CSci • B607 **Week #3 (b)** **Jan 25, 2001**

 Last edited Friday, February 2, 2001

The Primary Dialectic in the Five Construals

I. Computing

A. Review
   1. Today we will wrap up Part I of the course (Introduction) and start on Part II
   2. Formally: we introduced the project (2 promises, 3 formal criteria, 5–6 construals, etc.)
   3. Substantively: introduced two fundamental dialectics. The **primary dialectic** was between:
      a. **Meaning:** associated with "mind"—i.e., the non-effective, oriented/directed, reach of semantical phenomena across time, space, and possibility, characteristic of *symbols*, *semantics*, *intentionality*, *content*.
      b. **Mechanism:** associated with "body"—i.e., with *effectiveness*, with *getting things done*, the material, *potency* aspect of computation or machines
   4. The discussion of mechanism, or body, we talked about in terms of **potency predicates** (about which we'll have more to say today), which in turn raised a question about the appropriate level of *abstraction*.. In particular, it led us to the **secondary dialectic**: whether the potency or effective side of computing should be treated as
      a. **Concrete:** in the sense that the applicable notions of effectiveness, computability, "what can be done," etc., should ultimately be derived from physics
      b. **Abstract:** a characterisation of computing/computation as essentially *mathematical*
   5. In the AOS book series, there are two more dialectics I will be using, to analyse computing:
      a. Between what is **static** and what is **dynamic**: e.g., between a static program and the dynamic process that it engenders.
      b. Between the "**one**" and the "**many**" (e.g., between a single type and its many instances, a single file and multiple copies of it, a single program and multiple executions of it, etc.);
   6. Both of these pairs of issue will be of concern, throughout the class. But (for simplicity) I won't raise them into special named prominence, here.

B. How do these dialectics relate to computing?
   1. Mechanism
   2. That computing has to do with mechanism (mechanisation) is obvious. Computing is constitutively involved in answering questions like the following:
      a. (Leninesque) What can be done?
      b. What can be effectively accomplished?
   3. This fits with a general intellectual theme (since the Scientific Revolution): that what exists is what can be constructed out of ordinary physical ingredients
   4. Admittedly, as we've said (this is what the secondary dialectic deals with), the efficacy or causal organisation of computers is somewhat *abstract*
      a. Seemingly "above" the pure physical considerations of energy and material

   b. Nevertheless, computers are undeniably concrete material objects: "natural" things built out of physical parts.
5. Scientifically respectable, too—or so it is imagined—in an era in which "science" is equated, if not solely with matter and materials, then at least with cause and effect.

C. Meaning
1. But there is more to computing than body. That computing has to with meaning is equally obvious—so long as we construe 'meaning' sufficiently widely.
2. That is, computers also traffic in the semantic or–what philosophers call the **intentional**
   a. As we have seen (see figure 5 of the notes from lecture 2a), intentionality is a technical term used to cover all sorts of phenomena—language, symbols, information, interpretations, representations, codes, descriptions, models, and the like—that are *about*, or are *directed towards*, something (typically something *else*).
3. The intentionality of computing is betrayed in the language we use:
   a. Programming *languages*
   b. *Communication* systems
   c. *Knowledge representation* systems
   d. *Information* processing
   e. *Symbol* manipulation
   f. Programming *language semantics*
   g. Interpreters
   h. *Data* bases and *data* structures
   i. *Names*, *identifiers*, *terms*, etc.
   j. Pointers
   k. Call by *value*, call by *reference*
   l. *Referential* transparency
   m. Computational *models*
   n. *Meta-level* architectures
   o. *Specification* languages
   p. … etc. etc. etc.
4. This ubiquitous language suggests that computing is not only intentional at some level abstraction, but intentional *at the same level at which it is computational*

D. As promised, we will focus on both aspects, in the body of this course.

II. Objections

A. The are two main objections that some people will raise to this bivalent characterisation, in terms of which we are conducting the investigation

