The reductionist blind spot

Russ Abbott

Department of Computer Science, California State University, Los Angeles, California
Russ.Abbott@GMail.com

Abstract. Can there be higher level laws of nature even though everything is reducible to the fundamental laws of physics? The computer science notion of level of abstraction explains how there can be. The key relationship between elements on different levels of abstraction is not the is-composed-of relationship but the implements relationship.

1. Introduction

When a male Emperor penguin stands for two frigid months balancing an egg on its feet to keep it from freezing, are we to understand that behavior in terms of quarks and other fundamental particles? It seems somehow unreasonable, but that’s the reductionist position. Here’s how Albert Einstein (1918) put it.

The painter, the poet, the speculative philosopher, and the natural scientist … each in his own fashion, tries to make for himself … a simplified and intelligible picture of the world.

What place does the theoretical physicist’s picture of the world occupy among these? … In regard to his subject matter … the physicist … must content himself with describing the most simple events which can be brought within the domain of our experience … .

But what can be the attraction of getting to know such a tiny section of nature thoroughly, while one leaves everything subtler and more complex shyly and timidly alone? Does the product of such a modest effort deserve to be called by the proud name of a theory of the universe?

In my belief the name is justified; for the general laws on which the structure of theoretical physics is based claim to be valid for any natural phenomenon whatsoever. With them, it ought to be possible to arrive at … the theory of every natural process, including life, by means of pure deduction. … The supreme task of the physicist is to arrive at those elementary universal laws from which the cosmos can be built up by pure deduction. [emphasis added]

The italicized portion expresses what Anderson (1972) calls (and rejects) the constructionist hypothesis: the idea that one can start with physics and reconstruct the universe.
More recently Steven Weinberg (2001) restated Einstein’s position as follows. Grand reductionism is … the view that all of nature is the way it is … because of simple universal laws, to which all other scientific laws may in some sense be reduced. …

Every field of science operates by formulating and testing generalizations that are sometimes dignified by being called principles or laws. … But there are no principles of chemistry that simply stand on their own, without needing to be explained reductively from the properties of electrons and atomic nuclei, and … there are no principles of psychology that are free-standing, in the sense that they do not need ultimately to be understood through the study of the human brain, which in turn must ultimately be understood on the basis of physics and chemistry.

Not all physicists agree with Einstein and Weinberg. As Erwin Schrödinger (1944) wrote,

[L]iving matter, while not eluding the 'laws of physics' … is likely to involve 'other laws,' [which] will form just as integral a part of [its] science.

In arguing against the constructionist hypothesis Anderson (1972) extended Schrödinger’s thought.

[T]he ability to reduce everything to simple fundamental laws … [does not imply] the ability to start from those laws and reconstruct the universe. …

At each level of complexity entirely new properties appear. … [O]ne may array the sciences roughly linearly in [a] hierarchy [in which] the elementary entities of [the science at level n+1] obey the laws of [the science at level n]: elementary particle physics, solid state (or many body) physics, chemistry, molecular biology, cell biology, …. psychology, social sciences. But this hierarchy does not imply that science [n+1] is ‘just applied [science n].’ At each [level] entirely new laws, concepts, and generalization are necessary.

Notwithstanding their disagreements, all four physicists (and many others) agree that everything can be reduced to the fundamental laws of physics. Here’s how Anderson put it.

[The] workings of all the animate and inanimate matter of which we have any detailed knowledge are … controlled by the … fundamental laws [of physics]. … [W]e must all start with reductionism, which I fully accept.

Einstein and Weinberg argue that that’s the end of the story. Starting with the laws of physics and with sufficiently powerful deductive machinery one should be able to reconstruct the universe. Schrödinger and Anderson disagree. They say that there’s more to nature than the laws of physics—but they were unable to say what that might be.

Before going on, you may want to answer the question for yourself. Do you agree with Einstein and Weinberg or with Schrödinger and Anderson? Is there more than physics—and if so, what is it?
2. **Preview**

The title of this paper gives away my position. I agree with Schrödinger and Anderson.

The computer science notion of level of abstraction explains how there can be higher level laws of nature—even though everything is reducible to the fundamental laws of physics. The basic idea is that a level of abstraction has both a specification and an implementation. The implementation is a reduction of the specification to lower level functionality. But the specification is independent of the implementation. So even though a level of abstraction depends on lower level phenomena for its realization it cannot be reduced to that implementation without losing something important, namely its specification.

3. **Levels of abstraction**

A level of abstraction (Guttag 1977) is (a) a collection of types—which for the most part means categories—and (b) operations that may be applied to entities of those types. A standard example is the stack, which is defined by the following operations.

