Turing and Intelligence

by C. S. Schroeder

The following is the collection of short papers originally posted online at atheoryof.com in 2007. They regard the problem of modeling intelligent agency, specifically addressing Turing, and his surprisingly limited discussion of the matter. I find inspiration in his discussion of recognition, in his initial work in Computing, however, and argue that a more general concept of recognition could be most important to understanding intelligence generally, whether or not intelligence can be properly automated. The papers are presented here with all limitations, without modification, except possibly regarding formatting .

Turing and Intelligence

By *C. S. Schroeder*

If one is going to challenge a well-entrenched theory, it does well to go to the roots. And on the topic of intelligence, those roots are in the work of Alan M. Turing. Turing is most well known for his fundamental work on the nature of computation, but he is also known as a father of classic AI. In this article, I want to explore the insights in two of his most famous works, "On Computable Numbers"[1] and "Computing Machinery and Intelligence"[2], and show that assumptions made in the former are smuggled into the latter, when they ought not be.

Turing and His Machine

The Turing Machine is Turing's most lasting legacy. On the basis of this machine, one can define a class of functions, the *Turing Computable* functions – and because of the intuitive plausibility of the Turing Machine as a minimal model of *human computers*, a convincing case can be made that these Turing Computable functions are the class of *all computable functions*. The class of Turing Computable functions is equivalent to a number of other classes differently defined (i.e. Church's lamda-definable class), and some of these were claimed to also characterize the class of all computable functions before Turing's ideas were published[3]; but none of these other definitions were so *intuitively convincing* – since none of them presented an actual machine which modeled the human computer, i.e. a basic model of the *computer*.

Turing's model of the computer is astoundingly simple. First, there is a tape. This tape extends in both directions as far as we need it to go (i.e. we can always add more). On this tape are squares. In the squares are 0's, 1's, or nothing. The *computer* consists of a head, which reads from the square that it is over. Depending on the "state" of the computer and depending on the value in the square it reads, the computer will do something. What the computer can do is 1) erase the current symbol and optionally write a 0 or 1, together with optionally changing its state; or it can 2) move left or right along the tape, one square, and optionally change its state. The relations between observed symbols, states, and actions could be specified in a finite table (i.e. a *program*), which consisted of columns *symbol, state,* and *action*, where the last would contain a specific example from 1 or 2. The symbols would be 0, 1, and blank, naturally; while the *states* can be labeled simply qn, n replaced with some number (as in q4544), but where, q1 and q0 are distinguished as *start* and *stop* states visited at most once[4].

The input to the machine over some run, can simply be considered the sequence of 0's and 1's to, e.g. the immediate left of the head when it is in q1 (starts); and the

---

[1] In it's entirety, "On Computable Numbers, with an Application to the Entscheidungs Problem"

[2] The former available in "Collected Works of A.M. Turing: Mathematical Logic" and the latter in "Collected Works of A.M. Turing: Mechanical Intelligence"

[3] i.e. Church's, again. It should be noted, however, that Turing actually framed the question as one of what *numbers* are computable – i.e. which numbers can have their decimal expansions computed – but it is standard to frame the importance of Turing's insights as regarding computable functions, especially in a comparative context.

[4] This is a more or less standard presentation of the Turing machine, similar to Sipser's in *Introduction to the Theory of Computation*, 1997, and not Turing's itself.

output of some run can be considered the sequence of 0s and 1s to e.g. the immediate right of the head when it is in q0 (stops). One can see, then, how these machines can be understood to compute functions, simply reading these sequences as numbers in binary notation. But it must be noted that there is no guarantee that given some Turing machine, it will necessarily enter q0 given some input at q1. All the *computable* functions will have a Turing machine which stops on every such input. But a given machine may not represent such a function.[5]

Turing and Intuition

The place of intuition in Turing's argument that his machine aptly characterizes the general concept of a *computer* cannot be overstated. Turing in fact explicitly says that one of his primary arguments for it is "a direct appeal to intuition". In this direct appeal, Turing reflects on the *essential characteristics* of a human computer, qua computer; and ends up, more or less, with his Turing machine model.

To start, he argues that there is nothing essential to the typical two-dimensional piece of paper that calculations are commonly performed on when done manually, so that, rather, we can view the external memory as a one dimensional tape (an assumption that was later formally proved correct given the rest of the framework). He then moves to argue for a number of *finiteness* constraints. In particular the finiteness of

a) symbols
b) observed squares
c) states ("of mind")

a) refers to the printable symbols, and is assumed because otherwise "there would be symbols differing to an arbitrarily small extent"; which presumably shouldn't be allowed, since otherwise we'd have to rely on the human computer to make arbitrarily fine distinctions; furthermore, and as regards to (c), the finiteness of the states of mind that the computer can possibly enter into, he says,

> "If we admitted an infinity of states of mind, some of them will be "arbitrarily close" and will be confused. Again, the restriction is not one which seriously affects computation, since the use of more complicated states of mind can be avoided by writing more symbols on the tape" (p.250)

It is important to understand that Turing is here firmly focused on the issue of computation, and for a human computer to carry out a calculation mechanically, they should not be relying on subtle differences in their states of mind, which they may confuse. But despite these limitations, we do not lose any of the expressiveness of language, as this can be made up for by stringing together basic symbols; and despite (b), that many of these strings "cannot be observed in one glance", we do not loose any power of *computation*, since human computers themselves cannot *rely on* such recognition "at a glance" for arbitrarily long strings of symbols.

Having established the finiteness of the ontology, he tries to analyze the *actions* of the computer. He seeks to reduce these to the most basic actions possible, or what he

[5] All the *partially* computable functions will have a Turing machine that stops for all the inputs for which it is defined. Note also that finite decimals can be inputs or outputs with the restriction that the decimal be separated (e.g. by a blank) from the non-fractional part.

calls the "simple operations". He finds two: change of symbol and shift in observed squares, which when combined with changes in state of mind, form (quoted from p. 251):

A) A possible change of symbol together with a possible change of state of mind.
B) A possible change of observed squares, together with a possible change of state of mind.

Regarding A, we can assume that not more than one symbol is changed, since multiple changes can be done in sequence rather than all at once; and we can assume that this changed symbol is an "observed" symbol, "without loss of generality". Regarding B, the distance with which the observed squares can shift along the tape (i. e. the head move) must not be greater than some fixed distance, L, according with the limits of the human computer shifting their gaze.

I have moved rather quickly in this section, but the fundamental point is that these basic characteristics of the *human computer* are the basic characteristics of the *Turing Machine*[6]*,* and is crucial to Turing's case that the Turing computable functions are the computable functions.

Turing and Recognition

Crucial to understanding my critique of Turing on the topic of intelligence (in the following sections) is understanding Turing's "theory" of recognition as presented in "On Computable Numbers". I here isolate five primary features of recognitions, as Turing understands them there:

[6] The Turing Machine is a further simplification, but to no essential loss of *computing* power.

Recognition is *immediate:* Turing says, in his words, that there are certain "squares" which are "immediately recognisable". As I understand this, he is saying that at any given time there are certain squares that the computer is aware of, which may contain symbols, and when they do, these symbols are immediately recognized, which is to say that this process of immediate recognition *is not composed of simpler sub-processes*.

Recognition is *simple:* For Turing, each recognition is *of some one symbol in our finite alphabet* – that is, the given markings aren't recognized as both some x and some y in our alphabet, in the way that you may recognize your father as both a father and a man.

Recognition is *bounded:* This is to say that there is for Turing an event that is the recognition of a symbol that happens at a bounded location in space and time; it consists of a symbol with some specific continuous bounded space-time location, bearing a certain informing relationship to the computer, which is itself at a continuous bounded space-time location. The informing relationship itself is presumably a causal process occurring in space and time, thereby forming the link between the two space-time regions and creating one continuous bounded region.

Recognition is *restricted:* This is to say that a symbol that is recognized must reside entirely within one square. There is no recognition of "words" composed of markings in more than one square, except in an extended sense of "recognition".

Recognition is *binary-valued:* Recognition is – for his machine – an all or nothing affair. There are not grades of recognition. Given also the first feature (being simple), we can consider the recognition at some time and at some "immediately recognisable" square, to be sparsely populated vector, where each location in the vector represents some symbol in our finite alphabet, and one of these locations has value "1", with the rest "0".

