REPRESENTATIONS AS CONTROL SUB-STATES

Aaron Sloman
School of Computer Science
The University of Birmingham

(DRAFT VERSION mARCH 1994)

Abstract

Since first presenting a paper criticising excessive reliance on logical representations in AI at the second IJCAI at Imperial College London in 1971, I have been trying to understand what representations are and why human beings seem to need so many different kinds, tailored to different purposes. This position paper presents the beginnings of a general answer starting from the notion that an intelligent agent is essentially a control system with multiple control states, many of which contain information (both factual and non-factual), albeit not necessarily in a propositional form. The paper attempts to give a general characterisation of the notion of the syntax of an information store, in terms of types of variation the relevant mechanisms can cope with. Different kinds of syntax can support different kinds of semantics, and serve different kinds of purposes. Similarly concepts of semantics, pragmatics and inference are generalised to apply to information-bearing sub-states in control systems. A number of common but incorrect notions about representation are criticised (such as that pictures are in some way isomorphic with what they represent), and a first attempt is made to characterise dimensions in which forms of representations can differ, including the explicit/implicit dimension.

REPRESENTATIONS AS CONTROL SUB-STATES

Aaron Sloman
School of Computer Science
The University of Birmingham

(1) Introduction

This is a "position paper" introducing one key idea: The study of the nature and role of representations in intelligent systems can make most progress if based on the assumption that a mind (or a brain) is a sophisticated self-modifying control system with multiple control states, many of which contain information, of different kinds and in different forms (Sloman 1993b). A representation, then, is a sub-state in a control system. Different sorts of representations will be found in different control systems, and even within different sub-mechanisms in the same control system.

This idea, illustrating the "design-based" approach to the study of mind (Sloman 1993c), a development of Dennett's (1978) notion of the "design stance", is at odds with many common ideas of representation, which are usually abstracted from an inadequate survey of types of *external* representations (e.g. sentences and pictures). In particular, our analysis of representations as information-bearing control states undermines the idea that there are basically two kinds of representations (a) verbal/symbolic and (b) pictorial/analogical/iconic, and such notions as that representations can be unambiguously classified as declarative or procedural. By looking at the variety of types of information-rich sub-states in control systems, we'll find a much richer variety than such simple theories allow. (A recent survey, Narayanan 1993, includes both papers that assume some of these distinctions, and others that criticise them.) Further, the common assumption that mental representations are all consciously accessible is challenged by our analysis, since many of the information-bearing control states discussed below have nothing to do with consciousness. More surprisingly, the analysis of representations as sub-states in control systems also undermines the notion that symbols or representations are necessarily physical objects (as suggested by the "physical symbol system" hypothesis of Newell and Simon 1981), or even that every part of every representation must have a distinct underlying physical object as its implementation. Finally, this analysis leads to generalisations of the notions of syntax, semantics, pragmatics, and inference.

The "control system" viewpoint adopted here is also at odds with a standard notion of a "control system" that is limited to the kinds of systems normally studied by physicists and control engineers, in which the behaviour can be completely described by fixed systems of partial differential equations linking a fixed set of numerical variables. These conventional control systems all have a fixed degree of complexity, corresponding to a fixed set of dimensions or quantities that can vary, and a fixed collection of relations between these variables, usually expressed as equations or numerical inequalities.

By contrast, control systems that exhibit intelligence, such as humans and other hominids, appear to have architectures that are not only much richer, with far more functional differentiation between components than in standard control systems, but also do not have a fixed architecture, since the number and variety of components and connections between components can change over time. Furthermore, within such systems, many of the control states exhibit changes that are more like changing structures (e.g. trees and networks whose components and links change over time) than like changing values of numerical variables (e.g. voltage, pressure, velocity, etc.) The lack of a static architecture, and the use of information states that change in structure rather than in values of a fixed set of variables together imply that standard mathematics of control engineers will need to be enriched in order to cope with intelligent systems.

I am not suggesting that there is a sharp dividing line between intelligent and unintelligent systems, nor that we know in any detail how biological control systems actually work. Instead I am offering a theoretical framework within which different sorts of systems can be distinguished and classified, and which may turn out later to be useful for describing and analysing both natural and artificial behaving systems, in termsf of their architecture, their functional subdivisions, the types of information-bearing states and the types of causal interactions underlying their observable behaviour.