B. Non-semanticity
1. Some (many?) will object, essentially, to the first dialectic.
2. For various reasons, they will be opposed to the idea that a comprehensive theory of computing should include, or rest on, a theory of semantic (intentional) phenomena.
3. In defending that position (*not* to study semantics or intentionality), they are liable to argue one or more of the following four distinct positions:

   a. That computation *not a semantic phenomenon* (and that even if we sometimes *interpret* computational processes *as* semantic, or ascribe semantic significance *to* computers, such ascription is neither intrinsic to, nor constitutive of, their being computational)
   b. That even if computing does turn out to be semantic, a comprehensive theory of computation should nevertheless be *formulable* (perhaps reductively) *in non-semantical terms*
      i. This would be a naturalistic or reductionist theory
   c. That semantics and computation are *orthogonal*
      i. I.e., that a full theory of intentionality should be expected to consist of two parts
         α A *theory of semantics*
         β A non-semantical *theory of computing* (or—which is perhaps the same thing, but perhaps not—a theory of non-semantical computing)
      ii. So we should identify "computation" with mechanism ("body")*as opposed* to meaning ("mind"), rather than as including both.
      iii. This is a view that is more common in philosophy than in computing per se.
   d. That computer science has shown that semantics can be *explained mechanistically*—implying that meaning and semantics should be expected to fall within the scope of mechanism, rather than existing in dialectical tension with it
4. Replies
   a. Though these positions have merit (as we will see), I still believe that it is impossible to do justice to *use* of computing (CITW) unless one realises that people use computers to *model*, *represent*, *analyse*, *simulate*, *predict*, *explore hypotheses about*, and stand in a myriad other intentional relations to their task domains.
   b. Hence: any theory of computation that meets the first (empirical) criterion must take on semantical issues directly
   c. Note that this commitment (to require a theory of computing to include an explanation of its semantical or intentional aspect) is for the moment *methodological*, not substantive.
      i. No particular theory of semantics is required
      ii. Some objections (e.g., b and d) are (reductive) positions *on the topic of semantics*
      iii. So they meet the condition of providing an account of semantics
   d. Moreover, the claim that computing is intentional will not just be a pre-theoretic tenet of this investigation, it will also be a "post-theoretic" result (as durable a result as any we come to)
   e. Finally—somewhat ironically—I believe we will be able to do *better justice* to the "non-semantical" intuitions by understanding computation in intentional terms than if we do not. In a way, our joint-aspect approach is the "easier" one, allowing each aspect or feature to have its natural place. That puts us in a better place to be able to understand how they all fit together (even if they were ultimately to be shown to fit together within a purely mechanistic framework).
5. Terminology
   a. (Note for philosophers) Because of our commitment to this primary dialectic, in this course we will use 'computational' in approximately the way one uses 'logical'—as encompassing both aspects— and not as analogous with 'syntactic.'

C. Simplicity
   1. The other objection (to a semantical characterisation of computing) is usually be framed in terms of simplicity, but in reality has to do with the secondary dialectic.
      a. Thus someone might say: "Look, this is easy. You are making everything much too difficult. Consider the following typical story:
      b. A computer is a physical device, operating according to standard mechanistic principles, that is (or can be) semantically interpreted. Like any physical machine, it works in virtue of its physical constitution. What makes it computational is the fact that there it sustains a level of description—typically called the *syntactic* level of description—with 2 distinctive properties:
         i. Corresponds to, or is directly reflected in, aspects of the device's functional organisation, based on its physical constitution, which enables it to work in accord with ordinary causal laws; and
         ii. It is also a level at which the device (or device's behaviour) can be coherently semantically interpreted.
      c. That is (as one says), the concrete causal structure "mirrors" the abstract functional or "computational" structure.
      d. For example, consider a simple adding machine. The syntactic level of description is formulated in terms of the binary numerals (not numbers!) 0 and 1, along with the standard syntactic rules for binary addition. The 0s and 1s are implemented by functional properties, reflected in the device's physical structure (signals of 0.0 and 3.2 volts, say). The device is so constructed that, in virtue of this mapping of syntactic properties (via function) onto physical properties, it obeys the aforementioned rules. This means, in turn, that if we interpret the numerals as standing for the numbers zero and one, in the ordinary way, then we can interpret the machine as performing addition."
   2. Replies
      a. This isn't a bad characterisation, on the face of it.
      b. For example: it makes explicit reference both to semantics and to "working"—both principles that we are committed to take seriously
      c. But there are several problems with it—or at least issues that it leaves open:
         i. There is a question of how the syntactic properties relate to the semantic ones. According to the FSM construal (see below), they are *independent*. In fact if you add that independence claim, the above story essentially turns into the FSM story. But as stated above, the story doesn't include that constraint—what in philosophy is often known as the "formality condition." So there is a question as to exactly what systems the foregoing story would apply to, without that restriction.
         ii. There is also a question as to what properties are syntactic. When different logics are presented, one is typically given a *particular* system, with relatively evident syntactic properties. But we are interested in the general case: what *is it to be a syntactic property*? That question would have to be answered, in order for the foregoing story to be a candidate theory.
      d. The main thing, though, is that this story is essentially (modulo issues of formality) the FSM construal. And that is the construal we are going to look at first. So in a way, we can take this story as leading into the next major substantive part of the course.

