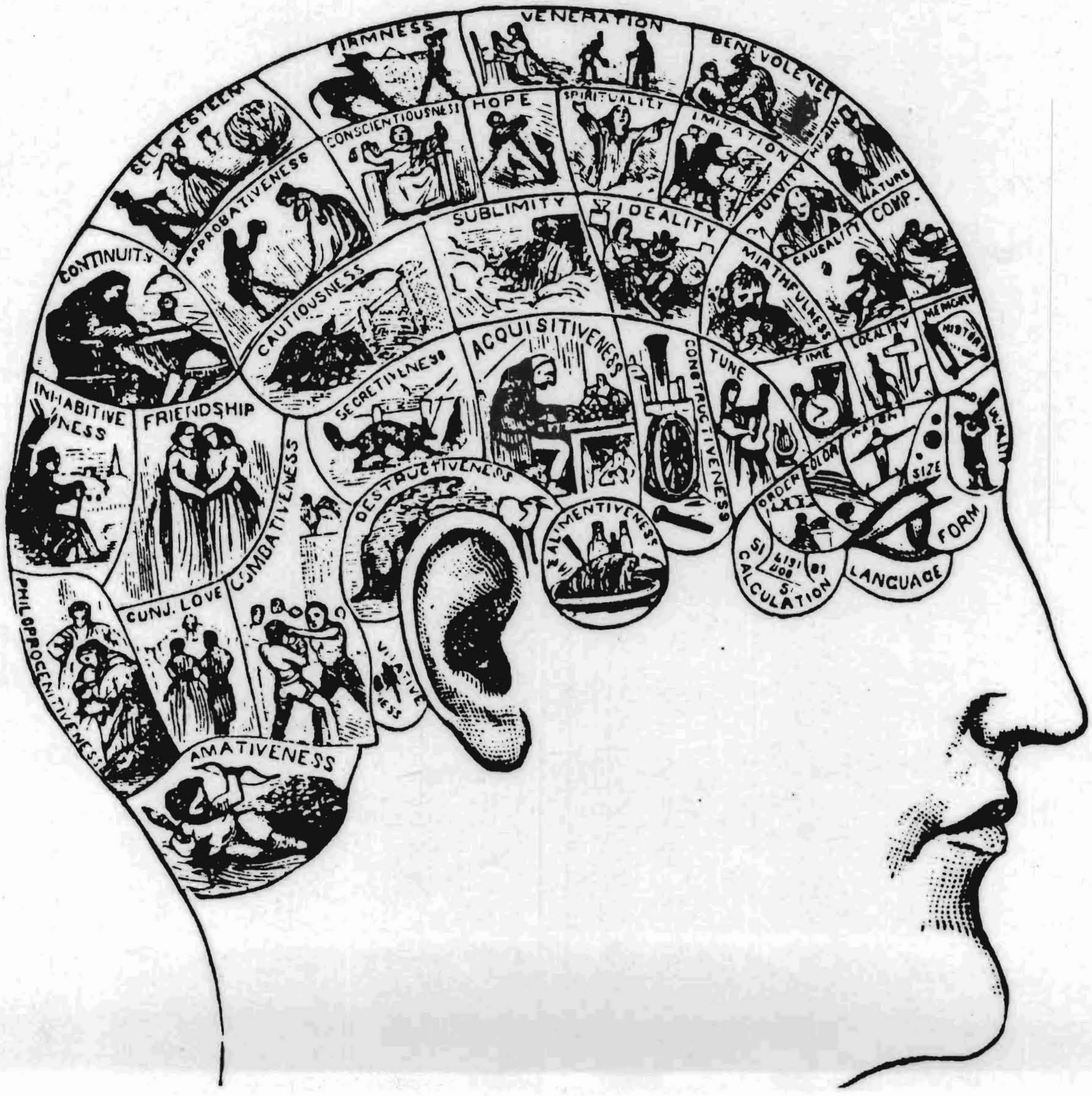
The Seventh Annual Conference of The Cognitive Science Society

Irvine, California
1985



Program of the

Seventh Annual Conference of the Cognitive Science Society

15-17 August 1985
Irvine, California

MACHINE UNDERSTANDING AND DATA ABSTRACTION IN SEARLE'S CHINESE ROOM

William J. Rapaport

Department of Computer Science and Graduate Group in Cognitive Science
University at Buffalo, State University of New York, Buffalo, NY 14260

