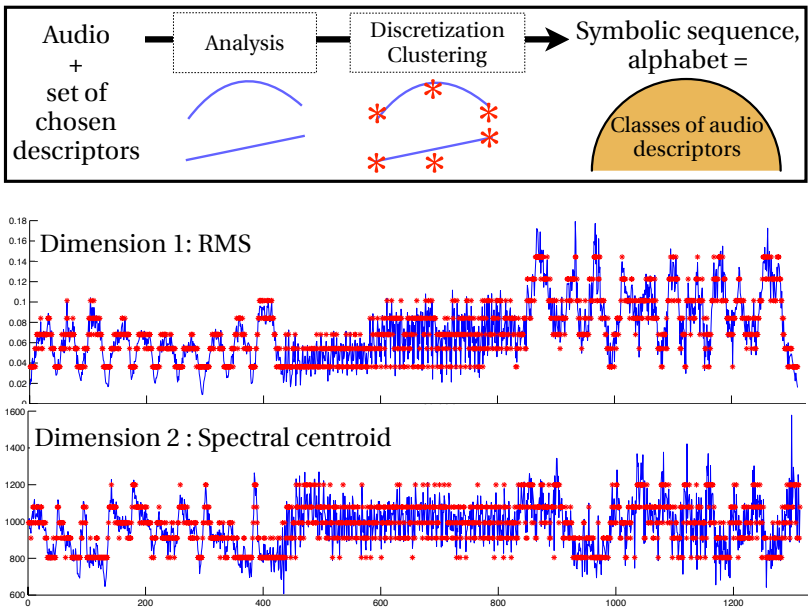


Figure B.5: Example: building a symbolic sequence from *Electronic counterpoint* (Steve Reich).

Figure B.5 gives a simple example of symbolic sequence built from *Elecronic counterpoint* (Steve Reich). In this figure, the chosen descriptors are RMS and Spectral Centroid, and the required number of clusters is quite low (7 classes for RMS, 4 classes for spectral centroid 4). In this example, the size of the alphabet is therefore $7 \times 4 = 28$.

When generating from such an annotated memory, as introduced in Chapter 6 (Figure 6.1), the comparison methods for generation can be defined in order to give different weights to the different descriptors, for instance defining an equivalence modulo transformation for one of them.

ONLINE When using a content-based alphabet online, the current process is to compare the inputs analyzed in real time to the centers of the clusters obtained from the offline analysis of an approaching material. This way, the new inputs are assigned to classes, and therefore to a letter of the alphabet. (When the scenario is defined on an idiomatic alphabet such as the chord progression of a jazz standard, there is no need for such an analysis since the scenario is a common reference for the musicians and the machine.) Future development will introduce music information rate (Dubnov et al., 2011) to get a relevant number of clusters automatically.

B.2 Reactive Improvisation Handler

The improvisation handler architecture presented in Chapter 9 (Part II) is integrated to the CommonLisp modular library described in B.1.1 and implemented in the OpenMusic environment (Bresson et al., 2011). When it is used in a patching environment, it can take advantage of the reactive features of the OpenMusic environment (Bresson and Giavitto, 2014). It gives the possibility to define easily:

- new reactive parameters,
- new output methods to be connected to different dynamic rendering modules,
- a external source of time markers to be informed of the current performance time.

Figure B.6 shows an example where the improvisation handler communicates with a renderer implemented in the Max environment. In this case, the output method is defined to send indexes to read in an audio buffer to Max via OSC protocol, and the current time of performance is sent back by the renderer. This configuration corresponds to the case of performance-oriented rendering (Chapter 13) as in the ImproteK system. The default configuration is the connection with a composition-oriented rendering module implemented in the OpenMusic environment (Section 15, Chapter 12).

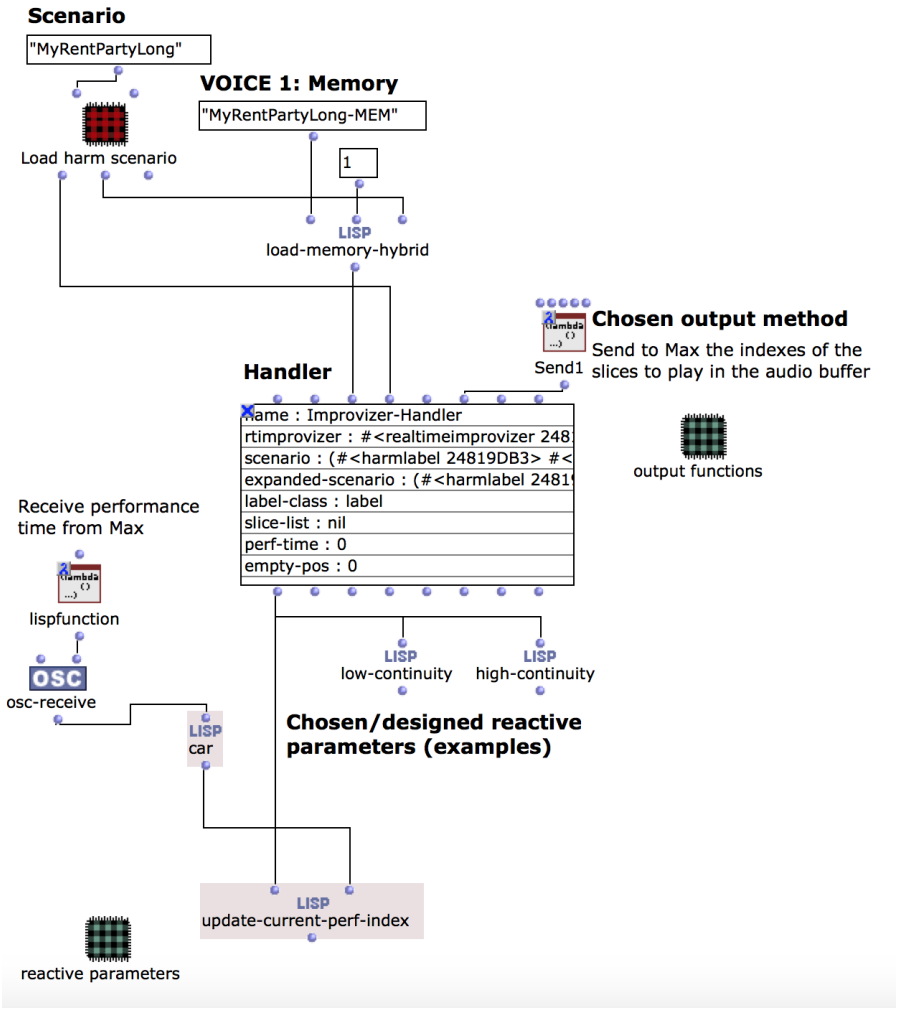


Figure B.6: Using the improvisation handler in a reactive patch (OM 6).

B.3 Dynamic Performance-Oriented Sequencer

The Dynamic Score presented in Chapter 10, the performance-oriented sequencer described in Chapter 13 and a graphical interface (Figure B.7) offering the controls mentioned in Chapter 14 are implemented in the graphical programming environment Max (Puckette, 1991) using the score follower Antescofo (Cont, 2008a) and its associated programming language (Echeveste et al., 2013a,b). It will be available at http://repmus.ircam.fr/nika/code.

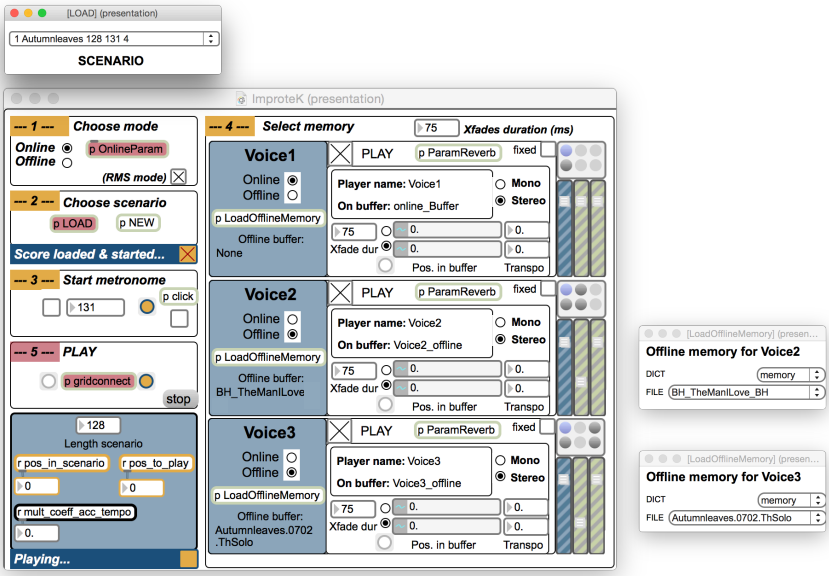


Figure B.7: Performance-oriented module to record, map, sequence, render, and synchronize multimedia sequences.

