Knitting and Weaving: Using OpenMusic to Generate Canonic Musical Material

Serge Lemouton and Asbjørn Schaathun

This chapter presents some compositional procedures developed by Asbjørn Schaathun for a generation of melodic material using canonic processes and for the calculation of rhythmic interpolations. These processes are derived from earlier computer programs elaborated by the composer and recently ported to OpenMusic. The violin concerto *Double Portrait* is an example of the use of such processes.

Early experiments

Asbjørn Schaathun is a long-time user of CAC systems; he first started to use this kind of software during the autumn of 1983 at the EMS-studios in Stockholm (at that time part of the Swedish Radio). There he met people who helped him understand Xenakis’ *Stochos-program*. He found in Xenakis a way of thinking that, he intuitively understood, could help him construct large-scale musical forms in which the rather microscopic constitutive parts of complex textures could be mapped out according to a global plan. Xenakis’ influence was crucial in Schaathun’s piano concerto *Musical Graffiti (II)*, notably in a passage of the piece where the latter created a cloud of vibrant, iterating rhythms for the orchestra, or in another passage where he composed a palette of intervallic relationships with the help of a stochastic algorithm.¹

During the summer of 1984, he attended the Darmstadt Ferienkurse where he met Magnus Lindberg and his friends. They were trying to notate improvisational rhythms with greater flexibility with the aid of Logo on an Apple II machine. So, from the autumn of 1984, when he purchased his first microcomputer, a colour Amstrad CPC 464, with double disc-drive(!), a CP/M operating system and a dot-matrix printer, Schaathun started to develop small routines for Computer-Aided Composition.

This resulted in the program package named *Rhythmmod* (from modulation of rhythmical streams). Since the computer had only a 64K memory, he had to load small routines in and out of the machine. This tedious work not only resulted in pieces like the Gaudeamus prize-winning concerto for bass clarinet and large ensemble *Actions, Interpolations and

¹Later on, with the Swedish composer and programmer Bo Rydberg, Schaathun designed a program, *Scha-tune*, to a certain extent similar to Xenakis’ *Stochos-program*, for a huge piece for percussion that never came to life.
Analyses (1988-90), but it made Schaathun realise that, in order to renew his language
and nurture his creative flow, he had to challenge the often intuitively rigid and habit-
orientated choices one repeatedly makes. In other words, he discovered that the computer
was the perfect medium to break one’s own patterns and to upset one’s aesthetical habits.

The small routines in Rhythmmod dealt with the two main compositional parameters,
duration and pitch, in order to generate surprises or “accidents” when merging different
streams of these two parameters. This parametrical stream was often created by impro-
vising rhythms on the keyboard of the computer and by playing some favourite chords
or webernesque melodic fragments on the synthesizer. The main goal was to achieve a
kind of mediation between improvisation and structural work.

Schaathun’s very first application for PatchWork was written early in 1994, following
his first stay at IRCAM in 1991-92. His first usage of this system involved the Esquisse
library, with which he attempted to transfer and extend his ideas of harmonies and pitches
to the quarter-tone realm. This first attempt was not very successful, and he finally had
to accept that he was much more interested in developing material from a chromatic and
horizontal outset into what, for the time being, he had coined “twisted melodies”. This
concept was, amongst others, inspired by Stockhausen’s new interest in melodies as a
form-building element as in his formula compositions from the early seventies. Was it
possible to develop new musical material and compositional methods from this melodic
point of departure?

Recent works using OpenMusic: Double Portrait

In 2006, Ashbjørn Schaathun was invited to IRCAM to create a new version of his violin
concerto Double Portrait. On that occasion, we decided to port some of the routines of
his compositional environment to OpenMusic.²

While in his earlier attempts he had concentrated on the transformation of rhythms
and chords through interpolation [1], he recently explored other avenues of research.
Instead of using stochastic processes to build material, he now elaborates textures from
complexes of multiplied (melodic) and polyphonic lines. In the end, if the lines grow
dense enough, they can even constitute a “block” of sound. Using different methods of
filtering, it is then possible to “carve out” simpler material in various ways from this
block. So the composer can also experiment with various degrees of recognition of the
underlying material. (In fact in this case “composing” resembles “de-composing” in the
manner of a sculptor.)

