Book Review:

Allen Newell, *Unified Theories of Cognition*[*]

Jordan B. Pollack

Laboratory for AI Research, The Ohio State University,
2036 Neil Ave. Columbus, OH 43210

1. Introduction

Besides being a status report on the Soar project, *Unified Theories of Cognition* is Allen Newell's attempt at directing the field of Cognitive Science by example. Newell argues that his approach to "unification," which involves the programmed extension of a single piece of software-architecture-as-theory to as many psychological domains as possible, is the proper research methodology for Cognitive Science today:

> In this book I'm not proposing Soar as *the* unified theory of cognition. Soar is, of course, an interesting candidate. With a number of colleagues I am intent on pushing Soar as hard as I can to make it into a viable unified theory. But my concern here is that cognitive scientists consider working with *some* unified theory of cognition. Work with ACT*, with CAPS, with Soar, with CUTC, a connectionist unified theory of cognition. Just work with some UTC. (Newell, p. 430)

Over the past decade, Newell and his colleagues at numerous universities (including my own) have applied Soar to a number of different domains, and have adopted a goal of making it toe the line on psychological results. This is a very ambitious goal, and Newell knows it:

> "The next risk is to be found guilty of the sin of presumption. Who am I, Allen Newell, to propose a unified theory of cognition...Psychology must wait for its Newton" (Newell, p. 37)

Newell is clearly entitled by a life of good scientific works to write a book at such a level, and in my opinion, it is the most substantial and impressive, by far, of recent offerings on the grand unified mind. My entitlement to review his book is less self-evident, however—who am I to stand in judgement over one of the founding fathers of the field? And so I fear I am about to commit the sin of presumption as well, and to compound it, moreover, with the sin of obliqueness: Because my argument is not with the quality of Newell's book, but with the direction he is advocating for Cognitive Science, I will not review his theory in detail. Rather, I will adopt a bird's eye view and engage only the methodological proposal. I will, however, belabor one small detail of Newell's theory, its name, and only to use as my symbolic launching pad.

2. Artificial Intelligence and Mechanical Flight

The origin of the name Soar, according to high-level sources within the project, was originally an acronym for three primitive components of the problem-space method. But shortly, these components were forgotten, leaving the proper noun in their stead, a name which evokes "grand and glorious things," and also puts us in mind of the achievement of Mechanical Flight, AI's historical *doppelgänger*.

---

[*] (Cambridge: Harvard University Press); 549 pages. This review is to appear in *Artificial Intelligence*.

> "Among those who had worked on the problem [of Mechanical Flight] I may mention [da Vinci, Cayley, Maxim, Parsons, Bell, Phillips, Lilienthal, Edison, Langley] and a great number of other men of ability. But the subject had been brought into disrepute by a number of men of lesser ability who had hoped to solve the problem through devices of their own invention, which had all of themselves failed, until finally the public was lead to believe that flying was as impossible as perpetual motion. In fact, scientists of the standing of Guy Lussac ... and Simon Newcomb ... had attempted to prove it would be impossible to build a flying machine that would carry a man."[1]

I will leave the substitution of contemporary scientists to the reader. Simply put, the analogy "Airplanes are to birds as smart machines will be to brains," is a widely repeated AI mantra with several uses. One is to entice consumers by reminding them of the revolution in warfare, transportation, commerce, etc. brought about by mechanical flight. Another is to encourage patience in those same consumers by pointing to the hundreds of years of experimental work conducted before the success of mechanical flight! A third is to chant it, eyes closed, ignoring the complex reality of biological mechanism.

Although it is quite likely that the analogy between AI and mechanical flight arose spontaneously in the community of AI pioneers, its earliest written appearance seems to be in a "cold war for AI" essay by Paul Armer, then of the Rand Corporation, in the classic collection *Computers and Thought*:

> "While it is true that Man wasted a good deal of time and effort trying to build a flying machine that flapped its wings like a bird, the important point is that it was the understanding of the law of aerodynamic lift (even though the understanding was quite imperfect at first) over an airfoil which enabled Man to build flying machines. A bird isn't sustained in the air by the hand of God--natural laws govern its flight. Similarly, natural laws govern what [goes on inside the head]. Thus I see no reason why we won't be able to duplicate in hardware the very powerful processes of association which the human brain has, once we understand them." (Armer, 1963 p.398)

We all agree that once we understand how natural law governs what goes on in the head, we will be able to mechanize thought, and will then have the best scientific theory of cognition, which could be refined into the technology for "general intelligence." But our field has basically ignored natural law, and settled comfortably upon methodologies and models which involve only the perfect simulation of arbitrary "software" laws. I believe that we are failing to integrate several key principles which govern cognition and action in biological and physical systems, and that the incorporation of these should be the priority of cognitive science rather than of the writing of large programs.

