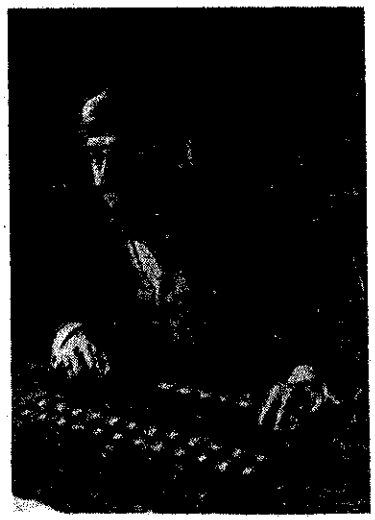
## REFERENCES


1. Barry Truax, "A Communicational Approach to Computer Sound Programs," *Journal of Music Theory*, 20 (1976), pp. 227-300.
2. For instance, two programs may resemble each other structurally, but if one costs less to run, is more accessible in terms of time, has a greater degree of real-time interaction, presupposes less familiarity with computers by users, and can be afforded by any school, then all those diverse characteristics will strongly influence, if not improve as well, the behaviour of the system, and the communicational model it reflects.
3. Allen Newell, "Heuristic Programming: Ill-Structured Problems," in *Progress in Operations Research*, ed. J. J. Aronofsky (New York: Wiley, 1969), pp. 371-72.
4. Otto E. Laske, "Towards a Theory of Musical Cognition," *Interface*, 4 (1975); id. "The Information-Processing Approach to Musical Cognition," *Interface*. 3 (1974).
5. Lejaren A. Hiller and Leonard Isaacson, *Experimental Music: Composition with an Electronic Computer* (New York: McGraw-Hill, 1959); L. A. Hiller, "Some Compositional Techniques Involving the Use of Computers," in *Music by Computers*, ed. H. Von Foerster and J. W. Beauchamp (New York: Wiley, 1968).
6. Hubert S. Howe, Jr., "Composing by Computer," *Computers and the Humanities*, 9 (1975): 281-90.
7. Herbert A. Simon and Richard K. Sumner, "Pattern in Music," in *Formal Representation of Human Judgment*, ed. B. Kleinmuntz (New York: Wiley, 1968) pp. 219-50.
8. B. Truax, "The POD System of Interactive Composition Programs," *Computer Music Journal*, vol. 1, no. 3, 1977, pp. 30-39; idem, "Computer Music Composition: The Polyphonic POD System," *IEEE Computer*, vol. 11, no. 8, 1978, pp. 40-50.
9. John E. Rogers, "The Uses of Digital Computers in Electronic Music Generation," in *The Development and Practice of Electronic Music*, ed. Jon H. Appleton and Ronald C. Perera (Englewood Cliffs, N. J.: Prentice-Hall), p. 189.
10. B. Truax, "A Communicational Approach", p. 269.

Barry Truax
Dept. of Communication
Simon Fraser University
Burnaby, B.C., Canada V5A 1S6

Photography by Rick Etkin

Interface, vol. 9 (1980), pp. 49-57.

# The Inverse Relation Between Generality and Strength in Computer Music Programs*

Barry Truax

ABSTRACT

The effect of the design of interactive computer music systems on the behaviour of users is described within a communicational model that correlates user interaction with the generality and strength of the available program strategies. A range of strong and weak methods within a hierarchical data structure maintains a generality of applicability and maximizes user experimentation. Real-time interaction is stressed as important for the system as a learning tool.

