Chapter 4

Consequences of a Finite Brain

Do the ultimate limits on what it is possible to compute have a role to play in explaining our world? Can the ultimate limits on computers tell us anything about ourselves? Can recursion theory, the most theoretical discipline of computer science, be applied to the brain? One major point of this chapter is to suggest that the answers are "Yes": Studying the limits of computation can give one a handle on principles governing any sufficiently intelligent agent, whether the agent is meaty or metal. If you are a finite machine, as we are, then there are certain necessary consequences. *Any* brain will have certain similarities. The similarity that I concentrate on is the phenomenon called *vagueness*, and I show why any finite-brained agent will have to deal with vagueness.

As a student, one of my interests was to understand the ultimate limits of thought, learning, understanding and intelligence. My interest was not really about brains, per se; my interest concerned the ultimate limits of *any* thinking machine. Now, it is primarily brains that think, learn, understand and have intelligence, and so one might expect that studying neuroscience might help one to discover these limits of thought. Not so, at least not with the state of neuroscience today. The reasons are that

(a) brains are not well-understood,

(b) brain activity is not easily describable, and

(c) brains are not susceptible to proving theorems about them and their limits.

And even if someday (a), (b) and (c) are averted with a mature neuroscience, it may be that

(d) the brain, being just one of presumably infinitely many possible kinds of thinking machine, barely scratches the surface as to what the ultimate limits are.

This reasoning led me to learn computer science, and, more specifically, logic, theoretical computer science, complexity theory and recursion theory. Why? Because

(a) computers *are* well-understood,

(b) their activities *are* nicely describable, and

(c) computers *are* susceptible to proving theorems about them and their limits.

Furthermore,

(d) in theoretical computer science one may study what is arguably the class of *all* thinking machines, not just one specific kind of thinking machine.

Understanding theoretical computer science and logic not only illuminates the limits of machines, I believe it can tell us interesting things about one particular machine: the brain. A computer is, alas, not a brain. The fine details of how the brain works will probably not be illuminated merely by understanding computers and their limitations; those interested in the fine details need to study real brains in addition to computers. But I have never been interested in fine details—I care only about general, foundational principles—and this goes for the brain as well. Given the plausible notion that the brain is a kind of computer, understanding computers and their limits may provide us with insights into the brain. That is, using computers as a model of the brain may have a payoff.

There is a danger that one might take me to mean that "using computers to model the brain may have a payoff." Now, *this* is uncontroversially true; computational models in neuroscience and psychology are widespread. Generally, the idea is to concoct a program that captures relevant aspects of the system of interest. But this is not what I mean. When I say that the brain is to be modeled as a computer, I do not mean that I have devised some particular kind of program that seems to capture this or that aspect of the brain or behavior. Instead, I mean that the brain is to be treated as a computational device—whether a neural network machine, a random access architecture, or other—and is thus subject to the same computational limits as computers.

This, too, is almost entirely uncontroversial: nearly everyone considers the brain a computing machine. (Although Penrose (1994) is one counterexample.) What has not been given sufficient attention, however, is that the brain-as-computer hypothesis—all by itself and without reference to any more detailed hypothesis about the program it computes or what its specific limitations are—has certain empirical implications. Rational, computer-brained agents—although they may be radically different in behavior, personality, intelligence, likes, and so on—are going to have certain similarities. These similarities are things true of any rational computationally bound agent; they are "behavioral invariants" for such machines.

Our limitations

Consider the informal plots in Figure 4.1. Each plot has human "activities," or behaviors, along the x axis. Along the y axis in each plot is the human "ability"; $f(x)$ is the degree of ability humans possess in doing activity x. For example, perhaps we are interested in the activity x of running, in which case $f(x)$ represents the human ability to run so and so fast. Or, consider the activity x of holding items in working memory. Humans have the ability to hold around seven (plus or minus two) items in working memory, and $f(x)$ represents this ability.

The top horizontal line in each graph represents the line at which greater ability become logically impossible. Graph A is what I will call the "usual conception," where our human abilities always fall short of the logical limits of what is possible. For example, having a working memory space of seven is presumably not explained by reference to any logical limits on what is possible. The explanation for it is likely to be brain-specific and historical in nature. Graph B, on the other hand, depicts an activity (the arrow) for which the limits of logical possibility are the active constraint in explaining the human ability in that activity.

For example, consider the activity x of determining truths of Peano Arithmetic. First, what is "Peano Arithmetic"? For now, you just need to know that it is a short list of obviously true sentences of arithmetic, from which one may derive infinitely many other truths of arithmetic (but not all the truths of arithmetic). And, note that "arithmetic" just refers to what you might think it does: mathematical sentences concerning addition and multiplication on the natural numbers $0, 1, 2 \ldots$. Thus, I am asking you to consider the activity of determining whether or not a sentence in arithmetic follows from Peano's axioms of

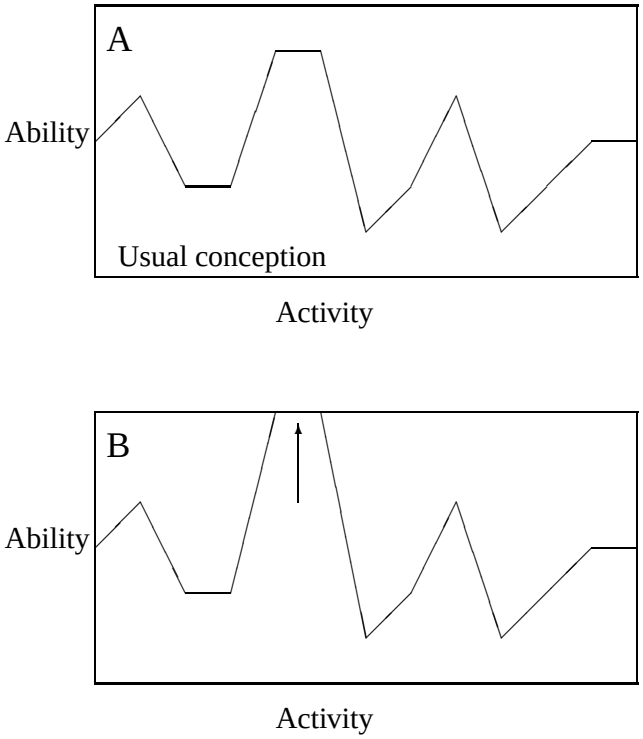

Figure 4.1: Ability to do a given activity. The top horizontal line in each plot demarcates the boundary between logical possibility and, above the line, logical impossibility. The usual conception is to view our ability as falling short of the line of logical impossibility. That is, the explanation for our ability being what it is usually refers to hosts of contingent physical or biological details about us. On the other conception, perhaps there are abilities of ours (the arrow) that are, in some sense, as good as they possibly can be, where "possibly" refers to logical possibility. The explanation for our ability (or inability) being what it is refers entirely (or almost entirely) to the limits of what is logically possible.

arithmetic. It turns out that we humans are not perfect at this; we are not perfect determiners of which sentences follow from Peano's axioms and which do not. This is not the kind of thing that anyone has actually experimentally tested, mind you, but no savant has come forth able to, without error, tell whether or not an arithmetic sentence follows from Peano's axioms. We humans seem to have a limited ability in this regard.

Another place we have a similar inability concerns the problem of determining whether or not some program is going to halt on some given input. That is, I hand you some software for your computer, and I give you a copy of the programming instructions, or code, underlying the software. I then give you something to input into the program. Two things can possibly happen. First, the program could take the input, grind it through some instructions and so on, and eventually terminate. This is what we almost always *want* our software to do: to eventually stop running without us having to reboot the system to force it to stop. The other possible result of placing the input into the program is that the program may never stop running; it is said to never halt. It just keeps carrying out more and more computations, going on and on and on without end. This is when one must definitely resort to rebooting the computer, or breaking out of the program somehow. Your task, with the program code in one hand and the input in the other, is to determine whether the program will or will not halt when that input is entered. This problem is called the *halting problem*, and people are known to be notoriously bad at solving it. Again, it is not as if there have been psychological experiments in this regard (not that I know of); it is just known within computer science circles, for example, that we are *all* prone to error in solving this problem.

One possible source of our limitations: logic itself

What is the source of our limited abilities in determining the truths of Peano arithmetic and in determining whether a program halts on some given input? It is thought that our limited abilities in these activities are explained by the undecidability of the problems. In particular, the set of truths of Peano Arithmetic is undecidable, as is the set of pairs of programs and inputs for which the program halts on the input. What do we mean by "undecidability"? We say that a set is *undecidable* if there is no computer program that exists, even in principle, that can take elements as input and always correctly output whether or not the element is a member of the set. If one did have such a program, the program would be said to *decide* the set; and the set would be decidable since

there exists at least one program that can decide it. (We will talk more about this later.) Thus, when we say that the set of truths of Peano arithmetic is undecidable, we mean that there is no program that can be run on a computer that will take as input a sentence of arithmetic and output whether or not it is true. And when we say that the halting set—i.e., the set of pairs of programs and inputs for which the program halts on that input—is undecidable, we mean that there is no program Q that can be implemented on a computer that will take a program code P along with an input x together as input (i.e., the pair $\langle P, x \rangle$ is the input to the program Q) and output YES if program P halts on input x, and outputs NO if program P does not halt on input x.

What does the undecidability of these problems have to do with our limited ability in solving them? Since they are undecidable, no computing machine can solve them perfectly. And since we are just computing machines, we, too, cannot solve them perfectly. This argument depends on something called Church's Thesis, which states that if something is intuitively computable—i.e., if it seems in some sense as if one is able to compute it—then it is computable, in principle, by today's computers. In other words, it says that there is no other notion of computing something that we have not already captured in our understanding of computers. (We'll be discussing this at more depth at the appropriate time later.) With Church's Thesis in hand, it is argued that we can compute nothing a computer cannot also compute, and since a computer has limited ability with the Peano arithmetic and Halting problems, so must we.

Such an explanation, if true, utilizes in an essential way the logical limits of what is possible for finite agents (by which I will mean for now computationally bound agents, although I shall mean something more precise in Section 4.2.3), and thus $f(x)$ in the plot from earlier would be depicted as reaching the top horizontal line.

Vagueness and logical limits

Vagueness—the phenomenon that, roughly, natural language words have borderline regions (see Section 4.1)—is a phenomenon, not an activity, but phenomena and activities are sometimes related. The activity of human running is associated with the phenomenon that humans run only so and so fast. The activity of remembering via working memory is associated with the phenomenon that humans can only hold seven items. The activity of determining truths of Peano Arithmetic is associated with the phenomenon that humans are incapable of acting as perfect Peano Arithemetic truth-determiners. And the

activity of discovering if a program halts on a given input is associated with the phenomenon that humans are not very good halting-determiners, or "bug-checkers."

In this vein, the phenomenon of vagueness will be seen to be associated with two certain logical limits on the ability of finite, sufficiently powerful, rational agents:

1. their inability to determine whether or not a program halts on a given input, and
2. their inability to generally acquire algorithms—programs that eventually halt on every input—for their concepts.

Thus, in explaining vagueness in this chapter, I will be arguing for a picture of explanations of human behavior as in graph B of Figure 4.1, where vagueness is connected with an activity for which the human ability is bound by the ultimate limits on what is possible; mathematics shapes us. We will see that vagueness is not due to any particularly human weakness, but due to a weakness that any computationally bound agent possesses; even HAL from *2001: A Space Odyssey* and Data from *Star Trek* will probably experience vagueness.[1]

4.1 Vagueness, the phenomenon

One of the main points of this chapter is to show why one particular, very important, all-pervasive, long-known, and not well-understood phenomenon is explained primarily by the fact that the human brain is finite. That phenomenon is *vagueness*. Vagueness applies to *predicates* of natural language. A *predicate* is a word that applies to a subset of the set of all possible objects or events. For example, the predicate 'dog' applies to all possible dogs, the predicate 'bald' applies to all possible bald heads, and the predicate 'eat' applies to all possible occasions of eating. Nouns, adjectives and verbs are predicates, but many other kinds of words are not predicates, such as logical connectives like 'and', 'or', and 'therefore', or other "function" words (as they are called in linguistics) such as 'after', 'each', 'in', 'must', 'he', 'is', 'the', 'too', and 'what'.

A predicate is vague if it has borderline cases. Yul Brynner (the lead in *The King and I*) is definitely bald, I am (at the time of this writing) definitely not,

[1] Very early ideas of mine along the lines presented here appeared in Changizi (1995). A paper on my theory was presented at the 1998 vagueness conference in Bled, Slovenia (Changizi, 1999a), and at the 1998 Irish Conference on Formal Methods (Changizi, 1999b). The latter concentrates on logics and semantics of vagueness motivated by my theory of vagueness. The main ideas were published in Changizi (1999c).

and there are many people who seem to be neither. These people are in the "borderline region" of the predicate 'bald', and this phenomenon is central to vagueness. Nearly every predicate in natural language is vague. From 'person' and 'coercion' in ethics, 'object' and 'red' in physical science, 'dog' and 'male' in biology, to 'chair' and 'plaid' in interior decorating; vagueness is the rule not the exception. Pick any natural language predicate you like, and you will almost surely be able to concoct a case—perhaps an imaginary case—where it is unclear to you whether or not the case falls under the predicate. Take the predicate 'book', for example. The object from which you are reading this is definitely a book, your light source is definitely not a book. Is a pamphlet a book? If you dipped this book in acid and burned off all the ink, would it still be a book? If I write this book in tiny script on the back of a turtle, is the turtle's back a book? We have no idea how to answer such questions. The fact that such questions appear to have no determinate answer is roughly what we mean when we say that 'book' is vague.