C

Interviews with Hervé Sellin

This chapter gives some extra material about the "Three Ladies" project (Hervé Sellin and Georges Bloch) presented in Section 18.2. Section C.1 presents transcriptions of interviews with Hervé Sellin during the listening sessions corresponding to the videos A.4.3 to A.4.8 (translated into english). Section C.2 contains a statement of intent by Hervé Sellin and the improvisation plans of the two pieces by Hervé Sellin and Georges Bloch (in french).

C.1 Transcriptions of Interviews and Listening Sessions

In this section, we present transcriptions of interviews with Hervé Sellin during the listening sessions corresponding to the videos A.4.3 to A.4.8 (translated into english).

> We decided to follow the chronological outline of the discussion since it enables to see the evolution of the discourse of the musician. Indeed, these interviews were realized using successive recording sessions of the pieces, from the least to the most "successful" according to the musician. Furthermore, the analysis of the recordings of the second piece (*The Man I Love*) refined his opinion on the first piece (*Autumn Leaves*).
>
> Margin annotations precise the scope of the quotations:
>
> - the recording session being listened: *[About the session...]*,
> - the improvised piece (*Autumn Leaves* or *The Man I Love*): *[About the piece...]*,
> - the project using ImproteK in general: *[About the project]*,
> - thoughts about improvisation in general inspired by the interactions with the system: *[About improvisation]*.

C.1.1 Listening Session *Autumn Leaves #1*

This section presents translated transcriptions of some extracts of an interview carried out with Hervé Sellin during a listening session focusing on the first studio recording of the improvised piece based on *Autumn Leaves* of the "Three ladies" project described in Section 18.2 (see **Video A.4.7**).

Video A.4.7

Hyperlink video *(or vimeo.com/jeromenika/interview-sellin-1)*
Description: Appendix A.4.7.
Musical part IV: Section 18.2.

C.1.1.1 *First impression about the session*

"It is not so bad ! It is interesting because it really looks like I'm discovering it for the first time. It is not so bad [in the sense that] there is nothing 'out'. If I had to evaluate it regarding the intentions of this project, I would say that I do not have enough interaction with the elements that I receive from the machine. Here, I tend to play in my normal way and there could be more effects of answers to the material."

C.1.1.2 *The bass line and the "standard"*

In this first session, the system played a bass line in addition to the "singers". Hervé said it tends to "pulled him" to the actual standard and that the process of "hybridization" in his own playing was stronger without this accompaniment. The initial goal was to set a reference to play "in and out", but for the following sessions, he decided to play without this bass to go further into hybridization.

[About the session Autumn Leaves #1]

"Yes, for sure, because it is a permanent call to order regarding the chords, the tempo... Yet, there are two different approaches: playing without support or base obviously creates a kind of freedom, but we all know that in jazz and in improvisation in general, freedom can be found within a restricting structure. It does exist. This is the phenomenon of playing 'in' and 'out', and playing 'out' is defined in relation to 'in'. The interest lies in the superimposition."

C.1.1.3 *Playing in and out*

[About improvisation]

"For the movie *Bird* by Clint Eastwood, about Charlie Parker's life, the real solos were extracted from the original recordings and the accompaniment rhythmics were replaced by new recordings because of the poor quality of the sound. The new rhythmic sections stick to the solo and it is horrible, because what was brilliant was this rhythmic playing like our bass line here, and Charlie Parker going in and out and in and out... And it terrific. The new accompaniment follows him when he goes in and out - because now we now where he goes when he goes out, maybe it was not the case in 1945, but in 2015 we "understand" that - and it is not good at all."

C.1.1.4 *"I do my job"*

[About the session Autumn Leaves #1]

"What I'm doing here does not step out of the line [i. e.the harmony is 'correct']... but it I am getting a little bored, I

am not having a lot of fun, playing it or listening to it. [...] In inverted commas, my "sense of craftsmanship" makes me want to get in this thing straight away, but it is not very exciting. I do my job. [...] [There is no particular stimulation coming from the interaction, even if] there is a connection between what I do and what happens. [...] This is why I say that it is not so bad, this is skillfully realized, by someone with job experience."

C.1.2 Listening Session *Autumn Leaves #2*

Video A.4.6

Hyperlink video *(or vimeo.com/jeromenika/interview-sellin-2)*
Description: Appendix A.4.6.
Musical part IV: Section 18.2.

This section presents translated transcriptions of some extracts of an interview carried out with Hervé Sellin during a listening session focusing on the second studio recording of the improvised piece based on *Autumn Leaves* of the "Three ladies" project described in Section 18.2 (see **Video A.4.6**).

C.1.2.1 *A connection at the 'event level'*

"This take is much more interesting, even if it is not a complete success. We are developing a connection. At least a connection at the 'event' level. [...] It's worth what it worth, but at least there is an interaction. Here, I am thinking that I have to do something even if it is not very original. [Listening to it, I see that] in my approach, the main thing is a wish to do something thinking 'If I don't do it, I will just play chords, jazz phrases, I will just play the theme, the song...' [...] This one is much more better. We can see a draft of the contacts that are being established."

[About the session Autumn Leaves #2]

C.1.2.2 *"The weight of what we know / of the standard"*

"It's started to have interaction, but it can be really better. The problem is that [in this kind of experiments] we carry the weight of what we know, of what we know how to do, of what we planned to do... And one can think that true improvisation should be performed with an empty brain, reacting to what happens, here, right now, but we have lots of ready-made things which come up. In fact, improvisation comes from how quickly you use one of other material. Here you can find an 'art', or a talent."

[About the session Autumn Leaves #2 / Improvisation]

"I have mixed feelings here, because we play a quite known and famous standard which is a song with a form, and from a certain point, the standard may become invasive. [...] This is a version that we did not really do, that is to say, never playing the chords or the theme nor going

[About the piece based on Autumn Leaves]

into the form of the tune, but only reacting to sound and events"

C.1.2.3 *History, training and background*

[About improvisation]

"Here we meet with something else, my history, my training, the way I work. I am quite pragmatic, and quite traditional in my modus operandi: I like the themes, I like when it is straight, I like when it is precise, I like nice harmony... I am not mad about 'discovering' or wild improvisation, even if I know how to do it. This raises a huge issue [...] it has been discussed a lot through the history of jazz, with all the people who had supposedly assimilated the culture and who claimed being modernist, and those who said 'No, we don't want no culture or tradition to be 'blank' and then really improvise', and it stands up."

[About improvisation]

"I come from a very strict and 'well-defined' background. I took me years to be able to do this [showing the video]. I had friends of my age who already were funambulists when we were twenty [...]. I was still listening to Armstrong, Ellington, Basie,... and they were already beyond Coltrane and Coleman, they played jazz rock at full speed... Not only I was not interested by all this, but I did not understand anything to it! My own process for really getting 'free' has been very long, but somehow this is an advantage because I developed deep foundations, and today I can take advantage of them. But it has been long, and I am still burdened with all these things."

C.1.2.4 *"My first preoccupation is to listen"*

[About the project / Improvisation]

The idea is not to play always 'with'. But the worst is to play 'without', in this case there is no point. I thought about that: I know why it works a little with me. It is because my approach of music is that I never listen to myself. I listen to the others, and then it generates reactions within me. These reactions are consistent thanks to my knowledge, but I absolutely never ask myself things like 'I heard that, so the best thing to play is E natural...' There is an 'anarchist' side to me, and this is why it is not so bad, even if it can be better. In a jazz orchestra, whatever it is, my first preoccupation is to listen to the others and to play according to what I ear. There is not a single moment when I decide alone what I am going to do."

C.1.2.5 *A Perspective: Modal Improvisation*

"I am thinking about a tune, we should try that: *So What*. It is modal, there is one chord. Maybe we could get to something a little more "sustained" in the long term. Yet, it is interesting to seize something and focus on it until the end of the tune, whatever the three [virtual] singers do. It is a way to play with it actually. At least you do not play anything that is in the bag you came with. [...] In this case the system would discover much more possibilities, and everything would be multiplied by time. We should work on both aspects: [...] see what the machine can do and what the musician can do, then analyse the difference, it must be easy to do."