In a recent paper,[1] I have tried to outline a theoretical and practical basis for evaluating computer music programs, in particular those where users operate in a compositional task environment aided by sound synthesis. The importance for me of using a communicational model rather than a purely structural one is that it considers both the program structure and the way that it interacts with users; that is, the model is concerned with the *behaviour* of the combined user/program system when engaged in musical problem solving. Further, the communicational approach considers the psychological and sociological environment in which this activity takes place as it affects the process.[2]

A major thesis of the paper referred to is that any computer system embodies both an implicit and explicit model of the musical process that can be inferred from the program's data structure and implied or actual user behaviour. Part of that model is revealed by the stage where the program

Manuscript received September 1979

* This paper was delivered at the First International Conference on Computer Music at Boston in 1976.

enters the individual musical process and the effect or influence it has on that process. The general importance of the model for system design is that a system should be conducive to the user task environment at every level, ranging from the sociological and pragmatic context, through to the cognitive requirements of the musical process. For instance, turn-around time has traditionally been regarded as a variable that is a *result* of the behaviour of a program or system, not as a *criterion* for its design. It is a simple but crucial distinction to realize that turn-around time is also a determinant of user behaviour that has an important influence on the learning potential of the system; in the design of a given task environment it cannot be left to chance.

It is probably evident by now that I have less interest in compositional systems that use the computer as a labour-saving calculation device than in systems which participate in the user's process at many stages, i.e. interactive systems. The potential of these systems is only recently coming to be recognized and actually exploited by composers and theorists. My emphasis in this paper is on criteria for interactive program design which have a positive influence on user behaviour, and the implications that such systems have for music and music theory is general.

## GENERALITY AND STRENGTH

Some analytic criteria serve a better purpose when treated as a starting point than as absolute methods. Such is the case with the generality and strength dichotomy. First of all, we must keep in mind that generality and strength describe methods of problem solution and have no absolute value outside a particular context. Generality refers to the range of problems to which a method can be applied; strength refers to the efficiency of problem solution in terms of specific parameters (e.g. time, cost, storage). At the simplest level, an inverse relation holds between them as pointed out by Newell[3]: weak methods have a high generality of applicability, whereas more powerful methods inevitably restrict the range of tasks to which they apply. A weak method such as generate-and-test may apply to any problem, albeit inefficiently and with low probability of success, whereas a powerful algorithm that guarantees a solution does so by restricting itself to a specific type of problem.

In applying this useful and simple distinction to sound synthesis and composition systems, my main concern is the influence that a given method has on user behaviour, and specifically on the learning potential of the system. Let us examine first the relation of the amount of information the user is required to



furnish as a function of the weakness or strength of a given method. In general the weaker the method, the more information the user will need to specify in order to obtain a given result (see Figure 1). Moreover, the well-formedness of the result will very much depend on the intelligent choices exhibited by the user in his specifications. The extreme weak case in sound synthesis would be where the user would have to supply all the pressure function data – clearly an impossibility. As stronger methods are used to produce that data, the amount of user input is reduced but so is the range of output. In the extreme case at the strong end, the user simply identifies a given sound in a catalog; user information is reduced to a minimum, but so is the range of output. What is changing reciprocally to the amount of user information is the degree and level of intelligence implemented in the program. The more intelligence built into the program, and the greater its degree of automation, the less low-level data that will be needed from the user (see Figure 1). Some criteria for program automation will be discussed later.

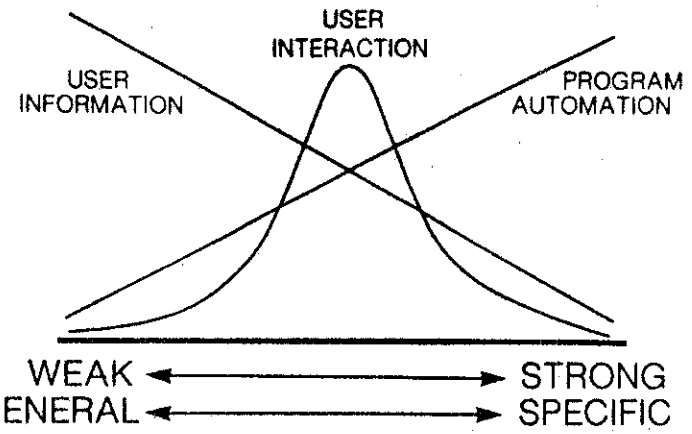



Fig. 1. Variation of user information, program automation, and user interaction as a function of the generality and strength of a system.