- `push(stack: s, element: e)` — Push an element e into a stack s and return the stack.
- `pop(stack: s)` — Pop the top element off the stack s and return the stack.
- `top(stack: s)` — Return (but don't pop) the top element of a stack s.

Although the intuitive descriptions are important for us as readers, all we have done so far is to declare a number of operations. How are their meanings defined? Axiomatically.

- `top(push(stack: s, element: e)) = e.` — After e is pushed onto a stack, its top element is e.
- `pop(push(stack: s, element: e)) = s.` — After pushing e onto s and then popping it off, s is as it was.

Together, these declarations and axioms define a stack as anything to which the operations can be applied while satisfying the axioms.

This is similar to how mathematics is axiomatized. Consider the non-negative integers as specified by Peano’s axioms.\(^1\)

1. **Zero is a number.**
2. **If A is a number, the successor of A is a number.**

\(^1\) As given in Wolfram’s MathWorld: [http://mathworld.wolfram.com/PeanosAxioms.html](http://mathworld.wolfram.com/PeanosAxioms.html)
3. Zero is not the successor of a number.

4. Two numbers of which the successors are equal are themselves equal.

5. (Induction axiom) If a set $S$ of numbers contains zero and also the successor of every number in $S$, then every number is in $S$.

These axioms specify the terms zero, number, and successor. Here number is a type, Zero is an entity of that type, and successor is an operation on numbers. These terms stand on their own and mean (formally) no more or less than the definitions say they mean.

Notice that in neither of these definitions were the new terms defined in terms of pre-existing terms. Neither a number nor a stack is defined as a special kind of something else. Both Peano’s axioms and the stack definition define terms by establishing relationships among them. The terms themselves, stack and a number, are defined ab initio and solely in terms of operations and relationships among those operations.

This is characteristic of levels of abstraction. When specifying a level of abstraction the types, objects, operations, and relationships at that level stand on their own. They are not defined in terms of lower level types, objects, operations, and relationships.

See the sidebar on how levels of abstraction function in different disciplines.

4. **Unsolvability and the Game of Life**

The Game of Life is a 2-dimensional cellular automaton in which cells are either alive (on) or dead (off). Cells turn on or off synchronously in discrete time steps according to rules that specify cell behavior as a function of their eight neighbors.

- Any cell with exactly three live neighbors will stay alive or become alive.
- Any live cell with exactly two live neighbors will stay alive.
- All other cells die.

The preceding rules are analogous to the fundamental laws of physics. They determine everything that happens on a Game of Life grid.

Certain on-off cell configurations create patterns—or really sequences of patterns. The glider is the best known. When a glider is entered onto an empty grid and the rules applied, a series of patterns propagates across the grid. Since nothing actually moves in the Game of Life—the concept of motion doesn’t even exist—how should we understand this?

Gliders exist on a different level of abstraction from that of the Game of Life. At the Game of Life level there is nothing but grid cells—in fixed positions. But at the glider level not only do gliders move, one can even write equations for the
number of time steps it will take a glider to move from one location to another. What is the status of such glider velocity equations?

Before answering that question, recall that it’s possible to implement Turing machines by arranging gliders and other Game of Life patterns. Just as gliders are subject to the laws of glider equations, Turing machines too are subject to their own laws—in particular, computability theory.

Game of Life gliders and Turing machines exemplify the situation described by Schrödinger. They are phenomena that appear on a Game of Life grid but are governed by laws that apply on a different and independent level of abstraction. While not eluding the Game of Life rules, autonomous new laws apply to them. These additional laws are not expressible in Game of Life terms. There is no such thing as a glider or a Turing machine at the Game of Life level. The Game of Life is nothing but a grid of cells along with rules that determine when cells go on and off. In other words, Game of Life gliders and a Game of Life Turing machines (a) are governed by laws that are independent of the GoL rules while at the same time they (b) are completely determined by the GoL rules.

5. Evolution is also a property of a level of abstraction

Evolution offers another example of how levels of abstraction give rise to new laws. Evolution is an abstract process that can be described as follows.

Evolution occurs in the context of a population of entities. The entities exist in an environment within which they may survive and reproduce. The entities have properties that affect how they interact with their environment. Those interactions help determine whether the entities will survive and reproduce. When an entity reproduces, it produces offspring which inherit its properties, possibly along with some random variations, which may result in new properties. In some cases, pairs of entities reproduce jointly, in which case the offspring inherit some combination of their parent’s properties—perhaps also with random variations.

The more likely an entity is to survive and reproduce, the more likely it is that the properties that enabled it to survive and reproduce will be passed on to its offspring. If certain properties—or random variations of those properties, or the random creation of new properties—enable their possessors to survive and reproduce more effectively, those properties will propagate.

We call the generation and propagation of successful properties evolution. By helping to determine which entities are more likely to survive and reproduce, the environment selects the properties to be propagated—hence evolution by environmental (i.e., natural) selection.