[About the project]

C.1.3 Listening Session *Autumn Leaves #3*

Video A.4.5

Hyperlink video *(or vimeo.com/jeromenika/interview-sellin-3)*
Description: Appendix A.4.5.
Musical part IV: Section 18.2.

This section presents translated transcriptions of some extracts of an interview carried out with Hervé Sellin during a listening session focusing on the third studio recording of the improvised piece based on *Autumn Leaves* of the "Three ladies" project described in Section 18.2 (see **Video A.4.5**).

C.1.3.1 *Complementarity*

"What I often try to do is to be complementary in comparison to what I ear, but it is not necessarily the better thing to do. It would be worthwhile to use the material in itself instead of trying to create a harmony, a balance. At the beginning, this is what I did: I play fast when it plays slow, I play the theme when it is confused, I play rhythmic when it begins to go around... This is my 'craftsman / Good Samaritan' side, this is my background, my training. I am a craftsman, who might have discovered that he is able to do a little more,..."

[About the session Autumn Leaves #3 / Improvisation]

C.1.3.2 *Stimulation: "a beautiful event"*

"It is funny because here I used a pattern which belongs to what I usually do, but what I heard from the machine made me develop it longer. Here I thought 'OK, well done !'. [...] I try to uncover what belongs to my own playing. [...] But I would not have done it without the stimulus of the machine. [...] At a given moment I played 'ka-ding' within my own discourse and then I received 'ba-doum ba-doum ba-doum' so I went on with my 'ba-doum ba-doum', but [without it] I would not have done it. Here, we

[About the session Autumn Leaves #3 / the project]

are in the good story. [...] This is a good sequence, something that happened spontaneously. I played something which is not usual for me, and it created a beautiful event with what happened in the machine, and I didn't have to search for something in the machine."

C.1.4 First Comparison Between the Two Pieces

Video A.4.8

Hyperlink video *(or vimeo.com/jeromenika/interview-sellin-4)*
Description: Appendix A.4.8.
Musical part IV: Section 18.2.

This section presents translated transcriptions of some extracts of an interview carried out with Hervé Sellin during a listening session focusing on the comparison of the two improvised pieces (*Autumn Leaves* and *The Man I Love*) of the "Three ladies" project described in Section 18.2 (see **Video A.4.8**).

C.1.4.1 *"It is good for me to watch this again"*

[About the project / Improvisation]

"It is good for me to watch this again. I see what works, what does not. I can see when I totally screwed up in my own habits, my own contradictions, my fears... There are so many things involved in this: I have to keep control, but I would like to lose it all the same to be surprised: find this bloody balance, with the sound, the coherence, the complementarity,... all this makes loads of parameters. [...] And it is impossible to react differently, unless you decide to forget everything, to forget the tune... but then you run the risk of only making noise."

C.1.4.2 *"Autumn Leaves: a chord progression, The Man I Love: a story"*

[About the project / Improvisation]

"Somehow, *Autumn Leaves* is more of a chord progression whereas *The Man I Love* is more of a story. That's for sure, even from an historical point of view. First the history is not the same, for *The Man I Love* there is a context, it has been written for a reason. *Autumn Leaves* went through all the colors of the rainbow. Of course, at first, it was a song, but it is not approached like that anymore, and the chord progression we use is not exactly that of the original song. It is the chord that the jazzmen seized. It would be like playing the blues, or *I got rhythm*, or any piece that the jazzmen use in their routine. *The Man I Love* is different, it is another canvas which ties harmony together in a different way. So it generates other elements at every level."

C.1.4.3 *His expectations regarding the system at the beginning of the project*

[About the project]

"It should become 'intelligent', structure itself within chord progressions, and not chose an element for a chord, then another for the following chord... The chord progression in *Autumn Leaves* is the famous *ii-V-I*, in *The Man I Love* it is more complicated because it develops on eight bars in a row with a guiding thread which is an inner counter-melody. The machine should be able to handle it. [...] Playing on a theme such as *The Man I Love*, it is much more complex because the progressions are not standard."

C.1.5 Listening Session *The Man I Love #1*

***Video* A.4.3**

Hyperlink video *(or vimeo.com/jeromenika/interview-sellin-5)*
Description: Appendix A.4.3.
Musical part IV: Section 18.2.

This section presents translated transcriptions of some extracts of an interview carried out with Hervé Sellin during a listening session focusing on the first studio recording of the improvised piece based on *The Man I Love* of the "Three ladies" project described in Section 18.2 (see **Video A.4.3**).

C.1.5.1 *Horizontality*

[About the session The Man I Love #1 / project]

"Here, we are in a complete process of complementarity but it is interesting. [...] The feeling I have straight away is that it is much more into horizontality than in *Autumn Leaves*. The tempo has something to do with it too. It allows everyone, the machine included, to spread a little more. [...] I find it much more flowing. In [*Autumn Leaves*], we are really given a hard time ! The machine does really better here."

C.1.5.2 *"I am playing with an orchestra"*

[About the session The Man I Love #1]

"From the very beginning, I play the introduction, then the system starts and it is already well engaged. [...] It is fun ! [...] This is great. This is a totally different way of functioning. Now I have the impression that I am playing with an orchestra: I react according to what is being played by the second pianist of the orchestra, then the singer comes back, the rhythm section comes back... It is not the same experience at all. Somehow, I feel much more comfortable here, and the elements fit together in a very nice way."

C.1.5.3 *"Music-driven" vs "Event-driven" reaction*

[About the session The Man I Love #1 / project]

"Here, my reactions are 'music-driven', and not really 'event-driven'. In the other [*Autumn Leaves*], the only thing that can make it take off is this event-driven reaction, the musical element must not be used otherwise it would be a little 'corny'. [...] A 'musical element', here, is for example when a piano plays something and I play something which is complementary, and then a singer arrives and she sings to me at least 8 bars so I know where to place myself..."

"[This one] is really beautiful. Really beautiful and very subtle. In *Autumn Leaves* it was extremely boring, but here I find it very subtle."

C.1.5.4 *"I am the guard" vs "What do I suggest ?"*

[About the project / Improvisation]

"There is still work to do, to tend to something a little more original, given the fact that the machine sends unexpected information. [...] When something happens either you think "I am the guard, I have to ensure safety" or you play the game and you think "what do I suggest ?" [...] There could have been more unexpected suggestions from me. What the machine plays on this one is really great. There is a nice organisation in the musical space."

C.1.5.5 *Anticipatory behavior*

[About the piece based on The Man I Love]

"It is indeed more long-term, in particular when the three ladies sing together. And the tempo has to do with it too, necessarily a bar lasts longer so... For the elements retrieved from Piaf, Holiday, and Schwarzkopf, we have the time to ear 3 notes, 4 notes, a phrase... It fills you somehow."

C.1.5.6 *"Bring the song out of Autumn Leaves was a mistake"*

[Step back: About the piece based on Autumn Leaves]

"We approached the performance on *Autumn Leaves* in the same way, because it is a song all the same. We often tried to bring out the song out of it, and this may be the mistake. [...] [If you decide that it] will go in all directions, then you have to go in all directions too. Because even if you decide to catch it off balance - e. g.the machine sends 4 elements during a bar so at the opposite you play something that lasts 8 bars to create a kind of balance - I am not sure that it will work, it is not that simple. Anyway, for sure between this piece and the other one it's like night and day. I am not saying that one is good and the other

is not, they are two separate universes and we have to be aware of it."

C.1.5.7 *"Fireworks and confetti"*

"When the three singers come in at the same time. I have to find something to do. I could always play the theme at some point, but I cannot do it until the end so I play 'huge chords', to make it a sounding finale like 'And they lived happily ever after !' Why not ? It had a lot of success in Italy [the first concert], the audience was like 'wonderful', [with large gestures, mimicking fireworks] we just needed fireworks, balloons, and confetti ! Yet do we really do this kind of things for that... ?"

[About the piece based on The Man I Love]

C.1.6 Listening Session *The Man I Love #2*

***Video* A.4.4**

Hyperlink video *(or vimeo.com/jeromenika/interview-sellin-6)*
Description: Appendix A.4.4.
Musical part IV: Section 18.2.

This section presents translated transcriptions of some extracts of an interview carried out with Hervé Sellin during a listening session focusing on the second studio recording of the improvised piece based on *The Man I Love* of the "Three ladies" project described in Section 18.2 (see **Video A.4.4**).