Turing's assumption at the outset is that the notion of recognition that should be allowed in the process of computation has the above features. And his example from the recognition of numbers does well to support his point with respect to computation. It seems clear that for the human computer to do their job and "get it right", they have to, for instance, do digit-wise comparisons of, e.g. 11011010101111010110100001010 and 11011010101111010110100001010, and cannot simply recognize them as equivalent at a glance. What is at issue here, however, is the significance of this model to *intelligent agency*. AI seeks to apply computational systems as models of intelligent agency – doing so however, assumes that, at root, *intelligence* starts with immediate, simple, bounded, restricted and binary-valued recognitions. What reason do we have to assume that it starts with these?

Of course, the proponent of AI would say that "recognition" as we commonly understand it is a combinatorial property that emerges from many applications of such Turing-recognitions… This is, I agree, an *assumption* of classic AI, but I don't think it is one we should make in trying to understand what it is to be an *intelligent agent*. Before we can come to consider the proper course in analyzing the notion of intelligent agency, we do well to consider the course of analysis first set by Turing in "Computing Machinery and Intelligence".

Turing and His Test

In "Computing Machinery and Intelligence", Turing addresses the question "Can Machines Think?" He finds, however, that addressing this question is hopeless in the abstract, so he "refines" the inquiry to instead ask whether a computer could perhaps pass a certain test – what we will call the Turing Test. Turing's test can be refined to the

following scenario: you have a person, the tester, who sits in front of a monitor; on the other side of a wall from the tester are two cubicles A and B; in one is a person and in the other a computer. The tester can pose questions to either cubicle A or cubicle B, and the person or the computer – whichever happens to be in that cubicle - will answer these questions (or not) as they can, with a response printed to the tester's monitor. The goal of the tester is determine which cubicle houses the computer and which the person, based on these responses. If, over many iterations with different testers, the computer is generally indistinguishable from the person, we say that the computer "passes the test", and should have "thought" attributed to it, if anything should.

Turing typically faces two types of objections to his position. First, there are objections that machines just can't think; and second, there are objections that if a machine passes his test, it does not follow that the machine can think. For my part, x passing the Turing test would be good evidence that x *can* think (even if not tantamount to it) and would in itself be remarkable feat of engineering. Moreover, I don't think that we should rule out, *a priori,* the possibility that there is (could be) a thing that thinks that *at some level of description* can be understood as a machine in Turing's sense of mechanism - and I think that Turing does a good job of replying to such objections. On the other hand, what *is* astounding to me is the altogether different tack that Turing takes in addressing the question of thinking machines from the one he took in addressing the question of computation. In particular, whereas so much of the support for the Turing machine being an apt model of the computer was *intuitive reflection on what it was to be a human computer, qua computer*, intuitive support for an apt model of a *thinker* in the form of reflection on what it is to be a human thinker, qua thinker, was cast aside entirely for *empirical evidence* – namely, the ability to pass the Turing test.

One could, of course, imagine Turing's article being written entirely differently. Suppose that Turing actually had some model of a thinker. He could have proceeded to present that model. After showing us something of what that model could do and how it was to be handled, he could have then shifted to providing intuitive support for that model. By reflection on what is essential to human thinkers, qua thinkers, he may have even convinced some of us as to the aptness of his model as, e.g. a minimal model of what it is to be a *thinker*. Of course, Turing had no such model, so he could not write this article in the fashion of "On Computable Numbers", but suppose someone *had* a model and could make such a case for it – wouldn't *that* be equal, if not better, support for that model of the *thinker*, in contrast to passing the Turing test?

It can perhaps be argued that what may have held Turing back from coming to an intuitively satisfying model of intelligence was his passion for *mechanizing* any such model. Perhaps, even if at some level of description such a thinker could be understood mechanically, we should not restrict our thinking to the mechanistic model when trying to uncover what is surely more abstract. Coming back to where we ended the previous section, we can ask, why should we be forced to look within the basic, limited framework of Turing-recognitions for a model of intelligence? Remember, Turing did not argue that *all* basic recognitions must be as described – just the ones relevant to a fundamental model of computation. But nonetheless, the primacy of such recognitions to *thinking* is smuggled in with the restriction to the mechanistic model. Suppose that reflecting on the general nature of recognition, we find intuitive support for alternative models? Should these be cast aside if they may not clearly fit within the mechanistic model? Perhaps it is

such reflection that can lead us to the model of the thinker, in the way that Turing was lead to the model of the computer. And perhaps only with such a model in hand, can we come back to answer the question: *can machines think?*

Recognitions and Intelligence

If we allow ourselves to reflect on the nature of recognition in general, we realize that it is clearly not *binary-valued.* There are certainly cases where we recognize things to *some degree*, and it seems we should not limit ourselves to such binary values when modeling *intelligence.* Also, if recognition is necessarily *simple*, certainly more of an argument for that needs to be made; it certainly seems I can recognize my copy of "Mechanical Intelligence" on the desk, as well as recognize a "red book" on my desk, even though they are *at the same location*. Do we have any reason at all to assume *a priori* that these recognitions are reducible to Turing-recognitions? I don't see any reason to make such a claim.

It is important to note that one of the major failings of intelligence computing is in the area of *pattern recognition*. Researchers have tried repeatedly to get the computer to recognize faces and other ordinary things, but only with very limited success. It is perhaps better to *assume* certain basic pattern recognition abilities in our model at the start, for the construction of intelligent systems, and after such a model of intelligence is justified by "a direct appeal to intuition" we can move from there to finding the technology to support the implementation of such a model. In particular, I think the property of being *restricted* needs to be reconsidered as well.

The property of being restricted is directly defined in terms of Turing's *squares* on the tape, but we might understand this to mean that what gets recognized for Turing e.g. the symbol, is *continuous* and *bounded*. If we are to recognize patterns, however, we may care to lift this restriction. And on reflection, it seems clear to me that I can recognize patterns in experience which are not continuous, e.g. the pattern of a row of parked cars, the pattern of windows on the wall, the pattern of stars in the autumn night sky, the printed italics word *here*. (Furthermore, the property of being *bounded* may be questioned as well, though I will save that analysis for another time.)

Of course, how such an understanding of recognition could be fleshed out and then *used* to understand intelligence in general is a much larger task than can be achieved here; but the most important point that can be made here is that researchers of intelligence should not be forced to reside within the computational model, when we know that model has certain restrictions that we have no reason to believe hold of ourselves, as thinking things. In fairness to Turing, he does not say that we *must* work within this model when addressing intelligence; he is more concerned with the adequacy of computational models for implementing intelligence. Nonetheless, it is the computational model which has dominated thinking about intelligence, and I think that reflecting on Turing's method in "On Computable Numbers" we can see holes in the justification for this project – namely that that by a "direct appeal to intuition" on what it is to be a human thinker, qua thinker, we can see that we should not be restricted to primitive Turing-recognitions.

In my next essay, "Recognition and Computation", I will further analyze arguments in favor of the restriction of models of intelligence to computational models, and present an example of a hypothetical non-computational process, which may for all

we know be a *natural* process, and which may factor into a non-computational understanding of recognition.

Recognition and Computation[7]

By *C. S. Schroeder*

In my previous article I reflected on Turing's insights into computation. I then argued that restricting our understanding of intelligence to the computational model smuggles in, as fundamental, *Turing-recognitions*, i.e. immediate, simple, bounded, granular, continuous, and binary-valued recognitions, where these properties are defined as follows:

Immediate: in the sense that the process of recognition that produces these recognitions is not composed of simpler sub-processes;

Simple: in the sense that what is recognized over some region is some one symbol among a finite alphabet;

Bounded: in the sense that the recognition and what is recognized jointly occur in a bounded space-time region;

Granular: (previously called "Restricted") in the sense that symbols from the alphabet are what get recognized, and not compositions of such symbols, i.e. no symbol is a proper part of another symbol;

Continuous: this property, not mentioned in the previous article, actually does not appear to be part of Turing's reflective analysis, in that he seems to leave open the possibility that a "symbol" could extend over a discontinuous part of the observed region (multiple discontinuous squares) at any time[8]. Nonetheless, it is a property of the recognitions - that there are no such "symbols" - in all accounts of Turing Machines, so we state it here as well.

Binary-valued: in that a particular symbol is either recognized or it is not.[9]

I want to maintain here that we should not restrict our understanding of intelligence to such recognitions. This freedom from Turing-recognitions comes in two parts. First, we should not restrict our thinking about intelligence to the concept of Turing-recognitions, because recognition, as we intuitively understand it, is not restricted to such recognitions; and a more general notion of recognition can better be used to understand intelligence. Second, we should not assume that at root the processes supporting these non-Turing-recognitions are "really" computational – since we can alternatively accept a basic notion of *similarity* (or *dissimilarity*) to understand the process by which such recognitions occur, which may be non-computational. Before moving to the positive arguments for

---

[7]

[8] An example could be a phrase, with squares in between the words which get "ignored" and whose contents are not properly part of the "symbol" which is here the phrase.