3. Deconstructing the Myths of Mechanical Flight

There are two myths in Armer's analogy which are important to correct. The first is that flight is based mainly upon the principle of the airfoil. The second is that the mechanical means by which nature solved the problem are irrelevant. Translating through the analogy, these two myths are equivalent to believing that cognition is based mainly upon the principle of universal computation, and that the mechanical means by which nature solved the problem are irrelevant.

---

[1] Orville Wright, *How we invented the airplane*(New York: Dover, 1988), p. 12. This book is a reissued collection of essays and photographs about the Wright's research and development process. It includes three essays by Orville Wright, and two interpretive essays by Fred C. Kelly. Subsequent citations are to this edition.

Although my favorite sections of Newell's book are those in which he emphasizes the importance of constraints from biology and physics, he conducts his research in a way which is consistent with the myths. Indeed the myths are principally supported by his arguments, both the Physical Symbol System Hypothesis [16], which is the assertion that Universal Computation is enough, and the Knowledge Level Hypothesis [17], which legitimizes theories involving only software laws, even though their very existence is based only upon introspection.

In order to see why these myths are a stumbling block to the achievement of mechanical cognition, I will examine several aspects of the solution to mechanical flight, using the reports by Orville Wright.

3.1. The Airfoil Principle

The principle of aerodynamic lift over an airfoil was around for hundreds of years before the advent of mechanical flight. The Wright Brothers just tuned the shape to optimize lift:

> The pressures on squares are different from those on rectangles, circles, triangles or ellipses; arched surfaces differ from planes, and vary among themselves according to the depth of curvature; true arcs differ from parabolas, and the latter differ among themselves; thick surfaces from thin...the shape of an edge also makes a difference, so thousands of combinations are possible in so simple a thing as a wing...Two testing machines were built [and] we began systematic measurements of standard surfaces, so varied in design as to bring out the underlying causes of differences noted in their pressures (Wright, p. 84)

Assume that the principle of Universal Computation is to AI what the principle of aerodynamic lift is to mechanical flight. In chapter 2 of this book, Newell reiterates, in some detail, the standard argument for the status quo view of cognition as symbolic computation:

- Mind is flexible, gaining power from the formation of "indefinitely rich representations" and an ability to compose transformations of these representations.(p 59-63)
- Therefore mind must be a universal symbol processing machine (p 70-71)
- It is believed that most universal machines are equivalent (p. 72)

If one holds that the flexibility of the mind places it in the same class as the other universal machines (subject to physical limits, of course), then the mathematics tells us we can use any universal computational model for describing mind or its subparts (and the biological path is irrelevant.) So, for example, any of the 4 major theories of computation developed (and unified) this century — Church's Lambda calculus, Post's Production System, Von Neumann's Stored Program Machine, and Universal Automata (e.g. Turing's Machine) — could be used as a basis for a theory of mind. Of course, these theories were too raw and difficult to program, but have evolved through human ingenuity into their modern equivalents, each of which is a "Universal Programming Language" (UPL): Lisp, OPS5, C, and ATN's, respectively.

If we plan to express our UTCs in UPLs, however, we must have a way to distinguish between these UPLs in order to put some constraints on our UTCs, so that they aren't so general as to be vacuous. There are at least five general strategies to add constraints to a universal system: architecture, tradition, extra-disciplinary goals, parsimony, and ergonomics.

The first way to constrain a UPL is to build something, anything, on top of it, which constrains either what it can compute, how well it can compute it, or how it behaves

while computing it. This is called architecture, and Newell spends pages 82-88 discussing architecture in some detail. In a universal system, one can build architecture on top of architecture and shift attention away from the basic components of the system to arbitrary levels of abstraction.