The preceding description introduced a number of terms (in italics). As in the case of stacks and Peano numbers, the new terms are defined ab initio at the evo-
olution level of abstraction. The independent usefulness of evolution as a level of abstraction is illustrated by evolutionary computation, which uses the abstract evolutionary mechanism to solve difficult optimization problems. It does so in a way that has nothing to do with biology or natural environments.

6. The reductionist blind spot

Physics recognizes four fundamental forces. Evolution is not one of them. Similarly there is no computational functionality in a Game of Life universe. In other words, both evolution and Turing machine computation appear as phenomena within frameworks that are blind to their existence. Nevertheless, both evolution and Turing machine computation can be completely explained in terms of phenomena that operate as primitives within those frameworks. Given that, do we really need concepts such as evolution and Turing machine computation?

In some sense we don’t. Echoing Kim (1984), Schouten and de Jong (2007) put it this way.

If a higher level explanation can be related to physical processes, it becomes redundant since the explanatory work can be done by physics.

In this sense both evolution and computations done by Game of Life Turing machines are redundant. After all, Game of Life Turing machines as such don’t do anything. It is only the Game of Life rules that make cells go on and off. Reductionism has not been overthrown. One could trace the sequence of Game of Life rule applications that transform an initial Game of Life configuration (that could be described as a Turing machine with input \( x \)) into a final configuration (that could be described as a Turing machine with output \( y \)). One could do this with no mention of Turing machines.

Similarly one could presumably—albeit with great difficulty—trace the sequence of chemical and physical reactions and interactions that produce a particular chemical configuration (that could be described as the DNA that enables its possessor to thrive in its environment). One could do this with no mention of genes, codons, proteins, or other evolutionary or biological terms.

One can always reduce away macro-level terminology and phenomena and replace them with the underlying micro-level terminology and phenomena. It is still the elementary mechanisms—and nothing but those mechanisms—that turn the causal crank. So why not reduce away higher levels of abstraction?

Reducing away a level of abstraction produces a reductionist blind spot. Computations performed by Game of Life Turing machines cannot be described as computations when one is limited to the vocabulary of the Game of Life level. Nor can one explain why the Game of Life halting problem is unsolvable. These concepts exist only at the Turing machine level of abstraction. Similarly, biological
evolution cannot be explicated at the level of physics and chemistry. The evolutionary process exists only at the evolution level of abstraction.

Furthermore, reducing away a level of abstraction throws away elements of nature that have objective existence. At each level of abstraction there are entities (see sidebar)—such as Turing machines and biological organisms—that instantiate types at that level. These entities are simultaneously causally reducible and ontologically real—a formulation coined by Searle (2004) in another context. Entities on a level of abstraction that are implemented by a lower level of abstraction are causally reducible because the implementation provides the forces and mechanisms that drive them. But such entities are ontologically real because their specifications, which are independent of their implementations, characterize what they do and how they behave.

The goal of science is to understand nature. Reducing away levels of abstraction discards both real scientific explanations—such as the evolutionary mechanism—and objectively real entities—such as biological organisms. Denying the existence of biological organisms as entities, for example, requires that one throw away the notion of species. What is a species after all if there are no biological entities for it to collect as a distinguishable type? But without species one is forced to dismiss the grand taxonomy of life—with a place for all life forms from \textit{E. coli} to elephants—whose structure and history biology has been so successful in describing. What would be left of biology? Not much. Reducing away levels of abstraction is simply bad science.

Reducing away levels of abstraction is bad science from an information theoretic perspective as well. Chaitin (2003) points out that Leibniz anticipated algorithmic information theory when he characterized science as developing the simplest hypothesis (in the algorithmic information theory sense) for the richest phenomena. Throwing away levels of abstraction typically reduces the algorithmic complexity of a description of some phenomenon.

7. \textit{Constructionism and the principle of ontological emergence}

Game of Life Turing machines and biological evolution illustrate Schrödinger’s perception that although higher level phenomena don’t elude the laws of physics they are governed by new laws. Because the higher level laws are not derived from the laws governing the implementing level, knowledge of the lower level laws does not enable one to generate a specification and implementation of the higher level. That is, one would not expect to be able to deduce computability theory from knowledge of the Game of Life rules, and one would not expect to be able to deduce biological evolution from knowledge of fundamental physics. In other words, as Anderson argued and contrary to Einstein, constructionism fails.
No matter how much deductive power one has available, one should not expect to be able to start with the fundamental laws of physics and reconstruct all of nature.

In some ways the preceding statement is a bit of an exaggeration. Computability theory, after all, can be deduced from first principles. Since it can be deduced from first principles, throwing in the rules of the Game of Life as extra premises can’t hurt.