And, by 'vague' we do *not* include conundrums such as whether redness is or is not bald; the word 'bald' does not apply within the domain of colors, and so some might say that neither 'bald' nor 'not bald' "nicely applies" to redness. If 'bald' is vague—and it is—it is *not* because of the colors that it is vague. In cases of vagueness, the inability to nicely apply the word and its negation is *not* due to the word not applying to objects in that domain. 'bald' is vague because there are heads which do not nicely fall under 'bald' or 'not bald', even though there are lots of other heads which *do* fall nicely under one or the other of 'bald' or 'not bald'.

For the uninitiated, why should we care about vagueness? There are a number of reasons.

Most importantly for my interests here, the fact that natural language is vague needs to be *explained*. Why are natural language users unable to draw a single sharp line between definitely bald and definitely not bald? Why do natural language users seem to find cases that are borderline bald? Why cannot natural language users determine the boundaries of the borderline region? Is it possible to have a non-vague language? If so, under what conditions?

The most central phenomenon of vagueness is the borderline region, where for a vague predicate P there are objects which are not clearly classifiable as either P or 'not P'. This borderline region phenomenon seems to stab at the very heart of the idea that our concepts divide the world into two parts: those objects to which the predicate applies—the predicate's *extension*—and those objects to which the predicate does not apply—the complement of the pred-

icate's extension. Although some would like to argue that the true meanings of P and 'not P' are as in classical two-valued logic where the extension of 'not P' is the complement of the extension of P (such a semantics is called *determinate*), the concepts as we actually use them do not *seem* to be like this, lest there be no phenomenon of vagueness at all. What are our concepts like, if they are not as in classical logic? If we are not carving up the world a la classical logic, how *do* we carve up the world?

A third reason the study of vagueness is important is related to the previous one, but now the concern is one of logic rather than the question of what is a concept. The issue is that classical two-valued logic seems to be at stake, for classical two-valued logic says (i) that the meaning of a predicate is a precise set and (ii) that the meaning of the negation of the predicate is the complement of that precise set. Prima facie, (i) and (ii) do not seem to be consistent with the existence of the phenomenon of vagueness. This threatens the usefulness of over seventy years of technical work in classical logic. What *is* the proper model of our interpretations and use of natural language predicates? And what *is* the logic of our inferences if we are not "classical logic machines"? Given that we humans are the paradigm example of rationality, it would serve us well to understand the logic we engage in for the purposes of (a) better understanding what is rationality and (b) building artificial machines that can mimic the intelligence and humanity of our inferences and utterances—how better to do this than to first understand how we do this?

Now that we care about vagueness, it is necessary to become clearer about what exactly the phenomenon of vagueness is? What is it that is in need of explanation? Some names of phenomena comprising the phenomenon of vagueness are 'borderline region', 'higher-order vagueness', 'sorites paradox', 'ineliminability', and 'essentialness'; the first two are central. All things equal, a theory that satisfies more of the phenomena is more favorable. But what are these phenomena? It is dangerous to attempt to precisely and formally define them since we have no clear pre-theoretic agreement on what exactly are the data, and any such definition is likely to be theory-laden to some extent. Accordingly I want to remain open-minded. I will give the rough idea for each phenomenon with the understanding that I am in no way defining what it is exactly. The best I hope for is an independently motivated theory that results in certain plausible phenomena that seem to match closely with the rough definitions of those named above. On to the phenomena.

What is the borderline region?

The *borderline region* phenomenon is roughly the phenomenon that for a vague predicate P we find ourselves with objects for which P neither clearly applies nor clearly does not apply; these objects are in the borderline region. Or, an object is borderline if it does not fit neatly into just one category. Alternatively, an object is borderline P if when we are given the choice "Which is it, P or not, and not both?" we do not know quite how to respond, and our non-response is seemingly not because we simply do not know, but because it seems fantastic to suppose there is exactly one correct response; this is partially what distinguishes vagueness from other sorts of unknowabilities. I say "seemingly" above because otherwise I exclude the possibility of an epistemic determinist theory of vagueness being correct, i.e., a theory where every object either falls into the extension of the predicate or the complement of the extension, and vagueness is due to our problems in seeing the boundary. The borderline region is also connected with the phenomenon that we are incapable of drawing a single sharp line distinguishing things P from things not P, and more than this, it is that any line drawn would seem ad hoc, arbitrary and wrong. Sometimes the borderline region is defined more epistemically as that region for which knowledge concerning membership in P is unattainable. The phenomenon is probably best communicated by example: wolves are borderline dog, violet is borderline blue, and so on.

What is higher-order vagueness?

Higher-order vagueness, second-order vagueness in particular, is the phenomenon that we find ourselves incapable of determining boundaries of the borderline region. Alternatively, imagine pulling hairs out of the head of a man who is definitely not bald. Second-order vagueness is exemplified by the fact that we do not find ourselves pulling out a single hair for which we are able to determine that the man suddenly becomes borderline bald. We find objects, or states of this man's head, which are borderline borderline bald. More explicitly epistemically, knowledge of the boundaries—if there are any—of the borderline region is unattainable. Higher-order vagueness, more generally, is the phenomenon that we find ourselves incapable of determining any semantically distinct boundaries at all between definitely bald and definitely not bald.

The "no boundaries" dogma

On these first, most central, two phenomena of vagueness—the borderline region and higher-order vagueness—what needs to be explained is not necessarily the real existence of a borderline region and higher-order vagueness, but rather why there seems to be a borderline region and higher-order borderline regions. The borderline region could be defined as the region which is semantically distinct from the definite regions, but a less theory-laden and more scientific approach would be to say that any adequate theory of vagueness must explain why there is a region which *seems* to be semantically distinct; this still leaves it open as to whether there is semantic indeterminacy. Also, sometimes (very often, in fact) higher-order vagueness is taken to be the phenomenon that there is no sharp semantic boundary between the definite regions and the borderline region, and even more radically it is sometimes taken that there are no semantic lines at all to be drawn, no matter how small the semantic difference or how impossible it is to see the lines. To have any lines posited by one's theory is, so it is reiterated, "not to take vagueness seriously." This seems straightforwardly bad science. There are our experiences of vagueness, and there are our theories about them; only the former can possibly count as data. Theories may well explain the data by positing that there is a semantically distinct borderline region, or that there are no sharp semantic lines, and perhaps such theories can in the end be victorious over epistemic theories like Sorensen's (1988), Williamson's (1994), a version of Koons (1994) and mine. What one cannot do is criticize epistemic theories on the basis that they do not posit a semantically distinct borderline region, or that they do posit sharp lines, for to do so is effectively to criticize epistemic theories for not being non-epistemic theories. In fact, if we are to be biased by our metaphysical prejudices, we have a long history of success in the drawing-sharp-lines business (e.g., classical logic) and should therefore be biased toward the existence of sharp lines.

Not only are epistemic theories unfairly criticized, so are many-valued theories (truth-valuational or probabilistic). I have never understood the criticism of fuzzy logic, for example, that it does not adequately handle higher-order vagueness. The charge is that there is a sharp, and let me suppose knowable, line between 'a is P' having truth value 0, where a is definitely not P, and having truth value > 0 (and < 1), where a is borderline P. This is criticized as being fantastic just as is a semantics with a sharp line between 'a is P' having truth value 0 and having, say, indeterminate truth value. Furthermore, it is argued, fuzzy logic is full of sharp lines everywhere, and this is just crazy.

Edgington (1992), who proposes a probabilistic theory, nicely states the feeling I have always had on this when she writes,

> A twofold classification into the true and the false is inadequate where things hover on the edge of this great divide. A threefold classification is little improvement, I agree, with things hovering on the edge of the still-substantial divide between the true and the indeterminate. But in a manyfold classification, the difference between close degrees of truth, *including the difference between clear truth and its near neighbors,* is (almost always) insignificant, and a good theory must preserve this fact—must not deliver significantly different verdicts for insignificantly different cases. It does not matter if things still hover on the edges of inconsequential divides. (Edgington, 1993, p. 198.)

The motto "no boundaries" has become a dogma when theories postulating inconsequential divides are dismissed out of hand.

Why does so much of the vagueness community strongly believe that there are no sharp lines? I understand the intuition of there being no semantic lines of any kind drawn since we "feel" like there are no such lines. But we also feel a lot of ways counter to our current physics, say, but we can explain why we feel that way, and we allow ourselves to conclude that it is just a feeling. For example, we do not feel like we are spinning around as the Earth rotates, but we know now that we are, and we can explain why we feel the way we do. What is it about vagueness that so many—indeed most—in the field are so bent on not only explaining the feeling, but making it a real part of the semantics?

And it is not even the case that the "no boundaries" dogma has proved fruitful. I do not believe there is even one extant theory satisfying the "no boundaries" constraint that is remotely adequate as a description, much less an explanation. One wonders whether the dogma is really just a skeptical "no possible theory of vagueness" dogma, given that the "no boundaries" constraint seems simply impossible to satisfy in a coherent fashion. Horgan (1994) goes so far as arguing that one should accept the incoherence of the "no boundaries" motto—that the incoherence is not vicious. Not all "no boundaries" theorists are so ready to take this route of Horgan; they hold out hope, presumably, for some theory satisfying their constraint. For example, Sainsbury (1990) proposes an attempted such theory that Edgington (1993) shows does have boundaries.

The sorites paradox

Moving on, there is a third phenomenon linked to vagueness: the *sorites paradox*. Its standard form is exemplified by the following two-premise argument

and conclusion: (i) 1 grain of sand cannot make a heap, (ii) for all n, if n grains of sand cannot make a heap, then $n+1$ grains of sand cannot make a heap, (iii) there are no heaps of sand. (i) is obviously true.[2] (ii) is very compelling, since to deny it means that there is some n such that n grains of sand cannot make a heap but $n+1$ grains can make a heap—that there is a to-the-grain distinction between being a heap and not—and this seems fantastic. (i) and (ii), though, imply (iii), which is obviously false. The sorites paradox is part of the phenomenon of vagueness in that it may be built using any vague predicate; and it may not be built with non-vague predicates.

There are two related constraints on a theory of vagueness. The first is that it locate the fault in the sorites argument, and do so without having to make fantastic claims. I do not mean to imply that the classical negation of the induction step cannot possibly be the solution to the sorites paradox. Epistemic, determinist theories do exactly this, but it is incumbent upon the theory to say why it is not so fantastic; for example, that we cannot ever determine or know the boundary, and this is why the suggestion that there is a boundary seems incredible. The second constraint is that a theory's post-theoretic notion of the phenomenon of vagueness should be such that a sorites argument built around a predicate displaying the phenomenon is paradoxical; i.e., denying the induction step must seem paradoxical. If the argument loses its paradoxical aspect, then the phenomenon claimed to be vagueness has a lesser claim to vagueness. For example, if we cannot ever determine or know the boundary but still believe quite reasonably that there is one, then there is nothing paradoxical in the sorites argument since the induction step can be (classically) denied readily without intuitive difficulties.

Ineliminability and essentialness

The final two phenomena are less central to vagueness.

The first of these is *ineliminability*: it is often felt that vagueness is not something we can simply eliminate from natural language. For example, it is sometimes said that any attempt to eliminate vagueness through precisification (i.e., making predicates precise) would, at best, radically undermine the meanings of natural language concepts. Also, restricting oneself to some delimited context is also thought to be unhelpful in eliminating vagueness—

[2] Although some have sought to save us from paradox by denying the base case. Unger (1979) and Wheeler (1979) deny that there are non-heaps by denying that there is a concept heapness at all.

vagueness occurs *within* contexts. The Undecidability Theory—i.e., my theory of vagueness—explains and accommodates a variety of ways in which vagueness is ineliminable (Subsection 4.3.5). I am less confident about this phenomenon being a necessary constraint on a theory of vagueness, and accordingly I do not criticize other theories on the basis that they do not explain or accommodate ineliminability. I do think that some degree of ineliminability must be addressed, however, lest we be left to wonder why we have not cured ourselves of vagueness.

Or a theory may alleviate this worry just mentioned by giving a good reason for why we should not want to "cure ourselves" of vagueness. This is the final phenomenon of vagueness: its seemingly *essential* nature, in the sense that it is often felt that even if we could eliminate vagueness, we would not want to because it fills an important and essential role for us. Perhaps it quickens our communication, or makes our utterances more informative, or gives us more power to correctly carve up the world, etc. There need not be any such reason, but if a theory has no reason, then it ought to say why vagueness is ineliminable lest, as mentioned above, we wonder why we have not cured ourselves long ago.

In addition to our shared pretheoretic notions which serve as our guide to saying roughly what is vagueness, we have stronger shared intuitions concerning what predicates are vague. If a theory says that a vague predicate is not vague, or that a non-vague predicate is vague, then this counts against the theory.

Explanatoriness

The phenomena from the previous subsections need to be explained, not just modeled. That is, a logic devised just to accommodate these phenomena is not sufficient to have the status of an explanatory theory of vagueness. I want to know why there is vagueness, not just how to describe it. In this section I mention a handful of accounts of vagueness that are not explanatory.