[9] This amounts to a clearer, more refined expression of the definitions given in the last article, and should be substituted for those definitions everywhere.

the place of non-Turing recognitions in understanding intelligence, however, we can start by deflecting some objections to such a non-computational viewpoint.

Restriction to Computational Models

First of all, it does well to note that it is not necessarily the case that *everything* is a computational process. You will hear the occasional professor proclaim that everything is a Turing Machine at a fine enough grain,[10] but Turing himself would have never suggested such a thing, since he thought the world, at root, was *continuous*[11]. The question of whether everything, at root, can be understood in computational terms rests on more than just whether the world is continuous or not. But if the world is continuous, it is not properly a computational process, since Turing machines simply do not have continuous inputs or outputs. And it remains an open question whether the world is continuous, so we can't say it *is* computational.

Putting the grand assumption - that everything is a computational process - to the side, we might still ask whether *intelligence* is necessarily computational. Of course, the question of whether intelligence is computational has two sides, as indicated in the introduction:

(a) Must we *understand* intelligence in computational terms?
(b) Is any process which has intelligence *at root* a computational system?

It should be clear that these are separable; by analogy, it may be that geology can be reduced to quantum physics, yet we do not demand that we understand geology in these terms, as that would be practically impossible.

To Daniel Dennett's mind, in *Brainstorms*, the answer is yes, at least to the first question. We start with a quote from Dennett in his chapter "Why the Law of Effect Will Not Go Away"

"the supposition that there might be a non-question-begging non-mechanistic psychology gets you nothing, unless accompanied by the assumption that Church's Thesis is false." (*Brainstorms*, p. 83)

Dennett here seeks to belittle the possibility of a "non-mechanistic" psychology by maintaining it conflicts with the heart of the theory of computability. Remember, Church's Thesis – a.k.a. the Church-Turing Thesis – is just the position discussed in the previous article: that a function is *computable* if and only if it is computable by a Turing Machine[12]; which is hardly subject to doubt in most minds. A "non-question-begging psychology" is just one which "makes no ultimate appeals to unexplained intelligence". So Dennett claims that if you want to maintain a psychological theory which makes no appeal to unexplained intelligence and is non-mechanistic, you have to believe that there

[10] In particular, and ironically, John Searle, a staunch critic of AI (Chinese Room), has said that everything at root is a Turing Machine – though this can be taken as a hedge on a larger wager.
[11] See "All machinery can be regarded as continuous, but when it is possible to regard it as discrete it is usually best to do so," in "Intelligent Machinery" (p. 5), and elsewhere.
[12] Or equivalently, expressible in the Lambda Calculus – as was originally stated by Church, giving him the foremost title to the thesis.

are computable functions which are not Turing computable. Of course, it stands to wonder here what Dennett means by "mechanistic".

It seems Dennett means by a "mechanistic" process, just one which is "Turing-computational"; yet to be non-question-begging a theory has to make use of only *computable* functions, he assumes; in which case, to be non-mechanistic (non-Turing-computational) and non-question-begging (computational), there would have to be computable functions which were not Turing-computable. But we can take issue with this argument at the assumption that to be non-question-begging, we can only make use of computable functions. Perhaps intelligence *is* best understood by a non-computational model (i.e. best modeled by non-computable functions); and perhaps, even at its roots, intelligence *is* a non-computational process. There would be nothing question-begging in maintaining this, provided the functions we use to model intelligence are mathematically comprehensible; and surely there are many continuous functions which are.

To be fair to Dennett, he is trying to attack a certain kind of non-computational understanding of intelligence. Dennett views psychological theory, from a functionalist perspective, as breaking down the agent into distinct smaller agents which carry out distinct purposes, and he is concerned that such a theory would be question-begging if it relied on the intelligence of the distinct, smaller agents, without a total reduction of their intelligence in some way. Dennett believes the only way to achieve that (presumably) is by getting rid of "intentional" language altogether (talk of beliefs, desires, purposes, etc.) in the specification of our theory, through an ultimate reduction of such language to a machine language. Within this framework, Dennett is right to stress the need to not beg the question, but a non-computational understanding of intelligence is perfectly possible even within this framework – i.e. if the distinct, smaller agents are explicable in a mathematical model which uses non-computable functions, e.g. continuous functions.

Dennett has clearly given us no reason to believe that the answer to question 2 above is *yes*. At root, perhaps intelligent processes in some cases are non-computational systems, as Turing would presume. Furthermore, Dennett has failed in his attempt to show that we must understand intelligence in computational terms. He *has* shown that one way to avoid begging the question is to reduce our theory to a compiled program; but he has not shown that this is the *only* way to avoid begging the question; and it is this latter claim that needs to be made if we are going to say we *must* understand intelligence in computational terms. Moreover, Dennett has given us no reason to believe that the promise of a computational psychology will be fulfilled. In what follows, I wish to suggest an alternative research program.

Pattern Recognition

In Douglas Hofstadter's behemoth, *Godel, Escher, Bach*, it is remarkable that when he comes to saying something constructive (rather than merely suggestive) about the nature of intelligence he points in the direction of *pattern recognition* (p. 647-676). This is remarkable in that he seems correct in doing so and yet the area of pattern recognition is one of the great failures of AI. As one researcher has written, stating the platitude of disappointment: "Computers are notoriously horrible at the kind of pattern recognition that comes so naturally to people"[13]. Tracing backwards to the root of the problem, I think we see that the culprit is the restriction to Turing-recognitions at the start, which

[13] Alan Gilchirst, *Scientific American Mind* June/July 2006, p.44

any computational understanding of intelligence is stuck with. As mentioned in the introduction, there are two reasons we should not restrict our understanding of intelligence to Turing-recognitions. The first is that a more general notion of recognition is intuitively available; and the second is that the more general notion of recognition is quite useful. To elaborate the more general notion, we will address the five properties of Turing-recognitions in turn.

Immediate: The property of being *immediate* is properly understood as a relative property. Recognition may be immediate relative to our understanding of the agent as a whole, in that we take it as primitive within our theory of intelligence; but within an implementation of intelligence, there will be sub-processes to recognition which nonetheless "make it up". Of course, within a computer, a Turing-recognition's sub-processes are so basic that they can be effectively ignored in our understanding of the computer, whereas in the more general notion of recognition, they are less basic. Nonetheless, in either case, sub-processes exist, and whether the recognitions are "immediate" will depend on what we choose to ignore. I want to say that when tackling the first issue, that of *understanding* intelligence, we can effectively ignore the sub-processes involved in recognition; just as we can effectively ignore the sub-processes involved in Turing-recognitions when trying to understand *computation*. And in this sense, even the more general recognitions can be *immediate*[14].[15]

Simple: The property of being *simple* has no place within the more general notion of recognition. First, it is clear that something, e.g. a collection of markings on a page, can be recognized as more than one thing among the many things that it could be recognized as, at any one time; just as you can recognize, e.g. in a picture, you and your Dad, and a father and son. Second, we have no reason to limit possible recognitions to a finite alphabet; intuitively, there is no limit to the degree to which we can refine our distinctions, so the recognitions that are possible are infinite and beyond.

Bounded: the property of being *bounded* would seem to have a clear place within any understanding of recognition, since our *experiences* are bounded in there spatial and temporal dimensions – and our experiences, essentially being the markings on the tape (the grounds for recognitions and determining *where* we recognize things), have a clear connection to the question of whether recognitions are bounded. There are many things to consider before passing judgment on the boundedness of recognitions, but we can accept it tentatively within our initial model, if only for simplicity.

Granular: this assumption effectively states, in the case of Turing-recognitions, that the markings which can be recognized as a symbol do not form proper parts of markings which could be recognized as another symbol. Of course, in the more general notion of recognition, we want to allow that the grounds for one recognition may be part of the

[14] Though an important amendment to this claim will have to be made at a later date when we tackle the topic of abstract recognitions.

[15] Though note that this provides no indication of what the processes underlying the general notion are; i.e. they needn't be computational processes.

grounds for others, e.g. a phrase can be recognized just as well as the words, although the markings that make them up do overlap.

Continuous: this assumption is perhaps the most important to see your way past. It is quite common to think that what gets recognized is continuous, but there are clearly patterns which can be recognized which are not, e.g. planes flying in formation, constellations, etc. In fact, I would go so far as to say that discontinuous regions of any form can be the location of a recognition, provided they fit within the "window of experience" (boundedness). Loosening this constraint opens the door for true *pattern* recognition, as opposed to *object* recognition.