	Mechanism	Meaning
FSM	Computers *work* in virtue of the *syntactic* (non-semantic) aspects of the ingredient symbols, where the notion of "syntax" is constrained to be effective	Computation is the manipulation of *symbols*, whereas symbols (at least arguably!—see the FSM critique) are intentional or semantically interpreted (or interpretable) entities.
EC	*Effective* computability—notion of *effectiveness* is exactly what we are talking about	Unclear—*watch this space!*
RF	What it is to *follow* a rule (as opposed to the rule's merely *holding* of something, has to do with *doing something substantive* or *effective*	Unclear—does the "following" require understanding, or some other form of intentionality? or is it just blind "acting in accordance with"? or "being causally driven by"? we'll have to figure out
IP	Notion of *processing* gets directly at the effective dimension	Information is (again—arguably!) already an intentional notion. But (as we will see) a number of theoretical approaches deal only minimally, if at all, with informational *content*.
DSM	Least clear of the lot. But still there is an effective, "do-something" assumption in the notion of a *machine* (that is: we are not taking DSM to be an (even) more general "digital system" construal, so as to allow a digital system like set theory to count as a computer	Doesn't look semantic at all

Figure 1 — The Primary Dialectic, in the Five Main Construals

III. Plan

A. Three more things need to be talked about briefly, before we take on that first construal.
B. Strategy
   1. Our plan is to look at each of the two primary aspects (meaning and mechanism) in each of the five construals
      a. For starters, note that semantic terms figure in several of the construal's names:
         i. Formal *symbol* manipulation
         ii. *Information* processing
         iii. Physical *symbol* systems
         iv. *Rule-following*
         v. Maybe even: effective *computing*
   2. A brief synopsis of how the issues figure, on the surface, is given in figure 1 (above).
   3. It's already clear what we will discover: we are in far better shape on the *mechanism* or "body" side of the house than on the semantical / "mind" side.
   4. We can't be complacent about the "body" side, though. We're not in *very* good shape there, either!

C. Semantics in computing
   1. We need to distinguish *programs* and *processes*, to understand how semantics' is used
   2. Ontologically, make a 3-way distinction, among (see figure 2):
      a. **Program:** the static, textual entity (as in an Emacs buffer)
      b. **Process:** dynamic, active behaviour, that happens when the program is run or executed.
      c. **Task domain:** the (typically external) world or subject matter that the program/process is about.
   3. In terms of this picture, distinguish *two semantic relations*:
      . **Program semantics:** α, from *program* to *process*
      b. **Process semantics:** β, from *process* to *world*
   4. To confuse them is essentially an issue of use and mention
   5. However, two complexities make dealing with them tricky:
      a. We have virtually no ontologically sound way of treating processes—as entities in their own right. So processes (and behaviour) are therefore usually *modelled*, mathematically.
      b. Identifiers in programs are often used *as if they referred to entities in the task domain*—i.e., as if the process ingredients were (semantically) *transparent*. This makes separating out the two relations very difficult.
   6. For more details, cf. my "One Hundred Billion Lines of C++", available on the website.
   7. In what follows, I'll try to identify which relation—α or β—is being talked about, whenever semantical issues come up.