The point is that higher level abstractions are typically creative additions to lower levels. To focus more clearly on the creativity aspect, the notion that one could start with lower level elements and deduce higher level elements is similar to the notion that one could start with a mountain of granite and deduce the faces of Washington, Jefferson, Lincoln, and Roosevelt. The granite can be carved and molded into those faces. But given the normal interpretation of deduce it makes little sense to say that one could start with the granite and deduce the faces. The idea of carving those faces into the granite was a creative leap and not what would normally be considered a deduction.

With this in mind, though, constructionism can be said to succeed. It has taken billions of years, but nature has implemented biological organisms. And the faces of Washington, Jefferson, Lincoln, and Roosevelt, are on Mt. Rushmore. Nature accomplished this trick starting from quantum mechanics. If one considers nature as a mechanism that generates and implements new levels of abstraction, then nature embodies constructionism.

Nature does its work as a random enumerator of possibilities and not in the deductive/explanatory sense suggested by Einstein and Weinberg. Nonetheless, nature is a constructive process. Normally one doesn’t refer to nature’s processes as deductive. But just as software theorem provers work by searching the space of possible proofs until they find one that works, nature too proceeds by search, keeping levels of abstraction that work and discarding those that don’t. If software theorem provers are deductive, then so is nature.

Nature is continually generating new levels of abstractions. Which persist? It depends on the environment at the time. Molecules persist only in environments with low enough temperatures; biological organisms persist only in environments that provide nourishment; and hurricanes (the only non-biological and non-social dynamic entity of which I’m aware) persist only in environments with a supply of warm water. This can be summarized as the principle of ontological emergence: extant levels of abstraction are those whose implementations have materialized and whose environments enable their persistence.

It’s important to realize, though, that in generating new levels of abstraction nature does not build layered hierarchies. New entity types may interact with any existing entity type. The types are not partitioned into disjoint layers that don’t interact. This is nicely illustrated by the fact that the gecko, a very macro organism, makes direct use (Kellar 2002) of the quantum level van der Vaals force to cling to vertical surfaces.
8. Constraints and downward entailment

Higher level laws generally have lower level implications. Because the halting problem is unsolvable, for example, it is unsolvable whether an arbitrary Game of Life configuration will ever reach a stable state. And because the Game of Life can implement a Turing machine, the Game of Life can compute any computable function. In other words, computability theory, a law that applies to a Game of Life Turing machine, applies to the Game of Life itself. Similarly, at the cell-by-cell level, velocity equations for Game of Life gliders can be used to predict when a glider will “turn on” a particular cell.

The same phenomenon applies to evolution. When Darwin and Wallace described the evolution level of abstraction, they knew nothing about DNA. But their model required some mechanism for recording and transmitting properties. In other words, it makes a prediction that any implementation of the evolution level of abstraction must provide a mechanism for transmitting properties to offspring. Because biology implements the evolutionary level of abstraction one can conclude that biological organisms must have a means to transmit properties. We now know that DNA is that mechanism. Prediction confirmed.

When autonomous higher level laws apparently effect lower level phenomena the result has been called (Andersen 2000) downward causation. But downward causation doesn’t make scientific sense. It is always the lower level phenomena that determine the higher level. The Game of Life rules, not glider equations, are the only things that determine when and whether cells go on and off. How can computability theory and glider equations let us draw conclusions about Game of Life cells? How can evolution let us draw conclusions about biological organisms? In (Abbott 2006) I call this downward entailment. Autonomous laws that apply at a higher level of abstractions can have implications for elements at a lower level as long as the lower level is implementing the higher level.

When frozen into ice cubes, for example, H$_2$O molecules form a rigid lattice and are constrained to travel together as the ice cube that they implement is moved about. This is only common sense. As long as molecules of H$_2$O are implementing a solid, they are constrained by laws that govern solids. Once the ice cube melts and the H$_2$O molecules are no longer implementing a solid, they are no longer bound by the laws of solids. This parallels the argument made by Sperry (1970).

The fate of the entire population of atoms, molecules, and other components [that constitute a wheel rolling downhill] are determined very largely by the holistic properties of the whole wheel as a unit.

Putnam (1975) makes a similar argument. He asks how one should explain why a square peg won’t fit into a round hole whose diameter is the same length as the peg’s side. Should the explanation be based on quantum physics or on geometry? Putnam’s answer is that the explanation should be based on geometry. An explanation at the quantum physics level explains only the one particular peg-and-hole
pair under consideration whereas one based on geometry explains all peg-and-hole pairs of incompatible dimensions. Putnam argues in particular that the quantum-level explanation must consider the elementary particles (and hence the materials) of which the peg and hole are made. But the specific particles and materials are not (or should not be) relevant; only the relative dimensions of the peg and hole matter. Thus the geometrical explanation is superior. A complete examination of the peg-and-hole problem is beyond the scope of this paper, but here is a preliminary analysis.