Take many-valued theories such as multiple valued logic—fuzzy logic (Zadeh, 1965) in particular—and probabilistic degrees such as Edgington's (1992). Multiple valued logics allow sentences to have truth values besides simply true and false, or 1 and 0. They allow truth values in between true and false, e.g., a truth value of 1/2, say. For vague predicates R there will be objects c falling in the borderline region of R, and the sentence 'c is R' accordingly has truth value in between 0 and 1. Probabilistic models of vagueness, on the other

hand, accommodate vagueness by saying that the probability that 'c is R' is true is somewhere in between 0 and 1. Probabilistic degrees have superiorities over many-valued logics with respect to modeling natural language (Edgington, 1992), many-valued logics whose deficiencies are well catalogued [for starters see Williamson (1994, Chapter 4), and Chierchia and McConnell-Ginet (1990, pp. 389 ff.)]. One problem with many-valued descriptions of vagueness is that it is not clear that they describe vagueness. To be sure, many-values are a good description in many cases: many vague properties come in degrees, like baldness or redness. But even some non-vague mathematical concepts have been shown to come in degrees to subjects, like 'even' (Armstrong et al., 1983), so degrees are not uniquely connected with vagueness. My principal problem with many-valued theories is that even if we agree that they provide a satisfactory model, or description, of natural language semantics—and allow useful applications in computer science and engineering—they do not make for an explanation for vagueness. Many-valued theories are silent on explanatory questions, and, in fairness, description and not explanation is their aim. They do not tell us why natural language is vague, and they do not even tell us why natural language predicates tend to come in degrees.

Consider briefly supervaluations (see Fine (1975) and Kamp (1975); see also Williamson (1994) for some history), which is the model of vagueness wherein, roughly, a sentence 'c is R' is "super-true," or definitely true, if it comes out true on every precisification of the borderline region of R, "super-false" if it comes out false on every precisification, and borderline, or indeterminate, truth value otherwise. Despite its problems concerning whether it is an adequate description of higher-order vagueness and more generally natural language, my problem is that I want to know why the non-linguistic facts do not determine a single precise extension for natural language predicates. What is it about us or the world that makes meanings incomplete? As in many-valued theories, description is the main task, not explanation; supervaluation aims to be a logic of vagueness, not a reason for why vagueness exists.

Sorensen (1988, pp. 199–216) puts forth an epistemic, determinist account in which vagueness is due to ignorance of the sharp line separating the positive and negative extension, but his theory is not aimed at explanation. Vagueness is, he argues, identical to a phenomenon he calls blurriness. Let $N_1, \ldots, N_{100}$ be mystery natural numbers, and consider the mystery sentences 'N_i is even' for every i from 1 to 100. Now say that an integer is *miny* if and only if it is less than or equal to the number of true mystery sentences. 'miny' is blurry. 0 is definitely miny. 1 is almost certainly miny (i.e., its probability of being miny,

presuming some natural assumptions, is $1 - (.5)^{100} \approx 1$), 101 is certainly not miny, and somewhere in between things are difficult to say. Sorensen pushes the idea that vague predicates possess the phenomena they do for the same reasons 'miny' possesses the phenomenon of blurriness; i.e., he pushes the idea that vagueness is blurriness. Even if I were to agree that blurriness is a perfect description of the phenomenon of vagueness, I still would want to understand why natural language predicates are blurry, i.e., why predicates are built as if from mystery numbers, and so on.

My point in this subsection is not to seriously entertain these theories on which I have touched, but to emphasize that much of the work on vagueness has concentrated on describing vagueness rather than explaining it; Hyde's (1997) defense of a "subvaluational" logic and Putnam's (1983) intuitionistic solution are two others.

4.2 Unseeable holes in our concepts

In this section I present the guts of my theory of why there is vagueness. I will become precise about what is a "finite, sufficiently powerful, rational agent" and I will argue that any such agent will have concepts with "unseeable holes" in them. What this all has to do with vagueness will not be discussed until Section 4.3. I believe it is useful to separate out this "unseeable holes" thesis from the explanation of vagueness, because, for me at least, that there are unseeable holes in our concepts—and also for any rational, computationally bound agent—is just as interesting and important as that there is vagueness.

We will see that, in my view, vagueness is definitely *not* a good thing for us, in the sense that it is not as if language has evolved to be vague because it was *independently* useful. Vagueness is something we are stuck with because we are rational, finite entities. If you were a computationally bound, rational alien agent given the task of figuring out what our natural language predicates mean, you would very probably end up with vagueness. I will explain how vagueness could, in principle, be avoided: it would require that we either have inaccessible meanings for our predicates, or that we radically confine our possible meanings for predicates to a very reduced set. Given that we do *not* want inaccessible meanings and *do* want to have a rich choice of meanings for predicates, vagueness is thrust upon us. Vagueness is a cost, but it is worth it because of the benefits it brings. In *this* sense, vagueness is good for you, since without vagueness you would be worse off.

My theory for why there is vagueness is simple, and in order that you not

miss the overall point in all the details (not technical details, just details), I first give you a quick introduction to my theory.

4.2.1 Brief introductions to the theory of vagueness

Very brief introduction

Here is the theory: When you or I judge whether or not a predicate P applies to an object a, we are running a program in the head for P on input a. This program for determining the meaning of predicate P we may call C_P. For every one of our natural language predicates P, we have a program for it, C_P, in the head. The job of program C_P is to output YES when input with objects that are P, and NO when input with objects that are not P.

The central problem, though, is that we are unable to always have these programs give an answer to every input; these programs will often take an input, but never respond with an answer of YES or NO. Instead of responding, the program will just keep running on and on, until eventually you must give up on it and conclude that the object does not seem to clearly fit under either P or 'not P'.

The reason we are susceptible to this difficulty is that it is a general difficulty for *any* finite-brained entity. And the reason this is true is because the problem of determining if a program halts on every input is undecidable. Thus, generally speaking, there will be, for each predicate P, objects x for which your program in the head for P, C_P, does not ever halt and output YES or NO. These objects will appear to be in the borderline region of P. In sum, we have borderline regions because we are not computationally powerful enough—no finite-brained agent is—to make sure our programs for our predicates always halt. We have holes in our concepts.

The other major feature of vagueness is that it is not generally possible to see the boundary between the definite regions and the borderline regions; this is called higher-order vagueness. This falls out easily from the above, and goes as follows. The borderline region for a predicate P is the set of all x such that the program $C_P(x)$ does not halt. How accessible is this set? For computational agents, the answer is, "not very." To determine that an object a is borderline P, one must determine that $C_P(a)$ does not halt. This, though, is the halting problem which we discussed a little bit in the introduction to this chapter. We mentioned there that the halting problem is undecidable, and so it is not generally possible for computational entities to solve. In sum, then, you will not generally be able to see the boundary of the borderline region because

it is too hard for you to determine which things are and are not borderline P—too hard, in fact, for any finite-brained entity. Not only, then, do we have holes in our concepts, we have *unseeable* holes!

This is the explanation of vagueness, as simple as I can make it. There are more intricacies to the story. For example, being finite-brained is not completely sufficient for the conclusion; one actually needs finite-brained and rational. But it gets across the main idea, which is, I think, embarrassingly simple.

A less brief introduction

I now give a little less brief primer on my theory of vagueness. The "vagueness is good for you" arguments will still not appear in this introduction. I will take *you* to be my example natural language user.

There are three hypotheses.

(1) The first hypothesis is the *Church-Bound Hypothesis*, and it states that you can compute no more and no less than what a computer can compute.

(2) The second hypothesis is the *Programs-in-Head Hypothesis*, and it states that what natural language predicates extensionally mean to you is determined by programs in your head. For example, an object is a dog to you if and only if your program in the head for 'dog' outputs YES when the (name of the) object is input into the program. It is much less plausible that many scientific, mathematical and technical predicates get their meaning to you via programs in the head, and this difference is what prevents my theory from concluding that such predicates, many which are not vague, are vague.

(3) The third and last hypothesis is the *Any-Algorithm Hypothesis*, and it states that you allow yourself the choice of any algorithm when choosing programs in the head for determining your natural language predicate meanings. (An *algorithm* is a program that halts on every input; programs sometimes do not halt on some inputs.)

Informally and crudely, the three hypotheses are that (1) you are a computer, (2) you have programs in the head determining what natural language predicates mean to you, and (3) you allow yourself the fullest range of possible meanings for natural language predicates.

If these three hypotheses are true, what follows? The Programs-in-Head Hypothesis says you choose programs to determine your meanings of natural language predicates. The Any-Algorithm Hypothesis says that the set of programs from which you are choosing is a superset of the set of all algorithms. But here is the catch: one of the basic undecidability results implies that any

such set of programs is undecidable. (A set is *decidable* if and only if there is program that outputs YES whenever input with an object from the set and NO whenever input with an object not in the set.) Because of the Church-Bound Hypothesis, this undecidability is a difficulty for you: in choosing from the set of programs you cannot always obtain algorithms. In fact, because picking algorithms is computationally more difficult than picking non-algorithms, you will "usually" pick non-algorithms; "most" of your programs determining the meanings of natural language predicates will not be algorithms. So, in an attempt to acquire a meaning for 'dog' via a program in the head that outputs YES when something is a dog to you and NO when something is not a dog to you, *there will be objects on which your program does not halt at all.* This does not mean that you will actually run into an infinite loop; it just means that you will eventually give up when running the program on such inputs.

What does this have to do with vagueness? Consider the set of objects for which the program for 'dog' does not halt. For any object in this set the program will neither say YES nor NO; the object will neither be a dog to you nor not a dog to you. My first theoretical claim is that this is the set of borderline cases for the predicate.

What about higher-order vagueness, the phenomenon that the boundaries of the borderline region are vague? Consider trying to determine exactly which objects are part of the borderline region. To determine that some object is in the borderline region of 'dog' requires that you determine that your program for 'dog' does not halt on that object. But now we have another catch: possibly the most well-known undecidability result is the "halting problem," which says that whether or not a program will halt on a given input is undecidable. This undecidability is a difficulty for you because of the Church-Bound Hypothesis: objects in the borderline region are generally difficult to determine as such, and where the boundaries of the borderline region are is not generally possible for you to determine. Imagine moving from 'dog' cases to borderline cases. Your program for 'dog' will no longer output YES, and will, in fact, never halt; but you will not know it will never halt. You will be unable to see the boundary. My second theoretical claim is that this inability is the phenomenon of higher-order vagueness. Here is a simple representation of the behavior of your program for 'dog', where 'Y' denotes YES, 'N' denotes NO, and '↑' denotes "does not halt".

Y Y Y Y Y Y ↑ ↑ ↑ ↑ ↑ ↑ ↑ ↑ ↑ ↑ ↑ N N N N N N N

{ - - 'dog' - - }{- - borderline - - }{- - 'not dog' - - }

So, the three hypotheses entail that for "most" of your natural language predicate meanings there are objects for which your program for that predicate does not halt. Add to this my two theoretical claims just mentioned and it follows that "most" natural language predicates are vague. That is the Undecidability Theory of Vagueness in a nutshell. Now to develop and defend it in more detail. The remainder of this section presents the Undecidability Theory and Section 4.3 discusses how it explains vagueness.

4.2.2 Theory

In this subsection I discuss the three hypotheses comprising the Undecidability Theory of Vagueness and show how they lead to what I call the "Thesis," which is central to the Undecidability Theory of Vagueness's characterization of vagueness. Here is the Thesis, followed by an explanation of the terminology used.

Thesis: For "most" natural language predicates P

1. your interpretation of P is determined by a program in the head that is capable of semideciding but not deciding it,
2. your interpretation of 'not P' is determined by a program in your head that is capable of semideciding but not deciding it, and
3. there are objects neither in your interpretation of P nor in your interpretation of 'not P'.

I will explain the scare quotes around 'most' later. By a "program in the head" I mean the method used by you to determine whether or not a given object is in your interpretation of P. One may usefully and informally think of the program as your intension of the predicate P. The "interpretation of P ('not P') determined by a program" is the set of objects on which the program in the head for P ('not P') outputs YES. A set is *decidable* by a program C if and only if for all x, C on input x outputs YES if x is in the set, and outputs NO otherwise. A set is *semidecidable* by a program C if and only if for all x, C on input x outputs YES exactly when it is in the set; if x is not in the set then C may well not halt at all, though. Do not confuse the notion of a set being semidecidable but not decidable by the program for it with the notion of an underdefined or incompletely specified set. The former, which appears in my theory, is a precise set that happens to be computationally difficult for the program to identify nonmembers, whereas the latter is not a well-defined set at all. Also, do not confuse a set's being semidecidable but not decidable by a program C with a set's being semidecidable but not decidable simpliciter. The latter means the set is computationally complex (in fact, it means it is

recursively enumerable but not recursive), but the former, which appears in my theory, only means that the set is complex as far as the program C is concerned; C is unable to decide it, even though it may well be decidable.

There is a simpler, equivalent way of stating the Thesis, one I implicitly used in the brief introduction to the theory. I had written there about a single program in the head, call it $C_{P/nonP}$, doing the work for both a predicate and its natural language negation: the interpretation of P was the set of objects on which $C_{P/nonP}$ outputs YES, and the interpretation of 'not P' was the set of objects on which the *same* program outputs NO. In the Thesis and throughout the remainder of the section the single program is treated as two distinct programs: one program, C_P, for P; and another, C_{nonP}, for 'not P'. The interpretation of P is the set of objects on which C_P outputs YES, and the interpretation of 'not P' is the set of objects on which C_{nonP} outputs YES. Each of these two programs can output only a YES, if they halt at all; they do not ever output NO. Realize that there is no difference in these approaches: running $C_{P/nonP}$ on an input is equivalent to simultaneously running both C_P and C_{nonP} on the input and seeing who halts first (if any); if C_P halts first then $C_{P/nonP}$ would have output YES, but if C_{nonP} halts first then $C_{P/nonP}$ would have output NO. In terms of a single program, the Thesis would be the following: *for "most" natural language predicates P there are objects for which $C_{P/nonP}$ does not halt.* Although this is simpler than the statement at the start of this section, the two-programs version helps to clearly identify distinct aspects of the Thesis.

In the subsections that follow I indulge in a sort of fantasy. I imagine that you are a rational, computationally bound agent who has entered into our culture. Your task is to learn language for the first time and to determine what our natural language predicates mean. We will see that the Thesis is very likely to be true of any such agent. Such an agent will likely choose to have vagueness because its costs are less than the costs of avoiding it. I will also show that it is plausible that the Thesis does, in reality, apply to you.

