Theory of Complexity

Kevin McGuire
Ottawa, Canada
e-mail: kevin@mcguire.name

Abstract

Dice throwing and similar techniques are valuable tools for moving the creative process out of a rut or for suggesting previously overlooked directions. Random number generators are often used in generative processes to produce variety.

Yet randomness and complexity have different meanings to different communities. Sometimes the term “random” is used to mean “of high complexity”, sometimes “of unrecognisable structure”. To discuss “unrecognisable” in a concrete manner we must discuss human perception and cognition, subjects which aren’t well understood. Meanwhile, highly complex artifacts can emerge from simple rules, adding to the confusion. Informal experiential notions of randomness and complexity differ in important ways from formal definitions derived from Information Science. These approaches are contrasted in an attempt to arrive at a shared model of complexity.

While randomness can be used as a creative trigger, its use in generative processes can impede the progress towards a desired solution. Therefore it is important to understand what a random number generator is providing. Since generative processes encode considerable structure and complexity into artifacts, we show that surprising variety can emerge without the need of random number generators. Complexity, surprise and variety are all possible without randomness.

1. The Role of Surprise

In many creative disciplines, chance is used to further the creative process. This can provide new insight and new solutions may appear. One famous manifestation of this is Brian Eno and Peter Schmidt’s Oblique Strategies [1], a deck of cards with cryptic remarks which one chooses for inspiration.

In music, the use of chance can be traced back to the 18th and early 19th century Musikalisches Würfelspiel (musical dice game) [2], one example of which is attributed to Mozart [3]. In modern dance, Merce Cunningham uses dice throwing just prior to a performance to determine the order of the choreography, costumes, lighting, décor, and music [4]. The mid-1900’s saw the establishment of aleatoric (or ‘chance’) music, defined as “music in which some element of the composition is left to chance or some primary element of a composed work's realization is left to the determination of its performer(s)” [2].

As a technique for unblocking creative potential, the use of randomness is clearly valuable. However in aleatoric music, chance is taken one step further and purposely removes some decision and control on the part of the composer (e.g. through the throwing of dice), although often within a limited number of possibilities. In so doing the composer hasn't simply written one single piece of music, but rather a family of music, being all the combinations of the elements combined.

In modern terms, aleatoric music consists of a process by which a random number is used to select combinations from within a predetermined set of elements. It follows an algorithm, and is thus an early example of generative art.

More recently, John Cage used various elaborate approaches to the application of chance in composition and performance [5]. His techniques were more sophisticated than simply throwing dice. For example in one, the performer must interpret the 'musical meaning' of lines on a sheet of paper. In another he leaves instructions to the players on how to turn the volume and tuning knobs on a set of radios (which would play whatever happened to be broadcast at that time). We could thus consider his algorithms to be more complex.

2. Approaches to Complexity

But what do we mean by structure, by complexity? These subjects can be tricky to discuss because one often ends up discussing the *apparent* complexity of an artifact. This could be the subtle layering of a piece of music, the sophisticated composition of a painting, or the intricate details of a sculpture.

The problem is that these are all qualitative notions of complexity. Understanding them with more precision requires understanding perception and cognition. Our senses reduce and encode information prior to processing by our cognitive centers [6], so a comprehensive theory on pattern and complexity likely must take those encodings into account. Then the cultural context and knowledge of the individual must be factored in. Our lack of precision in understanding these areas leads us to conundrums like whether Fractals [7] are complex or not. Most would state that they are visually complex. Yet Fractals are derived from a very compact algorithm with no randomness. They are actually quite simple! Which is correct?

In math and computer science, there's a relatively recent definition of complexity. Kolmogorov complexity [8] defines the *algorithmic information content* of a string as being the smallest Turing machine[1] capable of producing that string. In simple terms, it's the smallest program that can produce that output.

For example, the following string is quite long, so seems to have lots of information content: "123123123123123123123123123"

[1] A Turing machine is a simple theoretical computer that reads a tape of symbols which it interprets as instructions. Every real software program has an equivalent Turing machine, and every Turing machine can be realized as a real program.