The second method of constraining a universal theory is by sticking to "tradition," the practices whose success elevates them to beliefs handed down orally though the generations of researchers. Even though any other programming language would serve, the tradition in American AI is to build systems from scratch, using LISP, or to build knowledge into production systems. Newell is happy to represent the "symbolic cognitive psychology" tradition (p. 24) against the paradigmatic revolutions like Gibsonianism [10] or PDPism [21].

A third way we can distinguish between competing universals is to appeal to scientific goals outside of building a working program. We can stipulate, for example, that the "most natural" way to model a phenomenon in the UPL must support extra-disciplinary goals. Algorithmic efficiency, a goal of mainstream computer science, can be used to discriminate between competing models for particular tasks. Robust data from psychological experiments can be used to discriminate among universal theories on the basis of matching an implemented system's natural behavior with the observed data from humans performing the task. Finally, as a subset of neural network researchers often argue, the model supporting the theory must be "biologically correct." Newell, of course, relies very heavily on the psychological goals to justify his particular UPL.

Fourth, parsimony can be called into play. Two theories can be compared as to the number of elements, parameters, and assumptions needed to explain certain phenomena, and the shorter one wins. Newell makes a good point that the more phenomena one wishes to explain, the more benefit is gained from a unified theory, which ends up being shorter than the sum of many independent theories. This is called "Amortization of theoretical constructs" (p. 22) and is one of Newell's main arguments for why psychologists ought to adopt his unified theory paradigm for research. However, such a unification can be politically difficult to pull off when different sub-disciplines of a field are already organized by the parsimony of their own sub-theories.

Fifth, we might be able to choose between alternatives on the basis of ergonomics. We can ask the question of programmability. How easy is it to extend a theory by programming? How much work is it for humans to understand what a system is doing? The Turing Machine model "lost" to the stored program machine due to the difficulty of programming in quintuples. Systems which use explicit rules rather than implicit knowledge-as-code are certainly easier to understand and debug, but may yield no more explanatory power.

However, unless the constraints specifically strip away the universality, such that the cognitive theory becomes *an application of* rather than *an extension to* the programming language, the problem of theoretical underconstraint remains. Following Newell's basic argument, one could embark on a project of extensively crafting a set of cognitive models in any programming language, say C++, matching psychological regularities, and reusing subroutines as much as possible, and the resultant theory would be as predictive as Soar:

> Soar does not automatically provide an explanation for anything just because it is a universal computational engine. There are two aspects to this assertion. First from the perspective of cognitive theory, Soar has to be universal, because humans themselves are universal. To put this the right way around— Soar is a universal computational architecture; therefore it predicts that the human cognitive architecture is likewise universal. (Newell, p. 248)

Thus, just as the Wright brothers discovered different lift behaviors in differently shaped airfoils, cognitive modelers will find different behavioral effects from different universal language architectures. The principle of the airfoil was around for hundreds of years, and yet the key to mechanical flight did not lie in optimizing lift. Using a UPL which optimizes "cognitive lift" is not enough, either, as practical issues of scale and control will still assert themselves.

3.2. Scaling Laws

> "Our first interest [in the problem of flight] began when we were children. Father brought home to us a small toy actuated by a rubber [band] which would lift itself into the air. We built a number of copies of this toy, which flew successfully, but when we undertook to build a toy on a much larger scale it failed to work so well." (Wright p.11)

The youthful engineers did not know that doubling the size of a model would require 8 times as much power. This is the common feeling of every novice computer programmer hitting a polynomial or exponential scaling problem with an algorithm. But there were many other scaling problems which were uncovered during the Wrights' mature R&D effort, beyond mere engine size:

> We discovered in 1901 that tables of air pressures prepared by our predecessors were not accurate or dependable." (Wright, p. 55)

> We saw that the calculations upon which all flying-machines had been based were unreliable, and that all were simply groping in the dark. Having set out with absolute faith in the existing scientific data, we were driven to doubt one thing after another...Truth and error were everywhere so intimately mixed as to be indistinguishable (Wright p. 84)

Thus it is not unheard of for estimates of scaling before the fact to be way off, especially the first time through based upon incomplete scientific understanding of the important variables. Estimates of memory size [12, 13] for example, or the performance capacity of a brain [15, 24] may be way off, depending on whether memory is "stored" in neurons, synapses, or in modes of behaviors of those units. So the number of psychological regularities we need to account for in a UTC may be off:

> Thus we arrive at about a third of a hundred regularities about [typing] alone. Any candidate architecture must deal with most of these if it's going to explain typing...Of course there is no reason to focus on typing. It is just one of a hundred specialized areas of cognitive behavior. It takes only a hundred areas at thirty regularities per area to reach the ˜˜3000 total regularities cited at the beginning of this chapter...Any architecture, especially a candidate for a unified theory of cognition, must deal with them all--hence with thousands of regularities. (Newell, p. 243)

There is a serious question about whether thousands of regularities is enough, and Newell recognizes this:

> In my view its it time to get going on producing unified theories of cognition -- before the data base doubles again and the number of visible clashes increases by the square or cube. (Newell, p 25)

While Newell estimates 3000 regularities, my estimate is that the number of regularities is unbounded. The "Psychological Data" industry is a generative system, linked to the fecundity of human culture, which Newell also writes about lucidly:

> What would impress [The Martian Biologist] most is the efflorescence of adaptation. Humans appear to go around simply creating opportunities of all kinds to build different response functions. Look at the variety of jobs in the world. Each one has humans using different kinds of response functions. Humans invent games. They no sooner invent one game than they invent new ones. They not only invent card games, but they collect them in a book and publish them...Humans do not only eat, as do all other animals, they prepare their food... inventing [hundreds and thousands of] recipes." (Newell, p. 114)

Every time human industry pops forth with a new tool or artifact, like written language, the bicycle, the typewriter, the rubik's cube, or rollerblades, another 30 regularities will pop out, especially if there is a cost justification to do the psychological studies, as clearly was the case for typing and for reading. This is not a good situation, especially if programmers have to be involved for each new domain. There will be a never-ending software battle just to keep up:

> Mostly, then, the theorist will load into Soar a program (a collection of productions organized into problem spaces) of his or her own devising...The obligation is on the theorist to cope with the flexibility of human behavior in responsible ways. (Newell, p. 249)

If the road to unified cognition is through very large software efforts, such as Soar, then we need to focus on scalable control laws for software.

3.3. Control in High Winds

Although we might initially focus on the scaling of static elements like wing span and engine power, *to duplicate the success of mechanical flight, we should focus more on the scaling of control.* For the principal contribution of the Wright brothers was not the propulsive engine, which had its own economic logic (like the computer), nor the airfoil, a universal device they merely tuned through experiments, but their insight about how to control a glider when scaled up enough to carry an operator:

> Lilienthal had been killed through his inability to properly balance his machine in the air. Pilcher, an English experimenter had met with a like fate. We found that both experimenters had attempted to maintain balance merely by the shifting of the weight of their bodies. Chanute and all the experimenters before 1900, used this same method of maintaining the equilibrium in gliding flight. We at once set to work to devise a more efficient means of maintaining the equilibrium... It was apparent that the [left and right] wings of a machine of the Chanute double-deck type, with the fore-and-aft trussing removed, could be warped... in flying...so as to present their surfaces to the air at different angles of incidences and thus secure unequal lifts....(Wright p.12)

What they devised, and were granted a monopoly on, was the *Aileron Principle*, the general method of maintaining dynamical equilibrium in a glider by modifying the shapes of the individual wings, using cables, to provide different amounts of lift to each side. (It is not surprising that the Wright brothers were bicycle engineers, as this control principle is the same one used to to control two wheeled vehicles—iterated over-correction towards the center.)

Translating back through our analogy, extending a generally intelligent system for a new application by human intervention in the form of programming is "seat of the pants" control, the same method that Lilienthal applied to maintaining equilibrium by shifting the weight of his body.

Just as the source of difficulty for mechanical flight was that scaling the airfoil large enough to carry a human overwhelmed that human's ability to maintain stability, the source of difficulty in the software-engineering approach to unified cognition is that scaling software large enough to explain cognition overwhelms the programming-teams' ability to maintain stability.

It is well known that there are limiting factors to software engineering [5], and these limits could be orders of magnitude below the number of "lines of code" necessary to account for thousands of psychological regularities or to achieve a "general intelligence." Since software engineers haven't figured out how to build and maintain programs bigger than 10-100 million lines of code, why should people in AI presume that it can be done as a matter of course? [7].

