Re-Generating Stockhausen's "Studie II" in Csound

A Study About Algorithmic Composition

Joachim HEINTZ
Incontri - Institute for new music, HMT Hannover
Emmichplatz 1
30175 Hannover, Germany
jh@joachimheintz.de

Abstract

Stockhausen's "Studie II" (1954) is one of the classical works of electronic music. Written at a time when Computers played no role in the production of sound, it exhibits a way of composing which is quite similar to programming. This can be shown by re-programming the complete piece just with the input of five numbers. Beside this reduction - which is made in Csound - the compositorial decisions come to the fore, showing a flexibility and variability of algorithms[1] which can be inspiring and challenging still today.

Keywords

Music Composition, Audio Programming, Csound, Algorithmic Composition, History of Electronic Music

1 Introduction

Stockhausen's *Studie II* which was composed and realized in 1954, is well known as the perhaps earliest and strictest application of the serial technique of composition[2] to a piece of electronic music. The interest of Stockhausen and other serial composers on working with synthesized sounds suggests itself, because only in this field can also the timbre be composed in a serial way. But this paper wants to point out yet another relationship between this piece and electronic music: the close correlation between serial composing and programming. As the serial technique can be understood as a bundel of definable rules, it can be programmed.

By doing this, two different goals are persued. One is more sportive: proving the thesis of programmability by re-programming *Studie II* in practice, based on a series of 5 numbers as input. The second one is more important as a study about composing in general and about algorithmic composing in particular: How is the "machine" built? Is there something like "put n formulas in it and let it play for m seconds", or are there changes? Did the composer react to the result he sees (or hears) in the process of composition, or is there an automatism? And how can the irregularities we see in the score be judged: as errors, or intentional decisions, or both?

2 Building the Musical Material

When a composer wants to write a piece of purely electronic music, a word from the bible may come to his mind: "And the earth was without form, and void" (Genesis 1, 2). There is nothing which is self-confident; no ambitus, no scales, no chords, no times. If one doesn't want to use scales and durations "because they are there", one has to build ones own structure from the "void". So the first task in re-generating *Studie II* is to repeat what Stockhausen did in preparing the musical material:

a) to build a collection of series which are used for many different purposes, and

b) to build a pool of distinct values for frequencies, durations and intensities.

---

[1] I use the terms "algorithm" or "algorithmic composition" in a wide, general way. Algorithm in my use is an instruction which can be executed in a definite way to come from a state A (the input) to a state B (the output). I call "algorithmic composition" a way of composing music in which the whole generation of pitches, durations, sounds etc. can be described by a set of such definite executable instructions. In terms of computers, it means: it can be programmed. So "algorithmic composition" is in my understanding in no way restricted to a simple method like: "apply this formula(s) to these notes and you get the piece." In contrary, the algorithms can be different and complex, and they are derived from musical decisions. If the term algorithm is used in a strictly formula-like mathemathical way of changing state A to state B, it should not be applied to Stockhausen's Studie II (thanks to Gottfried Michael Koenig for pointing me on the issue of the use of "algorithmic" etc. in this piece).

[2] "Serialism" means "integral serialism" here: organizing all relevant parameters (not just the pitch) by series.

2.1 Building the Series (Number Squares)

The main input for *Studie II* is a serie of 5 numbers: 3 5 1 4 2. By a number of procedures, this series is expanded to a set of 2 x 5 "Number Squares", each consisting of 5 x 5 numbers. The first one, as an example, is the following:

3	5	1	4	2
5	2	3	1	4
1	3	4	2	5
4	1	2	5	3
2	4	5	3	1

For the re-generation of *Studie II*, these collections of 25 numbers (each building one series) are stored in an array for further use. In Csound, an array can be built as a "function table". So practically the program does at this point the following:

a) It defines functions (in Csound: "User Defined Opcodes") which perform the various modifications: from the starting 5 numbers to the first square (series), and then by a different method from the first square to the next four, and then by an again different method to the next five ones.

b) It fills 10 function tables with the results.