C.1.6.1 *"I want to surprise myself"*

"I am rather satisfied with what I do, except that I also want to surprise myself and to meet the level of demand of such a project. I know what I am able to do, but what is exciting is to look for something else."

C.1.6.2 *Go further in the "patchwork"*

"This base [voice generated and played by the system: bass/accompaniment] in the other [*Autumn Leaves*] is embarrassing. It is really good, that is not the issue, it is really reassuring... but what is the point, artistically speaking ? It is a kind of crutch that we inserted in here but which doesn't tell any story, even if it is well realized. It is just a safeguard."

[Step back: About the piece based on Autumn Leaves]

"It is like using training wheels. Yes, that is exactly it! We have to remove them! Anyway, we needed it, I started working on it on July [5 months before the interview]! I think it would be worthwhile to go through with this intention, including the 'patchwork' aspect. [...] We have to remove everything that does not help to go to an optimal result on the pretext of providing a false security. It is now obvious to me."

[About improvisation / the piece based on Autumn Leaves]

C.1.6.3 *Perspective: Compare "vertical" and "horizontal" scenarios*

[About the project]

"Trying to work on a more continuous material while working within a very vertical chord progression and a fast tempo may be interesting too as a research experiment regarding the machine. The same way, on a modal scenario we would have to see what happens if we try to tale the opposing view. We have to try all directions, and then have various possibilities."

C.1.6.4 *"It has a meaning"*

[About the session The Man I Love #2 / the piece based on The Man I Love]

"It is not the same anymore, I am not playing an accompaniment, I am really playing the song, with chords, etc. Here, we propose something, and there is no a safeguard. [...] Because, you see, this has a meaning for me. It could be another pianist playing [talking about the machine] and it has a meaning, it makes sense."

C.1.6.5 *"Try and try again"*

[About the session The Man I Love #2 / project]

"There are more interesting things to find. I do the job because we have to remember that this is *The Man I Love*, so I play the song and I finish with fireworks with everybody, but there are more interesting things to do. We have to play, try, try, and try again... And record and film to be able to say 'this was ok, we have to throw this out...' There are so many ways we can take, there is not a right one and wrong one, there are 3000 right and 3000 wrong. Some make sense regarding what we are here for. We are not here only to play 'nice' things, or to play without mistakes... let's try to forget all this. If it is nice, successful, interesting and flawless in the end, then it is great. But here the goal won't be reached once it has been nice or flawless."

C.1.6.6 *"Find in real time"*

[About the project]

"The interest [with the machine] is to play: I could write 10 pages of music and it would be monstrous because I would exactly know what to do regarding [harmony ? the chord progression ?]... This is not the interest, the interest is to find in real time."

C.2 “Three Ladies” Project: Statement of Intent and Improvisation Plans (in French)

C.2.1 Statement of intent by Hervé Sellin

> “En tant que seul musicien “vivant” - en “chair et en os” ! - (mais je n’oublie pas “l’homme” derrière la machine...) dans cette création, la difficulté était essentiellement la cohérence générale. Cohérence artistique et cohérence technologique.
>
> L’auditeur devait ne pouvoir suivre qu’une seule entité de discours composée de l’ensemble des éléments utilisés aussi bien par la machine que par le pianiste lui-même. Cela suppose une maîtrise des éléments de volume(s), nuances et dynamiques d’intention ainsi qu’une faculté de réaction et d’inspiration adaptée. Un autre difficulté a été la maîtrise des tempos. Dans l’improvisation en solo, le pianiste reprend le cours du discours à son compte et le retour des éléments proposés par la machine doit se faire en parfaite symbiose. Ce qui n’était pas évident...
>
> Une remarque sur le choix des 2 supports thématiques et donc, structurels : *Autumn Leaves* et *The Man I love*. Le schéma harmonique du premier offre plus de possibilités, de liberté et, donc, de souplesse d’improvisation. Cela provient des enchainements harmoniques qui présentent un caractère plus “universel”. Ils sont donc, en quelque sorte, plus faciles à oublier et l’on peut mieux se concentrer sur l’interactivité avec les éléments provenant de la machine.
>
> *The Man I love* présente, lui, un schéma harmonique plus spécifique et plus contraignant. Je dirais que les enchainements d’accords sont plus typés “chanson” que “standard de jazz à improviser”. C’est, néanmoins, ce qui fait la richesse de cette composition qui demeure un chef d’œuvre de référence.”

C.2.2 Improvisation plans, scenarios, and memories used in the project (by Georges Bloch and Hervé Sellin)

C.2.2.1 *Piece based on Autumn Leaves: improvisation plan*

On note “grille I, grille II,...” les occurrences successives de la grille jouée intégralement.

- Mémoires et paramètres pour les différentes voix au début de la performance :
  - Voix 1 : Hervé Sellin “clone”, tabou sur la première occurrence de la grille.
  - Voix 2 : *Das irdische leben*, 4e symphonie de Mahler avec Elisabeth Schwarzkopf et Bruno Waltern, grande continuité avec le futur du scénario.
  - Voix 3 : Des Knaben Wunderhorn, *Lob des hohen Verstands* avec Elisabeth Schwarzkopf et Szell.
- Introduction par Hervé Sellin, puis pont.
- Grille I :
  - Démarrage de la Voix 1 (Hervé Sellin “clone”).
  - Sur le pont : entrée de la Voix 2 (mémoire 4e symphonie de Mahler).
- Grille II :
  - Voix 2 (mémoire 4e symphonie de Mahler) continue.
  - Baisser la Voix 1 (Hervé Sellin “clone”), remplacée par Hervé Sellin live.
  - Sur le pont : Voix 3 (Mémoire Elisabeth Schwarzkopf, *Lob des hohen Verstands*) en arrêtant Voix 1.
- Grille III :
  - Voix 1 : Mémoire piano d’Hervé Sellin, continuité faible.
  - Voix 2 et 3 : Mémoire Elisabeth Schwarzkopf, *Lob des hohen Verstands*, continuités respectivement faible et forte.
- Grille IV :
  - Arrêt des voix pour laisser place à un solo d’Hervé Sellin.
- Grille V :
  - Nouvelle Voix 3 : Mémoire Elisabeth Schwarzkopf, Mozart, Don Giovanni, air de Donna Anna : *Mi tradi*.
  - Hervé Sellin “live” et interventions ou non du “clone” en Voix 1.

C.2.2.2 *Piece based on The Man I Love: improvisation plan*

La performance est basée sur 3 cycles de deux grilles légèrement différentes (partie 1 et partie 2):

- Mémoires et paramètres pour les différentes voix au début de la performance :

  - Voix 1 : Mémoire hybride : Billie Holiday et Hervé Sellin sur The Man I Love.
  - Voix 2 : Edith Piaf, *Milord*, grande continuité.
  - Voix 3 : Elisabeth Schwarzkopf, *Air de Liù* : *Tu che del gel sei cinta*, Puccini, Turandot, orchestre et chœurs de la Scala dirigés par Tullio Serafin. Nombreuses zones interdites pour éviter les passages où le tempo est trop lent et pose des problèmes de timbres en utilisant le vocodeur de phase.

- Introduction par Hervé Sellin, puis pont.
- Grille I (partie 1) :
  - A : Entrée de la Voix 1.
  - Pont : Sortie de Billie Holiday et entrée d’Edith Piaf et Hervé Sellin (live).
  - A’ : Entrée Hervé Sellin (Clone) accompagnant Piaf (Hervé Sellin “live” ne joue pas).
- Grille II (partie 2) :
  - A : Hervé Sellin “live” remplace Hervé Sellin “clone”, la Voix 2 (Mémoire : Edith Piaf, *Milord*) continue.
  - Pont : Voix 3 (Elisabeth Schwarzkopf, *Air de Liù*) remplace Voix 2 (Mémoire : Edith Piaf, *Milord*).
  - A’ : Retour du “clone” d’Hervé Sellin, la Voix 3 (Elisabeth Schwarzkopf, *Air de Liù*) continue (Hervé Sellin “live” ne joue pas).
- Grille III (partie 1) :
  - A : Hervé Sellin “live” remplace Hervé Sellin “clone”, la Voix 3 (Elisabeth Schwarzkopf, *Air de Liù*) continue.
  - Pont : Faire rentrer une nouvelle Voix 2 : Mémoire *Mon Dieu*, Edith Piaf.
  - A’ : “2 ladies” et entrée de Hervé Sellin “clone”.
- Grille IV (partie 2) :
  - A : Solo Hervé Sellin “live”.
  - Pont : Duo Hervé Sellin “live” et Hervé Sellin “clone”.
  - A’ : Sortie Hervé Sellin “clone” et entrée Billie Holiday.
- Grille V (partie 1) :
  - A : Entrée de Billie Holiday seule.
  - Pont : Entrée nouvelle Voix 3 Mémoire : Elisabeth Schwarzkopf, Mozart, Don Giovanni, air de Donna Anna : *Mi tradi*, ou Voix 2 (Mémoire *Mon Dieu*, Edith Piaf), ou l’une puis l’autre.