This is easily exemplified and illustrated by employing already known material. The
examples shown below are tested with small melodic fragments (extracted from the Gold-
berg Aria). This material is used as a demonstration and to test the functions of the
computer program. In real compositions, Schaathun usually elaborates himself the pri-
mary material.

²For the first version of Double Portrait (1991/92) a lot of Schaathun’s early routines were ported
to Max. These routines were also extended with a view to serve as programs for signal processing.
However, the idea of deriving the “drivers” from the actual compositional programs of the piece gave
rise to far too complex and refined live-electronics. Because of the complexity and because of some
important miscalculations in the balancing and grouping of the orchestra, Schaathun was not satisfied
and withdrew the score from publication right after its premiere in Paris in April 1992.
Canons and weaving

The example in Figure 1 is a patch that allows the realisation of classical canonic imitations. This patch generates a simple canon from a melodic subject. The first parameter controlled by the user is of course the number of voices in the canon. Four other parameters can be modified in this patch. They are given the following names in French:

- **diminutions**: a time factor which controls the pace of each voice (1 corresponds to the original tempo, 2 to two time faster, etc.)
- **entrées**: delays in milliseconds before the start of each voice.
- **transpositions**: transpositions in microns (it is possible to experiment with microintervallic transpositions).
- **formes**: assigns a melodic form to each voice (1 = original, 2 = retrograde, 3 = inversion, 4 = inversion-retrograde).

![Diagram of patch for the realisation of simple canons.](image)

**Figure 1.** Patch for the realisation of simple canons.

Each voice in the canon is thus a transformed version of the initial musical subject. Each transformation (diminution, delay, transposition, etc.) is computed by the “lambda-mode” patches as noted in Figure 1. It is also possible to program some other, more elaborate transformations that will be automatically applied to each voice in the canon.
The patch Goldberg 1087 (Figure 3) shows an example based on the last enigmatic canon in Bach’s Goldberg Variations (BWV 1087), the Canon 14 (Canon a 4 per Augmentacionem et Diminutionem), processed through the patch presented above. Figure 2 shows the melody Bach composed for this canon.

**Figure 2.** An enigmatic canon on the Goldberg ground: Bach’s proposition.

This canon is for four voices in rhythmic proportions. Bach labelled it “canon for 4 voices in augmentation and diminution”. Contrary motion is also applied to make the Goldberg ground visible. With the canon generation patch (Figure 3), it is possible to modify heuristically the parameters of the canon, and to try to find some other solutions that even Bach himself might not have imagined!

**Figure 3.** OpenMusic realisation of BWV 1087-14.
The process of generating canons is developed further in the *veving*\(^3\) patch (Figure 4). In this example there are more parameters (i.e. functions to apply to the different voices) than the four ones mentioned above. These are:

- **time map**: the tempo of each voice can be modulated by a sinusoidal function, of which it is possible to set the period (in seconds), and the amplitude (in %). In this way one can modulate the rhythm of a given melodic stream with a kind of imaginary very unstable conductor!
- **loop voices**: the length of time (in seconds) during which the subject is repeated.

---

**Figure 4.** Example of a *veving* process.

\(^3\)Norwegian for weaving.
This patch takes as a point of departure the canon and the structure of the patch in Figure 1. However, the aim here is not to generate contrapuntal polyphony but instead to create a single melodic line from the merging of all the independent voices. (This possibility is of course available in canons too.) The \textit{lus-merger} box is responsible for the blending of all the voices into one. This Lisp function is a generalisation of the OpenMusic \textit{merge} method that allows the merging of lists of voices.