When considering the peg-and-hole question, a fundamental issue concerns the language and concepts one should allow oneself to use. For example, at the quantum level, there is no such thing as a peg. So how can one even begin to approach the question? A peg can only be laboriously constructed by describing the elementary particles that make it up and the relationships among them. But if one does that and then makes an argument from geometry based on such a constructed peg and hole is one not really using the argument from geometry rather than the argument from quantum physics?

The fundamental relationship between levels of abstraction is the implementation relation. An argument that describes how a peg and hole may be implemented from quantum phenomena and then claims based on the geometry of the resulting peg and hole that one cannot be inserted into the other is really making an argument at the geometric level. The only role that the quantum level plays is to show that it is possible to implement pegs and holes from quantum phenomena.

On the other hand, if one does not construct a peg and a hole from quantum phenomena but simply shows that a particular configuration of atoms that we would describe as a peg and a hole cannot be manipulated so that they would fit the description that we would call having the peg inside the hole, then one must make that argument for every configuration of elementary particles that one wishes to cover. Even then, it isn’t clear how one could claim that one has said anything about pegs and holes in general or that one could even define the terms peg and hole.

One might approach the problem from a different direction. Since at the quantum level one can make use of spatial language, one can define peg and hole shapes of the appropriate dimensions. One could then argue that if these shapes are presumed not to be inter-penetrable, then the peg shape could not be positioned within the hole shape. One would then describe how such shapes could be filled with quantum material so that they become non-inter-penetrable, i.e., solids. But in doing so, isn’t one again showing how one could use the quantum level to implement pegs and holes and then making a geometrical argument?

It seems to me that any argument showing that a peg and hole of incompatible sizes cannot fit one within the other must be made at the geometrical level and that when one starts at the quantum level, one finds oneself describing how to implement the level of geometrical solids and then making the argument at the geomet-
tical level. Perhaps the problem is that one simply cannot talk about pegs and holes in any language other than at the level of geometrical abstractions.

Laughlin (2005) uses the term *protectorate* to refer to phenomena of this sort. The solid state of matter is one of his examples. A protectorate exhibits properties that simply don’t have a meaning at lower levels. The solid state of matter includes concepts—such as hardness, shear strength, torque, tensile strength, load bearing ability, etc.—that are meaningless at the level of elementary particle physics. Laughlin emphasizes that protectorates are new conceptual domains that obey new laws, in other words, levels of abstraction.

In these examples, constraints may be seen to be operating in two directions. First, the lower level system is constrained so that it implements some higher level abstractions. The Game of Life is constrained to behave like a Turing machine; water molecules are constrained to behave as a solid; or in the example cited earlier, granite is constrained to form the features of four American presidents.

Once those constraints are in place, the properties of the higher level objects constrain the implementing components. This second sort of constraint may be misleading if it suggests downward causation. There is no downward causation. But the properties and behaviors of the higher level object necessitate properties or behaviors of the lower level elements that implement them. As long as the lower level continues to implement the upper level the lower level is necessarily constrained by whatever constraints apply at the upper level.

The peg-and-hole and similar examples are frequently used to argue the functionalist position that multiply realizable properties are not reducible. After all if there are multiple realizations, to which one is the higher level property reducible? I believe that this argument misses the point. A realized level of abstraction, like a Game of Life Turing machine, exists at the abstraction level because it is independently specifiable—not because it is multiply realizable. A level of abstraction with only one implementation but with an independent specification would be just as real as one with multiple realizations.

This perspective turns the question of realizability around. In exploring what exists, the question is not whether any particular abstraction is multiply realizable. It is what new levels of abstraction can one implement given the currently existing levels of abstraction? Does it really matter, for example, whether the eye evolved once or multiple times? What really matters is that each time it evolved, it enabled its possessors to see. The fact (if it is a fact) that vision is more or less the same in each case is not important. What is important is that a vision capability was created, whether that happened once or many times.
9. Summary

The need to understand and describe complex systems led computer scientists to develop concepts that clarify issues beyond computer science. In particular, the notion of the level of abstraction and their implementation by pre-existing levels of abstraction explains how higher level laws of nature help govern a reductionist universe.

References


Abbott, Russ (2007) “Bits don’t have error bars,” Workshop on Engineering and Philosophy, October 2007. To be included in the selected papers from the conference. http://cs.calstatela.edu/wiki/images/1/1c/Bits_don’t_have_error_bars.doc


Kellar Autumn; Metin Sitti; Yiching A. Liang; Anne M. Peattie; Wendy R. Hansen; Simon Sponberg; Thomas W. Kenny; Ronald Fearing; Jacob N. Israelachvili; Robert J. Full. “Evidence for van der Waals adhesion in gecko setae.” Proceedings of the National Academy of Sciences of the USA 2002, 99, 12252-12256.