Binary-valued: clearly, we can have degrees of recognition; and though we are to strive for binary-valued recognitions in the process of computing functions, we have no reason to ignore such recognitions in the general model of intelligence. We will come to understand that degrees of recognition are closely related to degrees of belief, and the latter cannot be dispensed with without an elegant reduction.

Of course, the computationalist would seek to reduce the general notion of recognition to computation, but I am skeptical that what is lost in the idealization to Turing-recognitions can be made up by the complexity of machine states. This, of course, is part of classic AI's lingering promise, but I should like to think that the proper way to *understand* intelligence is by using the general notion of recognition; and if the computationalist perspective can't "keep up" so to provide the basis for engineering the objects of our theory, so much the worse for that perspective.

On the other hand, we can bolster the importance of this alternative framework if we can at least suggest a novel implementation of agents with the capacity for non-Turing-recognitions, outside of the computational perspective. In the articles and booklets which follow, I will be developing a general theory of intelligent agency based on such recognitions and a few other general notions. In the next section, however, I want to construct a theoretical bridge from the general understanding of non-Turing-recognitions, which will be used in that theory, to a scheme for their possible implementation outside of the computational perspective.

A Theoretical Bridge

Recognition, of course, has its roots in *similarity*. Generally speaking, we recognize something as something in virtue of the similarity it has to the other things in that category. Unfortunately, from the computational perspective, the success or failure of a pattern recognition system in this or that domain is very sensitive to the measure used to compute similarity. But if we are willing to consider stepping outside of the computational perspective, perhaps we would do better to take the notion of similarity, or better, *degree of similarity* as a primitive notion.

In taking this notion as primitive, I intend to accomplish two things. First, I intend to *use* this notion in our understanding of non-Turing recognitions, and thereby, in our theory of intelligence. This idea is not entirely new, of course, since empiricist philosophers dating back at least to Hume, have been calling concepts "faint copies of sensation", highlighting the idea that concepts are applied to experience, in a recognition,

based on a certain similarity to it. The second thing that is intended, however, is that the basic property of similarity serve as a bridge from the general, intuitive understanding of recognition to its implementation in a physical system, which almost certainly escaped the empiricists.

The fundamental idea of a pattern matching system based on similarity is this: there is a field of values, Q, and a collection of entities, C, such that a member of C *moves to, is applied to,* or *covers* a part of Q in virtue of a similarity between that member of C and that part of Q. That is to say, it is a basic property of our system that the member of C is *attracted to* those parts of Q to which it is similar, and in virtue of that attraction, moves to, is applied to, or covers that part of Q. Intuitively, the members of C are *concepts* and the field Q is experience, e.g. our visual experience at some time. The idea, then, is that a member of C, e.g. [Marylyn Monroe] (brackets meaning the concept), is a certain pattern of visual experiences, and gets applied to some part of Q, our visual experience, when sufficiently similar to it, e.g. when viewing a certain Andy Warhol painting, and thereby forms a *recognition*.

The full significance of such matching to intelligence, as I've indicated, will be spelled out at a later time. For now, we are concerned to move, to some extent, from psychological talk to talk of a physical system. It must be said that I do not *know* of any natural system which exhibits the properties described above; instead, I will simply describe a *hypothetical* system, which may, of course be a *real* system yet unknown. To start to do this, it is best to move away from such a high level understanding of the entities in C, as ordinary concepts. Instead, we will hypothesize that the members of C are unit-patterns.

A unit-pattern-generator (which can be considered a concept) generates unit-patterns. A unit-pattern is of course a *pattern* of some actual length and width, though to be thought of as quite small. In an application, a unit pattern gets generated, and then is attracted to some portion of a field based on similarity to it. Roughly, one side of these patterns is multi-colored. This side "matches up" with some portion of the field and moves to where there is such a match. This unit-pattern does not block the qualities it matches to, so other patterns can be applied to the underlying pattern as well. You can think of simply, a flat panel television screen facing up from the floor with some static picture; these patterns are sprinkled over the picture and fall into place according to their similarity to the pixel-patterns on the screen.

Of course, this matching might be looked on as "magic" to some engineers, but the basic idea is that *if similarity is a basic natural property then there is no reason to believe that matchings between similar things cannot simply, naturally occur*. Of course, similarity may only be a property that makes sense in certain domains, but if it ever does hold naturally, there is no reason rule out *something happening naturally as a function of similarity*. And finally, it should be clear that although the structure of Q and members of C will have something to say about whether this process can be simulated by a computer, even if the field Q and entities C have discrete structures, the matching process is of very high complexity, such that for a system to be practical – as we humans are – a natural matching process would be welcome which overcomes these bounds.

Human Computing

One way of viewing the mismatch between intelligence and computation is as a call for new physical systems that can implement intelligence, as addressed in the previous section. An alternative, however, is to view the mismatch between intelligence and computation as calling for computational systems which support and use *our* intelligence. The idea of "Human Computing" is one such idea in line with this alternative.

Luis von Ahn, who received a MacArthur Grant last year for his work on Human Computing, is the foremost researcher in this field[16]. Effectively, Human Computing is a means of solving large-scale problems by harvesting the intelligence of people through collecting information regarding their responses ("plays") in a game. In one such game, a two player game called *ESP,* the participants are shown an image and have to label it in the same way as the other player to get points (without any contact with that player other than through responses). The details of the game are less important than the basic point here: that one can more reliably categorize digital images by letting the people recognize the image than trying to infer it from, e.g. the linguistic context it is embedded together with image recognition algorithms; but still, the computer has a role in fixing the rules of the game and collecting the data.

Of course, once we simply face up to the idea that our computational infrastructure *is not going to cut us out of the loop*, then can start to build programs which truly support, enhance, and use our intelligence, rather than trying to replace it. Of course, by this I mean more than a UI design with a series of well-structured menus. We need processes for visualization of data which support our pattern-recognition abilities; we need interaction with visualized data so that it can morph appropriately at our suggestion; we need systems that can quickly *verify* patterns as holding on a global scale, when we recognize them on a local scale; and we need to be able to use or communicate such recognized and verified patterns effectively.

All of this is lost on the lazy, who presume the computer will eventually do their thinking for them; but it should not be lost on us. There are many great strides to be taken in "Human Computing" – now more broadly construed – which could well define the role of the human vs. the role of the computer, and thereby help the day to day keyboard laborer understand their role or perhaps, when it does not involve pattern recognition, better understand how to automate it.

Conclusion

The purpose of this article has largely been to help the reader overcome a conceptual bind to the computational perspective. I have argued that we can free ourselves from this bind without contradiction; analyzed the more general notion of non-Turing recognitions and suggested we start by using it in our analysis of intelligence; suggested a path toward novel hardware which could fill a role that computation may not be capable of filling; and suggested we rethink the relationship between human and computer, when it comes to the topic of intelligence. All of this, however, only takes us so far. What is needed now is the hearty explanation of how we may use non-Turing recognitions to understand intelligence.

---

[16] See http://www.cs.cmu.edu/~biglou/research.html for Von Ahn's research, including the very readable "Games With a Purpose".

To most people, there are a great many processes involved in intelligence, of which recognition is only a part. To me, recognition is one of only a few centrally important ideas in understanding intelligence. To get a feel for the theory that will be presented in the following publications, I will have to give some indication of how it is that recognition is connected to *action*. Without action, after all, there would be no intelligent *agency*. Remarkably, the theory that I will present can be introduced with a quote from a book, *On Intelligence*, whose topic is more precisely the *brain*.

"Move your arm in front of your face. To predict seeing your arm, your cortex has to know that it has commanded the arm to move. If the cortex saw your arm moving without the corresponding motor command, you would be surprised. The simplest way to interpret this would be to assume your brain first moves the arm and then predicts what it will see. I believe this is wrong. Instead I believe the cortex predicts seeing the arm, and this prediction is what causes the motor commands to make the prediction come true. You think first, which causes you to act to make your thoughts come true."(p.102)

This passage frames the issue in terms of "prediction", but we will see in the theory to come that prediction is just *recognition* which extends into the future. Your hand moves because it is part of a recognized pattern that extends into the future; and the foremost law is that when the pattern involves movement, and the recognition is strong enough, then *you move*.

In the next article I will elaborate on the basic idea of *the pattern-recognition agent*. It will provide us the opportunity to reflect on research in machine learning and its role in a theory of intelligence, which Turing himself – despite his central stance in "Computing Machinery and Intelligence" – had strong intuitions about.