As a simple example, this is one of the functions for transforming:

```
  opcode SS2_TP1Val, i, iiii
;calculates a new value from ival and the
transposition interval interv, within the
scope of iminval and imaxval
ival, interv, iminval, imaxval xin
ivalneu   =         ival+interv
ires      =         (ivalneu>imaxval?\
  ivalneu-imaxval : (ivalneu<iminval?\
  ivalneu+imaxval : ivalneu))
          xout      ires
  endop
```

If this function is applied in a loop which looks for the distance ("transposition") between one number and the next, it returns the new value. So by

```
ival      tab_i          indx, ift
indxnext   =          (indx+1 < iftlen ?\
             indx+1 : (indx+1) - iftlen)
ivalnext   tab_i      indxnext, ift
interv     =          ivalnext - ival
ires       SS2_TP1Val itransval,\
             interv, iminval, imaxval
           tabw_i  itransval,indx,ifttrans
itransval  =          ires
           loop_lt   indx, 1, iftlen, loop
```

the function table ift = [3 5 1 4 2] is transformed to the new function table ifttrans = [5 2 3 1 4], if the starting value is 5. If this procedure is repeated with the starting values 3 5 1 4 2 (by which the starting sequence of 5 numbers are simply read vertically), the first number square (see above) is produced.[3]

2.2 Building the Pools for Frequencies, Durations and Intensities

For the generation of possible values for frequencies, Stockhausen used a method which is quite similar to the well-known approach of an equal tempered scale. The partition of the octave in 12 equal steps can be described as 12th root of 2 as the relation for one frequency to the next possible. Stockhausen used essentially the same method, but as he was possessed with the number 5 in this piece, and also wanted to build new harmonies instead of the usual cromatic scale, he decided to take the 25th root of 5 (or in other words, the 5*5th root of 5 ...). This division of the double octave plus pure major third in 25 equal steps gives a multiplier of 1.066495... - which is quite close to the cromatic semitone of 1.059463.... So his gain is to have something rather common which nevertheless leads to new sounds.

It's easy to build this scale by recursively multiplying the starting value of 100 Hz with $5^{1/25}$. It is stored again in a function table.

The durations are built by the same method, starting with 2.5 cm (0.5 * 5 ...) as the minimal length of the tape,[4] and recursively multiplying by $5^{1/25}$ for getting 61 different durations.

For practical reasons, the intensities are built from 0 to -30 dB in one dB steps. For bringing this table to the same length as the frequency and duration table, the dB values (exept 0 dB) are mirrored, so that also the intensities are collected in a table of 61 values.

3 Building the Events in the Five Parts

The collection of 10 series - each of them containing sequences of the numbers from 1 to 5 - and the pool of frequencies, durations and intensities is now used to build the events. The whole piece is clearly divided in five parts; the length of each part is given by a certain run of series and different meanings or interpretations of them.

---

[3]The methods for building the next four number squares from the first one are different and more complicated. I can't describe them here. The number squares 6-10 are built by reversing the first five ones.

[4]The tape speed was 76.2 cm/sec, so 2.5 cm equates to approximately 1/30 second.

3.1 First Part

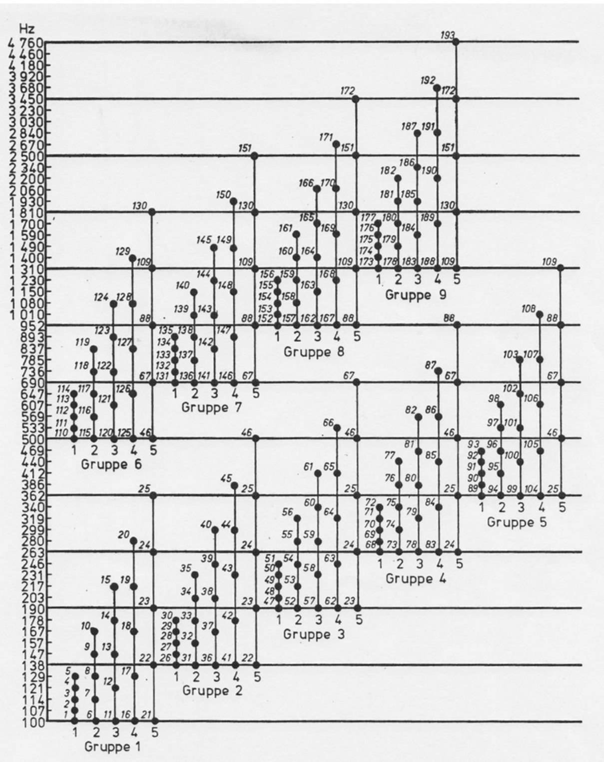
The whole piece can be considered as a collection of in total 5 * 75 + 5 = 380 single events. Each event needs six determinations:

a) the base frequency
b) the mixture built on this frequency
(there are *5* mixtures ...)
c) the duration
d) the start time
e) the maximum intensity
f) the envelope

The method which Stockhausen used for obtaining the data for the events can't be seen as simple and straightforward. It is rather complex, giving the impression of a kind of organic structure, organic movement, as if there are worms which move under certain circumstances. I will describe the first part more in detail, so that then the other parts can be discussed in relation and deviation to it.

3.1.1 Frequencies and Mixtures

For obtaining a base frequency for an event, the frequency table (cf. 2.1) is structured by 9 groups[5]:



Now four series are used to get a frequency and a mixture. The first series (A) determines one of the 9 groups in abstract. Let the first value of this serie be 3. Then a "transposition" value (B) is applied to this. This "transposition" works like a transposition in music: a transposition by 1 (prime) lets the note as it is, a transposition by 2 transposes it 1 step up, and so on. Let this transposition value be also 3. The result of the abstract group value, transposed by the transposition value (A t B) is then 5. This is the number of the group. Then two other series are used to determine the column of the group (C) and the position in this column (D). Let the values of the column and the position be 5 respective 3, then we have as the base frequency of the first mixture 690 Hz. The kind of mixture equals the column value, so we get mixture number 5 and by this the frequencies of the other four sine tones above 690 Hz as 952, 1310, 1810 and 2500 Hz.

It is important that the four series are not running at the same speed but asynchronously. There is one "superior" series (S) which determines how many repetitions of A and C are performed. For instance, if the first five values of A are 3 5 1 4 2, and the values of S are 2 4 5 3 1, the result is A with these repetitions:

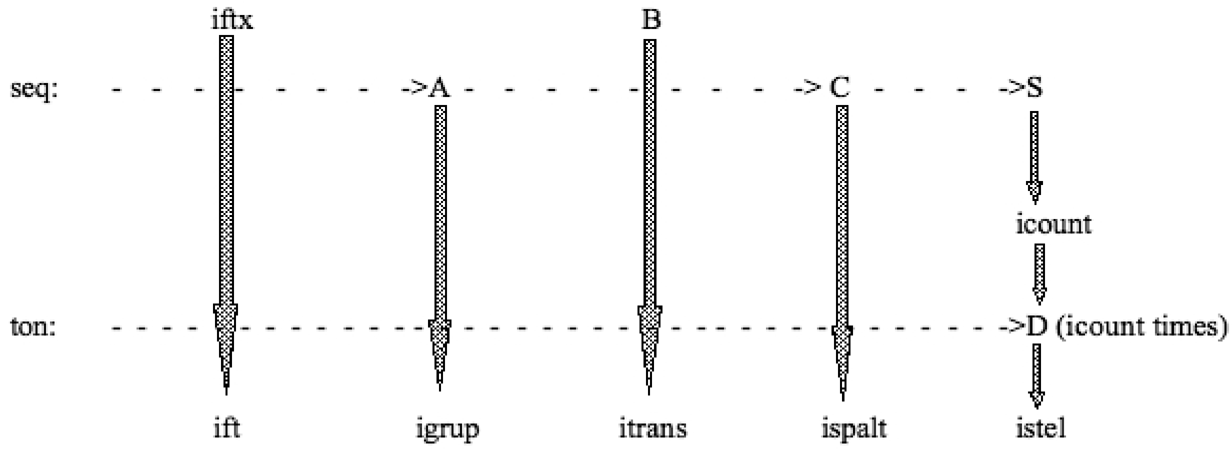
A: 3 3 | 5 5 5 5 | 1 1 1 1 1 | 4 4 4 | 2
S: 2 4 5 3 1 (repetitions)