As part of the fantasy I suppose that the true extensions (as opposed to your interpretations) of natural language predicates are determinate; that is, every object is either in the extension of the predicate or its complement. [For defenses of a determinate semantics within the vagueness literature see Campbell (1974), Cargile (1979), Sorensen (1988, 1994) and Williamson (1994).] I use 'extension' to refer to the true meaning of a predicate, and 'interpretation' to refer to whatever you mean by the predicate. All capital letters will be used to signify the extension of a predicate; e.g., 'bald' has the set BALD as its ex-

tension, and 'not bald' has the complement of BALD as its extension. In trying to figure out your interpretations for the language in the fantasy scenario, I suppose that you are presented with examples, somehow, from the true extensions. What I wish to communicate by this is that even if the true semantics of natural language predicates were determinate (via, perhaps, a semantic externalist account), you would still very likely end up with interpretations as specified in the Thesis (and thereby end up with vagueness). Thus, while standard classical two-valued logic would be a correct model of natural language true semantics, we will see that it is not a correct model of the way we actually interpret natural language predicates. On the question what *really* is the true semantics of natural language predicates my theory can remain agnostic.

4.2.3 Church-bound

The Undecidability Theory of Vagueness applies only to those agents that are computationally bound. Specifically, it applies only to those agents that are "finite" and "sufficiently powerful."

By a *finite agent* I mean an agent (i) that has a finite but possibly unbounded memory, (ii) that has an upper bound on the speed at which it can compute, (iii) whose primitive computations are simple (e.g., adding 1 to a number), and (iv) who cannot (or at least does not) utilize in its computing any aspects of the universe allowing it to achieve supertasks (i.e., to achieve infinitely many steps in a finite period of time finite "brain"). To defend my use of the term 'finite' in this way, I informally rephrase these requirements as follows: by a finite agent I mean an agent that is finite in (i) memory, (ii) speed, (iii) degree of complexity of primitive computations and (iv) resourcefulness in utilizing nature to achieve supertasks.

Without (i), an agent could have infinitely large look-up tables in the head. Such an agent could compute any function at all by simply storing the entire function (i.e., storing every pair $\langle x, f(x)\rangle$) in its head, so long as the function has domain and range with cardinality no greater than the cardinality of the look-up table. The agent could merely check his look-up table to see what $f(x)$ is. Without (ii), an agent could compute the first step of a computation in half a second, the next in a fourth, the next in an eighth, etc., thereby computing infinitely many steps in one second (such an agent is called a Plato Machine). Without (iii), an agent may have primitive computations that are themselves as mathematically computationally difficult as one pleases; of course, from such an agent's point of view these computations would *seem* utterly simple, requir-

ing only the least amount of "thinking" to compute (see, e.g., Copeland 1998). Finally, without (iv), it is logically possible that the laws of physics might make it possible to compute supertasks (despite (ii)) (see Earman and Norton (1993, 1996) and Hogarth (1994)). Being a finite agent severely constrains what an agent can compute, as I now describe.

We have an informal, pre-theoretic notion of what it is to compute something. Such an intuitive notion of a computation typically connotes that there be only a finite number of steps involved, that the amount of memory (and scratch paper) required also be finite, that each primitive step be relatively simple (enough to understand), and that one cannot engage in supertasks. That is, the intuitive notion of a computation exactly corresponds to those computations a finite agent can compute. We are inclined to say that a function f from the natural numbers to the natural numbers is intuitively computable if, for each natural number n, $f(n)$ is intuitively computable.

The Turing machine formalism provides an abstract, precise notion of what a computation is and leads to a particular set of functions on the natural numbers as the set of Turing-computable functions. Any computation a modern computer can do, a Turing machine can, in principle, do also; and vice versa. There is the well known hypothesis that the set of functions that are intuitively computable just is the set of Turing-computable functions; this hypothesis is referred to as *Church's Thesis* (or the Church-Turing Thesis).

The hypothesis is not a mathematical assertion; it refers to our intuitions and it does not make sense to ask whether it has been mathematically proven. Nearly everyone believes in Church's Thesis, though, as do I. One reason for this is that no one has yet provided a convincing case of an intuitively computable function that is not Turing-computable; the longer we go without such a case being found, the higher our inductive probability goes toward one that the sets are identical. A second, more slippery, reason nearly everyone believes in Church's Thesis is that half a dozen very different formalizations of computation have been concocted by different people and each leads to precisely the same set of computable functions.

If a finite agent can compute a function on the natural numbers, then the function must be intuitively computable. But then by Church's Thesis that function must be Turing-computable. Therefore, the only functions on the natural numbers a finite agent can possibly compute are those that are Turing-computable.

But any finite agent worth considering carries out computations on objects besides the natural numbers. What constraints are these computations under?

Although there are objects besides natural numbers that are objects of such computations (i.e., such an agent computes functions over objects besides the natural numbers), we can encode all of the objects the finite agent can grasp—including natural numbers—onto the natural numbers. Supposing each different possible state of the finite agent's mind is finitely describable, the set of all such finite descriptions can be bijectively encoded onto the natural numbers (hopefully in an intuitively computable fashion). (Such an encoding is *bijective* if and only if each object gets assigned to a unique natural number and each natural number is used in the encoding.) '4' may now be the code for mental state p_1 which holds the information of the finite agent's mother, '37' the code for mental state p_2 which holds the information of a particular fist fight the finite agent once witnessed, '18' the code for mental state p_3 which holds the information of the feeling of love-at-first-sight the finite agent felt upon meeting its spouse, '103' the code for the mental state p_4 which holds the information of the natural number 5, '1000' for the mental state p_5 which holds the information of the finite agent's favorite shade of blue, etc. Intuitively, every possible dog, every possible shade of color, every possible action, etc., is given its own natural number. With such an encoding, all of the finite agent's computations may be interpreted as computations on the natural numbers, and *the finite agent's computational power is constrained in such a way that it can compute only the Turing-computable functions on this set of codings.*

One should not find this too fantastic, given that the same sort of thing is true about every computer. In a physical computer, as opposed to an abstract model, there are no numbers actually input into the machine nor output from the machine; numbers are abstract objects. Rather, an input or output is some physical state and it encodes certain information. Each physical state is finitely describable and can be coded onto the natural numbers. '4' may be the code for physical state p_1 which holds the information of a black and white picture of a rooster, '37' may be the code for physical state p_2 which holds the information of natural number 5, '18' may be the code for physical state p_3 which holds the information of the sentence "Press any key to continue," etc. It is only through such means that one can meaningfully say that computers are subject to the same ultimate computational constraints as Turing machines, and it is also only through such means that one can meaningfully say that a finite agent is subject to the same ultimate computational constraints as Turing machines.

Worries over which coding is being employed for the finite agent are sometimes raised. For example, what if the coding makes intuitively uncomputable problems computable by having a non-intuitively computable coding? Or, is

there a privileged coding and, if so, what determines it? I wish to sidestep all such issues. To whatever extent these are legitimate worries, they are worries for anyone claiming that even computers are bound by Church's Thesis. This latter claim is uncontroversial, however, and so I am under no special obligation to explain or address issues of coding with respect to finite agents.

One might complain that the universe has uncountably many possible objects, and so no bijection is possible onto the natural numbers. Supposing for the moment that there are indeed uncountably many possible objects, I only care about what possible objects the finite agent can hold before its mind. Since it is finite, it can only entertain countably many possible objects. Its universe is countable, regardless of the cardinality of the real universe. This brings in its own trouble: if the universe is uncountable and the finite agent's universe countable, is not it going to have a false model of the world? The Downward Lowenheim-Skolem Theorem can help to alleviate this worry to an extent: as long as the finite agent notices only first-order properties of the universe, it is possible for its model to be such that the set of all truths is the same as God's (whose model is the true uncountable one). Should we, however, believe that it is confined to first-order properties? Perhaps, perhaps not; there are many things that can be said on this issue, but I have no need to pursue them here since nothing hinges on the agent's model being true.

Thus, I am confining discussion to finite agents, which means that I am confining discussion to agents capable of computing *only* the Turing-computable.

By "sufficiently powerful" I mean that the finite agent is capable of computing *at least* the Turing-computable.

Together, "finite" and "sufficiently powerful" imply that the computational powers of the agents I wish to discuss are bound by Church's Thesis and only bound by Church's Thesis. I sometimes say "Church-bound" instead of "finite and sufficiently powerful." I record this as the Church-Bound Constraint.

Church-Bound Constraint: *The agent can compute any function (over natural numbers coding mental states, which in turn represent objects in the world) so long as it is Turing-computable.*

Related to this constraint is the Church-Bound Hypothesis, which states that you are under the Church-Bound Constraint. The Church-Bound Hypothesis is, by assumption, true of the fantasy you. Is it true of the real you? Yes, and here is why. It is plausible that you are finite in the four senses discussed above (although see Penrose, 1994) and so cannot compute the Turing-uncomputable. Furthermore, you are, in principle, able (given enough time and scratch paper)

to compute any Turing-computable function. We know this because any of us can easily mimic the simple actions of a Turing machine as long as we please.

4.2.4 Programs in the head

Suppose that you, in the fantasy, are exposed to enough examples of things you have reason to believe are in the true extension of 'bald' (BALD) and others that are not that you acquire an educated guess as to what BALD is. Your guess determines some set as your "shot" at BALD, and this is your interpretation of 'bald'. In what can such a guess consist? There are infinitely many (possible) objects in your universe, infinitely many of them are bald and infinitely many are not. You are not, then, able to simply guess what the extension is, for you cannot store the extension since you are finite.

You must employ some sort of intension. You need to find some finite description of the set that determines your interpretation of 'bald' and your educated guess at BALD, and some finite description of 'not bald'. Recalling that these sets may be considered to be sets of natural numbers, one may wonder whether your interpretation of 'bald' can be described "in your head" as, say, a first-order sentence in the language of arithmetic (such sets are called *arithmetically definable*). For example, you may interpret 'bald' to be the set $\{n \mid \exists x \forall y\ R(n,x,y)\}$, where $R(n,x,y)$ is some recursive formula without quantifiers. The problem with this is that although it is indeed a finite description, the set is not recursively enumerable and since you are Church-bound it is generally too difficult for you to handle. (A set is *recursive* if and only if it is decidable by some program. A formula is recursive if and only if the set of objects satisfying it is recursive. A set is *recursively enumerable* if and only if it is semidecidable by some program.)

The same is true for any arithmetically definable set... except those that are recursively enumerable. For a recursively enumerable set it is possible for you to have a program in the head that says YES when and only when presented with objects in the set (although the program may never halt at all when presented with objects not in the set), but sets any more computationally difficult than recursively enumerable are beyond your reach. A program in your head, then, is what you must be employing to determine your interpretation of 'bald' if you wish to have an interpretation that is accessible to you. Your interpretation would then be the set of objects for which the program for 'bald' outputs YES, and this is recursively enumerable. This motivates the first rationality principle.

Principle of Program-Favoring: *Without good reason to the contrary, you should assume that the extension of natural language predicate P and its natural language negation 'not P' are capable of being correctly determined using programs in the head.*

This does seem to be a compelling principle of rationality: why choose interpretations that are not generally possible for you to actually use unless you have a good reason?

Supposing we believe that the Principle of Program-Favoring really is a constraint on rationality, is there good reason for believing that programs will not suffice to correctly interpret natural language predicates and their natural language negations? For example, in mathematics there *is* good reason for believing that programs do not suffice for certain predicates because there are predicates with interpretations that you know are not recursively enumerable. Consider the predicate 'not a theorem of Peano Arithmetic', for example. You know its extension is not recursively enumerable (since its complement is known to be recursively enumerable but not recursive). Your interpretation of 'not a theorem of PA' is set to its extension, regardless of the fact that you are incapable of generally recognizing things that are not theorems of PA. "To me, something is not a theorem of PA exactly if it does not follow from Peano's Axioms; I have no program for it, though." You might acquire a program in the head as a heuristic device aimed to approximately semidecide your interpretation of 'not a theorem of PA', but you are not confused into conflating your heuristic with your interpretation; you know that no such heuristic can possibly be the extension. Thus, you as a mathematician do have predicates for which you have good reason to believe the extension is not determinable via a program in the head, and your interpretations are, accordingly, not determined using programs in the head. (This is, in passing, why mathematical predicates such as 'not a theorem of PA' are not vague.) Given that you can acquire good reasons to believe programs are inadequate and can have interpretations that are not recursively enumerable, what reason is there for you not to do the same for natural language predicates?

The answer is that in the case of such a mathematical predicate you know what the definition of the extension is, and so you set your interpretation accordingly. For a natural language predicate, however, you have no God's eye view of its extension. The extension of 'bald' is learned via induction; you infer your interpretation of 'bald' from seeing objects you have reason to believe (somehow) are in BALD or its complement. You cannot easily acquire the definition for BALD, and as many examples of BALDness and its complement

you might confidently find, you still will not have access to its definition in the way you have access to that of 'not a theorem of PA', for you have no luxury of setting your interpretation to that determined by the definition written on paper before you as you do for mathematical predicates (and this has nothing to do with the fact that you are Church-bound). Given that you cannot have access to BALD in the way you have access to the extension of 'not a theorem of PA', it is also reasonable to suppose that you cannot learn that BALD is not recursive enumerable (supposing this were indeed true) in the way you learn that the extension of 'not a theorem of PA' is not recursively enumerable. I cannot discount the logical possibility of you, a Church-bound agent, learning (in the fantasy) through time that no recursively enumerable interpretation of 'bald' seems to fit the examples of BALD and its complement, and in this way assigning high probability to the hypothesis that BALD is not recursively enumerable, and therefore no program in the head is sufficient. The reasonable hypothesis, though, seems to be that for most (if not all) natural language predicates you have no good reason for believing that programs will not work. I record this as the following hypothesis.