Recognition and Intelligence[17]

By *Casey Schroeder*

In the previous article I elaborated on the reasons we should not restrict our thinking about intelligence to the computational perspective. I argued for this position from the fact that recognition is a general phenomenon which from the computational perspective is idealized and minimized to the point of possibly excluding a correct understanding of intelligence. I here want to provide the beginnings of an understanding of how the more general notion of recognition can start to provide us with a general theory of intelligence. I will do that by presenting the basic components of *the pattern recognition agent*. To properly understand that agent-model as an alternative to what is currently offered in AI, I will start by presenting a standard account of intelligent agency.

Standard Account

According to one popular book in AI[18], an *agent* is absolutely anything which

"can be viewed as perceiving its **environment** through **sensors** and acting upon that environment through **actuators**." (p. 32).

An agent then essentially has sensors and actuators and instantiates a mapping from sequences of perceptions to actions, by means of these sensor and actuator portals. Of course, by the definition, this mapping may simply be random, but for an agent to be considered intelligent, at least this mapping must be somehow "good". This goodness is embodied by the degree to which the agent approximates a *rational* agent. And a *rational agent* (according to R&N) is any agent that always

"select(s) an action that is expected to maximize its performance measure, given the evidence provided by the percept sequence and whatever built-in knowledge the agent has." (p. 36).

I take it here that "rational" is to be understood as "relatively ideally intelligent" – that is, the mapping is ideally intelligent, relative to the rest of the agents design, e.g. including their "built-in knowledge" (*assumptions*) and their performance measure. Of course, we can ask *whose performance measure* and *whose expectation of maximizing it?*

Typically, the "performance measure" in question is understood as the agent's *utility* and the expectation of maximizing it is understood as computed by the standard formula. So, acting rationally at that time would mean having our agents decisions conform to the formula for maximum expected utility:

$$MEU = \max(a, \sum(U(o)P(o \mid a, p, K))$$

i.e. choosing the action a, which maximizes the sum over outcomes o, of the product of the utility of o and the probability of o given a, percept sequence p, and built in

---

[17] © atheoryof.com

[18] Russell and Norvig (2003), *Artificial Intelligence: a modern approach*

knowledge K. And the upshot is that the smarter the agent the better the job it does at approximating the mapping defined by this function.

There is an important question of how we are supposed to understand where this utility function comes from, but it's typically thought that for the agent to make decisions in the direction of maximizing it, the agent has to be "aware of" more or less what it is. Similarly, there is an important question of where this conditional probability function comes from; and again, it is assumed that the agent has to at least calculate it to some approximation on the basis of things he more or less is "aware of". So typically, *beliefs* and *desires* here enter into the equation, where it is assumed that the utility function is a function of the agent's *desires*, which he is assumed to be more or less aware of; and where it is assumed that the conditional probability can be approximated on the basis of *beliefs*, which it is assumed he is also more or less aware of.

The reality of beliefs and desires and their influence on action, is to some degree supported by the ways in which we talk. It is not unusual for us to explain a persons behavior by saying that they did x, because they *wanted* y, and *believed* doing x would get it for them. In fact, it seems we talk as though desires and beliefs are both *reasons and causes* for our actions[19]; and that this dual nature allows this model of the intelligent agent to serve both a *normative and descriptive* function which might be seen as unique. This model of the intelligent agent, as one which does their best to maximize their desire fulfillment on the basis of their beliefs and percepts, is, I believe, the classic model of the intelligent agent.

Though such agents are not wedded to a computational view of agency, much of the support for this theory comes from its integration into a computational picture which has seemed philosophically appealing within the later half of the twentieth century. In particular, the "propositions" which are the contents of the propositional attitudes (beliefs, desires, etc.) are represented by symbols – or rather, compositions of symbols, i.e. sentences. These symbols are understood to have their contents fixed *externally* in virtue of correspondence or causal connection with the world (an issue which it is thought we have to face any way in the philosophy of language[20]). To have a particular attitude toward such propositions, is just to have the symbols which represent them in some location (black box) or in some role (functional role) within the architecture. The process of decision making, then, is understood as simply a form of computation over these symbols, analogous to logical derivations, of what action should be carried out.

The support for the classic model can be summed up as follows:

1. Its integration with a well established normative notion of rationality (MEU)
2. Its plausibility from the point of view of linguistic reflection and ordinary explanation of human action.
3. The appealing philosophical picture of the mapping of the agent to computing machinery.

But there are some very good reasons to question this view of agency as well. Some of the more important criticisms are:

---

[19] Davidson

[20] Putnam

C) The biological/neurological unreality of beliefs and desires (Churchland, "Eliminative Materialism and the Propositional Attitudes").
D) The limitations on instantiating such agents because of the constraints on computation, and in particular on computing probabilities (e.g. calculating the conditional probabilities within the popular framework of Bayesian Networks is NP-Hard (Pearl)).
E) The simple lack of fruit (applications) born from this conception of intelligence over the past half century.

Of course, without an alternative picture that provides the hope of achieving what the classic picture promised, the above criticisms might seem simply a hollow fact of life. I believe there are three things which an alternative needs to provide (abstracted from the support for the classic model above):

d) Integration with a normative notion of intelligence/rationality.
e) Intuitive plausibility.
f) A framework for understanding human action.

In fact, I believe that 2 is the bridge between 1 and 3. Without intuitive plausibility there is no ground on which to connect a descriptive theory as provided in 3, to a normative notion as provided in 1. This classic model of agency has its point of contact with intuition, at the linguistic level. When I reflect on the way I operate, I do indeed *say* that I "desire" that and "believe" this. And under the assumption of twentieth century philosophy that language was a clear window to the mind, we arguably had some intuitive plausibility for the reality of belief and desire.

But when I reflect on my experiences and thoughts, and how they lead to actions, I do not witness any such interplay of propositions which I have the attitudes of belief and desire toward. For me, this has always been so. When I was introduced to the idea of the "propositional attitudes" when I was 20, I thought then and there that people were taking their words too seriously. And I still do. Luckily, a theory of agency is not wedded to primitive beliefs and desires, nor even to an obvious mapping of beliefs and desires to the model. A theory of agency, to be a proper theory of agency, must provide an understanding of the reasons - both normative (1) and descriptive (3) - for action, which is comprehensible from the subjective point of view (2). And to attain this, we will be taking for our basic intuitive notions: *concepts, recognitions, experience,* and *values*; which I believe to be *more* comprehensible - from the subjective point of view or otherwise - than beliefs and desires. In the following sections I will elaborate on these elements.

The Spacious Present

Experience has a spatial-temporal structure. In fact, I assume here that our experiences have a spatial-temporal structure in virtue of *occurring in a space and time*, namely, the space and time of experience – which is to be understood here as entirely intra-subjective. Our experiences, as I understand them, are *qualitative* as well, but nonetheless, they can be represented mathematically.

The three domains that we will concentrate on here and in future works are the domains of *visual*, *tactile,* and *kinesthetic (motor)* experience. We idealize here away from scent, hearing, taste, and any other sense-domains one may wish to posit; we assume here that these three domains of experience *occur within the same space and time* (i.e. *the* space-time of experience)[21]; and we will understand these domains as having a very similar structure, i.e. I assume that experiences within these domains are to be modeled as *surfaces* with continuously varying quality-values over their points, where these quality values can be modeled by a continuous interval, e.g. (0,1).

This much, I hope, is fairly straightforward. There is, however, a part of my formulation which takes some getting accustomed to. Although it may be common to think of experience as *not* temporally extended, i.e. that our experiences occur in three dimensional space, and simply change with time, I maintain that there is a *time* dimension to experience as well, i.e. that our window of experience is at least to some extent, extended in time – the so-called spacious present. To stoke your intuition: think of such experience as "short-term experiential memory"; as when one shuts their eyes and continues to have the prolonged experience, until it fades outside the experiential window. This, on my view, is no less experience, and moreover, is assumed to occur even if new data is coming in, per usual. The idea that experience is extended in time is very important to my formulation of the pattern-recognition agent.[22]

Pattern Recognition

I assume that we recognize *patterns* in experience. A pattern in sensory experience can be thought of as *any* collection of qualitative surface points in experiential space-time. In particular, these experiences needn't be from one continuous region. They can be from multiple continuous regions. And there is no 'primacy' of patterns that are in one continuous region or of one quality. All patterns are treated as equal – ontologically and epistemologically speaking - i.e. 'object' patterns are just patterns, on equal footing with the pattern of all wave crests in a view out to sea, and both are recognizable.