The clever reader will notice though that this is simply the string “123” repeated 5 times. Another reader might disagree with that encoding and state instead that its “123123” repeated 5 times. Or its “123123123123123” repeated 2 times. Those are all true. But which is *more* true from an information content point of view?

1. ‘123123123123123123123123123’
2. ‘123123123123123’ repeated 2x
3. ‘123123’ repeated 5x
4. ’123’ repeated 10x

Intuitively we see that (4) is the shortest[2], and we’re pretty confident that there’s no shorter way of expressing it. If I wanted to transmit the string to you, I could send you all 30 digits, but it’s faster just to say, “’123’ repeated 10 times”. That’s its information content, “’123’ repeated 10 times”. You now know everything there is to know about the string.

Rather than a subjective measure of complexity (“hmm, that looks complicated”), Kolmogorov complexity provides an objective, algorithmic measure of complexity (“hmm, that turned out to be easy to do”). It provides a theoretical, well constructed, quantitative definition of complexity: the information content of an artifact is measured by the complexity of the program that produces it[3]. It’s irrelevant if the thing *looks* complicated, it only matters how hard it is to *make* it.

3. What it Means to be Random

The term “random” gets bantered about quite a bit. It, even more than complexity, requires more precision in our use. The mathematical definition is, “being or relating to a set or to an element of a set each of whose elements has equal probability of occurrence” [9]. Simply, anything could’ve happened, one outcome no more likely than the other. This is the classic dice throwing sense of randomness. Sometimes though we use it in the sense of “lacking a definite plan, purpose, or pattern” [9]. This is a kind of perceived randomness, as in “I couldn’t predict the outcome”.

Thus we have two approaches to randomness[4], one concerning the results, and the second concerning the behaviour.

[2] While these aren’t Turing machines, they are like pseudo code programs and for the sake of this discussion are sufficiently representative of the complexity of the required Turing machine.

[3] The bad news is that for any arbitrary string, we can’t know if we’ve in fact found the smallest Turing machine, there may always be a smaller one we just haven’t been clever enough to come up with. Thus the theory is of limited practical benefit but is a powerful conceptual model.

[4] Kolmogorov complexity defines randomness as a string whose smallest Turing machine is, in fact, the string itself. That is, there is no algorithm which reveals hidden structure, which compresses it; the shortest way to encode the string is to just remember all the digits.

4. Random Number Generators aren't Random

Ironically, random number generators aren't really random at all. They are referred to as "pseudo-random". Their outcome is somewhat random in the probabilistic sense above; the digits conform to an acceptable probabilistic distribution. But their behavior certainly isn't. Given the same seed, they will produce exactly the same sequence.

Thus when you reach for your random number generator, you aren't actually producing anything random, you're just producing an outcome which is very difficult to predict, with no recognizable pattern, whose values are well distributed probabilistically. There is in fact a pattern, you just can't spot it.

But pseudo random number generators tend to be relatively small programs. Thus the outcome is always of relatively low algorithmic information content: small Turing machine = low algorithmic information content. Their outcome could be considered simple, in fact.

Random number generators produce simple results.

5. Random Number Generators Considered Harmful

As discussed, randomness can be a valuable tool for permitting the artist or designer to cede control. However, when faced with set of design criteria, this randomness can do more harm than good. This is because without parameterisation and guidance of the randomness, results can just as easily move you further from your goal than closer to it. This is why approaches such as Genetic Algorithms [10] which use randomness to cross breed and for mutation require fitness functions to increase the probability of producing superior solutions. The randomness must be controlled.

Thus its not a matter of simply ceding control, but of ceding the *right* control, so that one can still achieve one's creative goals. To do so, one must guide the system to the desired solution. This task can be difficult once a random number generator has been introduced because it reduces predictability, yet predictability is required for us to navigate the space of possibilities. If you are unable to drive the system to the desired goal, then all you can do is chose from the results, hoping one matches. When control is lost, so too is subjectivity, and with it, access to one's cultural and aesthetical references [11]. One's role is relegated to that of a shopper [12].