1. INTRODUCTION.

In John Searle's Chinese-Room thought experiment, Searle, who knows neither written nor spoken Chinese, is locked in a room supplied with instructions in English that provide an algorithm allegedly for understanding written Chinese. Native Chinese speakers outside the room pass questions written in Chinese characters into the room; Searle uses these symbols, otherwise meaningless to him, as input and, following only the algorithm, produces, as output, answers written in Chinese characters, passing them back to the native speakers. The "answers . . . are absolutely indistinguishable from those of native Chinese speakers" (Searle 1980: 418). The experiment is used to support the following argument:

> [I] still don't understand a word of Chinese and neither does any other digital computer because all the computer has is what I have: a formal program that attaches no meaning, interpretation, or content to any of the symbols. [Therefore,] . . . no formal program by itself is sufficient for understanding . . . . (Searle 1982: 5.)

Many have disagreed over what Searle's argument is designed to show. The version I have just cited is clearly invalid: At most, the thought experiment might show that that particular program is insufficient for understanding, but not that a program that did attach meaning, interpretation, or content to the symbols could not understand. Such an attachment would also take the form of an algorithm (cf. Rapaport, forthcoming). But Searle denies this stronger claim, too:

> I see no reason in principle why we couldn't give a machine the capacity to understand English or Chinese, since in an important sense our bodies with our brains are precisely such machines. But . . . we could not give such a thing to a machine . . . [whose] operation . . . is defined solely in terms of computational processes over formally defined elements. (Searle 1980: 422; cf., also, the "robot reply", p. 420.)

And this is so because "only something having the same causal powers as brains can have intentionality" (Searle 1980: 423). These causal powers are due to the (human) brain's "biological (i.e. chemical and physical) structure" (Searle 1980: 422). The ***biological*** stance taken by Searle is essential: For even a simulated human brain "made entirely of old beer cans . . . rigged up to levers and powered by windmills" would not really exhibit intentionality (Searle 1982: 4), even though it appeared to. Searle, however, does not specify precisely what these causal powers are, and this is the biggest gap in his argument.

However, in his book, *Intentionality*, Searle tells us that "mental states are both ***caused by*** the operations of the brain and ***realized in*** the structure of the brain" (Searle 1983: 265), so we might hope to find an explication of these causal powers here. Indeed, a careful analysis of these two notions reveals (1) what the requisite causal powers are, (2) what is wrong with Searle's claim about mental states, and (3) what is wrong with his overall argument.

Moreover, it is consistent with my analysis that *some* intentional phenomena, e.g., pain and other qualia or "feels", ***need not*** be functionally describable, but, rather, might be the results of physical or chemical properties of the entity that experiences them. However, though these phenomena ***need*** not be functionally describable, they very well might be.

My theory, in rough outline, is this: Consider Searle's beer-can-and-windmill simulation of a human brain, programmed to simulate thirst. Searle says that it is *not* thirsty. We might reply that perhaps it feels *simulated* thirst; and we might then go on to wonder if simulated thirst *is* thirst. Even better, we should say that it *simulatedly* feels simulated thirst. Similarly, the Chinese computer system simulatedly understands simulated Chinese. But, so goes my theory, the simulated feeling of simulated thirst *is* thirst, and simulated understanding *is* understanding. The differences between such simulations and the "real" thing—or, more to the point, the *human* thing—lie in their physical make-up. The thirst implemented in the beer-can computer may not "feel" the way that thirst implemented in a human feels (what, after all, is it like to be a thirsty beer-can computer?), but they are both *thirst*. And so for understanding.

2. QUESTIONS ABOUT CAUSATION AND REALIZATION.

We need to clarify what Searle means by "causation" and "realization". In the passage cited above, he says that it is brain *operations* that *cause* mental states, whereas it is brain *structure* that *realizes* them. This difference proves important. Yet, earlier in *Intentionality*, Searle answers the "ontological" question, "What is the mode of existence of . . . Intentional states?" in two ways: by saying that they "are both caused by and realized in [a] the ***structure*** of the brain" and [b] "the ***neurophysiology*** of the brain" (Searle 1983: 15, my italics).

But which is it? Operations and structure are, arguably, things that can be shared by brains and beer-can contraptions. Since Searle clearly does not want to commit himself to that, the answer must be neurophysiology. Very well, then. What does Searle mean when he says that intentionality is *caused by* the neurophysiology of the brain? He means that "Intentional states stand in *causal* relations to the neurophysiological" (Searle 1983: 15). But what does *that* mean?