When we recognize these patterns, they are recognized *as* a type. And the result of this recognition of the pattern as a type is an entity we will call *the* recognition. A simple example is my recognition of one of the patterns in my sensory experience as of type *Mug*. We say, *the pattern is recognized*, but it is recognized *as a type Mug-pattern*, and the result of this is an entity *the recognition of a Mug*... But the structure of the blue-white-silver-black pattern is not all there is to the recognition in this case. In particular, *the space* of the recognition extends in experiential space-time to places I can't "see", behind the qualitative surfaces. In the case of object recognitions, I assume that the recognition has the spatial structure of what we would call *the object itself*, only *in experiential space*.[23] Such a basic idea that recognitions have extent greater than their

---

[21] Even though in the final analysis one must admit that these domains can occur in separate space-times.

[22] I have labeled my claims as assumptions, since in most cases it is not possible to *argue* for them; in fact, unless someone points out an inconsistency or a more appealing alternative, all I can say is that *you are not like me*. Despite the stubbornness on this point, I know that this is in every way an idealized model and that there is almost certainly a better formulation available. But the key points are that experience does have a structure of its own, that this structure can be mathematically modeled, and we can use this model as an *intuitive explanatory foundation* in the analysis of the intelligent agent, as we shall see.

[23] Note that I here am remaining neutral as to the existential status of ordinary objects

experiential surface is well established by gestalt phenomena. But this doesn't prevent us from simply recognizing *the surface*, which doesn't have a volume.

I assume that these recognitions, moreover, *have a location in experiential space and time.* In most cases, it is fixed (though perhaps not completely) by the location of its corresponding experiential surface – though possibly of greater extent (as in *my mug*). Nonetheless, the recognition is a matter of *degree*; and moreover, the location of the recognition may be *fuzzy*. We may recognize something to some degree and in such a way that its location isn't discrete, but graded within experiential space-time. But the points in this location have a *character* which constitutes the recognition as a type.

To reiterate: *any pattern that fits in the window of experience can be recognized*, which is to say that *any subset of surfaces in experiential space-time fulfilling the constraints in the previous section is a pattern and can be recognized*… Such a pattern can be through space and/or time; it can be a disjoint set of surfaces or one continuous surface through time. The patterns are recognized *as* a type. And the result of this is *a recognition* that has a *degree* and a (potentially fuzzy) *location,* with a *character* unique to the concept.

Concepts

I assume that recognitions occur in virtue of the application of *concepts*. Concepts are the bridge from sensory experience to recognition. They can for starters simply be thought of as a *function* producing a recognition of some type and to some degree at some location in experiential space-time, given certain experiences. For now we can assume that we are given a working set of such concepts to apply, and we can ask *what gets recognized?* It is my intent here to understand our recognition states as *composed* of applications of individual concepts. As such, we need a theory of individual concepts as well as a theory of how these distinct concepts come together to compose our recognition state in applications.

Any distinct concept, C, is defined by a function of the sort C:SP->R. Where SP is the set of all sensory patterns and R the set of all recognitions. But there are certain restrictions on which functions represent concepts.

First, it is to be understood that the recognitions in R that are produced by a particular concept, C, have a distinct *character*, unique to the concept in question.

Second, for any s in SP and any r in R, such that C(s)=r, r covers the area of s, i.e. the points in experiential space-time covered by a sensory pattern, s, are among the points covered by a recognition, r, if C(s)=r, for some C.

Although this second constraint clearly holds of most of our recognitions, it is nonetheless debatable. One might site, for instance, this scenario: you just ducked around the corner on the city street and you wait in order to startle your friend, just for kicks, as she walks by. You see her shadow approach, and it is her – almost unmistakably. In this case, we might want to say that you recognized your friend in virtue of a pattern in SP, which is not in the location of your recognition of her itself… Of course, there is some truth to that. But as we will see, we can understand concepts and their recognitions under the constraint given and still account for the recognition in this sense. As it will turn out,

this kind of "recognition" of your friend will be understood in terms of the recognition of the shadow as part of a *pattern* which your friend is also a part; and as we will see in later developments, it makes the life of our agent *so* much easier as to seem almost essential that we understand it in this way.

Of course, understanding individual concepts in this way, we are left with some questions for the bigger picture. And in particular, given a store of concepts, we have to be concerned with the composition of these applied concepts to reach an overall recognition state: i.e. given a store of concepts, what gets recognized, and according to what laws? There are two ways of answering this question. There is a *normative* way, and a *descriptive* way. From the normative perspective, we ask, *what is ideally recognized given our concepts*? From a descriptive perspective we ask *what would we really recognize, given our concepts?*

Given a set of concepts, we can take a stab and say: we would ideally *maximize our overall strength of recognition while still remaining consistent*. The basic idea is that some recognitions, though applicable by the function that is the concept itself, would nonetheless conflict with other recognitions which may be stronger taken as a whole; the set of concepts that apply with the most overall strength will get applied *ideally*… In real applications, on the other hand, it is likely that the concepts that we *try* to apply are to some extent ordered (though perhaps with an element of randomization) by their significance. As we try to apply them, when we get a match, we recognize, and then fill in our recognition state around what recognitions came before - such is the popular concept of *framing*. This is just a sample explanation of how real recognition might work; an example of how we might work to fill in the human agent model.

There are a few things that the above preliminary considerations ignore. Most importantly: what determines the *strength* of a recognition? What determines which concepts are *relevant* at any given time? How and when do we *form* concepts? Before understanding the full story, we must consider the importance of *values* within our agents.

Value State

It is common within psychology and philosophy alike, to assume that what is of positive value is pleasure or happiness and what is of negative value is pain or unhappiness. Maximizing rewards while minimizing punishments is then the goal of the game of life; and acting rationally is doing this to the best of one's ability.

Learning to best act in these ways given past rewards and actions is the subject of *reinforcement learning*, a sub-topic of AI which has its roots in Turing's positive intuitions about intelligence. But mistaking feedback, positive and negative, for the value of the *state,* is to mistake an artifact for its ends. To make the point that what is of value is the *state* we are rewarded or punished for achieving or being in, rather than the reward or punishment itself, it does well to consider two distinct types of *states,* which I will call *value states*.

First, suppose one value state simply consists of one variable. The feedback we get depends on the value of this variable; the greater the value the greater our reward; and if we maintain the value at a high level, we will get the continued high reward. In such a case, it seems quite clear that what we *should* do is increase this variable, all other things being equal. But we may ask *why* should we increase this variable? Is it because its

increased state provides us with more reward? *Or is it because the increased reward is an indication of the value of the increased state?* Considering such a case by itself, it is not clear to me where the value should be thought to reside.

However, consider a second example. This value state also consists of one variable, but in this system, you are rewarded for moving the value of the variable close to a particular value. If the value strays above or below the target value, you are rewarded for acting so as to change it to a nearer value, and rewarded proportional to the difference in proximity. *But by simply maintaining a value close to the target you are not rewarded or punished at all.* In such a case, it seems to me that what should be considered of value is the *state* of the variable, and not the feedback – since the feedback is clearly serving as an indicator of the positive value of having the variable close to that value.

Of course, we can treat both the first and the second case uniformly if we just admit that it is in both cases the *state* which is of value, rather than the feedback. This distinction between the value of the state and the role of feedback is crucial to a correct understanding of the pattern recognition agent.

The Basic Pattern Recognition Agent

We are now in position to appreciate the basic pattern recognition agent model. The basic understanding of our agent is this: the agent acts *randomly* within their scope of possible movements; when a satisfying event occurs, they construct a *concept* of their experiences – including the experiences of the random motions that took place - leading up to that event; at a later time, if they have similar experiences, and are in a similar value state, they will recognize the pattern in experience (through the application of the concept) and act in accordance with the recognition. Note here that I'm claiming *the action experiences themselves are part of the pattern recognized and it is in virtue of this recognition that the action takes place at the later time*.

The strength of these recognitions will have a lot to say about which among the possibly many conflicting actions will be carried out; and that strength will have a lot to do, not only with the fit of the concept to the current experience, but with the strength of the concept as determined by the prior feedback and the similarity of the current value state to the value state the agent was in when that pattern was previously successful.

Of course, with the recognition of a pattern, you have a recognition which may extend outside of experiential space-time. Such is to be expected in any pattern recognition agent. Experiential space-time becomes a sub-region of recognition space-time. And as the patterns involving actions flow into experiential space-time, those actions are carried out non-randomly. With this understanding, we can see that recognition, prediction, and intentional action are all unified.

Though some would take issue with this use of "intentional", it seems clear to me that such patterns involving action extending into the future and flowing to the present is much in line with the conception of intentional-states put forth as independent psychological states by Bratman[24], though he works within the more classical framework of propositional attitudes.