Therefore it would be better if we could avoid the use of random number generators. But can we do so without sacrificing our desire for variety, for complexity, for surprise?

6. Complexity without Randomness

Lets us examine a simple L-System[5] [13] with the following grammar:

S=F
F=F[-F]F[+F]F

Figure 1 shows the tree expanded to depth 4.

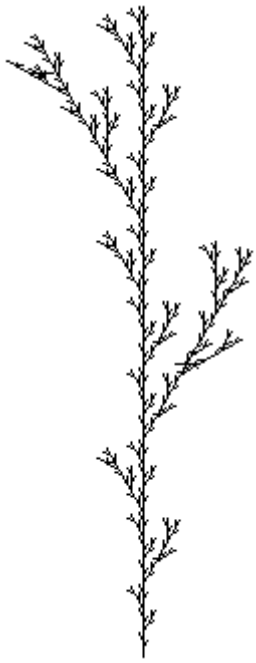

Figure 1 - Single rule

A lot of self-similarity and regularity are present. To add a bit of variety, lets add two more rules with LHS 'F', as shown in the discussion on Stochastic L-Systems in [13].

S=F
F=F[+F]F[-F]F
F=F[+F] F
F=F[-F] F

[5] L-Systems are rewriting graph grammars capable of producing branching, organic shapes. The grammar consists of a set of rules, each with a left hand side, an assignment (in our case, "="), and a right hand side. Starting with the start symbol S, the RHS is expanded as follows: for each symbol, find a rule with LHS that matches that symbol and replace the symbol with that rule's RHS. This expansion continues for several recursions. Finally, the resulting string is interpreted, in this case as turtle drawing commands (e.g. "F" is "draw a line forward N steps").

During expansion only one of these three ‘F’ rules can be applied, therefore we require a way of choosing which one to apply at any given time.

6.1 Random results

The obvious choice is to use a random number generator, with each rule having equal weight of being chosen. Figure 2 shows three randomly generated trees.

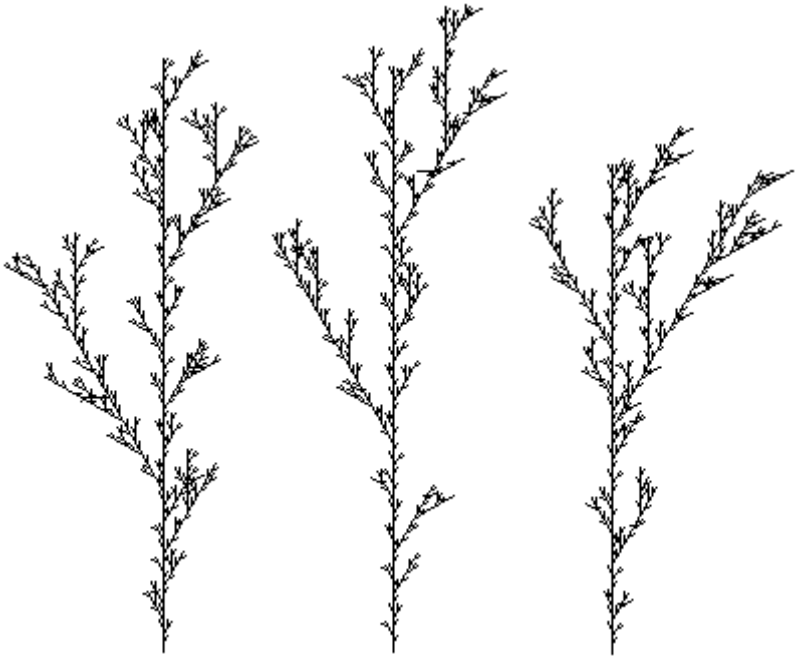

Figure 2 -Three random trees

These clearly exhibit more variation than Figure 1. Yet one could claim they are more alike than different. This is because the L-System itself, through the recursive application of a small set of rules, defines the topology. The random number generator just provides varieties. These look like they could be the same plant species, just grown under different conditions.