This is the way A and C are running. The transposition value B does not change during the whole part.[6] And the series D (giving the positions) is running entirely without repetitions. So there are three different movements: A and C are moving in a "regular speed", ruled by S; B is not moving at all; D is moving fast. In writing a function for this, it results in two nested loops:

---

[5] Heinz Silberhorn, Die Reihentechnik in Stockhausens Studie II, Rohrdorfer Musikverlag 1980, S. 17

[6] Obviously it was the plan to have one transposition value for each part: 3 5 1 4 2. But Stockhausen changed this plan later (see 4.3).



In Csound Code:

```
indxcount      =                   0
indxtonabs     =                   0
seq:
icount         tab_i               indxcount, iftcount
igrup          tab_i               indxcount, iftgrup
ispalt         tab_i               indxcount, iftspalt
indxton        =                   0
ton:
istel          tab_i               indxtonabs, iftstel
ival           SS2_ValsAusGTSS     iftbas, igrup, itrans, ispalt, istel
               tabw_i              ival, indxtonabs, iftout
indxtonabs     =                   indxtonabs + 1
               loop_lt             indxton, 1, icount, ton
               loop_lt             indxcount, 1, ftlen(iftcount), seq
```

3.1.2 Durations and Starts of Events

The method for obtaining the durations is essentially the same as for the frequencies. So, in the code the same function (called SS2_MkParamTab_Meth1) could be used. It returns a function table which contains the durations of the 75 events of the first part.

As for the starting times of the events, two different decisions have been made. The first one is a general one for all the event starts in *Studie II*: A second series of durations is generated. But these durations are not read as durations, but as starting points: they are imagined to succeed each other immediately, and the resulting times are captured. For instance, if these ghost-duration serie had the values 20.9, 45.3, 104.6, 71.2, 91.9, 91.9 (in cm), the resulting starts are 0, 20.9, 66.2, 170.8, 242.0, 333.9, 425.8.

The second decision is different for each of the five parts of *Studie II*. How are these values used?

In the first part, Stockhausen decided to give the events a somehow melodic structure. If one group consists of 3 events, just the time for the first one is taken from the ghost serie. Then event 2 starts at the end of event 1, and event 3 at the end of event 2:

In the code, instead of calculating all the 75 starts, just the 25 starts of the groups (sequences) are calculated. These values are then taken as the start values of a loop, in which the durations are shifted by the previous ones.

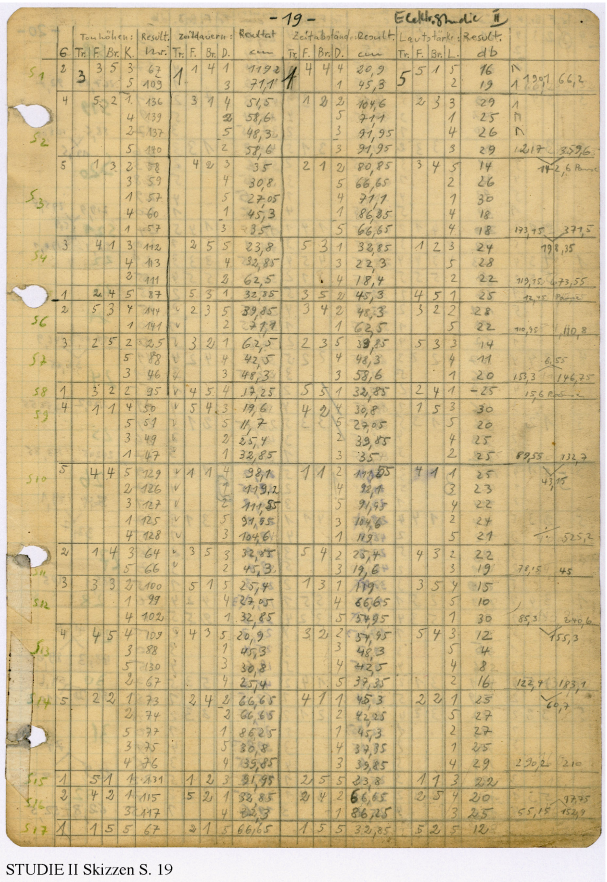
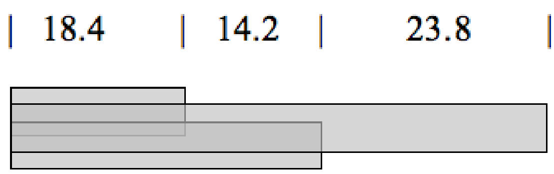