This leaves us to state how we are to understand the more classical psychological states of belief and desire. Understanding these, as well as many other things, within the

[24] *Intentions, Plans, and Practical Reason*

framework of pattern recognition agency will have to wait for a fuller treatment. I plan to start with that fuller treatment in my next article, where I will provide an idealized and minimized model of the pattern recognition agent for the sake of simplifying some complex issues; and consider how this idealized agent compares with some standard reinforcement learning agents.

Pattern Recognition Agency[25]

By *C. S. Schroeder*

Agency as I have begun to describe in previous essays can be idealized into a simple model. This model will naturally have reduced complexity compared with human agency, but will contain many of the most important elements of agency in general. *Our* experiences, of course, involve a four-dimensional space time structure, which we assume corresponds, to some extent, a four-dimensional space time structure "in the world". We can go a long way toward simplifying our issues if we reduce the dimensionality of our agent, and assume that our agent's experiences are one dimensional. That is, like ourselves, our agent has a temporal dimension to his experience, but unlike ourselves, our agent does not have a spatial dimension to his experience. He is, what I call, *the pinhole agent*, for our agents experiences are similar to what ours would be if we were to look through a pinhole; we would see one color at a time and these experiences would eventually move outside of our experiential window with time.

We will assume that our pinhole agent experiences colors and motion qualities, which because they may overlap in experiential space time, can be understood to occur at each point in experiential space-time, i.e. at each present moment a new quality pair <color, motion> enters our agents experiential space-time. We can do ourselves a favor now if we simplify things further by starting with the *discrete* pinhole agent. The discrete pinhole agent has a finite set of possible qualities it can experience, instead of the continuum that seems subjectively possible for us. So in particular, we assume the following qualities are possible:

Colors = {R, O, Y, G, B, I, V, null}
Motion= {L, R, U, D, S, null}

That is, we consider the colors of the rainbow, plus null; and we consider left, right, up, down, and stay to be the motion qualities, plus null. Furthermore, the discrete pinhole agent experiences these qualities in incremental space-time; so, instead of having an experiential space time like this:

———————

Our agent has one like this:

– – – – – – – – – –

Where the qualities enter into this experiential space-time pairs <color, motion> at a time.

Now, you can imagine our agent as being in the following world: The world is laid out like an infinite chessboard, only instead of white and black squares, our squares can be any of the colors from the set above in any pattern whatsoever. When our agent is over a square, our agent experiences the color in the square at that time. Now this world is not restricted to being static. That is, even if our agent does not move off of his square, it may change at the next increment; and if he does move, the square he moves to might be different than what it was a few moments before. In fact, the world, within these

constraints, can be *absolutely any way whatsoever possible*. It may be random, it may be strongly patterned, it may be a never ending continuous sea of orange…

To clarify, our motion experience at a time is the quality from the motion that was decided upon leading up to it. So if our agent "decides" to move left during real time r0, then the motion quality at time s0 will be L at r1. This also introduces the convention that we follow here, that the subjective present will be fixed with the label s0, and recent past experiences will grow in there s(t) labels with distance from the present, until they cease to be within the experiential window; e.g. in the above, they could grow to s9, until they drop off. But of course, in the whole window of experience exists at every instant of *real* time, and it is a substantive question if, e.g. the value of s0 at r(n) *must* equal the value of s6 at r(n+6), as we assume it *usually* does; as we shall see that this is not answered straightforwardly in the affirmative.

But before we get to this or other matters, we must specify something that *matters* to the agent. We will assume that our agent's motives, again unlike our own, can be represented by a single, one-dimensional interval, where its current state is simply represented as an index into this interval. We can understand this interval as being carried around with the head of the agent, or alternatively, we can understand this interval as represented on a *parallel* board, which can take various shades from black to white. However it is understood, we can say that this index can fluctuate, again, in *absolutely any way whatsoever possible.* That is, it may correspond in some way to the world of colors and motion, or it may not. Though the continuum from black to white does fine conceptually, it will do well for representing this interval if we give it numerical labels; so,

Dimension of Concern (DOC) = [-1, 1]

a subset of the real numbers.

Now, the existence of this dimension of concern in no way indicates how the agent is concerned with it. That is, does it "want" it to be white or black? Or does it "want" it to be middling gray? Of course, what it would take for us to legitimately say that the agent "wants" it a certain way is simply for the agent to be *as though designed to achieve that state*. So in order to engineer the agent so to "want" a particular state, we must have some idea of what the agent *should* do to attain a particular state, and then describe it in such a way that it to some fair degree does what it should. But this seems to be a remarkably hard thing to do when you do not know in advance what the world of color, motion, and motive (DOC) is like! Since, after all, the correspondence of motive to color and motion may be entirely random.

We will design our pinhole agent to "moderate" his dimension of concern, i.e. the closer the agent gets his state to the middle, the better. But we must make clear that our agent does not "know" that he wants to moderate his dimension of concern. And in fact, he doesn't *experience* this dimension of concern in the way that he experiences the colors and motions. The agent is quite simply *moved by* his dimension of concern – i.e. he changes his behavior based on the index into this dimension (in ways that will soon be made clear). He can only come to learn about this dimension through the effects of the feedback that it provides; and he only "knows" anything about it in a higher-order cognitive sense, much further down the line.

We suppose that our index starts off at the dead center, 0, medium gray. What should the agent do? We would probably say that the agent shouldn't do a thing. But he does – and for that, he's less biased than us. The agent, having no clue as to the world he lives in, has no reason to stay any more than he has to move in some direction; and when the agent has no reason to do something, he does *anything*. That is,

To the extent that an agent doesn't have a reason to act, that agent (at least to that extent) acts randomly.

This is a substantive hypothesis about agency in general - which, by the way, is not meant to indicate that the agent will never have a *reason to act randomly*, in which case the agent may act *more* random (i.e. it will turn out false that, to the extent that the agent acts randomly, that agent necessarily (at least to that extent) does not have reason to act.)

Of course, if the agent has no *a priori* reason to act in some way, what can *give* him a reason to act some way? The idea is simply this: the agent acts *randomly* within their scope of possible movements; when a satisfying event occurs, they construct a *concept* of their experiences – including the experiences of the random motions that took place – "leading up to" that event; at a later time, if they have similar experiences and recognize the pattern in experience through the application of the concept, they will thereby have a *reason* to act – they will have reason to act in accordance with the pattern because that pattern is linked with satisfaction.

It is open to argument whether such recognized patterns *really* give *any* reason for action at all. If it's possible that motives don't correspond in any way to motion and color, then what *reason* does the agent have to act in a way that was successful in the past? What is important to understand here – so I will reiterate - is that our agent is in no way *aware* of his motivational state. The agent does have qualitative experiences of his motions and colors, but he in no way experiences "qualities of satisfaction" - the mythical pleasure and pain. For all the agent can tell, the satisfaction "was", "came from", "exists in" the color and motion experiences he was having at the time of the satisfaction. We can then understand the reason an agent has to "act in a way that was successful in the past" like this: the satisfaction that our agent had at some point leading to the formation of the concept occurred at some *real time* r0. At this real time, our agent was experiencing all color and motion qualities from s0 to s9 in *subjective time*. Our agent, therefore, has no reason to associate the satisfaction with some particular quality or quality pair in s0 to s9, so he associates it – at least initially – *with everything* in s0 to s9 at that real time. Our agent, therefore, has a reason to carry out the actions embedded in the recognition, because it is the only way to fulfill the recognition, and recreate the experience that he has every reason to assume *is the satisfying event itself*. The "goal" in this case, is not what happened at s0 when the satisfaction occurred in real time; our agent has no reason to parse that out as of "real value"; rather, the agent takes everything that occurred for him at that real time as what is significant, for he knows no better – if it can at all be said he'd be better off for knowing - and has nothing else to go on.

Even if the agent has a reason to act some particular way, however, it doesn't mean that he *should* or *will* act in that way. First, it will be clear that different recognitions can give reasons for different and conflicting action. Second, despite having a reason for action, there remains a *random factor*, which decides among the actions to

take up; and this random factor, although giving greater weight to actions that have a reason to be carried out, nonetheless, will still give some weight to the other actions that are possible. We will now try to present the mechanisms by which our agent operates.

Presentation of Discrete, Pinhole, Pattern Recognition Agent

Crucial to our understanding of agency is three types of entities: *experience, concepts,* and *recognitions*. All three of these are represented by a more general entity: *patterns*. Patterns, for this agent, are as you may expect:

PPPPPPPPPP

Where the P's are to be replaced with pairs <color, motion>. The inclusion of *null* in the above definition of the color and motion qualities is to indicate that the patterns may be, as it were, incomplete. They may be, for instance, of the form (ignoring motion for the moment)

BR GIRO VV

Where we will understand the blank-space as null and null as blank-space. An experience, as I typically understand it, is a *complete* pattern; but a concept or recognition need not be. We will represent the set of current concepts the agent has available as C and we will represent the set of current recognitions as R.