What is the implication of this interplay between user and program intelligence? At the simplest level, we may draw a curve, as in Figure 1, representing the degree of user interaction that increases to a maximum somewhere between the weak, general end and the strong, specific end. That is, as more intelligence is implemented in program procedures, the less information the user will have to supply, and that which is suppled will be more effective because it applies to a higher level of organization within the program. Furthermore, the guarantee of a well-formed result increases when the program implements strong procedures, because these circumscribe a given range of more or less known results. As weak methods are progressively strengthened, there is increased user interaction and learning potential, first because the user is less burdened with data specification, but also because the process will be faster and the results more controllable. The user will quickly be able to make judgments at the semantic level that compare the present state with the goal state.

For instance, Fourier synthesis is a weaker method than frequency modulation synthesis. Consider the problem of producing time-dependent spectra by each method. The amount of user information required in Fourier synthesis is enormous, and moreover of a sort that few users would readily have at hand. A great deal of specific knowledge is required to produce a predictably well-formed result. If the information is derived from the analysis of an actual sound, the user's problem is eased but the range of output is reduced. Even a real-time interactive version of Fourier synthesis will only increase the level of interaction to a certain point unless the user develops strong control methods. By contrast, frequency modulation uses a much smaller amount of user information and guarantees as complex a result. The elegance is not only mathematical but also psychoacoustic, since every parameter change produces a predictable audible change. Although the range of all possible output may be smaller, it is more useful because all of it is accessible as a result of the efficiency of the learning situation.

As algorithm automation continues to increase, and user information is relied on less and less, user interaction decreases, or rather shifts to high-level decisions at the semantic level. Thus, our curve of interaction falls off beyond some critical point, as in Figure 1, until with full automation it is minimized. Of course, the learning process of the programmer or theorist in a more fully automated program will presumably increase,[4] even if the user of the program is eventually eliminated.

Although the examples given so far apply to synthes · the same pattern can be seen in compositional systems. A system based on a · ak generate-and-test



model, such as the Monte Carlo method of Hiller's early experiments,[5] requires a fair amount of well-chosen data from the user in the form of transition probabilities, and thus is characteristically non-interactive. Similarly, if the basic MUSIC 4/5 system is unequipped by strong methods, i.e. subroutines, to generate data systematically, it requires well-chosen data from the user in quite large amounts. Moreover, if the system does not run in real time, it provides a poor environment for the user to acquire any predictive knowledge. User interaction tends to shift to a generate-and-test process at the semantic level; try something and discard it if unacceptable, or more likely, keep it if minimally acceptable. Naturally, composers working with these systems develop their own subroutines to make the system more powerful with respect to their own musical goals.[6] This process in itself is useful, but usually only for the same composer and for a limited range of tasks.

## HIERARCHY

The model of generality and strength, correlated with user interaction as determined by an interplay between user intelligence and program automation, is suggestive as a basic means of system analysis or as criteria for system design. However, one factor of extreme importance has been partly ignored in the model thus far: hierarchy. By this I refer to the presence of different levels of information and control within a system. Hierarchy is a prime requirement for system automation, that is, for strengthening the intelligence of problem solution, but its implication for the user, and for musical knowledge in general, will depend on the type of hierarchy used in the automation. Some types will serve to constrain the range of system output that is easily accessible to the user; others will expand it.