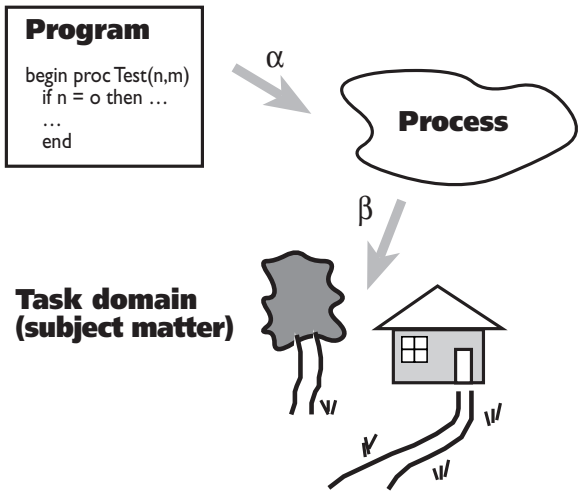

Figure 2 — Program, Process, World

B. Formality
   1. One other issue that we didn't address: that of *method*
   2. Primary methodological concern of the investigation is with **formality**
   3. Consensus that computation is somehow "formal" is astonishingly widespread
      a. Ontologically: computation is formal, involves the manipulation of formal symbols, etc.
      b. Methodological: computation is such that it should be studied formally, etc.
   4. Won't say much about it here, except to say that formality
      a. Is a major interest of mine
      b. Is an extraordinarily serious issue, with implications that will ultimately transform our understanding of the metaphysical basis of all of science.
   5. Keep it in mind, though in the background. We will come back to it from time to time, in each construal—especially in the last week of the semester.

Part II — Formal Symbol Manipulation

II. Introduction

A. Introduction
   1. Finally! Let's turn to Part II of the course: Formal Symbol Manipulation (FSM)
   2. This construal has "pride of place"
      a. Historically: Formal symbol manipulation gets at the intellectual conception, based on logic, from which computing arose.
      b. Substantially: of the five primary construals, FSM addresses the primary dialectic most directly.
      c. Methodologically: it also deals with formality explicitly.
   3. All three reasons recommend it for first-place treatment.
   4. FSM is the construal of computing most common in *logic*, *cognitive science*, *philosophy of mind*, and (to a lesser extent) *Artificial Intelligence*
      a. Issues we talk about for the next few weeks will therefore feel more familiar (perhaps seem more important) to people from these traditions, than to computer scientists.
      b. In Part III, on effective computability, Turing machines, etc., the tables will turn.
      c. That part of the course will be more familiar to computer scientists, but less so to cognitive scientists and philosophers.

B. Miscellaneous
   1. Readings (for Part II)
      a. Primary: the first few chapters of AOS Vol. II: are available on the web site
         i. Note: AOS Vol. II Chapter 2 (Explanatory Critique) is the most difficult of any of the AOS chapters (it is *very* philosophical).
         ii. Don't be disheartened. Subsequent chapters are much easier!
      b. Secondary: variety of papers, listed in the syllabus, and included in the class reader.

C. History
   1. Formality: old as the Greeks
      a. Won't here tell that story (though a story worth telling)
      b. Theme of the 20th century (in art and music, as well as intellectually)
   2. Focus in particular on the notion as it comes into computing through logic.

D. Plan
   1. I will argue that the construal fails in three ways:
      a. **Conceptually:** independent of its applicability to computation, it turns out to rest on a mixture of different notions and intuitions, some in tension with each other, some in outright conflict.
      b. **Empirically:** no one of these readings is sufficient to reconstruct practice—and some are demonstrably false. In the general case, real-world computer systems can neither be reduced to, nor explained by, any identifiably formal subspecies of generalised symbol manipulation.
      c. **Explanatorily:** Even if computation *were* formal, on any of the different readings of what formal symbol manipulation may mean, those facts would not achieve for comput-

ing the theoretical safety or naturalistic cleanliness and palatability that is sometimes imagined.