  - A' : Entrée Voix 2 (Mémoire *Mon Dieu*, Edith Piaf), ou Nouvelle Voix 3 (Mémoire Elisabeth Schwarzkopf, *Mi tradi*), ou l'une puis l'autre.

- Grille VI (partie 2) :
  - Finale : "Three ladies singing" et Hervé Sellin crescendo.

Bibliography

Andrea Agostini and Daniele Ghisi. Real-Time Computer-Aided Composition with bach. *Contemporary Music Review*, 32(1):41–48, 2013. (Cited on page 31.)

Alfred V. Aho and Margaret J. Corasick. Efficient string matching: an aid to bibliographic search. *Communications of the ACM*, 18(6): 333–340, 1975. (Cited on page 28.)

Cyril Allauzen, Maxime Crochemore, and Mathieu Raffinot. Factor oracle: A new structure for pattern matching. In *SOFSEM 99: Theory and Practice of Informatics*, pages 758–758. Springer, 1999. (Cited on pages 23, 59, and 63.)

Antoine Allombert, Myriam Desainte-Catherine, and Gérard Assayag. Iscore: a system for writing interaction. In *Proceedings of the 3rd international conference on Digital Interactive Media in Entertainment and Arts*, pages 360–367. ACM, 2008. (Cited on page 35.)

Jaime Arias. *Formal Semantics and Automatic Verification of Hierarchical Multimedia Scenarios with Interactive Choices*. PhD thesis, Université de Bordeaux, 2015. (Cited on page 35.)

Jaime Arias, Myriam Desainte-Catherine, and Camilo Rueda. Modelling data processing for interactive scores using coloured petri nets. In *Application of Concurrency to System Design (ACSD), 2014 14th International Conference on*, pages 186–195. IEEE, 2014. (Cited on page 35.)

Gérard Assayag. Computer Assisted Composition Today. In *1st symposium on music and computers*, Corfu, 1998. (Cited on page 31.)

Gérard Assayag and Georges Bloch. Navigating the oracle: A heuristic approach. In *International Computer Music Conference*, pages 405–412, Copenhagen, 2007. (Cited on pages 23 and 60.)

Gérard Assayag and Shlomo Dubnov. Using factor oracles for machine improvisation. *Soft Computing*, 8(9):604–610, 2004. (Cited on pages 23, 34, and 60.)

Gérard Assayag, Shlomo Dubnov, and Olivier Delerue. Guessing the composer's mind: Applying universal prediction to musical style. In *International Computer Music Conference*, pages 496–499, Beijing, 1999. (Cited on pages 23 and 34.)

Gérard Assayag, Georges Bloch, and Marc Chemillier. OMax-ofon. *Sound and Music Computing (SMC)*, 2006a. (Cited on pages 23 and 35.)

Gérard Assayag, Georges Bloch, Marc Chemillier, Arshia Cont, and Shlomo Dubnov. OMax brothers: a dynamic topology of agents for improvization learning. In *1st ACM workshop on Audio and music computing multimedia*, pages 125–132, ACM, Santa Barbara, California, 2006b. (Cited on pages 19, 23, 24, 34, and 59.)

Jean-Julien Aucouturier and François Pachet. Jamming with plunderphonics: Interactive concatenative synthesis of music. *Journal of New Music Research*, 35(1):35–50, 2006. (Cited on page 29.)

Derek Bailey. *Improvisation: its nature and practice in music*. Da Capo Press, 1993. (Cited on page 1.)

Gérard Berry and Georges Gonthier. The esterel synchronous programming language: Design, semantics, implementation. *Science of computer programming*, 19(2):87–152, 1992. (Cited on page 99.)

Christian Béthune. Le jazz comme oralité seconde. *L'Homme*, 171(3): 443–457, 2004. (Cited on page 148.)

J.A. Biles. GenJam: Evolutionary computation gets a gig. In *Conference for Information Technology Curriculum, Rochester, New York, Society for Information Technology Education*, 2002. (Cited on page 33.)

Tim Blackwell. Swarming and music. In *Evolutionary Computer Music*, pages 194–217. Springer, 2007. (Cited on pages 19 and 22.)

Tim Blackwell and Peter Bentley. Improvised music with swarms. In *wcci*, pages 1462–1467. IEEE, 2002. (Cited on page 22.)

Georges Bloch, Shlomo Dubnov, and Gérard Assayag. Introducing video features and spectral descriptors in the omax improvisation system. In *International Computer Music Conference*, volume 8, 2008. (Cited on page 133.)

Laurent Bonnasse-Gahot. Donner à omax le sens du rythme: vers une improvisation plus riche avec la machine. *Ecole des Hautes Etudes en sciences sociales, Technical report*, 2010. (Cited on pages 80 and 160.)

Laurent Bonnasse-Gahot. An update on the SOMax project. Technical report, Ircam - STMS, 2014. Internal report ANR project Sample Orchestrator 2, ANR-10-CORD-0018. (Cited on pages 19, 25, 33, and 210.)

Jocelyn Bonnerave. Improviser ensemble. de l'interaction à l'écologie sonore. *Tracés. Revue de Sciences humaines*, (18):87–103, 2010. (Cited on page 9.)

Dimitri Bouche and Jean Bresson. Planning and Scheduling Actions in a Computer-Aided Music Composition System. In *Proceedings of the Scheduling and Planning Applications woRKshop (SPARK)*, Jerusalem, Israel, 2015a. (Cited on pages 31 and 137.)

Dimitri Bouche and Jean Bresson. Articulation dynamique de structures temporelles pour l'informatique musicale. In *Actes du 10ème Colloque sur la Modélisation des Systèmes Réactifs (MSR)*, Nancy, France, 2015b. (Cited on page 135.)

Dimitri Bouche, Jérôme Nika, Alex Chechile, and Jean Bresson. Computer-aided composition of musical processes. *Journal of New Music Research*, 2016. (submitted). (Cited on pages xvi, 109, 135, 136, and 187.)

Pierre Bourdieu. *Esquisse d'une théorie de la pratique, précédée de trois études d'ethnologie kabyle*. Le Seuil, 1972. (Cited on page 9.)

Oliver Bown and Aengus Martin. Autonomy in music-generating systems. In *Eighth Artificial Intelligence and Interactive Digital Entertainment Conference*, 2012. (Cited on page 23.)

Robert S Boyer and J Strother Moore. A fast string searching algorithm. *Communications of the ACM*, 20(10):762–772, 1977. (Cited on page 58.)

Jean Bresson and Jean-Louis Giavitto. A Reactive Extension of the OpenMusic Visual Programming Language. *Journal of Visual Languages and Computing*, 4(25):363–375, 2014. (Cited on pages 31 and 211.)

Jean Bresson, Carlos Agon, and Gérard Assayag. OpenMusic. Visual Programming Environment for Music Composition, Analysis and Research. In *ACM MultiMedia 2011 (OpenSource Software Competition)*, Scottsdale, AZ, USA, 2011. (Cited on pages 41, 75, 109, 135, 187, 207, and 211.)

Joseph T Buck, Soonhoi Ha, Edward A Lee, and David G Messerschmitt. Ptolemy: A framework for simulating and prototyping heterogeneous systems. 1994. (Cited on page 30.)

John Cage. *Silence: lectures and writings*. Wesleyan University Press, 1973. (Cited on page 4.)

Clément Canonne and Jean-Julien Aucouturier. Play together, think alike: Shared mental models in expert music improvisers. *Psychology of Music*, 2015. (Cited on page 20.)

Dorian Cazau, Olivier Adam, and Marc Chemillier. An original optical-based retrieval system applied to automatic music transcription of the marovany zither. In *Proceedings of the Third International Workshop on Folk Music Analysis (FMA2013)*, page 44, 2013. (Cited on page 179.)

Joel Chadabe. Some reflections on the nature of the landscape within which computer music systems are designed. *Computer Music Journal*, pages 5–11, 1977. (Cited on pages xiv and 17.)