Our agent has three basic operations that he carries out: *store, match, and decide*. The *store* operation takes a pattern in experience and stores it as a concept in C. The *match* operation takes patterns in C and determines their applicability to current experience. If they are applicable, it results in a recognition at the location it is applicable. The *decide* operation takes all the recognitions in R, and chooses an action based on the strength of those recognitions and a random factor. There is in this, however, one basic thing I have left out which is important to everything so far described, and that is *motive*, i.e. dimensions of concern (DOC).

To start, DOC is relevant to the *store* operation: suppose you experience something; this in itself is not sufficient for the application of the store operation; to apply store, there must be some incentive. The incentive comes when there is a change in your DOC. If your DOC is indexed at -.5 and jumps from there to 0; then this is a fair indication that you should take note of what just happened. This "taking note" happens by applying *store*. But store doesn't just take whatever pattern is in experience and toss it onto a heap of other patterns that are in C; rather, it takes the pattern and *stores it along DOC* at -.5. Why the pattern is stored at -.5 and not simply thrown onto a heap is that our agent is not always looking to match this pattern, but rather, only when he is in a position with respect to DOC that he was in before, when the pattern was relevant – in this case, -.5.

This brings us to *match*. The match operation is applied over some concepts at each increment of time, but not every pattern in C is tried for a match - only the patterns *in the neighborhood* of the current index of DOC. So in particular, if at some later time

we were again at DOC index of -.5, then we would likely try to match the pattern we stored at -.5 to the present experience. If DOC was at .5, that pattern would likely be ignored. When a match occurs, a recognition is created. The recognition has the same pattern as the concept, and is fixed to the location where the match occurs. So in particular, if the current experience is (ignoring motion and fixing s0 at far left):

RGIVOYRROV

There may be a match between it and a concept:

VRYGVGRGIV

at the overlapping region RGIV. In this case, the recognition would be fixed to the front portion of the experience of RGIVOYRROV, at RGIV, and the VRYGVG portion of the recognition would extend beyond the present and outside of experiential space-time. This extension beyond the temporal bounds of experience I take to be an essential component of any intelligent agent. We will say that these extensions occur in *recognition space-time*, which has experiential space-time as a sub-region. It is important to note that the *strength* of a recognition is as much a function of its fit with present experience as it is with its proximity to the current index on DOC. But moreover, it is a function of its *relevance* – a measure that will be made more precise, but can be thought of as the count of the pattern at that index (i.e. how many times that pattern was stored).

Exactly how close a stored pattern needs to be to the index to be considered, and how to grade the notion of proximity among the patterns that are considered, will be considered shortly; for now, we simply state that the patterns in this neighborhood are said to be in *the working set,* WS. If the proximity to the index is measured as *p* between 0 and 1; the degree of match - similarity - between the pattern and the experience at some point is *s* between 0 and 1; and the relevance *r*, is unbounded; then we can hope to calculate the strength of the recognition as *p* x *s* x *r*.

The *decide* operation is applied after store and match, during each increment of time. While there are many methods that can be used for the decide operation, one simple method is to take R and conduct a weighted lottery, selecting an action to follow at that point in time; though with "random" as a separate option, where its selection indicates the selection will be made at random without weight. It is a crucial issue to determine the probability with which "random" gets selected – or uncertainty, *u*. I tend to think of it as a function of the *density* of the patterns along the DOC. We will have to consider how the *proximity (p), similarity (s), relevance (r),* and *uncertainty (u)* are determined – so now we turn to issues regarding these measures.

Issues in Pattern Recognition Agency

A number of issues in pattern recognition agency are closely tied to issues in reinforcement learning. In fact, nearly every issue in reinforcement learning has a correlate in pattern recognition agency. We will start by addressing these similar issues.

What I have called the "relevance" of a pattern is closely related to what is considered the *value* of a *state* in reinforcement learning. And the question of how to

compute the value of a state in reinforcement learning is the central question of that discipline. So we can expect to get some help from this theory when we ask the question: exactly what feedback is necessary for a pattern to be stored with some relevance and how is it we should change the relevance of a stored pattern?

On the question of which pattern should be stored with some relevance, we can assume simply that there is a threshold to the value attributed to it for it to be stored – so the question is how do we determine and change this value with time is the central question. We assume that the initial relevance of a pattern is proportional with the reward it is associated with – but the reward associated with the pattern directly, i.e. the reward achieved when the pattern was experienced, is not all there is to the relevance; for if it were, the relevance of the pattern would ignore, for instance, what might happen immediately following this reward, e.g. persistent high negative reward. Effectively, what we need to take this into account are *eligibility traces*, which determine how the reward is propagated back through previously experienced patterns – influencing its relevance[26]. In general, we will assume that the reward will be propagated to the extent that the directly associated pattern overlaps with the previous patterns – where this usually means that if your window is of length 10, then the strength of the reward at r(i) will be attributed 100% to the pattern at r(i) and (100-10j)% for each pattern at r(i+j) thereafter (until j=11).

The second important matter that we can look to reinforcement learning for support is the issue of *deciding* on an action. In reinforcement learning, the issue of decision making revolves around striking a balance between *exploitation* and *exploration*. In exploitation, one uses what they've learned – so in our case, we would exploit if we decided on an action according to the pattern which was recognized with the most overall strength. In exploration, on the other hand, one tries new things to see what might come of it – this is the factor of uncertainty that I've mentioned already, the uncertainty as to whether there may be something better. The most appealing resolution to this matter from my perspective is a lottery over the patterns recognized, weighted according to the strength of the recognitions, but with a distinct random participant, which has its own weight – of course, determining this weight is not so simple, but I tend to think of it as a function of the density of patterns around the index to the DOC… So we turn now to the questions surrounding this index.

The first issue is exactly how close a stored pattern needs to be to the index to be considered. I take it that the range of DOC has a lot to say about the scope of consideration; but I can say that I expect, more or less, a bell shaped curve applied over the index to DOC. From here, we can answer the second question, of how to grade proximity to the index, and say that one can determine how to grade the notion of proximity among the patterns to the DOC according to the value of that Bell shaped curve at the patterns location on the DOC.

I should say that there does not appear to be any help supplied by reinforcement learning on the topic of proximity to the DOC – in fact, the whole idea of a context is largely irrelevant within the framework of Markov Decision Processes[27]. So similarly, it will have little to say on our next set of questions, regarding pattern matching: Exactly

[26] In reinforcement learning, this process is an effective way of speeding up the learning process; but in our case it is a necessary component for learning the value (relevance) since we do not store the value for every state in advance (in some form).

[27] Where what came before is probabilistically irrelevant to what happens next.

how similar a stored pattern needs to be to the current experience to be applied, and how should we grade the notion of similarity among the patterns – thereby grading the value of the recognition? Of course, what this does, in large part, is ask for an understanding of *similarity* – an understanding that we are not in a position to give. We therefore assume that there is a property of similarity, which comes in degrees, and figures prominently in recognition – but relent in trying to explain its mechanics, which as I have argued previously, fall outside of the computational realm.

Finally, there remains the issue of how it is that general patterns are abstracted from the particular patterns our agent experiences. Of course, again, the patterns that are abstracted from those experienced are abstracted based upon the similar parts of the experienced patterns. (Again the notion of similarity has a crucial role to play in understanding the mechanics of our agent.) These abstracted patterns contain among them the "null" values mentioned earlier, but are stored like any other pattern along the DOC.

Final Remarks

These are the core issues in pattern recognition agency as it stands today. In work to come, I will present the discrete pinhole pattern recognition agent in a simplified interactive model, through an applet that will be available at this site. This demonstration, naturally, is for the purposes of understanding pattern recognition agency. But as many of the issues at the end of the prior section indicate – we are a long way from implementing this agent on computational hardware. In fact, the concluding issues of the last section may tend to indicate that my urging the reader in previous works to free themselves from the bounds of the computational perspective was not simply a matter of opening the readers mind for the sake of understanding. It may seem to indicate that in fact, we must understand pattern recognition agency independent from the computational perspective.

I am not inclined at this moment to give up on the computational perspective, of course – but I am inclined to view attempts to replicate the true pattern recognition agent within this perspective as only possibly *approximating* this agent. I am inclined to say that pattern matching, which is the core of any pattern recognition agent, can only be efficiently carried out by non-computational processes, and that is not to say it is never efficiently carried out at all… But there will be more to come on the relationship between tractability and pattern recognition in our June edition.