Leibniz, Gottfried Wilhelm (1686) *Discourse on Metaphysics*, Sections 5-6, as translated by Ariew and Garber [10, pp. 38-39].


All Internet accesses are as of April 3, 2008.

**(Sidebar) Mathematics, science, and engineering (including computer science)**

The notion of the level of abstraction clarifies some of the similarities and differences among mathematics, science, engineering, and computer science.

Mathematics is the study of the entities and operations defined on various levels of abstraction—whether or not those levels of abstraction are implemented. Mathematicians devise formal (or at least “rigorous”) specifications of levels of abstraction. They then study the consequences of those specifications—which in the case of Peano’s axioms is number theory.

Science is (a) the characterization of observed natural phenomena as levels of abstraction, i.e., the framing of observed phenomena as patterns, followed by (b) a determination of how those levels of abstraction are implemented by lower level mechanisms.

Engineering (including computer science) is the imagination and implementation of new levels of abstraction. The levels of abstraction that engineers and computer scientists implement are almost always defined informally—most real-world systems are too complex to specify formally. They are characterized as requirements, natural language descriptions of required functional and performance properties. Engineers and computer scientists implement systems that meet requirements.
Whereas engineers and computer scientists imagine and implement new levels of abstraction, scientists identify existing levels of abstraction and discover the mechanisms nature uses to implement them. In other words, science is the reverse engineering of nature.

Why did computer science rather than engineering develop the notion of level of abstraction? In (Abbott 2008) I discuss how computer scientists start from a well defined base level of abstraction—the bit and the logical operations defined on it—and build new levels of abstraction upwards from that base. Engineers work with physical objects implemented at multiple and arbitrary levels of abstraction. Since there is no engineering base level of abstraction, engineers construct mathematical models that approximate nature downwards as far as necessary to ensure that the systems they build have reliable physical foundations.

(Sidebar) Entities

The issue of what makes an entity is key to this entire discussion. A fundamental question is under what conditions will we recognize that an entity exists? A second question is what leads to the formation of entities? I plan to take these questions up in another paper. But briefly, here are my answers.

An entity is a persistent pattern of reduced entropy. It seems to me that it’s reduced entropy, i.e., increased internal organization, that we identify as being characteristic of an entity. (It’s an interesting question whether we recognize persistent patterns of increased entropy as entities. I can’t think of good examples of a persistent patterns of increased entropy that aren’t artificially bounded so that overall the entropy of the pattern along with whatever bounds it is reduced.)

How are entities created? There seem to be three ways, leading to three distinct types of entities.

1. Static entities. The new upper level elements exist in an energy well. In (2006 and 2008) I call this static emergence. Examples include atoms made from elementary particles, molecules made from atoms, solids made from atoms and molecules, etc. The constraints are simply the fundamental forces that hold components together to form larger structures. Phase transitions typically mark the imposition or removal of constraints of this sort.

2. Dynamic entities. Procedural processes tie elements together. In (2006 and 2008) I call this dynamic emergence. Examples include biological organisms and social groups. The processes of a social group, i.e., the ways in which the group elements behave and interact, causes the group to persist as a group. A social club, to take a very simple example, is held together by the fact that the members adhere to formal or informal agreements about how they will behave. They pay dues, and they attend meetings for example. So-
cial and biological groups have the interesting property that their components may change while the group as an entity persists. People may join and leave a club even though the club persists. One is a member of the club as long as one behaves according to the processes that define the club and that bind its members together—whoever those members are at any particular time. The same sort of analysis applies to animal groups like herds, colonies, etc. Similarly biological organisms gain and lose molecules while they persist as organisms. This property makes dynamic entities less amenable to analysis by supervenience than static entities. In another contrast to static entities (which require energy to tear them apart) dynamic entities require energy to keep themselves together since they cohere only so long as their members continue to behave as expected. And behavior requires energy. Consequently, dynamic entities must import energy from their environments to persist.

3. **Symbolic entities.** A symbolic framework provides the means to create new abstractions. Entities of this sort are (appropriately) called symbolic entities. (This is a class of entity that I haven’t mentioned before.) The obvious examples are the entities created within computational symbolic frameworks such as the Game of Life and computer programming languages. In symbolic frameworks, mechanisms are defined explicitly to support the creation of new abstractions. No special energy is required for any symbolic construction as long as the framework itself continues to exist. Presumably a similar mechanism enables us to conceptualize symbolic entities.