2. Quite a lot to have going against it.
3. Yet based on a *penetrating insight into the nature of intentionality* (applicable to both people and machines)
   a. Alone of the six construals to get at this
   b. Absolutely necessary to hang onto, in constructing a new account
4. Aim of the investigation: dig that positive moral out, and free it from its "formal" clutches.

E. Method

1. Two predicates under investigation in this critique: *formal* and *computational*
   a. Sometimes confusing which is under analysis
   b. Primarily, *computing*.
   c. Will assess readings of 'formal' (as with other proposals for what it is to be computation) *wrt their viability as an account of what computing is like*.
2. Given that, as we said a few weeks ago, there are two ways a proposal can fail (figure 3): too narrow, or too broad.

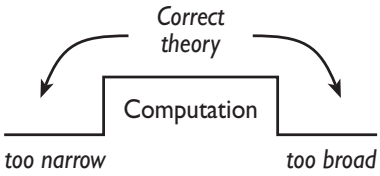

Figure 3 — Two ways to fail

III. Conceptual ("what does it say?")

A. Introduction

1. Three words: 'formal', 'symbol', and 'manipulation'
2. Look at them in turn, in this order:
   a. Symbol
   b. Manipulation
   c. Formal

B. Symbol

1. FSM takes computation to be *intentional—ineradicably semantic*
   a. If don't want an intentional construal of computing, have others to choose from
   b. Important, because some people view formal symbol manipulation as *non-semantical*
   c. Will look at what kind of "non-semanticity" in exhaustive and exhausting depth
   d. Bottom line:

> Formality's rejection of semantics is like the prohibitionists' dismissal of alcohol, not the physicists prohibition of the luminiferous ether. The position's integrity *requires the existence of that against which it is arrayed*.

   e. If semantics were *absent*, then FSM would collapse in vacuity
   f. Call such a construal **stuff manipulation**
2. Restriction
   a. Should 'symbol' be restricted (as in "non-symbolic" or "anti–symbolic" cog sci)?
   b. No. Because if we *do* restrict it, then we immediately fall off the "too narrow" cliff.
   c. For example, can't restrict it to:

i. Conceptual
ii. Explicit
iii. Derivative
α This one is of the wrong *kind*. Like secondary readings of computing. *If* computing does have derivative semantics, that should be *shown*, not assumed
β General problem of *preëmptive* analysis.

d. So take 'symbol' widely, to include representations, data structures, continuous fields and von Neumann stores and Lisp heaps and just about anything else you can imagine.

3. A note on people, in passing
   a. We are symbol manipulators at least in part (e.g., when doing mathematics)
   b. One might wonder whether we *always* are—i.e., whether that is what it is to think.
   c. If one weren't asking whether the formal symbol manipulation conception is applicable to computing, one might want to accept some of the above restrictions on the notion of 'symbol'—and then ask (as in cognitive science) whether people are symbol manipulators, or even whether they are formal symbol manipulators, given such a restricted conception.
   d. We won't pursue that tack here, but for the record, is would be a substantial inquiry.
   e. By not restricting the notion of a 'symbol,' you might think that our investigation won't have much impact on that project.
   f. However, there is this relation
      i. Given a broadened notion of 'symbol,' we will challenge (pretty hard) the idea of *formality*.
      ii. If one were to ask whether people are symbol manipulators, as just suggested, it would still remain open whether we are *formal* symbol manipulators.
      iii. It is very likely that the results we arrive at here, in challenging the formality of computing, might apply to the formality of people, even on that restricted conception.
   g. In sum, there is more "cross-over" from this analysis to that than might at first appear.
   h. Nevertheless, I won't consider that (human) question further, here.