What is missing is some control principle for maintaining dynamical coherence of an ever-growing piece of software in the face of powerful winds of change. While I don't pretend to have the key to resolving the software engineering crisis, I believe its solution may rest with building systems from the bottom up using robust and stable cooperatives of goal-driven modules locked into long-term prisoners dilemmas [2], instead of through the centralized planning of top-down design. The order of acquisition of stable behaviors can be very important to the solution of hard problems.

3.4. On the irrelevancy of flapping

> Learning the secret of flight from a bird was a good deal like learning the secret of magic from a magician. After you know the trick and know what to look for, you see things that you did not notice when you did not know exactly what to look for. (Wright (attributed) p. 5)

When you look at a bird flying, the first thing you see is all the flapping. Does the flapping explain how the bird flies? Is it reasonable to theorize that flapping came first, as some sort of cooling system which was recruited when flying became a necessity for survival? Not really, for a simpler explanation is that most of the problem of flying is in finding a place within the weight/size dimension where gliding is possible, and getting the control system for dynamical equilibrium right. Flapping is the last piece, the propulsive engine, but in all its furiousness, it blocks our perception that the bird first evolved the aileron principle. When the Wrights figured it out, they saw it quite clearly in a hovering bird.

Similarly, when you look at cognition, the first thing you see is the culture and the institutions of society, human language, problem solving, and political skills. Just like flapping, symbolic thought is the last piece, the engine of social selection, but in all its furiousness it obscures our perception of cognition as an exquisite control system competing for survival while governing a very complicated real-time physical system.

Once you get a flapping object, it becomes nearly impossible to hold it still enough to retrofit the control system for equilibrium. Studying problem solving and decision-making first because they happen to be the first thing on the list (Newell p. 16), is dangerous, because perception and motor control may be nearly impossible to retrofit into the design.

This retrofit question permits a dissection of the "biological correctness" issue which has confounded the relationship between AI and connectionism. The naive form, which applied to work in computational neuroscience but not to AI, is "ontogenetic correctness," the goal of constructing one's model with as much neural realism as possible. The much deeper form, which could be a teleological principle someday, is "phylogenetic

correctness," building a model which could have evolved bottom up, without large amounts of arbitrary top-down design. Phylogenetically correct systems acquire their behaviors in a bottom-up order that could, theoretically, recapitulate evolution or be "reconverged" upon by artificial evolution. Thus, while the airplane does not flap or have feathers, the Wrights' success certainly involved "recapitulation" of the phylogenetic order of the biological invention of flight: the airfoil, dynamical balance, and then propulsion.

4. Physical Law versus Software Law

By restricting ourselves to software theories only, cognitive scientists might be expending energy on mental ornithopters. We spin software and logic webs endlessly, forgetting every day that there is no essential difference between Fortran programs and Lisp programs, between sequential programs and production systems, or ultimately, between logics and grammars. All such theories rely on "Software Law" rather than the natural law of how mechanisms behave in the universe.

Software laws, such as rules of predicate logic, may or may not have existed before humans dreamed them up. And they may or may not have been "implemented" by minds or by evolution. What is clear is that such laws can be created *ad infinitum*, and then simulated and tested on our physical symbol systems: The computer simulation is the sole guaranteed realm of their existence.

An alternative form of unification research would be to *unify cognition with nature*. In other words, to be able to use the same kind of natural laws to explain the complexity of form and behavior in cognition, the complexity of form and behavior in biology, and the complexity of form and behavior in inanimate mechanisms.