And what does Searle mean when he says that intentionality is ***realized in*** the neurophysiology of the brain? I shall argue that he means that it is "implemented" in the brain, using a somewhat technical sense belonging to the computational theory of abstract data types; but that theory is more complex than Searle realizes.

3. DATA ABSTRACTION AND IMPLEMENTATION.

Computer programs describe *actions* to be performed on *objects* (cf. Lecarme 1984: 60-61). The actions are expressed in terms of the operations of the particular programming language used, and the objects are represented by *data structures*, each of which must be constructed out of the data structures available in the particular programming language. Data structures can be classified into different data *types*. An *abstract data type* is a formal (i.e., mathematical or abstract) data structure, together with various characteristic operations that can be performed on it (cf. Aho *et al.* 1983: 10-14). An *implementation* of an abstract data type is (usually) an actual data structure in a program, that plays the role of the abstract data type. This characterization is admittedly rough, but will serve my present purposes.

An example should help. A *stack* is an abstract data type consisting of potentially infinitely many items of information ("data") arranged ("structured") in such a way that new items are added only to the "top" of the stack and an item can be retrieved only if it is on the top. The usual image is that of a stack of cafeteria trays—the last item put on the stack is the first to come off. The programming language Pascal does not have stacks as a built-in data type, but they can be implemented in Pascal by arrays, which *are* built in. Unlike a stack, an item can be added to or removed from any cell of an array: But if one steadfastly refuses to do that and steadfastly treats an array in a last-in/first-out manner, then it is, for all practical purposes, a stack. Indeed, real stacks of cafeteria trays are more like arrays than stacks.

The relation between an abstract data type and an implementation of it is reminiscent of that between an Aristotelian genus or species and an individual of that genus or species, and implementation is reminiscent of instantiation—but not exactly: An Aristotelian individual has essential properties, namely, those had by the species to which it belongs. Its accidental properties are those that differentiate it from other individuals of its species.

But, whereas a stack has a top essentially, an array has one only accidentally. And two implementations of a stack can differ in more than merely accidental ways: Arrays are essentially different from linked lists, yet both can implement stacks. And stacks are essentially different from queues, yet arrays can implement them both.

These differences arise from the fact that not all properties of an abstract data type need be "inherited" by an implementation, nor does the abstract data type have to have all the essential properties of its implementation. For instance, stacks are infinite; arrays in Pascal are finite and of fixed size. Arrays that implement stacks *can* be accessed in the middle (even if they shouldn't be); stacks cannot. Finally, one abstract data type can implement another. Thus, e.g., the abstract data type *sequence* can be implemented by the abstract data type *linked list*, which, in turn, can be implemented by *symbolic expressions* in LISP. These can be thought of either as a "real" implementation, or as yet another abstract data type ultimately to be implemented by electronic signals in a computer.

Since these notions play an important role in my theory, a few applications of them to other areas may prove helpful. In doing this, I shall be likening certain things to abstract data types; I do not intend to argue that all of these things *are* abstract *data* types (although some are). Consequently, I shall use terms such as *Abstraction* to refer to the general category of such things.

When you "listen to music" on a record, are you *really* listening to music or merely to a recording—a simulation—of music? Clearly, both, because recordings of music *are* music. A musical score is an Abstraction that is implemented by, say, an orchestra. The relations between an abstract data type and an implementation of it are precisely those between a score and a performance of it. Even the "implementation" of an abstract data type by the mathematical notation used to describe it is paralleled by the relation between the abstract score and its printed copy.

Here is a mathematical example: Peano's axioms describe the abstract data type *natural numbers*. Any sequence of elements satisfying those axioms is an implementation of "the" natural numbers and, thus, *is* a sequence of natural numbers. Similarly, rational numbers can be implemented as equivalence classes of ordered pairs of (any implementation of) natural numbers.

And an example from science: Different collections of water molecules and collections of alcohol molecules are implementations of the Abstraction *liquid*.

And, more to the point, perhaps mental phenomena, like abstract data types, are Abstractions that can be implemented in different media, say, human brains as well as electronic computers.

4. REALIZATION AS IMPLEMENTATION.

We can now turn to what Searle means when he says that intentionality is "realized in" the brain. As a first approximation, let us say that:

(T1) *A is realized in B* means: *A* is an Abstraction implemented in *B*.

Since Searle claims that intentionality is also "caused by" the brain, we should inquire into the relationship between *being caused by* and our new understanding of *being realized in*.