Entities in all three classes are distinguished by their entropy. They all have less entropy than their surroundings—and their components are more highly correlated with each other than with outside elements and than outside elements are with each other.
Appendix. Dennett’s “Real Patterns”

In “Real Patterns” Dennett (1991) uses the fact that a Turing Machine may be implemented in terms of Game of Life patterns to argue for his The Intentional Stance (1987) position regarding beliefs—which he calls mild realism.

It has been suggested that “Real Patterns” has a significant overlap with this paper. I disagree. But to explore that issue, this appendix examines “Real Patterns” in some depth. My primary goal is to describe (in Dennett’s own words as much as possible) the primary points made in “Real Patterns.”

The fundamental issue discussed in “Real Patterns” is the status of beliefs. Much of the paper draws connections among beliefs, patterns, and predictions. Here’s an extract which is represented as the paper’s core content. It appears (as of July 1, 2008) on the Tufts Cognitive Study website: http://ase.tufts.edu/cogstud/papers/realpatt.htm.

Are there really beliefs? Or are we learning (from neuroscience and psychology, presumably) that, strictly speaking, beliefs are figments of our imagination, items in a superseded ontology? Philosophers generally regard such ontological questions as admitting just two possible answers: either beliefs exist or they don’t. There is no such state as quasi-existence; there are no stable doctrines of semi-realism. Beliefs must either be vindicated along with the viruses or banished along with the banshees. A bracing conviction prevails, then, to the effect that when it comes to beliefs (and other mental items) one must be either a realist or an eliminative materialist.

Dennett suggests that one way to evaluate a belief is by looking at predictions they allow one to make. He writes (p. 30) that “the success of any prediction depends on there being some order or pattern in the world to exploit. What is the pattern of a pattern of?”

Thus, Dennett acknowledges—unsurprisingly—that there are regularities in the world, which he tends to call patterns. Dennett does not seem to be asking how those regularities come about or what they consist of. He seems more interested in the relationship between such regularities and how we think about them. Dennett continues (p. 30).

Some have thought, with Fodor, that the pattern of belief must in the end be a pattern of structures in the brain, formulae written in the language of thought. Where else could it be? Gibsonians might say the pattern is “in the light”—and Quinians (such as Donald Davidson and I) could almost agree: the pattern is discernible in agents’ (observable) behavior when we subject it to “radical interpretation” (Davidson) “from the intentional stance” (Dennett).

When are the elements of a pattern real and not merely apparent? Answering this question will help us resolve the misconceptions that have led to the proliferation of “ontological positions” about beliefs, the different grades or kinds of realism. I shall concentrate on five salient exemplars arrayed in the space of possibilities: Fodor’s industrial strength Realism …; Davidson’s regular strength realism; my mild realism; Richard Rorty’s milder-than-mild irrealism, according to which the pattern is only in the eyes of the beholders, and Paul Churchland’s eliminative materialism, which denies the reality of beliefs altogether.”
Earlier (p. 29) he writes,

I have claimed that beliefs are best considered to be abstract objects rather like centers of gravity. ... My aim [in this paper] is not so much to prove that my intermediate doctrine about the reality of psychological states is right, but just that it is quite possibly right ... ."

His concern, he says (p. 30), is

not in differences of opinion about the ultimate metaphysical status of physical things or abstract things (e.g., electrons or centers of gravity), but in differences of opinion about whether beliefs and other mental states are, shall we say, as real as electrons or centers of gravity. I want to show that mild realism is the doctrine that makes the most sense when what we are talking about are real patterns, such as the real patterns discernible from the intentional stance.

Although earlier Dennett acknowledged that there are regularities (patterns) in nature, it is not clear from the final sentence in the preceding whether Dennett is now claiming that at least some of these patterns become apparent only when one takes the intentional stance.

In contrast to what seems like philosophical infighting, my concern is not with the ontological status of beliefs but with what I claim are real features of nature—whether anyone has beliefs about them or not.

Dennett cites an article by Chaitin that discusses Chaitin’s measure of randomness and says that a pattern is real “if there is a description of the data that is more efficient than the bit map,” i.e., more concise than a literal replication of the primitive elements of which the pattern is composed.

Yet this discussion about patterns and their efficient representation seems to be diluted by Dennett’s more general acknowledgement (p. 36) that science is “widely acknowledged as the final arbiter of ontology. Science is supposed to carve nature at the joints—at the real joints, of course.”

If that is the case, then to determine what is real, ask a scientist—or at least an expert in the field—who presumably has a more efficient (or more insightful?) way of describing data than an exhaustive enumeration.

And that is more or less the position that Dennett takes. Dennett then refers (p. 41) to the fact that a Turing machine can be built using Game of Life patterns.