No-Good-Reason-for-Non-Programs Hypothesis: *For most natural language predicates P and their natural language negation 'not P' you have no good reason to believe that programs in the head are inadequate for correctly determining their interpretation.*

The No-Good-Reason-for-Non-Programs Hypothesis together with the Principle of Program-Favoring imply the following hypothesis.

Programs-in-Head Hypothesis: *For most natural language predicates P and their natural language negation 'not P', their interpretations are determined by you using programs in the head.*

You in the fantasy scenario are, then, very likely to fall under the Programs-in-Head Hypothesis. "Very likely" because it is very likely that the No-Good-Reason-for-Non-Programs Hypothesis is true, and given that you are rational you will follow the Principle of Program-Favoring and thus fall under the Programs-in-Head Hypothesis.

Does the Programs-in-Head Hypothesis apply to the real you? Here is an intuitive reason to think so. For most natural language predicates P you are capable of recognizing, given enough time, any cases of P and 'not P'. E.g., given enough time you are capable of recognizing, for any bald-to-you person, that he is bald to you; and, for any not-bald-to-you person, that he is not bald to you. To suppose otherwise would imply, implausibly, that there is a person

that is bald (not bald) to you, but you are utterly incapable of recognizing him as such. The only way for you, who are Church-bound, to have this recognition capability is to have programs in the head doing the work.

4.2.5 Any algorithm

By the Programs-in-Head Hypothesis you have for most natural language predicates a program in the head as the intension determining your recursively enumerable interpretation of the predicate, and this interpretation is your attempt to fit the extension of the predicate. You would like to have a single program in the head capable of determining your interpretation of both 'bald' and 'not bald'; that is, a program that not only says YES exactly when an object is in your interpretation of 'bald', but says NO exactly when an object is not in your interpretation of 'bald'. This is just to say that you would like to have a program to decide the interpretation of 'bald', not just semidecide it. Such a program would be an algorithm since it would halt on every input, and the corresponding recursively enumerable interpretation of 'bald' would be recursive.

But alas, you are Church-bound, and a well-known undecidability result says that there is no algorithm for algorithmhood; there is no general procedure by which either you or a Turing machine can always choose programs (from the set of all possible programs) that are algorithms. It is not, then, generally the case that your programs in the head are algorithms, and your corresponding interpretations for natural language predicates and their natural language negations may generally be only semidecided by the programs for them. (And in fact things are even worse than this, for a related undecidability result says that the corresponding interpretations are not generally even recursive; semidecide is all that any possible program can do in these cases.) If the interpretation of 'bald' ('not bald') is determined by a program in the head that semidecides but not decides it, then supposing that 'bald' ('not bald') is one of the predicates covered by the Programs-in-Head Hypothesis, that program cannot be what is determining the interpretation of 'not bald' ('bald'). This is because the Programs-in-Head Hypothesis states that 'not bald' ('bald') must have a program semideciding its interpretation, and the program for 'bald' ('not bald') cannot possibly be that program. Thus, 'not bald' ('bald') must have its own program in the head. I have now shown 1 and 2 of the Thesis.

How about 3 from the Thesis? It is possible for the interpretation of 'bald' and that of 'not bald' to cover every object, but by the Church-Bound Hypothesis this is not generally possible for you to accomplish. If it were generally

possible, then the two programs semideciding each interpretation could serve as a single algorithm (run both programs simultaneously until one halts), and you could therefore always acquire algorithms. But this is impossible. Thus, it is not generally the case that your interpretation of 'bald' and that of 'not bald' cover every object.

Notice that none of this hinges on either interpretation being non-recursive; what matters is the program for the interpretation semideciding but not deciding it. Predicates with finite interpretations (arguably 'small natural number') are therefore subject to the same conclusion just made concerning 'bald'.

Except for the use of "most" in the statement of the Thesis, I now seem to have shown that you are subject to the Thesis. Concerning "most," it is easier to acquire non-algorithms than algorithms, since in order to achieve algorithmic status the program must halt on every input, whereas to achieve non-algorithmic status there needs to be only one input on which the program does not halt.[3] This observation makes it convenient and informally true to say that for "most" natural language predicates your corresponding programs are not algorithms. This is really just elliptical for the proposition that you are not generally able to acquire algorithms and that it is more difficult to acquire algorithms than non-algorithms. To differentiate this use of 'most' (or 'usually') with genuine uses of it, I always put scare quotes around it.

With this it appears I have now shown you are subject to the Thesis. There is just one remaining problem. I wrote above (second paragraph of this subsection) that "there is no general procedure by which either you or a Turing machine can always choose programs (from the set of all possible programs) that are algorithms." The parenthetic remark merits some examination. Why should you be required to choose from among the set of *all* possible programs? Although the set of all algorithms is not recursively enumerable, there do exist proper subsets of the set of all algorithms that are recursively enumerable, and even recursive. Could you be choosing your programs from one of these subsets? For example, the set of primitive recursive programs is recursive, and perhaps you are choosing from this. If so, you can be sure that every program you choose is an algorithm, and thus that every one of your interpretations for natural language predicates is decidable by the program responsible for it (and is therefore recursive). The Thesis would, then, not follow after all.

[3]More formally and in recursion theoretic terminology, this is captured by the fact that the set of algorithms is Π_2, and the set of non-algorithms Σ_2; the relative difficulty of acquiring algorithms versus non-algorithms is analogous to the relative difficulty of determining cases where a program does not halt versus when it does.

There is a good reason for the fantasy you not to confine yourself in such a fashion. Your interpretations of natural language predicates are a result of a learning process of some sort. You see cases you have reason to believe are in the extension of 'bald' (i.e., you are guided by the true semantics somehow), and you make an educated guess at the extension with your interpretation. *A priori*, you have no reason to believe that all concepts of the world can be correctly determined (or even adequately approximated) with algorithms from some recursively enumerable subset of the set of algorithms. Why should you believe that all extensions may be correctly determined with, say, primitive recursive interpretations? This motivates the following rationality principle.

Principle of No-R.E.-Subsets-of-Algorithms: *Without good reason to the contrary, you should not presume that there is a recursively enumerable subset of the set of all algorithms such that for all natural language predicates P (or 'not P'), algorithms from this subset supply the best interpretation for P ('not P').*

This is a compelling principle: why purposely choose a language with less rich interpretations without good reason? In fact, any recursively enumerable subset of the set of all algorithms is, in a certain real mathematical sense, infinitely less rich than the set of all algorithms.

Supposing we believe that the Principle of No-R.E.-Subsets-of-Algorithms is a constraint on rationality, is there good reason to believe that there are recursively enumerable subsets of the set of all algorithms sufficiently rich for natural language predicate interpretations? Although I am willing to suppose that it may be logically possible for you to acquire high probability in such a hypothesis (after, say, many years of searching for uses of algorithms outside of this recursively enumerable subset and not finding one), there would not appear to be actual evidence for such a supposition. This goes to support the following hypothesis.

No-Good-Reason-for-R.E.-Subsets-of-Algorithms Hypothesis: *There is no good reason for you to presume that there is a recursively enumerable subset of the set of all algorithms such that for all natural language predicates P ('not P'), algorithms from this subset supply the best interpretation for P ('not P').*

One might wonder whether there is nevertheless the following good pragmatic reason for confining algorithm choice to a recursively enumerable subset of the set of all algorithms: by so confining oneself one does indeed avoid the Thesis (and vagueness). The solution comes with a painful price, though. For

all you know there are algorithms that can provide the correct interpretation. Yes, not confining yourself to a recursively enumerable subset of the set of all algorithms brings with it the cost of there being objects in neither the interpretation of P nor the interpretation of 'not P'. However, it is possible for the interpretations to be only "finitely mistaken," where by this I mean that they are complements save for finitely many objects in neither interpretation. Constraining yourself to a recursively enumerable subset of the set of algorithms only for the pragmatic reason of avoiding the Thesis runs the risk that there are predicates, perhaps many, that are not only not correctly interpretable using algorithms from that subset, but will be infinitely mistaken. For example, if one constrains oneself to the set of primitive recursive functions without reason to believe that no predicates should best be interpreted using non-primitive recursive algorithms, then in all those cases where a predicate should be best interpreted using a non-primitive recursive algorithm you are guaranteed to incorrectly classify the objects on infinitely many occasions. Worse than this, it may be that no primitive recursive algorithm even "comes close" to the best algorithm for the predicate. It might be like using the set of odd numbers as an approximation to the prime numbers.

The No-Good-Reason-for-R.E.-Subsets-of-Algorithms Hypothesis conjoined with the Principle of No-R.E.-Subsets-of-Algorithms imply the following hypothesis.

No-R.E.-Subsets-of-Algorithms Hypothesis: *You do not confine your choice of programs to a recursively enumerable subset of the set of all algorithms when interpreting natural language predicates and their natural language negations.*

You in the fantasy scenario are, then, very likely to fall under the No-R.E.-Subsets-of-Algorithms Hypothesis. "Very likely" because it is very likely that the No-Good-Reason-for-R.E.-Subsets-of-Algorithms Hypothesis is true, and given that you are rational you will follow the Principle of No-R.E.-Subsets-of-Algorithms and thus fall under the No-R.E.-Subsets-of-Algorithms Hypothesis.

Does the No-R.E.-Subsets-of-Algorithms Hypothesis apply to the real you? There are reasons to think so. In fact, there is reason to think that the real you is subject to the following hypothesis.

Any-Algorithm Hypothesis: *You are free to choose from the set of all algorithms when interpreting natural language predicates or their natural language negations.*

If the Any-Algorithm Hypothesis is true of you, then so is the No-R.E.-

Subsets-of-Algorithms Hypothesis. This is because any set containing the set of all algorithms is not a recursively enumerable subset of the set of all algorithms.

What reasons are there to think that the Any-Algorithm Hypothesis is true of the real you? It is difficult to tell a plausible story about how (the real) you could have come to restrict program choice to exclude some algorithms, especially since by the Church-Bound Hypothesis you are capable of computing any algorithm. The man on the street does not know recursion theory, and even if he does, as I do, I cannot imagine attempting to restrict myself to, say, primitive recursive intensions for every new interpretation I acquire. Nor does it seem plausible to suppose that we humans might have evolved to exclude certain algorithms. It is, in fact, very difficult to avoid allowing yourself the choice of any algorithm since once you allow yourself the use of *'while' loops*—i.e., the ability to implement programs including statements like "while such and such is true, continue doing blah"—you are able to build, in principle, any algorithm (presuming you can also carry out some trivial basic operations).

To avoid this conclusion you would have to ban the use of 'while' loops, using only *'for' loops*—i.e., the ability to implement programs including statements like "for i becomes equal to 1 to n do blah", or "do blah n times"—which is very restrictive. One could argue that your 'while' loops are in reality bounded since you do not (and cannot) let them run forever; thus, it is not the case that every algorithm can be implemented. But this does not mean that the proper representation of your program does not use a 'while' loop. No real computer, after all, can actually implement unbounded 'while' loops, but it would be a mistake to say they cannot run unbounded 'while' loops and any algorithm.

It can be noted that the idea of animals employing 'while' loops has some empirical support, namely in the *Sphex ichneumoneus* wasp, which has been observed to enter into what is plausibly represented as an infinite loop. Consider the following often quoted excerpt from Woolridge (1963, p. 82).

> When the time comes for egg laying, the wasp *Sphex* builds a burrow for the purpose and seeks out a cricket which she stings in such a way as to paralyze but not kill it. She drags the cricket into the burrow, lays her eggs alongside, closes the burrow, then flies away, never to return. In due course, the eggs hatch and the wasp grubs feed off the paralyzed cricket, which has not decayed, having been kept in the wasp equivalent of deep freeze. To the human mind, such an elaborately organized and seemingly purposeful routine conveys a convincing flavor of logic

> and thoughtfulness—until more details are examined. For example, the Wasp's routine is to bring the paralyzed cricket to the burrow, leave it on the threshold, go inside to see that all is well, emerge, and then drag the cricket in. If the cricket is moved a few inches away while the wasp is inside making her preliminary inspection, the wasp, on emerging from the burrow, will bring the cricket back to the threshold, but not inside, and will then repeat the preparatory procedure of entering the burrow to see that everything is all right. If again the cricket is removed a few inches while the wasp is inside, once again she will move the cricket up to the threshold and re-enter the burrow for a final check. The wasp never thinks of pulling the cricket straight in. On one occasion this procedure was repeated forty times, always with the same result.

I am not suggesting that you are possibly subject to such infinite loops. I am only suggesting that 'while' loops are plausibly part of your computational grammar. In fact, one might say that the wasp has a "concept" of 'readied burrow' which is determined by the following program:

```
WHILE burrow not ready do
 IF burrow clear & cricket not moved when I emerge
 THEN burrow is ready;
```

As an example showing that you regularly engage in 'while' loops (or an equivalent) as well, in order to determine if the bath temperature is good, you may well keep increasing the hot until it is comfortable or too hot; if the latter then you keep decreasing until comfortable; and so on. That is, you implement the following program:

```
WHILE temperature not comfortable do
 IF temperature too cold
 THEN increase hot water;
 ELSE decrease hot water;
```

'while' loops seem to be an integral part of your (and my) computational grammar. And if this is true, the Any-Algorithm Hypothesis is sure to apply to the real you. Thus, the No-R.E.-Subsets-of-Algorithms Hypothesis also applies to the real you.