Suppose, first, that *A* is caused by *B*. Is *A* realized in *B*, in the sense of (T1)? The motion of billiard ball #1 may be caused by the motion of billiard ball #2, yet #1's motion is not realized in #2's. Kennedy's death was caused by, but not realized in, Oswald. The *Mona Lisa* was (efficiently) caused by, but surely not realized in, Leonardo. *But*: The *Mona Lisa* was (materially) caused by Leonardo's canvas, and it surely is thereby realized therein. An American flag might be caused by Fourth-of-July fireworks, and thereby realized therein. And the simulation of a stack can be caused by the execution of a program containing arrays; the stack is thereby realized by the execution of the program. So, *being caused by* does not imply *being realized in*, but it is not inconsistent with it. It is possible for *A* to be caused by, but not realized in, *B*.

Suppose, next, that *A* is realized in *B*, in the sense of (T1). Is *A* caused by *B*? Consider a stack realized in an array. Is the stack caused by the array? This is a strange way to express it, but not a bad way: A *real* stack *was* caused to be by a real array. *But* it is an *abstract* stack that is realized in the array. And similarly for the *Mona Lisa* and the American flag. Thus, let us add to (T1),

(T2) If an Abstraction *A* is realized in *B*, then a real *A* is caused by *B*.

Moreover,

(T3) If an Abstraction *A* is realized in *B*, then real *A is B "under a description"* (or, "in a certain guise").

Is this being fair to Searle? Is his notion of realization the same as the notion of implementation, as captured by (T1)-(T3)? There is evidence to suggest that it is. Here, I shall merely note that (T1) is consistent with Searle's claim that "mental phenomena cannot be reduced to something else or eliminated by . . . re-definition" (Searle 1983: 262), as long as by 'mental phenomena' we understand something abstract. For to implement an Abstraction is *not* to reduce it to something (because of the only partial overlap of properties) *nor* is it to eliminate it (for the same reason, as well as because of the possibility of distinct implementations). I would suggest that "implementationism" might prove to be a more fruitful, less constraining viewpoint in the philosophy of science than reductionism has been.

Before pushing on, let me remind you of my purpose. Searle argues that a computer running the Chinese program does not understand Chinese, because only human brains can thus understand. Understanding, even if describable in a computer program, is biological; hence, an electronic computer cannot understand. On the other hand, I am arguing that there can be an abstract notion of understanding—a functional notion, in fact, a computational one—that can be implemented in computers as well as in human brains; hence, both can understand. I gain my leverage over Searle by making a distinction that is implicit in his theory, but that he fails to make. Just as an array that implements a stack does not do so in the way that a linked list does, so a computer that implements the understanding of Chinese need not do so in precisely the way that a human does. Nevertheless, they both understand.

The distinction that Searle does not make explicit can be discerned in the following passage:

> [M]ental states are as real as any *other* biological phenomena, as real as lactation, photosynthesis, mitosis, or digestion. Like these other phenomena, mental states are caused by biological phenomena and in turn cause other biological phenomena. (Searle 1983: 264; my italics.)

The use of 'other', here, perhaps begs the question. But it also suggests that by 'mental states' Searle means *implementations of abstract mental states*. I shall refer to these as *implemented mental states.*

We are now in a position to see where Searle is led astray. It is simply false to say, as Searle does, that one kind of thing, "mental states[,] are both *caused by* operations of the brain and *realized in* the structure of the brain" (Searle 1983: 265). Rather, it is one thing—an implemented mental state—that is caused by the operations of the brain, and it is something else altogether—an abstract mental state—that is realized in the structure of the brain.

For example, a "liquid", considered as an Abstraction, *can* be realized (implemented) in a collection of molecules. But what that collection *causes* is *actual* (or implemented) liquidity. Similarly, the Abstraction, "liquid-properties-of-water", can be realized (implemented) in *different* collections of water molecules, but the *actual* liquid properties of (some particular sample of) water are caused by *different* actual behaviors. Moreover, the Abstraction, "liquid" (*simpliciter*) can certainly be realized (implemented) in different collections of molecules (water molecules, alcohol molecules, etc.), and the actual liquidity of these molecules is caused by *different* actual behaviors; yet *all* are liquids, *not* because of the *causal* relationship, but because of the *realizability* relationship.

So, too, the Abstraction, "understanding" can be realized (implemented) in both humans and computers; *actual* understanding can be caused by both humans and computers; but, because of the realizability relationship, *both are* understanding.