Since the universal Turing machine can compute any computable function, it can play chess—simply by mimicking the program of any chess-playing computer you like. ... Looking at the configuration of dots that accomplishes this marvel would almost certainly be unilluminating to anyone who had no clue that a configuration with such powers could exist. But from the perspective of one who had the hypothesis that this huge array of black dots was a chess-playing computer, enormously efficient ways of predicting the future of that configuration are made available. ... The scale of compression when one adopts the intentional stance toward the two-dimensional chess-playing computer galaxy is stupendous: it is the difference between figuring out in your head what white’s most likely (best) move is versus calculating the state of a few trillion pixels through a few hundred thousand generations. But the scale of the savings is really no greater in the Life world than in our own Predicting that someone
will duck if you throw a brick at him is easy from the folk-psychological stance; it is and will always be intractable if you have to trace the photons from brick to eyeball, the neurotransmitters from optic nerve to motor nerve, and so forth.

Dennett moves on from this observation to discuss Fodor’s position with respect to regularities and whether or not they must be mirrored in the brain. (p. 42)

For Fodor, … beliefs and their kin would not be real unless the pattern dimly discernible from the perspective of folk psychology could also be discerned (more clearly, with less noise) as a pattern of structures in the brain.

Dennett then returns (p. 43) to discussing regularities in the world. He claims that

Philosophers have tended to ignore a variety of regularity intermediate between the regularities of planets and other objects “obeying” the laws of physics and the regularities of rule-following (that is rule-consulting) systems. These intermediate regularities are those which are preserved under selection pressure: the regularities dictated by principles of good design and hence homed in on any self-designing systems. That is, a “rule of thought” may be much more than a mere regularity; it may be a wise rule, a rule one would design a system by if one were a system designer … . Such rules no more need to be explicitly represented than do the principles of aerodynamics that are honored in the design of birds’ wings.

It isn’t clear to me to which regularities Dennett is referring. Is he really saying that regularities that have been discovered by evolution (or those common to engineering or creative design) have been ignored by philosophers? It doesn’t seem to matter, though, because Dennett doesn’t discuss these regularities either.

Dennett then returns to relationships between individuals’ beliefs, the predictions they may make about the world, and the generally noisy patterns on which those beliefs and predictions are based. (p. 45)

Fodor takes beliefs to be things in the head—just like cells and blood vessels and viruses, … Churchland [with whom Dennett agrees on this point favors understanding beliefs as] indirect “measurements” of a reality diffused in the behavioral dispositions of the brain (and body). We think beliefs are real enough to call real just so long as belief talk measures these complex behavior-disposing organs as predictively as it does.

Much of the rest of the paper is devoted to arguing that two individuals may see two different patterns in the same data and that (p. 48) “such radical indeterminacy is a genuine and stable possibility.”

Dennett allows (p. 49) for the possibility of correctly deciding which of two such competing positions is “correct” by dropping “down from the intentional stance to the design or physical stances.” On the other hand, (p. 49) he says that there could be two different systems of belief attribution to an individual which differed substantially in what they attributed—even yielding substantially different predictions of the individual’s future behavior—and yet where no deeper fact of the matter could establish that one was a description of the individual’s real beliefs and the other not. In other words, there could be two different, but equally real, patterns discernible in the noisy world. The rival theorists would not even agree on which parts of the world were pattern and which were noise, and yet nothing deeper would settle the issue. The choice
of a pattern would indeed be up to the observer, a matter to be decided on idiosyncratic pragmatic grounds.

Dennett ends (p. 51) with the following.

A truly general-purpose, robust system of pattern description more valuable than the intentional stance is not an impossibility, but anyone who wants to bet on it might care to talk to me about the odds they will take.

What does all this show? Not that Fodor’s industrial-strength Realism must be false, and not that Churchland’s eliminative materialism must be false, but just that both views are gratuitously strong forms of materialism—presumptive theses way out in front of the empirical support they require. Rorty’s view errs in the opposite direction, ignoring the impressive empirical track record that distinguishes the intentional stance from the astrological stance. Davidson’s intermediate position, like mine, ties reality to the brute existence of pattern, but Davidson has overlooked the possibility of two or more conflicting patterns being superimposed on the same data—a more radical indeterminacy of translation than he had supposed possible. Now, once again, is the view that I am defending here a sort of instrumentalism or a sort of realism? I think that the view itself is clearer than either of these labels, so I shall leave that question to anyone who still finds illumination in them.

It seems clear from this discussion that although Dennett must approach some of the issues that arise when exploring questions of emergence and that he makes use of the fact that it is possible to emulate a Turing Machine by using Game of Life patterns, the focus of “Real patterns” is to offer guidance to philosophers regarding how his views about beliefs are positioned relative to those of other philosophers and not to discuss issues of emergence, levels of abstraction, or the reality of higher level entities.

Reference