What does the No-R.E.-Subsets-of-Algorithms Hypothesis tell us? There is a set of all possible programs you can attain (and this is recursively enumerable since you are bound by Church's Thesis). This set is not a subset of the set

of all algorithms, as the No-R.E.-Subsets-of-Algorithms Hypothesis requires. This means you are not generally capable of choosing algorithms. In particular, if the Any-Algorithm Hypothesis is true, then since the set of all algorithms is undecidable, you are not generally capable of choosing algorithms. We saw above that the Church-Bound Hypothesis and the Programs-in-Head Hypothesis "almost" imply the Thesis. What was missing was some reason to believe that the set of algorithms from which you determine your interpretations is not recursively enumerable. The No-R.E.-Subsets-of-Algorithms Hypothesis finishes the argument, and these three hypotheses together entail the Thesis.

4.2.6 Thesis

In this subsection I bring together the previous three subsections. Here is the Thesis again.

For "most" natural language predicates P

1. your interpretation of P is determined by a program in your head that is capable of semideciding but not deciding it,
2. your interpretation of 'not P' is determined by another program in your head that is capable of semideciding but not deciding it, and
3. there are objects neither in your interpretation of P nor in your interpretation of 'not P'.

There are two ways of arguing toward the Thesis. The first is via the fantasy scenario and is largely but not entirely prescriptive, concluding that the Thesis, and thus vagueness, follows largely but not entirely from rationality considerations alone (and the Church-Bound Constraint). The second is via the real you scenario and is descriptive, concluding that the Thesis follows from hypotheses that are true about us. The first would be rather worthless without the second, because a theory that claims that vagueness would exist in the fantasy scenario but says nothing about the real us would be incomplete at best, since it is us who experience vagueness, not some idealized, rational fantasy agents. The second, however, is made more interesting by the first. The Thesis, and thus vagueness, does not follow from some human irrationality or quirk, but is, on the contrary, something to which nearly any rational, sufficiently powerful, finite agent will converge.

I finish this section by (i) cataloguing the premises of both the prescriptive (fantasy you) and the descriptive (real you) argument, (ii) reminding us that the premises of the prescriptive argument entail those of the descriptive argument, and (iii) summarizing how the Thesis follows from the descriptive

argument (and thus by (ii) also from the prescriptive argument). Below are the two arguments.

Descriptive Argument (Real you)	Prescriptive Argument (Fantasy you)
Church-Bound Hypothesis.	Church-Bound Constraint.
Programs-in-Head Hypothesis.	Principle of Program-Favoring. No-Good-Reason-for-Non-Programs Hypothesis.
No-R.E.-Subsets-of-Algorithms Hypothesis.	Principle of No-R.E.-Subsets-of-Algorithms. No-Good-Reason-for-R.E.-Subsets-of-Alg Hyp

The prescriptive argument says that any rational (Principles of Program-Favoring and No-R.E.-Subsets-of-Algorithms), Church-bound (Church-Bound Constraint) agent is subject to the Thesis so long as (a) he has no good reason to believe that the extensions of most natural language predicates are not recursively enumerable (No-Good-Reason-for-Non-Programs Hypothesis), and (b) he has no good reason to presume that there is a recursively enumerable subset of the set of all algorithms that suffices for adequate interpretations of natural language predicates (No-Good-Reason-for-R.E.-Subsets-of-Algorithms Hypothesis). Because (a) and (b) are very difficult to imagine being false, the Thesis follows "largely" from the Church-Bound Constraint and the two principles of rationality. Supposing (a) and (b) are true, the Thesis (and thus vagueness) is good for you, fantasy and real.

The descriptive argument says that (α) we humans are Church-bound (Church-Bound Hypothesis), (β) for most natural language predicates and their natural language negations we use programs in the head to determine our interpretations of them (Programs-in-Head Hypothesis), and (γ) any algorithm may possibly be used by us as a determiner of the interpretations of natural language predicates or their natural language negations (Any-Algorithm Hypothesis), and thus we do not confine ourselves to a recursively enumerable subset of the set of all algorithms for interpreting natural language predicates or their natural language negations (No-R.E.-Subsets-of-Algorithms Hypothesis).

The prescriptive premises (for the fantasy scenario) imply the descriptive premises in the following sense. If you satisfy the Church-Bound Constraint then the Church-Bound Hypothesis is true. If you follow the Principle of Program-Favoring and the No-Good-Reason-for-Non-Programs Hypothesis is true, then the Programs-in-Head Hypothesis is true; the converse is not true. If

you follow the Principle of No-R.E-Subsets-of-Algorithms and the No-Good-Reason-for-R.E.-Subsets-of-Algorithms Hypothesis is true, then the No-R.E.-Subsets-of-Algorithms Hypothesis is true; the converse is not true.

The descriptive premises entail the Thesis as follows. The Programs-in-Head Hypothesis states that you use programs in the head to determine most of your interpretations of natural language predicates and their natural language negations. Most of your interpretations are therefore semidecidable by the programs responsible for them. The No-R.E.-Subsets-of-Algorithms Hypothesis states that the set of programs at your disposal for natural language predicate interpretations is not a recursively enumerable subset of the set of all algorithms. This entails that the set of algorithms from which you can possibly choose is not recursively enumerable. The Church-Bound Hypothesis states that you can only compute the Turing-computable, and thus you cannot generally choose programs for your interpretations of natural language that are algorithms (even if your interpretations are recursive, or even finite). For "most" of your interpretations of natural language predicates P (or 'not P') your program for it will be able to semidecide but not decide it. Since "most" predicates can only semidecide their interpretation, this means that for "most" predicates there must be one program for P, and another program for 'not P', and each can only semidecide the interpretation for which it is responsible. We have so far concluded that "most" natural language predicates satisfy 1 and 2 of the Thesis. Your interpretations of P and 'not P' cannot cover every object, because if they could be then it would be possible to take the two programs and use them as one algorithm to simultaneously decide the interpretations, and this would contradict the impossibility of generally acquiring algorithms. Thus, "most" of the time your interpretations of predicates and their natural language negations do not cover all objects; there are objects in neither interpretation. We now have that "most" predicates satisfy 1, 2 and 3; the Thesis follows from the three hypotheses.

It is important to note that the Thesis is an important claim about natural language whether or not one believes that the Thesis has anything to do with vagueness. If it is true, then, informally, our concepts have "unseeable holes" in them. As for vagueness, my theory's characterization of vagueness is that a predicate is *vague* if and only if it satisfies 1, 2 and 3 from the Thesis; the truth of the Thesis implies that "most" natural language predicates are indeed vague, as we know they are.

4.3 From theory to vagueness

In this section I demonstrate how the Undecidability Theory of Vagueness explains vagueness. Recall that the theory's characterization of vagueness from Subsection 4.2.6 is as follows: Predicate P is vague to you if and only if

1. your interpretation of P is determined by a program in your head that is capable of semideciding but not deciding it,

2. your interpretation of 'not P' is determined by a program in your head that is capable of semideciding but not deciding it, and

3. there are objects in neither your interpretation of P nor your interpretation of 'not P'.

The Thesis stated that "most" natural language predicates satisfy 1, 2 and 3, i.e., "most" natural language predicates are vague. In terms of a single program $C_{P/nonP}$ that outputs YES whenever an object is P and outputs NO whenever an object is 'not P', the characterization is that a predicate P is vague to you if and only if there are objects on which your program $C_{P/nonP}$ does not halt. The corresponding Thesis is that "most" natural language predicates have a region of objects for which the program does not halt. In what follows the Thesis is assumed to be true.

Please notice that in my theory's characterization of vagueness, vague predicates are not in any way required to be computationally complex. I have a great deal of difficulty with people thinking that my theory somehow equates non-recursiveness with vagueness. The only appeal to non-recursiveness has been to the non-recursiveness of the set of algorithms and the halting set (the set of all pairs of programs and inputs such that the program halts on that input), not to the non-recursiveness of natural language predicate interpretations. The interpretations of vague predicates may well be recursive, and even finite, and vagueness is unscathed. And even if a predicate's interpretation is not recursive, the vagueness comes *not* from this but, as we will see, from the facts that the interpretations do not cover all objects and that the programs are not algorithms.

4.3.1 Borderline region

Your interpretations of P and 'not P' do not cover all objects; there are objects c such that the natural language sentences 'c is P' and 'c is not P' are both false. These objects comprise the borderline region. This fits well with the datum of a borderline region: that there are objects which do not seem to fit

neatly into just one category. The development in Section 4.2 served in part to show that (i) any rational, Church-bound agent in the fantasy is very likely to have a borderline region for "most" natural language predicates, and (ii) you do, in fact, have such a borderline region for "most" natural language predicates.

In epistemic theories of vagueness the borderline region is characterized differently than merely "not fitting neatly into just one category." Rather, for epistemicists the borderline region is comprised of those objects for which knowledge of membership is unattainable, where "membership" refers to membership in the true extension. The Undecidability Theory explains this sort of borderline region as well. Suppose that BALD is the true extension of 'bald'. You are not generally capable of acquiring a program in the head that decides BALD, even if BALD is decidable, because you are not generally capable of acquiring algorithms. Your interpretation of 'bald' is semidecidable but not generally decidable by the program responsible for it, and even if you are so lucky to correctly interpret it (i.e., your interpretation is equal to the extension BALD), if you want to be able to respond to queries about 'not bald' you must acquire a second program in the head, and this program will not generally correctly interpret 'not bald' (i.e., the 'not bald' program will not semidecide the complement of BALD). Your interpretations of 'bald' and 'not bald' do not cover every object, and the programs for each only semidecide them. There are therefore objects for which you are incapable of determining or even knowing, using your programs in your head, whether or not it is a member of BALD.

4.3.2 Higher-order vagueness

Although you cannot draw a sharp line between 'bald' and 'not bald', can you draw a sharp line between 'bald' and 'borderline bald'? There is, in fact, a sharp line here posited by my theory, but are you generally capable of drawing it? No. The two programs in the head for baldness (one for 'bald' and one for 'not bald') are not powerful enough to determine the lines. To see this intuitively, imagine starting in the 'bald' region and moving toward the borderline bald region. While in the 'bald' region your program for 'bald' halts and says YES and the program for 'not bald' never halts. When you move into the borderline region the program for 'not bald' still does not halt, but the program for 'bald' suddenly now never halts as well. You are not, though, generally able to know that the program for 'bald' will never halt—you cannot generally know when you have crossed the line. This seems to be consistent with our observations of higher-order vagueness, and it solves the problem without

having to postulate semantically distinct higher-order borderline regions. This latter aspect is good since it puts a stop to the regress of higher and higher order semantically distinct borderline regions, all which amount to nothing if when one is finished there is still a knowable sharp line between the definite region and the non-definite region.

Can this phenomenon really be the phenomenon of higher-order vagueness? In my theory what does it "feel like" to not be capable of determining the boundaries of the borderline region? Well it feels like whatever it feels like to attempt to decide a set using a program that only semidecides it. One might try to make the following criticism: Let us take the set of even numbers and supply you with a program that only says YES exactly when a number is even, and is otherwise silent. Do the evens now seem vague through the lens of this program? There are a number of problems with such a criticism as stated. First, it is not enough that the program simply says YES when an input is even and is silent otherwise. When we say that the program semidecides but does not decide the set of evens we mean that if the program is silent we are not sure whether it will at any moment converge and say YES. The program's silence is not translatable to NO. Second, it is difficult to judge our intuitions with a predicate like 'even' for which we already have a program in the head for deciding it. We should imagine instead that it is some new predicate P for which we have no intuitions. The third problem is that even with these fixes the question the critic needs to ask is not whether P-ness seems vague, but whether P-ness seems to have whatever feel higher-order vagueness has. This is because P is not vague according to my theory since it does not satisfy part 2 of the characterization of vagueness (i.e., we are not given a program for semideciding 'not P'). On this modified question it is unclear that we have any compelling intuitions that the answer is NO. When using the given program to attempt to decide the extension of P, you will be incapable of seeing where exactly the boundary is, and therefore you will be unable to classify many objects. These objects plausibly are just like the borderline borderline objects (i.e., second-order borderline objects).

Another critic may ask the following: Let us suppose that you have two programs that only semidecide their respective interpretations, and let us also suppose that the interpretations do not cover every object. If these programs are for some predicate P and 'not P' then is P-ness necessarily vague? For example, let us take the predicate 'theorem of arithmetic', whose extension is not even recursively enumerable. You are surely capable of determining some theorems and some non-theorems, and you must therefore utilize a program

in the head for ‘theorem’ and another for ‘not theorem’. But surely ‘theorem’ is not now vague! There is a difference between this case and vague natural language predicates. You as a mathematician are conscious that you are not actually deciding theoremhood with your programs. You understand that they are only heuristics, and it is possible that each might even occasionally be incorrect, e.g., saying that a theorem is not a theorem. That is, your programs for theoremhood do not determine what you mean by ‘theorem’. You mean by ‘theorem of arithmetic’ whatever follows from its definition. 1 and 2 from the characterization of vagueness are not satisfied.

4.3.3 The sorites paradox

Finally we arrive at the sorites paradox, which I give here in the following form: (i) 0 hairs is bald, (ii) for all n, if n hairs is bald, so is $n + 1$, (iii) therefore you are bald no matter how many hairs you have. Notice that I have stated the argument in natural language; many researchers on vagueness state the paradox in some logical language, which is strange since the paradox is one in natural language. Presenting it in a logical language inevitably makes certain biased presumptions; for example that ‘not’ is to be translated to the classical negation ‘$\neg$’.