Marc Chemillier. Improviser des séquences d'accords de jazz avec des grammaires formelles. In *Proc. of Journées d'informatique musicale*, pages 121–126, Bourges, 2001. URL http://ehess.modelisationsavoirs.fr/marc/publi/jim2001/jim2001english.pdf. (English summary). (Cited on page 35.)

Marc Chemillier. Toward a formal study of jazz chord sequences generated by Steedman's grammar. *Soft Computing*, 8(9):617–622, 2004. (Cited on pages 35 and 46.)

Marc Chemillier. L'improvisation musicale et l'ordinateur. *Terrain*, 53, "Voir la musique":67–83, 2009. URL http://terrain.revues.org/13776. (Cited on pages 8 and 35.)

Marc Chemillier. Tutorial "oracle" and "improvizer" with "beats", 2010. URL http://ehess.modelisationsavoirs.fr/atiam/improtek/index.html. [Online; accessed 23-January-2016]. (Cited on page 35.)

Marc Chemillier and Jérôme Nika. Etrangement musical : les jugements de goût de Bernard Lubat à propos du logiciel d'improvisation ImproteK. *Cahiers d'ethnomusicologie*, (28), 2015. (Cited on page 150.)

Marc Chemillier, Jean Pouchelon, Julien André, and Jérôme Nika. La contramétricité dans les musiques traditionnelles africaines et son rapport au jazz. *Anthropologie et sociétés*, 38(1):105–137, 2014. (Cited on pages 161 and 180.)

Axel Chemla-Romeu-Santos. Guidages de l'improvisation. Master's thesis, Master ATIAM - Ircam, UPMC, 2015. Supervisors: Jérôme Nika and Gérard Assayag. (Cited on pages xiv, xvi, 25, 189, 209, and 210.)

Yves Citton. Politics as hypergestural improvisation in the age of mediocracy. In George E. Lewis and Benjamin Piekut, editors, *The Oxford Handbook of Critical Improvisation Studies*. Oxford University Press, Oxford, 2013. (Cited on page 9.)

Yves Citton. Manifeste pour les hypergestes d'improvisation, 2015. URL http://penserimproviser.org/wp/

atelier-2-intervention-de-yves-citton. Second Unexpected Workshop organised by Bernard Stiegler and IRI [Online; accessed 23-January-2016]. (Cited on page 7.)

Philippe Codognet and Daniel Diaz. Yet another local search method for constraint solving. In *Stochastic Algorithms: Foundations and Applications*, pages 73–90. Springer, 2001. (Cited on page 29.)

Darrell Conklin and John G Cleary. Modelling and generating music using multiple viewpoints. 1988. (Cited on page 30.)

Darrell Conklin and Ian H Witten. Multiple viewpoint systems for music prediction. *Journal of New Music Research*, 24(1):51–73, 1995. (Cited on page 33.)

Arshia Cont. Realtime audio to score alignment for polyphonic music instruments, using sparse non-negative constraints and hierarchical hmms. In *Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on*, volume 5, pages V–V. IEEE, 2006. (Cited on page 21.)

Arshia Cont. Antescofo: Anticipatory synchronization and control of interactive parameters in computer music. In *International Computer Music Conference*, Belfast, 2008a. (Cited on pages 75, 109, 187, and 213.)

Arshia Cont. *Modeling musical anticipation: From the time of music to the music of time*. PhD thesis, Université Pierre et Marie Curie-Paris VI, 2008b. (Cited on pages 6 and 29.)

Arshia Cont. A coupled duration-focused architecture for realtime music to score alignment. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(6):974–987, 2010. (Cited on page 123.)

Arshia Cont. On the creative use of score following and its impact on research. In *Proc. International Conference on Sound and Music Computing*, Padova, 2011a. (Cited on page 99.)

Arshia Cont. On the creative use of score following and its impact on research. In *SMC 2011: 8th Sound and Music Computing conference*, 2011b. (Cited on page 21.)

Arshia Cont, Shlomo Dubnov, Gerard Assayag, et al. Guidage: A fast audio query guided assemblage. In *Proceedings of International Computer Music Conference (ICMC)*, 2007. (Cited on page 29.)

Maxime Crochemore, Christophe Hancart, and Thierry Lecroq. *Algorithms on strings*. Cambridge University Press, 2007. (Cited on pages 56 and 57.)

Roger B. Dannenberg. Real-Time Scheduling and Computer Accompaniment. In MIT Press, editor, *Current Directions in Computer Music Research*, volume 225-261, 1989. (Cited on page 31.)

Roger B. Dannenberg and Christopher Raphael. Music score alignment and computer accompaniment. *Communications of the ACM*, 49(8):38–43, 2006. (Cited on page 21.)

Philippe Depalle and Gilles Poirot. A modular system for analysis, processing and synthesis of sound signals. In *Proceedings of the International Computer Music Conference*, pages 161–164. International Computer Music Association, 1991. (Cited on page 115.)

Jacques Derrida. Joue–le prenom. *Les Inrockuptibles*, 115:41–42, 1997. (Cited on page 8.)

Jacques Derrida. Entretien par Jerôme-Alexandre Nielsberg, "Jacques Derrida, Penseur de l'événement". *L'Humanité*, 28, 2004. (Cited on page 8.)

Myriam Desainte-Catherine, Antoine Allombert, and Gérard Assayag. Towards a hybrid temporal paradigm for musical composition and performance: The case of musical interpretation. *Computer Music Journal*, 37(2):61–72, 2013. (Cited on page 135.)

Marie E. Desjardins, Edmund H. Durfee, Jr. Charles L. Ortiz, and Michael J. Wolverton. A Survey of Research in Distributed, Continual Planning. *AI Magazine*, 20(4), 1999. (Cited on page 137.)

Alexandre Donze, Ilge Akkaya, Sanjit A. Seshia, Edward A. Lee, and David Wessel. Real-time control improvisation for the smartjukebox. Technical report, 2013. URL `http://chess.eecs.berkeley.edu/pubs/1065.html`. (Cited on page 30.)

Alexandre Donzé, Rafael Valle, Ilge Akkaya, Sophie Libkind, Sanjit A Seshia, and David Wessel. Machine improvisation with formal specifications. *Proceedings of the 40th International Computer Music Conference (ICMC)*, 2014. (Cited on page 30.)

Robert L. Douglas. Formalizing an african-american aesthetic. *New Art Examiner 18 (June/Summer 1991), 18*, 24, 1991. (Cited on page 20.)

Jon Drummond. Understanding interactive systems. *Organised Sound*, 14(02):124–133, 2009. (Cited on page 18.)

Shlomo Dubnov, Gérard Assayag, and Ran El-Yaniv. Universal classification applied to musical sequences. In *International Computer Music Conference*, pages 332–340, Ann Arbor, Michigan, 1998. (Cited on pages 23 and 34.)

Shlomo Dubnov, Gerard Assayag, and Arshia Cont. Audio oracle: A new algorithm for fast learning of audio structures. In *Proceedings of International Computer Music Conference (ICMC)*. ICMA, 2007. (Cited on pages 26 and 29.)

Shlomo Dubnov, Gérard Assayag, and Arshia Cont. Audio oracle analysis of musical information rate. In *Semantic Computing (ICSC), 2011 Fifth IEEE International Conference on*, pages 567–571. IEEE, 2011. (Cited on pages 27 and 211.)

José Echeveste. Un langage temps réel dynamique pour scénariser l'interaction musicien-machine. *TSI. Technique et science informatiques*, 33(7-8):587–626, 2014. (Cited on page 35.)

José Echeveste, Arshia Cont, Jean-Louis Giavitto, and Florent Jacquemard. Operational semantics of a domain specific language for real time musician–computer interaction. *Discrete Event Dynamic Systems*, pages 1–41, 2013a. (Cited on pages 31, 75, 109, 187, and 213.)

José Echeveste, Jean-Louis Giavitto, and Arshia Cont. A Dynamic Timed-Language for Computer-Human Musical Interaction. Research report RR-8422, INRIA, 2013b. URL `http://hal.inria.fr/hal-00917469`. (Cited on pages 75, 109, 187, and 213.)

José Echeveste, Jean-Louis Giavitto, and Arshia Cont. A Dynamic Timed-Language for Computer-Human Musical Interaction. Research report RR-8422, INRIA, 2013c. (Cited on page 21.)

José-Manuel Echeveste. *Un langage de programmation pour composer l'interaction musicale: la gestion du temps et des événements dans Antescofo*. PhD thesis, Université Pierre et Marie Curie-Paris VI, 2015. (Cited on pages 99, 113, and 187.)