6.2 Sequential results

But was a random number generator really required? What if instead one did the simplest thing one could think of, which is that every time a rule must be chosen, the system just chose the next one in order? That is, first time it picked F=F[+F]F[-F]F, next time F=F[+F] F, next F=F[-F] F, then back to the first again.

This can be represented with the sequence {1, 2, 3}, signifying which rule to pick. When the sequence is exhausted, the system starts over at the start (i.e. with ‘1’). There are six combinations of sequences of three digits, thus six possible results. Figure 3 shows three of them. They exhibit a surprising degree of variety.

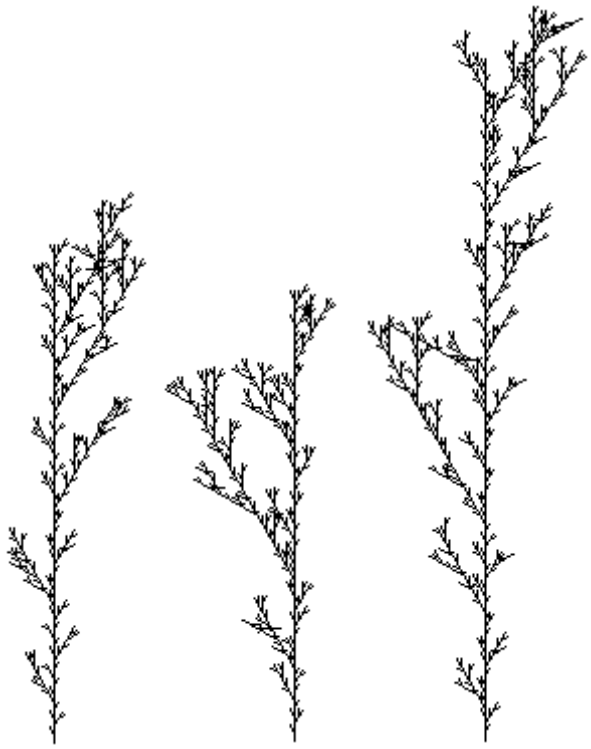

Figure 3 - The three variations of sequential rule selection

In fact, subjectively its difficult to guess that there was no random number generator involved. Yet there is very little information added above the existence of the three rules, just the simplest notion of a selection algorithm (i.e. “pick each in turn”).

6.3 Longer sequence

More algorithmic information, and thus variety, can be achieved by encoding more structure in the sequence. In Figure 4, on the left is the result of the sequence {3, 3, 2, 2, 1, 1, 2, 2, 3, 3}, and on the right is the author’s phone number. The results are rich though it’s not evident that they exhibit any more visual complexity than the previous examples.