I realize this is not currently a widely shared goal, but by applying Occam's razor to "behaving systems" on all time scales (Newell, p. 152) why not use the ordinary equations of physical systems to describe and explain the complexity and control of all behavior? In an illuminating passage, Newell discusses control theory:

> To speak of the mind as a controller suggests immediately the language of control systems -- of feedback, gain, oscillation, damping, and so on, It is a language that allows us to describe systems as purposive. But we are interested in the full range of human behavior, not only walking down a road or tracking a flying bird, but reading bird books, planning the walk, taking instruction to get to the place, identifying distinct species, counting the new additions to the life list of birds seen, and holding conversations about it all afterward. When the scope of behavior extends this broadly, it becomes evident that the language of control systems is really locked to a specific environment and class of tasks -- to continuous motor movement with the aim of pointing or following. For the rest it becomes metaphorical. (p. 45)

I think Newell has it backwards. It is the language of symbol manipulation which is locked to a specific environment of human language and deliberative problem-solving. Knowledge level explanations only metaphorically apply to complex control systems such as insect and animal behavior [4, 6], systems of the body such as the immune or circulatory systems, the genetic control of fetal development, the evolutionary control of populations of species, cooperative control in social systems, or even the autopoetic control system for maintaining the planet. Certainly these systems are large and complex and have some means of self-control while allowing extreme creativity of behavior, but the other sciences do not consider them as instances of universal computing systems running software laws, divorced from the physical reality of their existence!

We have been led up the garden path of theories expressed in rules and representations because simple mathematical models, using ordinary differential equations, neural networks, feedback control systems, stochastic processes, etc. have for the most part been unable to describe or explain the generativity of structured behavior with unbounded dependencies, especially with respect to language [8]. This gulf between what is needed for the explanation of animal and human cognitive behavior and what is offered by ordinary scientific theories is really quite an anomaly and indicates that our understanding of the principles governing what goes on in the head have been very incomplete.

But where might governing principles for cognition come from besides computation? The alternative approach I have been following over the past several years has emerged from a simple goal to develop neural network computational theories which gracefully admit the generative and representational competance of symbolic models. This approach has resulted in two models [18, 19] with novel and interesting behaviors. In each of these cases, when pressed to the point of providing the same theoretical capacity as a formal symbol system, I was forced to interpret these connectionist networks from a new point of view, involving fractals and chaos -- a dynamical view of cognition, more extreme than that proposed by Smolensky [23]. I have thus been lead to a very different theoretical basis for understanding cognition, which I will call the "Dynamical Cognition Hypothesis," that:

> The recursive representational and generative capacities required for cognition arise directly out of the complex behavior of non-linear dynamical systems.

In other words, neural networks are merely the biological implementation level for a computation theory not based upon symbol manipulation, but upon complex and fluid patterns of physical state. A survey of cognitive models based upon non-linear dynamics is beyond this review [20], however, I can briefly point to certain results which will play an important role in this alternative unification effort.

Research in non-linear systems theory over the past few decades has developed an alternative explanation for the growth and control of complexity [11]. Hidden within the concept of deterministic "chaotic" systems which are extremely sensitive to small changes in parameters, is the surprise that precise tuning of these parameters can lead to the generation of structures of enormous apparent complexity, such as the famous Mandelbrot set [14].

There is a clear link between simple fractals, like Cantor dust, and rewriting systems ("remove the middle third of each line segment"), and Barnsley has shown how such recursive structures can be found in the limit behavior of very simple dynamical systems [3]. The very notion of a system having a "fractional dimension" is in fact the recognition that its apparent complexity is governed by a "power law" [22].

The equations of motion of non-linear systems are not different in kind from those of simpler physical systems, but the evoked behavior can be very complicated, to the point of appearing completely random. Even so, there are "universal" laws which govern these systems at all scales, involving where and when phase transitions occur and how systems change from simple to complex modes, passing through "critical" states, which admit long-distance dependencies between components.

The logistic map $x_{t+1} = kx_t(1 - x_t)$ is a well studied example of a simple function iterated over the unit line where changes in k (between 0 and 4) lead to wildly different behaviors, including convergence, oscillation and chaos. In a fundamental result, Crutchfield & Young have exhaustively analysed sequences of most-significant-bits generated by this map[2] and have shown that at critical values of k, such as 3.5699, these bit

sequences have unbounded dependencies, and are not describable by a regular grammar, but by an indexed context-free grammar [9].

Without knowing where complex behavior comes from, in the logistic map, in critically tuned collections of neural oscillators, or in the Mandelbrot set, one could certainly postulate a very large rule-based software system, operating omnipotently behind the scenes, like a deity who governs the fate of every particle in the universe.