Arne Eigenfeldt and Philippe Pasquier. Considering vertical and horizontal context in corpus-based generative electronic dance music. In *Proceedings of the Fourth International Conference on Computational Creativity*, volume 72, 2013. (Cited on page 28.)

Aaron Einbond. CatOracle : l'improvisation concatenative avec Mubu, 2015. URL `http://medias.ircam.fr/xfb3c40`. Seminar at Ircam [Online; accessed 23-January-2016]. (Cited on page 27.)

Aaron Einbond, Christopher Trapani, and Diemo Schwarz. Precise pitch control in real time corpus-based concatenative synthesis. In *International Computer Music Conference Proceedings*, volume 2012, pages 584–588. International Computer Music Association, 2012. (Cited on page 27.)

Aaron Einbond, Christopher Trapani, Andrea Agostini, Daniele Ghisi, and Diemo Schwarz. Fine-tuned control of concatenative synthesis with catart using the bach library for max. In *Proceedings of the International Computer Music Conference*, pages 1037–1042, 2014. (Cited on page 27.)

Moses I Finley. *The world of Odysseus*. New York Review of Books, 1954. (Cited on page 5.)

R. James Firby. An Investigation into Reactive Planning in Complex Domains. In *6th National Conference on Artificial Intelligence*, pages 202–206, Seattle, WA, USA, 1987. (Cited on page 137.)

Alexandre RJ François, Isaac Schankler, and Elaine Chew. Mimi4x: an interactive audio–visual installation for high–level structural improvisation. *International Journal of Arts and Technology*, 6(2):138–151, 2013. (Cited on pages 19 and 24.)

Daniel J Fremont, Alexandre Donzé, Sanjit A Seshia, and David Wessel. Control improvisation. *arXiv preprint arXiv:1411.0698*, 2014. (Cited on page 30.)

Kyle Gann. *The Music of Conlon Nancarrow*, volume 7. Cambridge University Press, 2006. (Cited on page 190.)

Fiammetta Ghedini, François Pachet, and Pierre Roy. Creating music and texts with flow machines. In *Multidisciplinary Contributions to the Science of Creative Thinking*, pages 325–343. Springer, 2016. (Cited on pages 19 and 30.)

Jean-Louis Giavitto, Arshia Cont, and José Echeveste. Antescofo a not-so-short introduction to version 0.x. 2015. URL `support.ircam.fr/docs/Antescofo/AntescofoReference.pdf`. (Cited on page 124.)

Dorien Herremans, Stephanie Weisser, Kenneth Sörensen, and Darrell Conklin. Generating structured music for bagana using quality metrics based on markov models. *Expert Systems with Applications*, 42(21):7424–7435, 2015. (Cited on page 28.)

Gregory Hickok. Computational neuroanatomy of speech production. *Nature Reviews Neuroscience*, 13(2):135–145, 2012. (Cited on page 9.)

Mantle Hood. The challenge of "bi-musicality". *Ethnomusicology*, 4 (2):55–59, 1960. (Cited on page 143.)

David Brian Huron. *Sweet anticipation: Music and the psychology of expectation*. MIT press, 2006. (Cited on page 6.)

Steven Johnson. *The New York Schools of Music and the Visual Arts*. Routledge, 2012. (Cited on pages xiv and 3.)

Sergi Jordà. *Digital Lutherie: Crafting Musical Computers for New Musics' Performance and Improvisation*. PhD thesis, Universitat Pompeu Fabra, Barcelona, 2005. (Cited on page 18.)

Mark Kahrs. Dream chip 1: A timed priority queue. *IEEE Micro*, 13(4): 49–51, 1993. (Cited on page 31.)

Peter E Keller. Joint action in music performance. *Emerging Communication*, 10:205, 2008. (Cited on page 9.)

Donald E. Knuth, James. H. Morris, and Vaughan. R. Pratt. Fast pattern matching in strings. *SIAM journal on computing*, 6(2):323–350, 1977. (Cited on page 58.)

Edward W Large. Periodicity, pattern formation, and metric structure. *Journal of New Music Research*, 30(2):173–185, 2001. (Cited on page 117.)

Edward W Large and Mari Riess Jones. The dynamics of attending: How people track time-varying events. *Psychological review*, 106 (1):119, 1999. (Cited on page 117.)

Olivier Lartillot and Petri Toiviainen. A Matlab toolbox for musical feature extraction from audio. In *International Conference on Digital Audio Effects*, pages 237–244, 2007. (Cited on page 210.)

Arnaud Lefebvre and Thierry Lecroq. Computing repeated factors with a factor oracle. In *Proceedings of the 11th Australasian Workshop On Combinatorial Algorithms*, pages 145–158, 2000. (Cited on page 60.)

Arnaud Lefebvre, Thierry Lecroq, and Joël Alexandre. Drastic improvements over repeats found with a factor oracle. *Proceedings of the 13th Autralasian Workshop on Combinatorial Algorithms*, pages 253–265, 2002. (Cited on pages 23 and 59.)

Benjamin Lévy. *Principes et architectures pour un système interactif et agnostique dédié à l'improvisation musicale*. PhD thesis, Université Pierre et Marie Curie-Paris VI, 2013. (Cited on pages xiv and 24.)

Benjamin Lévy, Georges Bloch, and Gérard Assayag. OMaxist dialectics. In *International Conference on New Interfaces for Musical Expression*, pages 137–140, 2012. (Cited on pages 23, 24, and 35.)

George E Lewis. Too many notes: Computers, complexity and culture in Voyager. *Leonardo Music Journal*, 10:33–39, 2000. (Cited on pages 19, 21, and 65.)

George E Lewis. Improvisation and the orchestra: A composer reflects. *Contemporary Music Review*, 25(5-6):429–434, 2006. (Cited on page 4.)

George E Lewis. Foreword: After afrofuturism. *Journal of the Society for American Music*, 2(02):139–153, 2008. (Cited on page 20.)

Bernard Lortat-Jacob. Formes et conditions de l'improvisation dans les musiques de tradition orale. In Jean-Jacques Nattiez, editor, *Musiques, une encyclopédie pour le XXIe siècle. Volume 5: L'unité de la musique*. Actes Sud / Cité de la musique, 2007. (Cited on pages 2 and 9.)

Bernard Lubat. Interview in Marc Chemillier's seminar "Modeling of musical knowledge based on orality", 2010. URL ehess.modelisationsavoirs.fr/seminaire/seminaire09-10/seminaire09-10.html#bernard. [Online; accessed 23-January-2016]. (Cited on page 147.)

Pascal Maigret. Reactive Planning and Control with Mobile Robots. In *IEEE Control*, pages 95–100, 1992. (Cited on page 31.)

Fivos Maniatakos. *Graphs and Automata for the Control of Interaction in Computer Music Improvisation*. Thèse de doctorat, Université Pierre et Marie Curie, 2012. (Cited on page 63.)

Philippe Manoury. La note et le son. *L'Harmattan*, 1990. (Cited on page 4.)

James McCartney. Supercollider: a new real-time synthesis language. In *International Computer Music Conference*, Hong Kong, 1996. (Cited on page 99.)

Eduardo Reck Miranda. At the crossroads of evolutionary computation and music: Self-programming synthesizers, swarm orchestras and the origins of melody. *Evolutionary Computation*, 12(2):137–158, 2004. (Cited on page 22.)

Julian Moreira, Pierre Roy, and François Pachet. Virtualband: interacting with stylistically consistent agents. In *Proc. of International Society for Music Information Retrieval Conference*, pages 341–346, Curitiba, 2013. (Cited on pages 19, 26, and 33.)

James. H. Morris and Vaughan. R. Pratt. A linear pattern-matching algorithm. Technical report, University of California, Berkeley, 1970. (Cited on page 55.)

Jérôme Nika. Intégration de contrôles harmoniques et rythmiques dans un processus informatique d'improvisation musicale. Master's thesis, Telecom ParisTech - ENSTA ParisTech, 2012. (Cited on page 160.)

Jérôme Nika and Marc Chemillier. Improtek: integrating harmonic controls into improvisation in the filiation of OMax. In *International Computer Music Conference (ICMC)*, pages 180–187, 2012. (Cited on pages 35, 59, and 152.)

François Pachet. The continuator: Musical interaction with style. *Journal of New Music Research*, 32(3):333–341, 2003. (Cited on pages 23 and 33.)