Figure 1: Stockhausen's sketches for Studie II, part 1. *© Archiv der Stockhausen-Stiftung für Musik, Kürten (www.stockhausen.org)*

3.2 Second Part

The main difference of the second part to the first one is the way the single events are treated. They no longer succeed each other but are played simultaneously, building "chords". These chords have either common starts or common ends. The durations are working like additions; if the durations of a group of 3 events are 18.4, 14.2 and 23.8 cm, and the 3 events start at the same time, the first event has 18.4 cm, the second has 18.4 *plus* 14.2 cm, and the third has 18.4 + 14.2 + 23.8 cm:



The realization as Csound Code works again with two nested loops (see above 3.1.1). Here the inner loop has to add the durations:

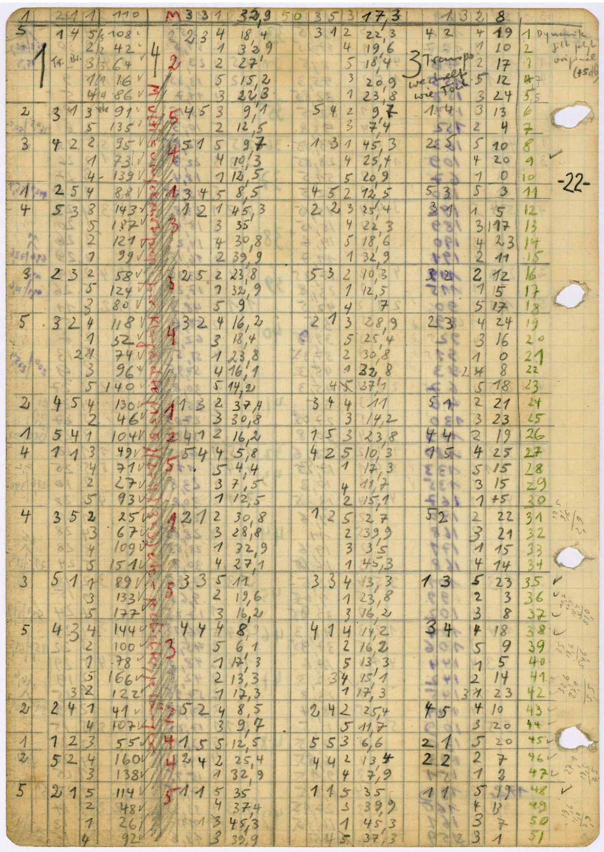
```
note:
;common start: first duration normal,
                  all others are added up
if ienv == 1 || ienv == 2 then
 ;durations from normal table
idurnorm      tab_i  indxnotabs, iftdur
 ;real duration = sum of preceding
  durations of this sequence plus idurnorm
idur          =      idurnorm + iduraccum
 ;this is the next value for the
                       assembled durations
iduraccum     =      idur
 ;write durations to iftout
              tabw_i idur,indxnotabs,iftout
;common end: last duration normal, all
             others are added up reversely
else
 ;counting down
indxback = indxabsseq+icount - indxnote-1
idurnorm      tab_i  indxback, iftdur
idur          =      idurnorm + iduraccum
iduraccum     =      idur
              tabw_i idur, indxback, iftout
endif
indxnotabs    =      indxnotabs + 1
       loop_lt   indxnote, 1, icount, note
```

3.3 Third, Fourth and Fifth Part

In order to not go into too much detail but to discuss more general questions in the next chapter, the focus here is taken on the main differences between the parts and its methods.