(2) The importance of “abstract”, or “virtual”, machines

These ideas do not fit in well with the notion of a “physical symbol system” because most of the representing structures that are of interest turn out not to be physical: they are not detectable or measurable by physical instruments, and they do not obey the laws of physics neither do they disobey them — the laws are irrelevant to them. The use of position of a screw to represent required temperature in a thermostat, or the state of a rotary “governor” on a steam engine to represent speed would be clear examples of physical symbols. By contrast many of the representing structures in AI systems are not physical, but are components of what computer scientists normally describe as “virtual machines” or “abstract machines” in the sense in which a programming language such as C, Lisp or Prolog defines a virtual machine with a specific ontology (permitted abstract data-structures) and behavioural capabilities (i.e. operations on those data-structures).

Typical software data-structures such as lists, networks, or arrays are not physical objects: their laws of behaviour are not those of physical objects: for instance it is commonplace for list **A** to contain **B** as an element whilst **B** contains **A** as an element, whereas such mutual containment is not possible for physical objects. If **cons** is the standard list constructor that creates a new list given an arbitrary item **X** and an existing list **L**, and **head** and **tail** are the standard functions for accessing the first element of a list and the rest of the list, then examples of such non-physical laws would be the following, which are not laws of physics, but much more like laws of mathematics:

$$\mathbf{head(cons(X, L)) = X}$$

$$\mathbf{tail(cons(X, L)) = L}$$

Computer programs typically manipulate states in *virtual* machines, e.g. machines containing numbers, strings, arrays, lists, tables, procedures, etc. rather than states in a physical machine, like voltage, current or physical location. Even machine code instructions manipulate bit patterns in abstract address spaces, rather than physical objects. Of course, the virtual machine processes are implemented using physical processes. However, there need not be any one-to-one mapping between virtual machine states or structures and those of the underlying physical machine. For example a very large sparse array in a computer can contain far more cells than there are atoms in the underlying physical machine, or even in the universe. (Sparse arrays use a technique in which array locations containing a “default” value are not explicitly recorded, whereas those with a non-default value are. This can save a lot of space if the vast majority of locations contain the default value. Whether a very large array uses this technique or explicit storage of the contents of all locations may be impossible to tell simply by accessing the array: the two implementation techniques can provide functionally equivalent abstract data-structures. However, if the array contains more locations than could be accommodated in the physical universe it must use some mechanism other than explicit storage.) Similarly, a logic-based database containing a theorem prover could contain infinitely many items of information, without using infinitely many physical components!

Virtual or abstract machines have many properties that make them suitable for building behaving systems that need to be able to take in and process complex and rapidly changing information. A network in an abstract machine, containing many nodes with many links, can be constructed or reorganised far more rapidly than any physical network of comparable complexity, which is one of the reasons why virtual machines are so important for intelligent systems. For example, by using *pointers* to complex structures it is very easy to have the effect of many copies of the same structure in different locations, and then to produce the effect of changing all copies simply by changing the single structure they all point to.

Nevertheless all the virtual machines to be found in actual behaving systems are (ultimately) *implemented* in physical machines, be they brains or computers. A virtual machine need not be directly implemented in a physical machine: it could use a simpler abstract machine. For example, the virtual machine corresponding to a high level programming language is typically implemented in terms of the lower level abstract machine corresponding to the "instruction set" of the host computer, such as a VAX, or HP-PA, or SPARC, all of which are themselves abstract machines capable of being implemented in physical machines in different ways.

This relationship of "implementation" that can hold between machines at different levels of abstraction, and between an abstract and a physical machine, could turn out to be one of the most important contributions of computer science to the study of mind: far more important than the concept of an algorithm, for example, even though the latter has received more attention (e.g. Penrose 1989). For now I shall assume that the reader is familiar with the typical hierarchies of implementation levels that can be found in computer-based systems.

Even if biological systems are very different and far more complex and subtle, it can be useful to think of implementation hierarchies in computers as providing a relatively simple, well-understood illustration of the concept of a behaving system that needs to be understood as having its behaviour controlled in part by an abstract machine. This is especially true if we consider internal as well as external behaviour: for the internal behaviour that is of interest in understanding problem-solving or goal-directed actions in humans is generally the behaviour of abstract, not physical machines. Processes of inference, reasoning, interpretation are not physical processes.