5. THE RELATIONSHIPS BETWEEN CAUSATION AND REALIZATION.

In an analogy, Searle (1983: 269) offers an analysis of a combustion engine (see Figure 1), consisting of high-level phenomena causing other high level phenomena, each of which is both caused by and realized in corresponding low-level phenomena, one causing the other, respectively, as in Figure 2.

But Searle's diagram (Fig. 1) is not detailed enough; the complexity hinted at in (T3) needs to be made explicit. Searle's diagram should be augmented by a mid level analysis, as in Figure 3. More distinctions appear in Figure 3 than may actually be needed; we can, however, distinguish the following:

- a certain kind of relationship (causal, according to Searle)—call it "causation-1"—between the low-level phenomena and the mid-level phenomena;
- a certain (arguably distinct) kind of causal relationship—call it "causation-2"—between low-level phenomena;
- a certain kind of causal relationship—call it "causation-3"—between mid-level phenomena, paralleling (and possibly distinct from) causation-2;
- a certain kind of relationship (arguably causal)—call it "causation 4"—between high level phenomena, paralleling (and possibly distinct from) causation-1 and causation 2;
- a certain kind of relationship—call it "R"—between mid and high-level phenomena; and

— the relationship of realization (or implementation) between low- and high-level phenomena.

How are causation-1, causation-2, causation-3, and causation-4 related, and what is R? Searle, we have seen, conflates the several causations. Instead, as the second stage of my theory, I propose the following:

— R is the relation of being an instance of.

— There are several possibilities for causation-1. The simplest, perhaps, is to define it in terms of realization and R-inverse. Another is to take it as what Jaegwon Kim has called "Cambridge dependency"—the sort of "non-causal causation" involved when a sibling's marriage "causes" you to become an in-law. I favor a third interpretation: Castañeda's relation of *consubstantiation* (cf. Castañeda 1972: 13ff, 1975: 145f)—a relation, somewhat like co-referentiality, holding between intensional entities. In any case, causation-1 is *not* a kind of causation at all.

— causation-2 is ordinary, physical causation; and

— causation-3 is definable in terms of causation-1 and causation-2.

Thus, I propose that when a low-level device realizes (implements) a high-level Abstraction, it produces a mid-level phenomenon. A general theory of the relationships of causation (Searlean and otherwise), realization (or implementation), and low-, mid-, and high-level phenomena is presented in Figure 4.

We can now apply these distinctions to the Chinese-Room thought experiment. Machine understanding is, indeed, understanding, just as human understanding is: They are both instances of the more abstract (functional or computational) characterization of understanding. Machine understanding is caused-1 by a computer (or computer program) in which abstract understanding is realized; that is, machine understanding is an instance of abstract understanding, which is realized in a computer.

6. ON THE BIOLOGICAL CHARACTER OF INTENTIONALITY.

This way of looking at the issues clarifies the "biological naturalism" (Searle 1983: 264) that underlies Searle's claim that non-biological computers cannot exhibit intentionality on the grounds that intentionality is biological.

But why must intentionality be biological? Because, according to Searle, only biological systems have *the requisite causal properties* to produce intentionality. What are these causal properties? Those that are "causally capable of producing perception, action, understanding, learning, *and other intentional phenomena* (Searle 1980: 422; my italics). This is nothing if not circular. Moreover, to implement an abstract data type, it is only necessary to have the (physically) realizable properties of the abstract data type. (E.g., for an array to implement a stack, it is sufficient that it can store data and have a "top"; it need not be infinite.) And Searle has offered us no reason to think that intentionality, abstractly conceived, *could* not be implemented in a beer-can and windmill device.

But most importantly, the theory presented here shows that "the requisite causal powers" means causation 1, which is not real causation at all. The requisite causal powers, then, are simply the ability to realize a species of an Abstraction (or to be consubstantiated with an instance of an Abstraction). The present theory makes it clear that "*causality*" *is a red herring*. Only realizability (and perhaps consubstantiation) counts.

7. CONCLUSION.

The relationship between an Abstraction and its implementations underlies machine understanding of natural language (and of AI in general). It is unlike the more familiar relationship between species and individuals (or universals and particulars). Nor is it the case that implementations are "reductions" of Abstractions. Thus, there is no need to advocate a *reduction* of the mental to the biological, eliminative or otherwise. Rather, the mental is *implementable* in the biological, as well as in the digital-electronic. Implementability is a notion worth further study.

REFERENCES

Aho, Alfred V., John E. Hopcroft, and Jeffrey D. Ullman, *Data Structures and Algorithms* (Reading, MA: Addison-Wesley, 1983).