In this way the composer can build up fairly complex melodic lines, where each line is composed (literally!) of elements which in a very subtle and musical way give clues about the melodic line from which they derive.

**Knitting : canons centred around a pivot**

The process called \textit{strikking} is a variation on the preceding one where all the voices are synchronized from a central point of reference (see Figure 6). For this purpose, the starting point for each voice is calculated automatically.

Figure 5 shows a patch implementing this “knitting” process: an additional sub-patch (\textit{calcul-offsets}) automatically computes the delays in the entries of the voices.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{patch_strikking.png}
\caption{Example of the \textit{strikking} process.}
\end{figure}

\footnote{Norwegian for knitting.}
This process is very well suited to more “abstract” material such as contours, or melodic threads (melodies without rhythms). In this way one can produce, at least in the rhythmic domain, organic macro-symmetrical textures.

**Figure 6.** Central synchronisation.

### Rhythmic interpolations

The rhythmic interpolation process developed by Schaathun (as described in [1]) allows to move from one rhythmic cell to the other by interpolation, no matter the content of the initial rhythmic cells. This process is illustrated in Figure 7. Given two lists of durations, some “void” durations are randomly inserted in the shortest one so that each list is of the same length. Each pair of durations from these lists is, in turn, interpolated.

![Diagram](https://via.placeholder.com/150)

**Figure 7.** Rhythmic interpolation process (illustration from [1]).

The short durations are not expressed in the interpolated rhythms until they reach a threshold value. Thus, when during the transformation the duration exceeds the threshold value, the flow of durations suddenly changes and creates outbursts of energy through grace notes and appoggiaturas.
The patch in Figure 8 implements this interpolation process in OpenMusic.

![Image of a patch with musical notation and labels](image)

**Figure 8.** Rhythmic interpolation patch.

**Applications in Double Portrait**

Almost all the pieces by Asbjørn Schaathun since 1983/84, beginning with the piano concerto *Musical Graffiti (II)*, involve computer-aided composition and/or live-electronics developed along the same lines as CAC programs.

In *Double Portrait*, the violin sound is transposed by five harmonizers. Each harmonizer plays some melodic sequences. The goal of these harmonizers is to create a kind of polyphony by demultiplying the violin line, which in itself is already a demultiplication and proliferation – via weaving and canonic processes – of an underlying opening pitch-formula. These modulating sequences or melodies, which are fed into the harmonizer as transposing intervals, are generated from melodic contours created with the help of all of the aforementioned processes.

The harmonizer-files-generator patch (Figure 9) shows how these melodic lines are created. Seven opening formulas (the basic shapes of which are here taken from the opening of Berg’s violin concerto) are reduced to constitute the initial melodic profiles
(sub-patch contours), while durations are generated using rhythm interpolation processes. The resulting lines are in turn subjected to a canonic process such as that in Figure 4 (in the sub-patch canon) where they are chained together (loop-voices control) in strings of randomly chosen “row-forms” (forme control: original, inversion, retrograde or inversion-retrograde) and variously transposed (transposition control) using the intervals of different four- or five-note chords. The resulting canon is then reduced to a single melodic line.

![Figure 9](image)

**Figure 9.** Generation of a melodic line for the harmonizers in *Double Portrait* using weaving processes.

The chords used in the transposition constitute the different harmonic fields and the different main sections of *Double Portrait*. The intervals in these four- or five-note chords get larger as the piece progresses: the small intervals and close positions in the beginning become larger intervals and thus wider transposing intervals near the end of the piece.

The speeds of the different basic melodies are also carefully controlled by a rhythmic interpolation process (described above), which increases the activity and thus the density of the added events. There is a gradual process of intensification in the piece as the pitches from the five harmonizers get more and more rhythmically active and the intervallic leaps get larger. However, the electronics representing the “other personality” of the violin, is never more than a colouring shadow in this *double portrait*.
Bibliography