C. Manipulation

1. For now, just note that this indexes the *potency* or "effective" aspect of computing
2. That is, the construal
   a. Makes a claim that these are not just (abstract, static) symbol systems, enjoying some kind of Platonic existence
   b. Rather, they involve machinery, the *doing* of something (just as we said in characterising the main mind/body dialectic)
3. Not a lot else to say yet. But that does not mean that this word is unimportant. Far from it.
4. Will get to more presently.

D. **Formal**

1. This is the only construal to explicitly use the term 'formal'
2. Because we haven't been able to restrict 'symbol,' *most of the weight of the construal hangs on this term*.
3. That is: it is the word 'formal,' in this construal, that "wears the trousers."

4. We will take it to be a predicate on *manipulation*, not on *symbols themselves*
   a. I.e., parse it as: (formal (symbol manipulation))
   b. Not as: ((formal symbol) manipulation)
   c. I.e., the formal manipulation of (perfectly semantical) symbols, not as the manipulation of formal symbols
5. The question is what the formality comes to
6. Of the space of possible symbol manipulation, *what is it that characterizes the "formal" species?*
7. Motivation
   a. Before looking at possible definitions, look at something formality is thought to convey
   b. A wonderful conceptual *cleanliness* or *safety*
      i. Cleanliness, naturalistic palatability denied non-formal systems (perhaps including us)
      ii. A kind of **middle way** wrt semantics
   c. On the one hand, as we've said, FSM assumes that computers are intentional (that's the point of the 'symbol' manipulation part)
   d. On the other, they are thought to be "rescued from all the *really difficult* semantical questions—how reference arises, what establishes the conditions for semantic success, what normative dimension of significance comes to, how truth relates to beauty and goodness (the sorts of questions that a full theory of human intentionality would have to wrestle with).
   e. I.e., a middle ground between the naturalistically palatable physical substrate, and the higher reaches of intentional mystery
8. As this first critique moves forward, we will want to keep an eye on various readings of formality, to see whether it secures, for formal symbol systems, this essential middle ground, this intermediate position.

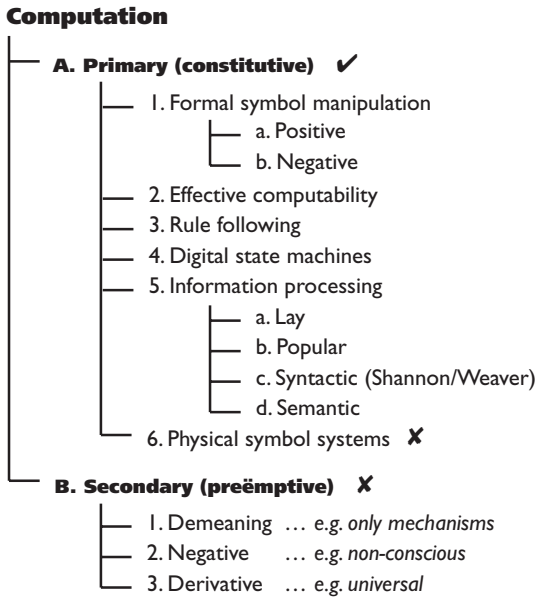

Figure 4—Investigative Structure

IV. Two readings

A. Strategy
   1. Remember what we did (in part I) with information processing: broke it down into (four, as it happens) *readings*, underneath the main construal.
   2. Now do the same for FSM (see figure 4).
B. Two readings
   1. **Positive**
      a. Works in virtue of the *syntax* or *shape* or *form* of the ingredient symbols
      b. Cf. *potency* predicates (grammar, syntax, form, shape, physical, causal, effective, etc.)

   c. The FSM construal puts the emphasis squarely on one of them.
2. **Negative**
   a. Works *independent of the symbol's semantics*
   b. Example: in logic, say, an inference about Socrates' mortality. Inference (in a formal system) would have *not to depend* on: Socrates, his mortality, the relation between the symbols that that person, etc.
   c. It is this negative characterisation that is thought to convey on the whole situation the wonderful conceptual cleanliness cited earlier (no irreducible mystery, no naturalistic unpalatability).