Castañeda, Hector-Neri, "Thinking and the Structure of the World," *Philosophia* 4(1974)3-40. Originally written in 1972. Reprinted in 1975 in *Critica* 6(1972)43-86.

__________, "Identity and Sameness," *Philosophia* 5(1975)121-50.

Kim, Jaegwon, "Noncausal Connections," *Nous* 8(1974)41-52.

Lecarme, Olivier, "Pascal," *Abacus*, Vol. 1, No. 4 (Summer 1984): 58-66.

Rapaport, William J., "Searle's Experiments with Thought," *Philosophy of Science* (forthcoming).

Searle, John R., "Minds, Brains, and Programs," *Behavioral and Brain Sciences* 3(1980)417-57.

_______, "The Myth of the Computer," *New York Review of Books* (29 April 1982): 3-6; cf. correspondence, same journal (24 June 1982): 56-57.

_______, *Intentionality: An Essay in the Philosophy of Mind* (Cambridge: Cambridge University Press, 1983).

Shoemaker, Sydney, "Functionalism and Qualia," in N. Block (ed.), *Readings in Philosophy of Psychology*, Vol. I (Cambridge: Harvard University Press, 1980): 251-67.

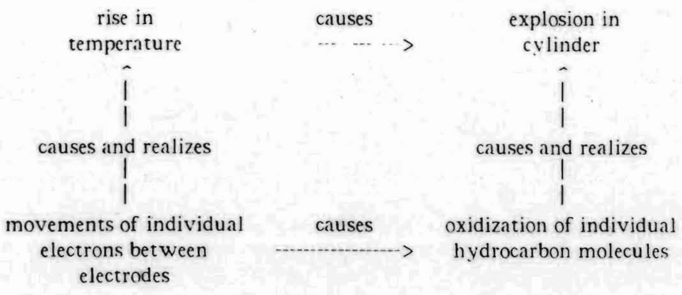


Figure 1

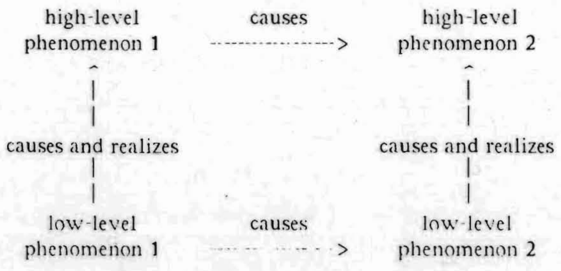


Figure 2

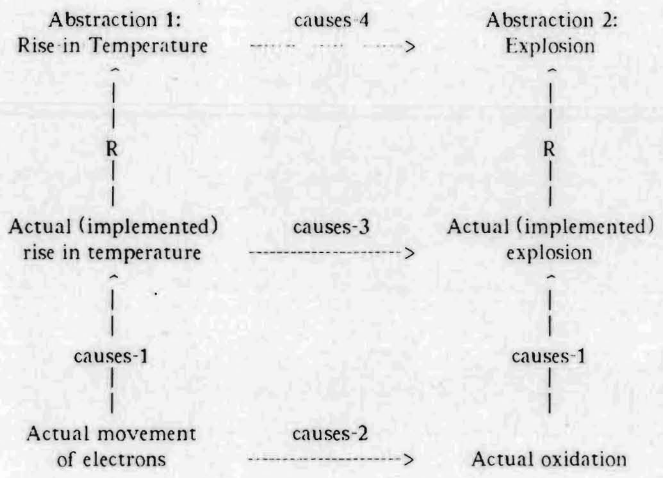


Figure 3

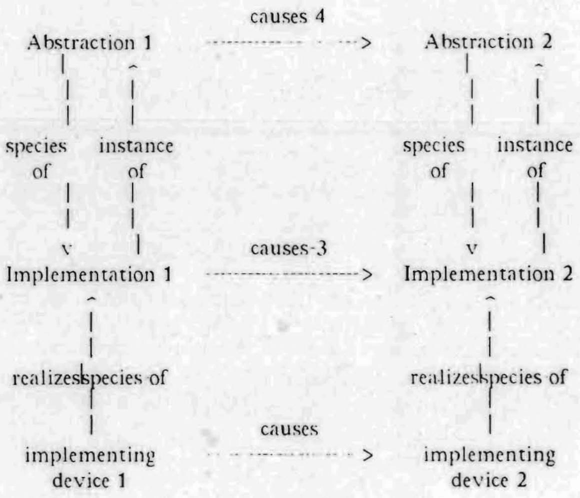


Figure 4