One type of hierarchical data structure serves to reduce redundancy. The unit generator concept, for instance, is typical of the usefulness of this approch. The generator, which is a subroutine, performs a given data operation whenever called. Therefore, instead of repeatedly calculating a function or waveform, it is generated once and used whenever necessary. Both user and machine data requirements are profitably reduced. Obviously this technique is appropriate at the level where redundancy is greatest, such as that of sound synthesis. However, if the same principle is applied at higher compositional levels (or as it has been to pattern recognition[7]), considerable predictability occurs which limits the range of output. Each level may be thought of as an accumulation of data at the next lower level. A change at any one level changes

the membership at higher and lower levels. The user optimizes the structure by editing single units. Although analogous to certain acoustic and musical processes based on iteration, periodicity or motivic development, this kind of hierarchy reduces redundancy of information input, but does not increase the range of output.

By contrast, in a different kind of hierarchy, each level *interprets* data on a lower level, instead of accumulating it, and therefore each level provides an effective control on the others. A simple example in traditional music is the tempo indicator which controls the speed of all rhythmic units. The composer or performer optimizes a realization of the work by adjusting the tempo to a suitable value. Its use implies that the lower level units (the note values) are not absolute but rather are open to higher level control. Giving the user controls at a new level of a hierarchy strengthens the system, increases user interaction, encourages optimization through "hill-climbing", keeps the amount of input minimized and, apparently in contradiction to our previous claims, increases the generality of the system by seeming to increase the range of output. However, this is not actually true, since the same result could have been realized previous to such an extension of the system hierarchy, but only with so much extra input that it is unlikely that the user would have taken the trouble, and if he had, he would probably have been diverted from his original goal by the complexity of the specification. Although the range of potential output has not been increased, the range of output that is *accessible* to the user has been enlarged. As a result of complex hierarchy, the user may work more efficiently at various structural levels, controlling systematic relationships that would otherwise have been cumbersome to deal with.

This latter approach is the basis of the compositional structure of my own POD system. As I have described it elsewhere,[8] each concept of the hierarchy interprets data collected at a lower level. Performance variables affect the realization of a syntactic distribution of events in actual sound and time. The distribution in turn interprets the mapping of sound objects, i.e. sets of timbral characteristics, onto events with specific frequencies within the time structure. As well, it controls the statistical character of these events within frequency and amplitude time fields. Degrees of control may be implemented at each level of the structure, and most parameters can be changed quickly enough that the real-time feedback situation encourages optimization of a given structure. Performance, structural and random variants offer further possibilities for larger scale compositional planning.



## CONCLUSION

Real-time interaction forms the necessary basis for an effective learning situation of powerful potential for music composition. However, it is best aided by strong acoustic and music structural methods which allow and encourage a user to work with complex control strategies, not simply with endless data specification. A survey of the literature will show that few such strong methods have been developed and implemented to date – an indication perhaps of a certain reluctance by composers and theorists to recognize the potential of the computer as a testing facility for musical concepts through their implementation as programmed procedures within a user-oriented task environment; in other words, the recognition that music theory can and should become an empirical science. On the contrary, many people have felt that any strong compositional procedure merely made a program idiosyncratic, that is, suitable only for single composer's needs.[9] This is a danger, but only if overly strong, limited methods are used. The fundamental point I wish to make very clear is that a *balance* between generality and strength may be achieved by using relatively weak methods at every stage of an intelligent hierarchy. Hierarchy satisfies the user's needs for powerful controls to generate complex musical structures, and make effective use of an interactive system. Weak methods at every control stage, or more preferably, a range of weak to strong methods, allows the widest range of musical output to be achieved and may satisfy the needs of a variety of composers.

A program implementing musical knowledge itself embodies a working musical competence in a form never achieved as explicitly before. The development of such programs not only increases music procedural knowledge, but its use also brings composers together out of the isolation of their respective working methods. As I concluded in the paper referred to earlier,[10] such programs imply "that a wide range of contemporary musical techniques have something in common at the level of control structure, even when the content or message level is diverse. To develop such a model that can be shared by composers is in direct opposition to the notion of the composer's activity as irrational, non-generalizable, non-teachable, and non-programmable. It points to a competence shared among music users, but recognizes the diversity of strategies devised for achieving different performance goals."