François Pachet and Pierre Roy. Markov constraints: steerable generation of markov sequences. *Constraints*, 16(2):148–172, 2011. (Cited on page 27.)

François Pachet, Pierre Roy, Julian Moreira, and Mark d'Inverno. Reflexive loopers for solo musical improvisation. In *SIGCHI Conference on Human Factors in Computing Systems*, pages 2205–2208, ACM, Paris, 2013. (Cited on pages 19, 26, and 33.)

Alexandre Papadopoulos, Pierre Roy, and François Pachet. Avoiding plagiarism in Markov sequence generation. 2014. (Cited on pages xiv and 27.)

Hermeto Pascoal. *Calendário do som*. Senac, 2000. URL `http://www.hermetopascoal.com.br`. [Online; accessed 23-January-2016]. (Cited on page 178.)

Andrea Perrucci, Francesco Cotticelli, and Thomas F Heck. *A Treatise on Acting, from Memory and by Improvisation (1699)*. Scarecrow Press, 2008. (Cited on page 7.)

Jeff Pressing. Cognitive processes in improvisation. *Advances in Psychology*, 19:345–363, 1984. (Cited on pages 5 and 20.)

Wolfgang Prinz. Perception and action planning. *European journal of cognitive psychology*, 9(2):129–154, 1997. (Cited on page 9.)

Miller Puckette. Combining event and signal processing in the max graphical programming environment. *Computer Music Journal*, 15:68–77, 1991. (Cited on pages 75, 87, 99, 109, 187, and 213.)

Mathieu Ramona, Giordano Cabral, and François Pachet. Capturing a musician's groove: Generation of realistic accompaniments from single song recordings. In *Proceedings of the 24th International Conference on Artificial Intelligence*, pages 4140–4141. AAAI Press, 2015. (Cited on page 30.)

Axel Röbel. A new approach to transient processing in the phase vocoder. In *Proc. of the 6th Int. Conf. on Digital Audio Effects (DAFx03)*, pages 344–349, 2003. (Cited on page 115.)

Xavier Rodet, Pierre Cointe, Jean-Baptiste Barriere, Yves Potard, Bernard Serpette, and Jean-Pierre Briot. Applications and developments of the formes programming environment. In *Proceedings of the 1983 International Computer Music Conference, Computer Music Association*, 1983. (Cited on page 137.)

Robert Rowe. *Interactive music systems: machine listening and composing*. MIT press, 1992. (Cited on pages 18 and 19.)

Robert Rowe. The aesthetics of interactive music systems. *Contemporary music review*, 18(3):83–87, 1999. (Cited on pages 33 and 65.)

Robert Rowe. Split levels: Symbolic to sub-symbolic interactive music systems. *Contemporary Music Review*, 28(1):31–42, 2009. (Cited on page 32.)

Pierre Roy and François Pachet. Enforcing meter in finite-length markov sequences. In *AAAI*, 2013. (Cited on page 27.)

William Russo. *Jazz composition and orchestration*. University of Chicago Press, 1997. (Cited on pages xiv and 6.)

Matthieu Saladin. La partition graphique et ses usages dans la scène improvisée. *Volume!. La revue des musiques populaires*, (3: 1):31–57, 2004. (Cited on page 4.)

Dario Sanfilippo and Andrea Valle. Feedback systems: An analytical framework. *Computer Music Journal*, 37(2):12–27, 2013. (Cited on page 32.)

Norbert Schnell, Axel Röbel, Diemo Schwarz, Geoffroy Peeters, Riccardo Borghesi, et al. *MuBu and friends–Assembling tools for content based real-time interactive audio processing in Max/MSP*. Ann Arbor, MI: MPublishing, University of Michigan Library, 2009. (Cited on page 27.)

Arnold Schoenberg and Leonard Stein. *Structural functions of harmony*. Number 478. WW Norton & Company, 1969. (Cited on page 5.)

Diemo Schwarz. *Data-Driven Concatenative Sound Synthesis*. PhD thesis, Ircam - Centre Pompidou, Paris, France, January 2004. (Cited on page 29.)

Diemo Schwarz. State of the art in sound texture synthesis. In *Proc. Digital Audio Effects (DAFx)*, pages 221–231, 2011. (Cited on page 28.)

Diemo Schwarz, Nicola Orio, and Norbert Schnell. Robust polyphonic midi score following with hidden markov models. In *International Computer Music Conference (ICMC)*, pages 1–1, 2004. (Cited on page 21.)

Diemo Schwarz, Grégory Beller, Bruno Verbrugghe, and Sam Britton. Real-time corpus-based concatenative synthesis with catart. In *9th International Conference on Digital Audio Effects (DAFx)*, pages 279–282, 2006. (Cited on page 28.)

L. Henry Shaffer. Analysing piano performance: A study of concert pianists. *Advances in Psychology*, 1:443–455, 1980. (Cited on page 5.)

Chuck Sher. The new real book: jazz classics, choice standards, pop-fusion classics: for all instrumentalists and vocalists. 2005. (Cited on pages xiv and 2.)

George Sioros and Carlos Guedes. Complexity driven recombination of midi loops. In *International society for music information retrieval (ISMIR 2011)*, pages 381–386, 2011a. (Cited on page 21.)

George Sioros and Carlos Guedes. A formal approach for high-level automatic rhythm generation. In *Proceedings of the BRIDGES 2011: Mathematics, Music, Art, Architecture, Culture Conference. Coimbra, Portugal.*, 2011b. (Cited on page 21.)

Jacques Siron. L'improvisation dans le jazz et les musiques contemporaines : l'imparfait du moment présent. In Jean-Jacques Nattiez, editor, *Musiques, une encyclopédie pour le XXIe siècle. Volume 5: L'unité de la musique*. Actes Sud / Cité de la musique, 2007. (Cited on page 8.)

John A Sloboda. Music performance. *The psychology of music*, pages 479–496, 1982. (Cited on page 5.)

David Sudnow. *Ways of the hand: The organization of improvised conduct*. MIT Press, 1978. (Cited on page 7.)

Greg Surges, Tamara Smyth, and Miller Puckette. Generative feedback networks using time-varying allpass filters. *Proceedings of the 2015 International Computer Music Conference*, 2015. (Cited on page 31.)

Gregory Surges. *Generative Audio Systems: Musical Applications of Time-Varying Feedback Networks and Computational Aesthetics*. PhD thesis, University of California San Diego, 2015. (Cited on page 32.)

Gregory Surges and Shlomo Dubnov. Feature selection and composition using PyOracle. In *Ninth Artificial Intelligence and Interactive Digital Entertainment Conference*, Boston, Massachusetts, 2013. (Cited on pages 19 and 26.)

Belinda Thom. *BoB: An improvisational music companion*. PhD thesis, Bell Labs, 2001. (Cited on pages 19 and 33.)

Margaret E Thomas. Nancarrow's canons: Projections of temporal and formal structures. *Perspectives of New Music*, pages 106–133, 2000. (Cited on page 190.)

Walter Thompson. *Soundpainting: the art of live composition. Workbook I*. Walter Thompson, 2006. (Cited on page 21.)

Christopher Trapani and José Echeveste. Real time tempo canons with antescofo. In *International Computer Music Conference*, page 207, 2014. (Cited on page 190.)

Eric Cesar Jr. Vidal and Alexander Nareyek. A Real-Time Concurrent Planning and Execution Framework for Automated Story Planning for Games. In *AAAI Technical Report WS-11-18*, 2011. (Cited on page 137.)

Cheng-i Wang and Shlomo Dubnov. Guided music synthesis with variable markov oracle. *Tenth Artificial Intelligence and Interactive Digital Entertainment Conference*, 2014a. (Cited on page 29.)

Cheng-i Wang and Shlomo Dubnov. Variable markov oracle: A novel sequential data points clustering algorithm with application to 3d gesture query-matching. In *Multimedia (ISM), 2014 IEEE International Symposium on*, pages 215–222. IEEE, 2014b. (Cited on page 29.)

Ge Wang. *The ChucK audio programming language." A strongly-timed and on-the-fly environ/mentality"*. PhD thesis, Princeton University, 2009. (Cited on page 99.)

Matthew Wright, Adrian Freed, et al. Open sound control: A new protocol for communicating with sound synthesizers. In *Proceedings of the 1997 International Computer Music Conference*, number 8, page 10. International Computer Music Association San Francisco, 1997. (Cited on page 96.)

Aymeric Zils and François Pachet. Musical mosaicing. In *Digital Audio Effects (DAFx)*, volume 2, 2001. (Cited on page 29.)