In the **third** part all events are isolated; like *staccato* tones on a piano. For widening the ambitus to potentially all frequencies, the method of using the series (cf. 3.1.1) changes in the way that the previously constant B now also changes like the A and C series.

(see figure 2, next page)



STUDIE II Skizzen S. 22

Figure 2: Stockhausen's sketches for Studie II, part 3. Note that the "Tr" (Transposition) is now changing. © Archiv der Stockhausen-Stiftung für Musik, Kürten (www.stockhausen.org)

In the **fourth** part this method is kept. Besides, the way of building chords from part 2 is repeated.

Part **five** is a mixture of all the different methods. It consists of "melodic" sequences like in part 1, "chord" structures like in part 2 and 4, and "staccato" passages like in part 3. So the inner loop of the start times has to differentiate between the cases:

```
ton:
istartdiff3 tab_i indxtonabs, iftstarts5a
;starts if direct from iftstarts5a/typ=3
istartabs   =      istartabs + istartdiff3
;typ1: concatenating the notes of one
                                  sequence
if ityp == 1 && indxton < icount-1 then
idur1       tab_i  indxtonabs, iftdurs
idurshift   =      idurshift + idur1
istart1     =      istartseq + idurshift
;value for the next note after the
                       duration of this note
      tabw_i istart1, indxtonabs+1, iftout
;typ2: chords
elseif ityp == 2 then
; for common starts
if (ienv == 1 || ienv == 2 || ienv == 5)
       && (indxton < icount-1) then
istart2     =      istartseq
;else starting point as difference to the
maximum duration of the sequence
else
idur2     tab_i   indxtonabs, iftdurs
istart2    =    istartseq + (imaxdur-idur2)
endif
;value for this note from this calculation
          tabw_i istart2,indxtonabs,iftout
;value for the next note from normal table
     tabw_i istartabs, indxtonabs+1,iftout
;typ=3: isolated events, or last note
                    from a typ=1 sequence
else
     tabw_i istartabs, indxtonabs+1,iftout
endif
indxtonabs    =      indxtonabs + 1
             loop_lt indxton,1,icount, ton
;indxenv++ if value was used
indxenv        =      (ityp == 2 ?
                    indxenv+1 : indxenv)
```

4 Results, Problems, Conclusions

Studying *Studie II* is not meant as an act of museal interest. The principal questions here are:

- Can this composition which was made entirely by hand - as a composition and as a realization - be regenerated by program code? If yes - why?
- Does everything run smooth or are there serious problems in the re-generation?
- What can be learned by Stockhausen's way of generation for today's algorithmic compositions?

4.1 Can *Studie II* be regenerated as program code?

Yes, it can. It can be proven that in principle all the events of the piece can be re-generated, by just starting with the numbers 3 5 1 4 2 and executing some (describable = programmable) methods. It has been done in Csound, and it should be possible to do it in any other audio programming language with a similar power as Csound. The reason *why* this is possible leads to the method of composing which Stockhausen applied in this piece (and probably tests for the first time so consistently), the so-called serial way of composing. Serialsm can be decribed as a method which situates the decisions of a composer in the space of structures and methods. *Studie II* structures the material (frequencies, durations, timbres, intensities), starting "from nothing", and defines methods of generating events, using series which are also generated in a regular way. So there is a very interesting interrelation between a compositional technique which came from a purely musical development (Schoenberg - Webern - Messiaen) and the emergence of the computer and mainly the first high-level programming languages (Fortran, Lisp) exactly at the same time in the mid-50s.