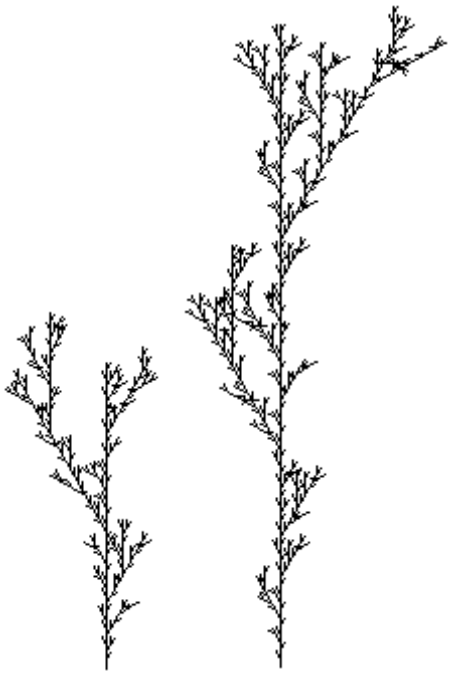

Figure 4 - Longer sequences

6.4 Discussion

This exercise could continue with more and more sophisticated sequences. A legitimate (and fun) area of exploration is the relationship between changes in the sequence and changes in the resulting picture. Patterns of digits, reflecting the sequence, etc., gradually adds more information to the sequence, which is then realized as visual complexity in the picture. Yet even the simplest algorithm for choosing the rules results in pictures with surprising variety. This suggests that the generative system itself is the primary source of the visual complexity. Similar rich results from generative systems that don't use random number generators have been shown in [14,15].

While random number generators provide a well distributed set of values, to achieve variety, a simple algorithm will do. Random number generators are overkill. Worse, they obscure the inherent rich complexity and variety of the generative process.

Each sequence above can be considered a computable function. There are an infinite number of computable functions. Most are not random number generators. Thus, there's a huge space of possible functions waiting to be explored.

7. Conclusion

Informal notions of complexity and randomness lead to ambiguity and misunderstanding in the generative community. One reason is that subjective complexity is experiential. To understand it, one must understand perception and cognition. Information theory provides a concise and quantitative definition of

complexity in terms of the smallest program capable of producing the result. This allows us to think of complexity differently, not solely as a visual experience, but also in terms of the sophistication of the process which produced it.

We've shown that a considerable amount of visual complexity can result from the simple addition of alternative rules to a generative system. We've also shown that while random number generators produce well distributed values, visually rich and "random looking" results can be achieved through much simpler processes.

Throwing dice, picking cards from a deck, are valuable techniques for breaking one's thought processes out of a rut. However, the use of random number generators in generative processes makes it difficult to guide that process in the desired direction. If one's goal is to produce complexity, then one should look to the process, since it contains all the algorithmic information content. If one's goal is to produce variety, than a random number generator is likely unnecessary if the generative process itself is sufficiently rich.

When looking for a function to produce variety, there are infinite available. Rather than reaching for the random number generator, pick a different one!

8. References

[1] Oblique Strategies. http://www.rtqe.net/ObliqueStrategies/

[2] Wikipedia. http://en.wikipedia.org/wiki/Chance_music

[3] University of Vienna. http://sunsite.univie.ac.at/Mozart/dice/

[4] NAC/CAN. http://www.nac-cna.ca/en/nacnews/viewnews.cfm?ID=1113

[5] Cage, John. "Indeterminacy", in his Silence: Lectures and Writings, 35–40. Middletown, Conn. Wesleyan University Press, 1961.

[6] Resnikoff, HL. Illusion of Reality. Springer-Verlag New York, Inc. New York, NY, USA, 1989. ISBN 0-3879-6398-7

[7] Mandelbrot, B. B.. The Fractal Geometry of Nature. W. H. Freeman and Company, 1982. ISBN 0-7167-1186-9

[8] Li, Ming and Vitányi, Paul. An Introduction to Kolmogorov Complexity and Its Applications, Springer,1997.

[9] Merrian-Webster. http://www.webster.com/dictionary

[10] Goldberg, DE. Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA, 1989.

[11] Soddu, Celestino. "Generative Design. A swimmer in a natural sea frame", *9th Generative Art Conference*, Milan, 2006.

[12] Soddu, Celestino. Remarks made during presentation of [Soddu] as remembered by the author, *9th Generative Art Conference*, Milan, 2006.

[13] Prusinkiewicz, Przemyslaw, and Aristid Lindenmayer. The Algorithmic Beauty of Plants. New York: Springer-Verlag, 1990.

[14] McGuire, Kevin. “Controlling Chaos: a Simple Deterministic Program for Drawing Complex Organic Shapes”, *3rd International Conference on Generative Art*, Milan, 2000.

[15] McGuire, Kevin. “ArpEgg: a Rewriting Grammar for Complex Arpeggios”, *9th Generative Art Conference*, Milan, 2006.