What is usually dangerous about rejecting premise (ii) is that it implies there is an n_0 such that n_0 hairs is bald but $n_0 + 1$ hairs is not; i.e., it usually leads to there being no borderline region. This is bad because borderline regions surely exist. In my theory’s case, though, what happens? A sorites series moves along a “path” that is most gradual from P to ‘not P’; it must therefore cross the borderline region lest it not be “most gradual.” Imagine starting in the ‘bald’ region and moving toward the borderline region. Eventually there will be a number n_0 such that n_0 hairs is bald but it is not case that $n_0 + 1$ is bald, and you cannot in general determine where this occurs. However, this in no way prevents $n_0 + 1$ from being borderline bald, i.e., being neither bald nor not bald. Eventually the denial of (ii) will occur—and you will not know when—but it does not imply the lack of a borderline region. The sorites paradox is thus prevented without losing vagueness.

4.3.4 Essentialness

There is a widespread feeling (since Wittgenstein, it seems) that vagueness is an essential part of natural language. That is, even if it were eliminable (see

Subsection 4.3.5 to see why it is not), we would not want to eliminate it since it serves an essential role.

My Undecidability Theory of Vagueness has its own explanation. Recall from Section 4.2 that the Undecidability Theory in largely prescriptive dress rested upon one constraint, two weak (weak relative to the three hypotheses in the descriptive argument) hypotheses, and two principles of rationality. The constraint was the Church-Bound Constraint, which states that I am concentrating only on agents that are bound by Church's Thesis and able to compute any computable function. The first of the two weak hypotheses is the No-Good-Reason-for-Non-Programs Hypothesis which says that we have no good reason to believe that programs are not sufficient to describe the world. The second of the two weak hypotheses is the No-Good-Reason-to-Exclude-Algorithms Hypothesis which says that we have no good reason to believe that some algorithms may not be useful in describing the world. Supposing the truth of these two weak hypotheses the truth of the two principles of rationality suffices to secure the Two-Programs Thesis and the resulting vagueness (as seen in Subsections 4.3.1 and 4.3.2). Principles of rationality claim that one ought to do something, where there is some implication that not doing that something would be very bad, whatever that might mean. The essentialness of vagueness is bound up with the rationality principles: vagueness is essential because the only way to avoid it is through irrationality, which would be bad. Avoid badness... get vagueness. Let us examine the two principles of rationality in turn.

The Principle of Program-Favoring says that without good reason to the contrary, you should assume that the extension of natural language predicate P and its natural language negation 'not P' are capable of being correctly determined using programs in the head. Recall that this helps lead to the Two-Programs Thesis and thus vagueness because 'not P' is required to be semidecidable by the program for it as well as P, and it is this dual requirement that is difficult to satisfy. How "essential" is this rationality principle; i.e., how "bad" would it be to act in non-accordance with it? You could, after all, avoid the vagueness of 'bald' if you were only willing to live with just one program in the head—the one for 'bald', say. However, this benefit would come at great cost since you would be generally able to identify bald things but not generally things that are not bald. Is seeing the other half of a concept really that essential? Alternatively, is not being able to see the other half of a concept so bad? Yes, it is so bad; I take this to be obvious. The utility gained by bringing in the program for 'not bald' is that it helps you see the "other half" of the concept.

Since it cannot do this job perfectly, vagueness is the result. [Or in "single program" form (see the discussion near the start of Subsection 4.2.5), as soon as you allow your single program to say NO and make your interpretation of 'not P' be the set of objects on which the program says NO rather than simply the complement of the interpretation of P, vagueness is the result since you cannot put in the NOs perfectly.]

Now let us look at the second principle of rationality, the Principle of Any-Algorithm, which says that without good reason to the contrary, you should not presume that there are particular algorithms such that for all natural language predicates P (or 'not P') the algorithm does not supply the best interpretation for P ('not P'). You could avoid the vagueness of 'bald' if you were willing to confine your choice of programs to some recursive subset of the set of algorithms. I spent a little time near the end of Subsection 4.2.5 defending why it is bad not to act in accordance with this principle, and I largely refer you to that discussion. The short of it is that violating the Principle of Any-Algorithm would be very costly since you would thereby confine yourself to much less rich interpretations for natural language predicates and you would not be as capable—possibly even incapable—of adequately classifying an in principle, classifiable world. Vagueness is essential, in addition to the earlier reason, because it is essential that we be capable of classifying our world.

4.3.5 Ineliminability

Vagueness is not to be easily circumvented, or so it is usually thought, and my theory of vagueness leads to several ways in which it may be said that vagueness is ineliminable. One major notion of ineliminability emanates from the fact that there is nothing particular to us humans assumed in the theory; ideal computing devices such as HAL from *2001 Space Odyssey* and Data from *Star Trek* are subject to vagueness as well. Greater powers of discrimination and computation cannot overcome the dilemma of a borderline region and higher-order vagueness. Why, though, is vagueness ineliminable for them?

Let us consider whether the borderline region may be completely eliminated. Once the two programs exist for, say, baldness, perhaps it is possible to find a single new algorithm for baldness that divvies up the universe into YES and NO in such a way that anything that is definitely bald (with respect to the two programs) falls into YES, and anything that is definitely not bald falls into NO (i.e., it respects the definite cases). This algorithm would act by classifying each member of the borderline region as either bald or not, and would serve

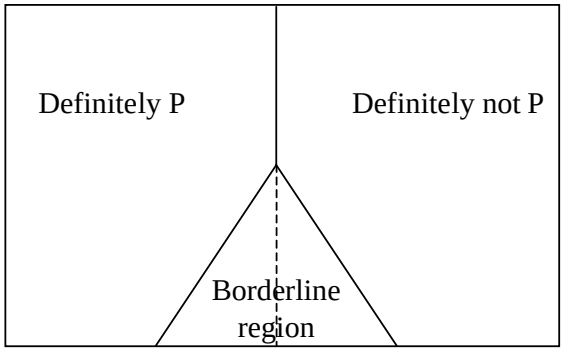

Figure 4.2: A successful precisification would recursively cut through the borderline region as shown by the dotted line, leaving the definite regions untouched. This is, however, not generally possible.

to redefine baldness so as to be non-vague all the while preserving the definite cases (see Figure 4.2). The algorithm would amount to a precisification of baldness. But if it were generally possible to precisify the borderline region and obtain an algorithm for a precisification of baldness, then it would have been generally possible to find such an algorithm in the first place (i.e., pick two programs and then precisify them), contradicting the non-recursiveness of the set of algorithms. Therefore it is not generally possible to eliminate the borderline region, and I call this sort of ineliminability *non-precisifiability*. [If, under supervaluationism, (super)truth is meant to be determined by running through all precisifications and checking to see if the sentence is true in each, then (super)truth of natural language utterances is not generally possible to determine since it is not generally possible to precisify at all.]

May you carefully restrict yourself to certain well-defined contexts, and within these contexts might vagueness be eliminated? We usually do not think so. For example, we do not seem to find ourselves able to identify a group of people (say, infants) such that baldness is no longer vague amongst that group. My theory explains this sort of ineliminability. What you would like to find is a subset of the universe of objects such that there is no longer a borderline region for baldness; you would like to eliminate vagueness by restricting the context

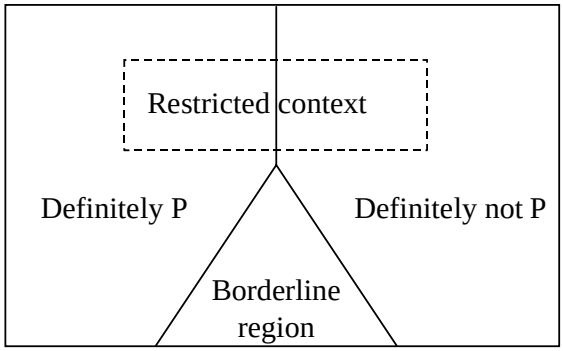

Figure 4.3: A successful restriction would consist of a recursive subset (a context) consisting of no borderline region as shown by the dotted box. This is, however, not generally possible.

to one where there is no borderline region (see Figure 4.3). Not just any subset (or context) will do—you need to be able to recognize (via a program in the head) when something is or is not in that subset, and this implies that you need an algorithm. But now you are back to the same old problem yet again: you cannot generally acquire algorithms. Your contexts are not generally decidable by the programs for them. You may then acquire a context which does not include any of the borderline region but be presented with objects for which you are incapable of knowing whether it is in the context. The objects may in actuality not be in the context, but you may then judge them to be borderline cases and thereby see vagueness. One might respond in two ways here. First, perhaps you only judge an object to be part of the context if the program for the context actually says YES that it is part of the context; if this were so then a single program only semideciding the context is sufficient. The difficulty with this response is that you may well be asked about some object whether it is part of the context and what its categorization is with respect to the vague predicate, and you cannot just refuse to answer. A second response is that we have no reason to believe that contexts require an Any-Algorithm hypothesis; perhaps the allowable programs for contexts are confined to a recursive subset of the set of algorithms. The difficulty with this is that contexts very often are natural language concepts; e.g., attractiveness among bald people, or quickness

among cats. Therefore, (a) the arguments from Section 4.2.5 toward allowing any algorithm apply here, and therefore (b) the context itself is vague. Even if you manage to secure a context that is decided by the program for it, because you cannot generally determine where the borderlines are it is not generally possible for you to be assured that the context does not include some of the borderline region. It is not, then, generally possible for you to restrict yourself to a context wherein the borderline region is eliminated, and I call this sort of ineliminability *non-restrictability*.

I have discussed two sorts of ineliminabilities concerning the borderline phenomenon. Higher-order vagueness is also ineliminable. To begin with, it is not generally possible to correct the two programs so that they decide their respect interpretations. If it were possible, the program determining the interpretation of P could have been "corrected" to an algorithm in the first place, and there would be no need for a second program in the head at all. But it is not possible to generally acquire an algorithm, and thus it is not possible to so correct the programs.

Although the two programs for baldness cannot determine the boundaries of the borderline region, it is possible for the borderline region to be recursive and thus it is in principle possible for you to have an algorithm deciding it. If you had such an algorithm there would be no more higher-order vagueness since you could determine the boundaries of the borderline region. However, it is not generally possible to find the algorithm. For one, it is not generally possible to pick an algorithm rather than a non-algorithm for the job of attempting to decided the borderline region. And two, you cannot be sure, even given that you have an oracle handing you any desired algorithm, whether what it is deciding is the borderline region since you cannot generally know what things are in the borderline region.

4.3.6 Degrees of vagueness

The previous section concludes the sections on the phenomena of vagueness. I want to discuss "degrees of membership" in this section. Membership seeming to come in degrees is not a phenomenon unique to vagueness; non-vague predicates such as 'even' have been shown (Armstrong et al., 1983) to come in degrees to subjects. Also, consider the set $HALT$ of ordered pairs of programs C and inputs x such that the program halts on that input. In a natural sense we are inclined to say that $\langle C_1, x_1\rangle$ is more $HALT$-like than $\langle C_2, x_2\rangle$ if $C_1(x_1)$ halts after fewer steps than does $C_2(x_2)$. Such judgements may depend

on a number of factors, such as typicality, probability, degree of difficulty in determining membership, and so on.

Prior to noticing such examples of precise sets that nevertheless display to us a phenomenon of degrees, one might be worried about the fact that since in my theory concepts are pairs of precise sets, the phenomenon of seeming degrees of membership might be precluded. If single precise sets are not inconsistent with the phenomenon, then it seems there is no prima facie inconsistency to two pragmatically related precise sets displaying the phenomenon—i.e., vague predicates within my theory—and it seems there is no particular responsibility for me to explain seeming degrees of membership. I do not know the explanation for degrees, and I do not care; I defer to whatever is the best theory.

4.3.7 Summing up

I have argued that it is very likely to be in a Church-bound agent's (i.e., finite and sufficiently powerful) best interest to have accessible (i.e., interpreted via programs in the head), maximally accurate (i.e., any algorithm a possible meaning determiner) interpretations of natural language predicates. I have also argued that such an agent having such interpretations experiences vagueness. Vagueness, then, is in such an agent's best interest. If we, too, are Church-bound, then vagueness is very likely *good* for us; the only possible ways for us to avoid vagueness are either to lose the accessibility of our natural language meanings or to confine our meanings to an "infinitely less rich" choice, each very likely to be more costly than the costs of vagueness.

4.4 Discussion

The Undecidability Theory of Vagueness is now out on the table. We have seen how it is motivated in Section 4.2 and how it explains the phenomena linked to vagueness in Section 4.3. There are number of issues to be discussed concerning it that I take up in this section.

Nonstandard concepts

Section 4.2 concluded with the Two-Programs Thesis, which says that "most" of your interpretations of natural language predicates P and 'not P' are semidecidable but not decidable by the programs for them ((i) and (ii)), and are not

complements ((iii)). This is a stunning conclusion irrespective of the vagueness to which we saw in Section 4.3 it leads. 'not P' cannot ("usually") be translated into logic as '$\neg P$' as it is usually thought. Rather, 'not P' should be represented as a distinct predicate of its own, $nonP$, the *dual* to P. $nonP$ has no logical connection to P (other than non-overlap), although, informally, they should be thought of as rough approximations of the complement of the other.

My theory leads to a nonstandard notion of what is a concept, where by 'concept' I mean your or my concept, extensionally construed. There is no single set of objects which fall under a given concept; the interpretation of 'bald' is not the concept baldness, and neither is the interpretation of 'non-bald'. Rather, a concept is comprised of two sets with only a pragmatic relation; the concept baldness is comprised of the interpretation of 'bald' and that of 'non-bald'. Your concept, or your semantics of, P-ness is the ordered pair $(A, nonA)$, where A and $nonA$ are the interpretations of P and 'not P', respectively. If a single two-sided coin represents the usual view of a concept—i.e., 'bald' on one side and '¬bald' on the other—my theory leads to a 'two single-sided coins' view of what is a concept: you have access to only one side of each coin, and the coins are independent of one another (although they are disjoint, and are likely to be rough approximations of the complement of the other's interpretation).