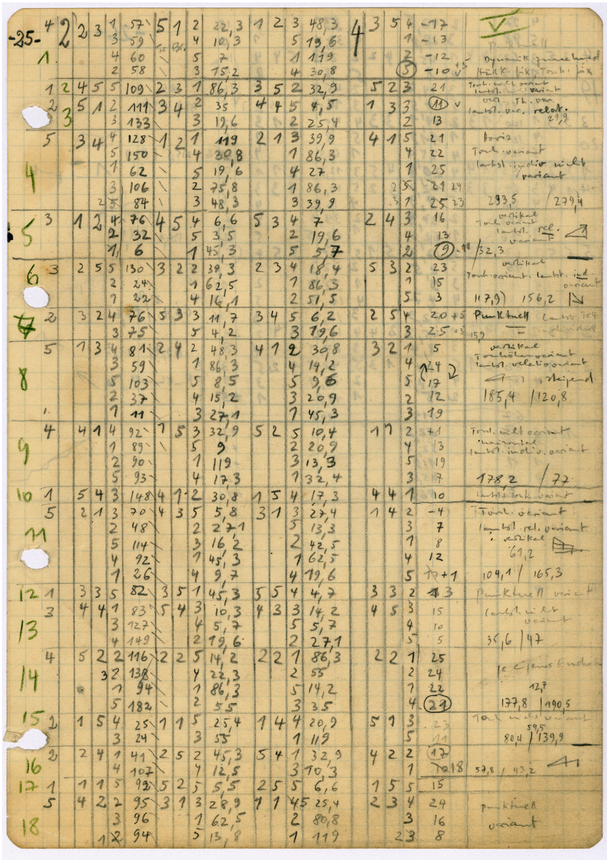
4.2 What are the difficulties in the regeneration of *Studie II*?

The problems in the regeneration of *Studie II* can be divided into three different categories.

First, there are obviously a number of errors in Stockhausens calculations of some frequencies or durations. These deviations of the correct calculation are recorded truly in the work of Silberhorn[7]. They are easy to understand in a work with 380 events, each event requiring at least four calculations, all done by hand and paper.

But, **second**, there are certain deviations from the "actual" result, which are more complicated. It seems that sometimes Stockhausen decided intentionally for a different single event. In some cases it seems he avoided a nearly-repetition, in other cases corrected densities[8]. And in part five one gets the impression that he often decided "on the fly", on the basis of the predefined structures, what he preferred for this situation. From the point of programming, these latter kind of deviations are crucial. If they are more than exceptions, the re-generation makes no sense, because it has just to translate the exceptions, and this is no more a re-generation but a transcription of the score. In my opinion, in the first four parts of *Studie II*, there are just deviations by error or small exceptions, but the fifth part comes close to the point of changing to a new quality. A closer look at the sketches proves it:

(see figure 3, next column)



STUDIE II Skizzen S. 25

Figure 3: Stockhausen's sketches for Studie II, part 5. Note the spontaneous side notes about the musical content. © Archiv der Stockhausen-Stiftung für Musik, Kürten (www.stockhausen.org)

Third, there are some incertainties with respect to the generation of the envelopes. It can be shown by accurate analysis that part two and four use series of five shapes for the envelopes.[9] This should be possible to show also for part one. Part three uses just one envelope, and part five is also in this respect a combination of all the previous shapes. So I believe the envelopes can be generated by series, too, and could show the series for parts two and four, but the origin of the series are obscure.[10]

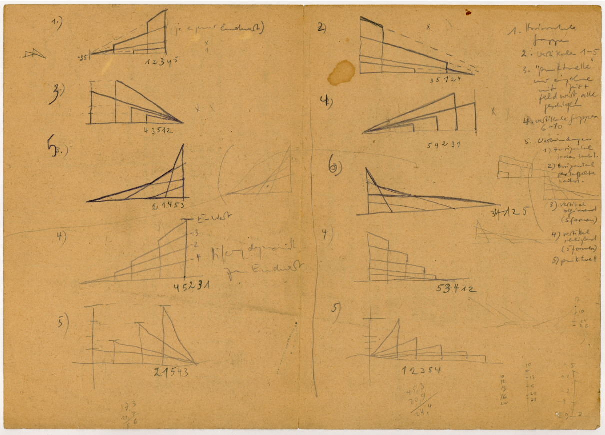
[7]pp 128-167

[8]cf Silberhorn pp 118-125

[9]In my analysis and transcription, the series for part two and four are the following:

1 2 4 3 5	2 5 4 3 1
1 4 3 5 2	2 1 5 4 3
1 4 5 2 3	2 3 1 4 5
4 3 1 5 2	2 5 1 3 4
3 1 4 5 2	2 3 5 1 4

[10]Unfortunately, nothing could be found in the sketches of Stockhausen which clarifies the origin of the series which are mentioned in the previous footnote.