5. Conclusion

I want to conclude this review with a reminder to the reader to keep in mind the "altitude" of my criticism, which is about the research methodology Newell is proposing based upon the status quo of myths in AI, and not about the detailed contents of the book. These are mature and illuminated writings, and Newell does an excellent job of setting his work and goals into perspective, and recognizing the limitations of his theory, especially with respect to the puzzles of development and language.

Despite my disagreement with Newell's direction, I was educated and challenged by the book, and endorse it as an elevator for the mind of all students of cognition. But still, I would warn the aspiring cognitive scientist not to climb aboard any massive software engineering efforts, expecting to fly:

> You take your seat at the center of the machine beside the operator. He slips the cable, and you shoot forward...The operator moves the front rudder and the machine lifts from the rail like a kite supported by the pressure of the air underneath it. The ground is first a blur, but as you rise the objects become clearer. At a height of 100 feet you feel hardly any motion at all, except for the wind which strikes your face. If you did not take the precaution to fasten your hat before starting, you have probably lost it by this time....(Wright, p 86)

References

[1] P. Armer, Attitudes towards Artificial Intelligence, in *Computers & Thought*, E. A. Feigenbaum and J. Feldman (ed.), McGraw Hill, New York, 1963, 389-405.

[2] R. Axelrod, *The evolution of cooperation*, Basic Books, New York, 1984.

[3] M. F. Barnsley, *Fractals Everywhere*, Academic Press, San Diego, 1988.

[4] R. Beer, *Intelligence as adaptive behavior: An experiment in computational neuroethology*, Academic Press, New York, 1990.

[5] F. P. Brooks, *The Mythical Man-Month*, Addison-Wesley, Reading, MA, 1975.

[6] R. Brooks, Intelligence without Representation, *Artificial Intelligence 47*, 1-3 (1991), 139-160.

[7] C. Cherniak, Undebuggability and cognitive science, *Communications of the Association for Computing Machinery 31*, 4 (1988), 402-412.

[8] N. Chomsky, *Syntactic structures*, Mouton and Co., The Hague, 1957.

[9] J. P Crutchfield and K. Young, Computation at the Onset of Chaos, in *Complexity, Entropy and the Physics of INformation*, W. Zurek (ed.), Addison-Wesley, Reading, MA, 1989.

[10] J. J. Gibson, The Ecological Approach to Visual Perception, , Houghton-Mifflin.

[11] J. Gleick, *Chaos: Making a new science*, Viking, New York, 1987.

[12] W. D. Hillis, Intelligence as emergent behavior; or, the songs of eden, *Daedelus 117*, (1988), 175-190.

[13] T. K. Landauer, How much do people remember? Some estimates on the quantity of learned information in long-term memory., *Cognitive Science 10*, (1986), 477-494.

[2] They analysed the bit string $y_t = floor(0.5 + x_t)$.

[14] B. Mandelbrot, *The Fractal Geometry of Nature*, Freeman, San Francisco, 1982.

[15] H. Moravec, *Mind Children*, Harvard University Press, Cambridge, 1988.

[16] A. Newell and H. A. Simon, Computer sciencs as empirical inquiry: Symbols and search, *Communications of the Association for Computing Machinery 19*, 3 (1976), 113-126.

[17] A. Newell, The knowledge level, *Artificial Intelligence 18*, (1982), 87-127.

[18] J. B. Pollack, Recursive Distributed Representation, *Artificial Intelligence 46*, (1990), 77-105.

[19] J. B. Pollack, The Induction of Dynamical Recognizers, *Machine Learning 7*, (1991), 227-252.

[20] R. Port and T. Van Gelder, Mind as Motion, , In Preparation.

[21] D. E. Rumelhart, J. L. McClelland and The PDP research group, *Parallel Distributed Processing: Experiments in the Microstructure of Cognition*, MIT Press, Cambridge, 1986.

[22] M. Schroeder, *Fractals, Chaos, Power Laws*, W. H. Freeman, New York, 1991.

[23] P. Smolensky, Information Processing in Dynamical Systems: Foundations of Harmony Theory, in *Parallel Distributed Processing: Experiments in the Microstructure of Cognition*, vol. 1, D. E. Rumelhart, J. L. McClelland and the PDP research Group (ed.), MIT Press, Cambridge, 1986, 194-281.

[24] J. Von Neumann, *The Computer and the Brain*, Yale University Press, New Haven, 1958.