There is an intuitive sense in which this notion of a concept is incoherent. By incoherent I do not just mean that it is non-classical; there are certainly many other non-classical notions of what it is to be a concept which are not incoherent in the way that I mean. For example, in fuzzy logic a concept is a fuzzy set, and in some connectionist models of mind a concept is a vector of the weights of the connections. In each of these cases there is a *single* intuitively natural object representing a concept—a single fuzzy set and a single weights vector. In my case though the single object representing a concept is an ordered pair of sets and this complex object is entirely unnatural. Rather, my notion of a concept consists of *two* intuitively natural objects, namely sets, unnaturally paired together. Whether one should share this intuition of incoherence is not particularly crucial, but to the extent that there is incoherence it helps to explain one of the aspects typically thought to pertain to vagueness. Since Frege (1970) (see also Dummett (1975) and Wright (1975)) there has been the thought that vague predicates are incoherent, and we can see from where the feeling of incoherence springs. This "incoherent" notion of a concept is, according to my theory, an *essential* part of language for any rational, sufficiently powerful, finite agent. Our concepts are essentially incoherent, and this is just the sort of

intuitive feeling people have had about vagueness since Wittgenstein.

The notion of "your concept" just discussed is really "your concept competently employed." In your actual performance you will sometimes incorrectly conclude, for example, that an object is borderline since you cannot afford to let your programs run forever. There are thus three levels of concepts that may be distinguished. First, there is the (fantasy) level of true concepts "out there in the world." I have been presuming they are determinate. Second, there is the level of your concepts "in the head" determined by the pair of programs for P and 'not P', respectively. Third, there is the level of your concepts as you actually perform using them; these will be, at best, approximations of the second-level concepts.

Associated with these distinctions is the following criticism. One might complain that my explanation of vagueness is too abstract to possibly be correct. The theory depends crucially on the notion that we run programs in our head that do not halt on some inputs. This, one could charge, cannot be the explanation for vagueness since our programs never actually do diverge forever. The critic can even admit that perhaps the methods in the head are indeed best represented as non-algorithmic programs in the head, but deny that this can be the explanation for vagueness since the programs are never actually allowed to run forever.

It is true that programs in your head certainly do simply give up after some point; you don't run into infinite loops. When presented with something in the borderline region, where both programs diverge, after some period of introspection you will inductively conclude that the object is borderline. You could have been too hasty, for on the very next step it could be that one program would have halted. All this is no difficulty for my theory. In fact I need it for my theory, for it is just these difficulties the actual agent runs into in dealing with his programs in the head that accounts for his inability to determine the boundaries of the borderline region, or higher-order vagueness. It is this third level mentioned above whose non-identity with the second level helps to explain higher-order vagueness.

Universality of vagueness

I now touch on three issues related to the universality of vagueness.

The first concerns whether all natural language predicates are vague. By the Thesis and the characterization of vagueness, "most" of your natural language predicates are vague. "most," however, does not mean all, and accord-

ing to my theory there may be some non-vague predicates. But are not all natural language (nonrelational) predicates vague? It is not clear that the answer to this question is 'Yes'. For example, Sorensen (1988, p. 201) cites 'flat', 'clean' and 'empty' as example non-vague predicates. These predicates are often applied in "restricted domains of discourse; not all bumps, dirt, and contents are relevant" (ibid.), but if I am asked if, strictly speaking, some surface is flat, I am sure that my answer is either YES or NO (i.e., not neither). "Strictly speaking," surfaces are either flat or not, whereas for 'heap' there is no such "strictly speaking" analog. I also believe 'mortal' and 'everlasting', for example, to be non-vague. There are explanations consistent with my theory for why non-vague predicates are rare at best. The first emanates from the observation made in Subsection 4.2.5 that the set of algorithms is much more difficult than its complement, and this is what motivated the scare quotes around 'most' in the first place. The second explanation helps to explain why there are few to no non-vague "complex" natural language predicates. By complex predicates I informally mean those predicates like 'dog', 'bald', 'people', 'chair', etc., that depend on a number of more "primitive" predicates like 'red', 'circular', etc., for their application. Most of our every day predicates—the ones we use to carve up the world—are complex. In order for one of these predicates to be non-vague, *every* more primitive concept it employs must be non-vague—although see Sorensen (1988, pp. 228–229) for some nice and unusual counterexamples—and this is probably never the case, given that "most" (primitive) concepts are, according to my theory, vague.

The second universality issue is that given that some predicates might be non-vague, we do not find in our experiences cases where, say, 'dog' is vague but 'cat' is not; similar sorts of predicates should either both be vague or neither. The observation just mentioned concerning complex versus primitive concepts explains this datum. Similar predicates make use of similar more primitive concepts, and thus inherit the vagueness (or lack thereof) of the more primitive concepts.

On the third issue of universality, the Thesis is about your interpretations, stating that "most" of the time your natural language predicates are vague. The Thesis obviously also applies to any of us individually. One datum of vagueness seems to be that we don't find ourselves disagreeing about the vagueness of predicates. What reason have we to believe, in my theory's sights, that you and I have the same vague predicates? Why should your "most" and my "most" coincide? The answer to this query is as follows: If you believe 'bald' is vague and I believe it is non-vague, then it is not the case that we have the same

concept of baldness save that one is vague and the other not. Your concept of baldness consists of two interpretations which do not cover every object. My concept of baldness, on the other hand, consists of just a single classical interpretation; I have no "hole" in my concept. We disagree about more than just baldness's vagueness since our concepts are genuinely different. Therefore, in order to explain why we all agree on which predicates are vague, it suffices to explain why we all tend to have the same concepts for predicates. Explaining *this*, however, is not something my theory is subject to any more than any other theory; any adequate account of our shared concepts suffices to explain why we agree about the vagueness of predicates.

Non-vague metalanguage

I do not equate precision and non-vagueness; you can be precise and vague since vague concepts in my theory are, after all, two precise sets of a particular sort. For convenience here is the Undecidability Theory of Vagueness's characterization of vagueness again: Predicate P is vague if and only if

1. your interpretation of P is determined by a program in your head that is capable of semideciding but not deciding it,
2. your interpretation of 'not P' is determined by a program in your head that is capable of semideciding but not deciding it, and
3. your interpretation of P is not the complement of your interpretation of 'not P'.

The metalanguage used to state the characterization is precise. Furthermore it is non-vague. Do not be confused into thinking my use of "your head" in the metalanguage brings in vagueness; it is no more vague within my model than is "the computer I am typing this on," and we may suppose that it is implicitly "your head right now" to allay worries about the identity of your head through time. Predicates are possibly vague, names (or individual constants) are not; or, at least, they are different issues.

Non-vague metalanguage does not necessarily imply that the characterization of vagueness, and thus 'vague', is non-vague, although it does imply that it is precise. Is 'vague' vague according to my theory? This question can be read in two ways: (a) is *your* concept of vagueness vague?, and (b) is *the* concept emanating from my theory's characterization vague. Let me answer (b) first. The concept of vagueness emanating from the characterization is the set of all predicates P satisfying the characterization; i.e., it is just a set. The extension of 'vague' is that set, and the extension of 'not vague' is the complement of

that set. Part (iii) of the characterization is thus violated and so *the* concept of vagueness is not vague.

What about question (a)? Although the true extension of 'vague' is not vague, might your concept of it possibly be vague? According to the characterization, 'vague' is vague to you if and only if

1. your interpretation of 'vague' is determined by a program in your head that is capable of semideciding but not deciding it,

2. your interpretation of 'not vague' is determined by a program in your head that is capable of semideciding but not deciding it, and

3. your interpretation of 'vague' is not the complement of your interpretation of 'not vague'.

The characterization is not just a characterization—it is also the explanation for why some predicates are vague. It says that vague P seems vague to you because your programs in the head for P and 'not P' semidecide but do not decide their respective interpretations and these interpretations are not complements. When you are presented with P and are asked whether it is vague, my theory's claim is that you do some introspection—running both programs on various inputs—and see if things "feel" like whatever things feel like when (i), (ii) and (iii) of the characterization obtain. Therefore, your interpretation of 'vague' according to my account might seem to be the same as the true extension. The problem with this suggestion is that it is not generally possible for you to successfully do such introspection. Your interpretations might be complements yet you not be able to know this, or not be complements and you not be able to know this. Also, your programs might not decide their interpretations but you may not be capable of verifying this, or vice versa. Thus, it is doubtful that your interpretation of 'vague' actually is the true extension. The question of whether 'vague' is vague to you is still open, and it comes down to a factual matter. As far as my characterization goes, your concept of vagueness may be vague or not.

Sorensen (1985) has argued that 'vague' is vague. His argument tactic is to show that one can build a sorites argument using 'vague'. He proposes the disjunctive predicates 'n-small' for each natural number n, each which applies to those natural numbers that are either small or less than n. The sorites argument is as follows:

(1) '1-small' is vague.

(2) For all n, if 'n-small' is vague, then '$n+1$-small' is vague.

(3) 'One-billion-small' is vague.

'1-small' is obviously vague, so (1) is true. (2) seems compelling, but (1) and (2) imply (3) which, supposing that our interpretation of 'small' is such that one billion is definitely and clearly not small, is false since it is now equivalent to the non-vague predicate 'less than one billion.' 'vague' is vague because "it is unclear as to where along the sequence the predicates with borderline cases end and the ones without borderline cases begin" (Sorensen, 1985, p. 155).

Such unclarity, or even unknowability, of the line is not sufficient for vagueness. The sorites argument for 'vague' is not paradoxical, since it is not fantastic to deny the induction step; such a denial means asserting that there is an n such that 'n-small' is vague but '$n+1$-small' is not. Is this difficult to believe? I do not see why. For 'bald', on the other hand, the proposition that n hairs is bald but $n+1$ hairs is not strains credulity, and we are accordingly unhappy to deny the induction step. It strains credulity because we feel, right or wrong, that there are borderline cases of baldness. Is there any such intuition for vague predicates? Are there cases of predicates we find are borderline vague? I do not know of any such examples; any natural language predicates I have encountered either have borderline cases or do not. (See also Deas (1989) for one who agrees.)

'vague' is not vague, then; or at least if it is vague it is not so merely by being able to be put it into what seems to be a sorites series. A sorites series is only genuine when it is paradoxical, and it is only paradoxical when the denial of the induction step seems counter-intuitive. Since the denial of the induction step for the sorites series above is not counter-intuitive, this suggests that any sorites series with 'vague' will also not be paradoxical. In fact, since there are no clear cases of borderline vague predicates, there is not even any prima facie reason to believe 'vague' is vague.

Hyde (1994) utilizes Sorensen's argument for the vagueness of 'vague' to argue, in turn, that higher-order vagueness is a pseudo-problem. If 'vague' is vague and vagueness is defined as the phenomenon that there are borderline cases, then since the existential 'there are' is not vague 'borderline case' must be vague because otherwise 'vague' would not be vague. I.e., if 'vague' is vague then 'borderline case' is vague. But higher-order vagueness is the phenomenon that there are borderline cases of borderline cases—it is that 'borderline case' is vague—and this is already built into the original concept of vagueness. Higher-order vagueness comes for free from the vagueness of 'vague', and thus one need not tell any special story concerning higher-order vagueness. Hyde's argument fails without the vagueness of 'vague', though, and with reason now to deny the vagueness of 'vague', there is reason to believe

that higher-order vagueness is a genuine problem needing possibly a separate explanation. (See also Tye (1994) for other criticisms of Hyde's argument.)

What kind of theory is this?

Where does the Undecidability Theory of Vagueness fit in amongst the spectrum of theories of vagueness?

(1) It is an epistemic theory. Vagueness exists in part because of your inadequacies: you are finite. Furthermore, no semantic indeterminacy concerning the true concepts is required. In this sense it is a full-fledged epistemic theory. (2) However, despite the consistency with a determinist account of true semantics, my theory has an indeterminist aspect in that the semantics for the natural language user's concept P-ness consists of two distinct interpretations, one for P and another for 'not P'. The borderline region is semantically distinct from the definite regions. The account of natural language semantics for your concepts is, then, indeterminist. If one holds that (true) meaning is competent use (of a community even), then the semantics at this level *is* the true semantics, and one would have to hold semantic indeterminism. (3) Finally, the underlying logic of your concepts is determinist in that P and 'not P' become P and $nonP$, each which gets its own determinist classical interpretation.

How does the Undecidability Theory compare to other explanatory theories of vagueness? Is there anything about the Undecidability Theory that is favorable? I think so: its paucity of assumptions. It rests on the three weak descriptive hypotheses from Section 4.2: the Church-Bound, Programs-in-Head, and Any-Algorithm Hypotheses. Each is, independent of vagueness, quite plausible. The Church-Bound Hypothesis (see Subsection 4.2.3) says that you are finite, bound by Church's Thesis, and capable of, in principle, computing any Turing-computable function. The Programs-in-Head Hypothesis (see Subsection 4.2.4) says that your interpretation of a predicate is determined by a program in your head for it. That is, natural language predicates are not like 'theorem of Peano Arithmetic' for which your interpretation is set to that given by its (not recursively enumerable) arithmetic definition, not the set determined by whatever program you might use as a heuristic for responding to queries about it. The Any-Algorithm Hypothesis (see Subsection 4.2.5) says that you allow yourself the use of any algorithm for your interpretations of natural language predicates. We saw that if you allow yourself 'while' loops in the building of programs, then it is difficult to reject this hypothesis. Each is compelling, and vagueness follows. Alternative theories of vagueness must either deny one of

these, or argue that the phenomenon my theory explains is not vagueness.