STUDIE II Skizzen #1.1

Figure 4: Stockhausen's early sketch for Studie II, containing a list of possible envelopes. © Archiv der Stockhausen-Stiftung für Musik, Kürten (www.stockhausen.org)

4.3 What does the regeneration of *Studie II* show regarding questions of algorithmic composition?

With a certain right, *Studie II* can be considered an early model of algorithmic composition, the application of serial compositional methods in the area of electronic sound (and today: computer music). Re-generating this composition means: coming to the root of musical decisions; being where the composer was while working. From this, some important conclusions can be drawn by this work.

1. *Change the Methods Depending on the Musical Structures you Want to Get*
   It is impressive and inspiring to see how Stockhausen arranges and adapts the flow of the series depending on the musical structures and differences he wants to get. This way of composing can be compared with an architect of landscapes, regulating the streams of water, planting this here and that there. It is totally different from a way of "composing" which can unfortunately not be considered as an exception in the field of algorithmic composition: put one algorithm in the machine and let the machine play for a long long time ...
2. *Listen to the Results and React as a Musician Instead of Sticking with Principles*
   It is extremely interesting to see how Stockhausen changed his own plans, obviously because he saw some musical problems if he would have kept them. One example is the generation of the frequencies. He started with a constant "transposition" (cf 3.1.1) value of 3 in the first part and 5 in the second part. We do not need too much inspiration to suspect that the values of the other three parts were planned to be 1, 4 and 2, so representing the king's crown (3 5 1 4 2) in this field. But then, at the beginning of part three, something exciting happens. Stockhausen decided to break the whole method: not to have one constant transposition in the whole part, but changing this value by running a series. Why? Because he saw that insisting on the method would have been "too consistent", this means: predictable, and wouldn't allow the staccato sequences of part three to have larger leaps. This is the paradigm of a musical reasonable decision which breaks the "purity" of a method.
3. *Listen Also to the Single Events and Change Them if Your Ear Judges So*
   As it has been discussed (see 4.2), Stockhausen changes not just methods, but also single events, or even small groups of events, according to the musical scope or a certain situation. Generalizing this, we can say that a balance must be found in algorithmic compositions between the coherence of methods, of the generation of musical structures on one hand, and the need of listening to the results - also in a small context - and consequently changing some of them by decisions on the other hand. Both extremes are to avoid: just generating in an algorithmic way, even in an intelligent and changeable way, can result in unmusical structures or "untempered" relations at a scope; too much deviations gets lost of the structure and the large context.

5 Conclusion

Re-Generating Stockhausen's *Studie II* forces one to take to the inner movement and compositorial method of this piece. It reveals meaningful correlations between a compositorial technique and a formalized way of thinking which becomes reality in programming, and it shows important qualities of a non-dogmatic but musical way of working in this domain. In this early piece, Stockhausen has shown two virtues (which he was not famous for later in life): to be modest ("I'm doing a study") and appropriate, and to be short, knowing that stopping before "all has been sayed" can often be more than showing all the material one has generated. This work is a worthy model exactly in this way, and a re-generation can convey a modern formulation for this exemplarity.

6 Acknowledgements

Thanks to Anna Buschart for reading the manuscript, and to the students of the analysis seminar of electronic music at the HMT Hannover in winter 2009/2010 for their patience and interest.

Special thanks to Kathinka Pasveer and the Stockhausen Stiftung for the generousity of not just letting me study the sketches but also giving the permission for printing some of the sketches here.

References

The Re-Generation of Stockhausen's *Studie II* is part of the regular QuteCsound distribution since version 0.4.5. See
http://sourceforge.net/projects/qutecsound
(in the Examples -> Music menu).

It is based of the analysis of Heinz Silberhorn:
Heinz Silberhorn, Die Reihentechnik in Stockhausens Studie II, Rohrdorfer Musikverlag 1980