C. Schematise it as follows
1. Identify three classes of predicates (cf. AOS pages II·1·23 and II·2·24)
   a. **Syntactic:** those properties involved in what might be called a system's grammatical regularities;
   b. **Effective:** those properties in terms of which the mechanical operation of the system is defined—i.e., that play a causal role in engendering behaviour; and
   c. **Semantic:** such properties as truth, reference, meaning, etc.
2. In terms of this division, the positive reading of the formal symbol manipulation claim can then be taken as claiming that the syntactic properties are the effective ones:

   Positive: SYNTACTIC = EFFECTIVE

3. The negative reading, in contrast, can be understood as a claim that the effective and semantic classes do not overlap:

   Negative: EFFECTIVE ∩ SEMANTIC = ∅.

4. Assumption that the two readings, positive and negative, come to the same thing, in any or all instances, depends in part on what is perhaps *the most universal and unquestioned claim in the vicinity*: that a formal symbol systems' syntactic and semantic properties are disjoint:

   Disjointness: SYNTACTIC ∩ SEMANTICS = ∅

D. Fodor, reigning apologist of formality in cog. sci, recognises this strange conceptual structure:
1. "What makes syntactic operations a species of formal operations is that being syntactic is a way of *not* being semantic. Formal operations are the ones that are specified without reference to such semantic properties of representations as, for example, truth, reference, and meaning. Since we don't know how to complete this list (since, that is, we don't know what semantic properties there are), I see no responsible way of saying what, in general, formality amounts to. The notion of formality will thus have to remain intuitive and metaphoric, at least for present purposes: formal operations apply in terms of the, *as it were, shapes* of the objects in their domains."[1]
2. I.e., Fodor:
   a. Identifies formality negatively
   b. Then takes the positive (syntactic) reading to be a *species* of nonsemanticity, implicitly relying on the near-universal assumption ("disjointness"), expressed above, that syntac-

[1] From *Methodological Solipsism* (emphasis added)

tic and semantic properties do not overlap.

c. By the end, equates formality with effectiveness—which he recognises, in his hedged reference to "as it were, shapes", to be as-yet unreconstructed.

E. Discussion of positive
   1. Notion of "work" is effective
   2. Not 'work' as in "how does it work that n/7 is of the form .(2n)(4n)(8n)?"
   3. Means work *effectively*, or *physically*
   4. Threat, though, is that it will reduce to vacuity, in virtue of "syntax" or "form" reducing to *anything that is causally effective*, and "work" being interpreted as "works, effectively", so that the whole thesis becomes: a formal system is one that works, mechanically, in virtue of the possession of effective properties—i.e., to materialism (or stuff manipulation again)
   5. Won't spend more time on it now.
   6. Defer the intuitions underlying it (especially the *shape* intuitions) to the second construal (and associated critique), which has to do with *how things work*
   7. See figure 5.

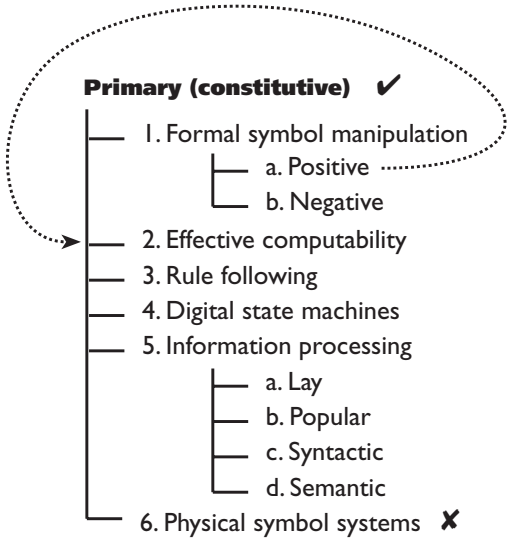

Figure 5—Deferring "positive" formality

F. Next Tuesday: we'll pick up discussion of *negative* reading(s) of formality

—— *end of file* ——