I am not saying that it is *only* these complex abstract machines that have information-bearing sub-states. Even a simple thermostat controlling a room heater has a sub-state that can be thought of as representing the current ambient temperature and another sub-state corresponding to the required temperature. These sub-states can vary independently of each other, and they have different causal roles in the system, but they are part of an integrated system that behaves as a whole. Thus, although the control systems that are of most interest in the study of mind are far more complex than this, it can still be illuminating for a survey of control systems to include the simpler systems, including systems that can easily be represented by a fixed set of physical measurements and a fixed set of equations linking them. Limiting cases can be part of a concept even though most instances are very different, just as a circle is an interesting limiting case of the concept of an ellipse, despite not having a major and a minor axis, and zero is a limiting case of the concept of a number that can be the result of counting or measuring.

(3) Causal relations in abstract machines

A full discussion of the causal relations between processes in virtual and physical machines would require a lengthy paper on its own: I am going to assume the common sense view that computer events best described in software or program terms can cause events both in the virtual machines they manipulate (e.g. adding a new subnet to a network, or re-formatting a page of text) and in the underlying physical processes (e.g. getting many transistors to change their state). More obviously, physical events, such as arrival of an electrical signal at an interface can cause software events, such as transfer of control to a new sub-program or creation of a new data-structure such as a character string. In other words, causal influences can cross implementation boundaries.

Philosophical objections to this assumption tend to use arguments claiming that only physical events can be real causes or effects. There are many problems with such objections, partly because they unjustifiably assume that there is a unique and well-defined *physical* layer of reality, and also because they imply rejection of most of our patently useful ordinary ways of talking about causation, e.g. when we describe social events as causing political events, or bad news as causing distress and distress as causing a flood of tears.

The claim then is that sub-states of a control system contain information, and can be described as being "representations" in a new technical theory-based sense of the word that is intended ultimately to replace and explicate the ordinary notion of a representation. More precisely: each state of a component of a control system is a representation-instance, and the range of states that are possible for that component defines a representational system, or what Donald Peterson (1994) calls a "form of representation", more commonly referred to as a "notation" or "formalism". A thermostat uses particularly simple formalisms, because the sub-states of a thermostat that are relevant to its function vary only in particularly simple ways, whereas some other control systems, such as computer operating systems, use far more complex formalisms in their information-bearing sub-states.

(4) Preliminary overview

My ideas on all this are still in a half-baked form, and this paper does not so much report results as invite contributions to an ongoing investigation. The basic idea that representations are control states, or aspects of control states. can be elaborated as follows.

- Many complex behaving systems have diverse independently variable interacting sub-states, with different functions (different causal roles). Sometimes different sub-states are in physically separate sub-mechanisms, and sometimes they are superimposed in one mechanism (like superimposed wave-forms).
- The functional differentiation between different sub-states is typically far greater for systems that we would regard as intelligent than for non-intelligent systems (examples of both are given below).
- Some of the sub-states "contain" information, used for different purposes: sensory buffers, perceptual summaries, long term memories, plans, feedback control signals. I call these "information" states. At this stage there is no presumption about *how* the information is embodied in the state: different embodiments are to be found in different sorts of plants, animals and machines. In a simple thermostat, information about temperature might be embodied in the degree of curvature of a bi-metallic strip, whereas quite different embodiments can be found in more sophisticated controllers and in biological systems, including chemical states, neural states and software states.
- The notion of "information" used here will not be defined in advance. This is one of those many concepts that can only be made completely clear in the context of a fully developed explanatory theory (in deep science definitions come last, not first). While we lack a comprehensive theory characterising control architectures, the kinds of sub-states that can occur within such architectures, and the kinds of causal roles such substates can fulfil, we cannot yet give a precise definition of "information-bearing sub-state". However, it is easy to give examples of machines and organisms with sub-states that contain information about the current environment, information about previous events, information about states to be achieved, information about how to achieve or prevent or maintain states of affairs, and information about what to do next. In each case the sub-state contains information *about* something, information that can influence processes within the same or other sub-components of the architecture.

- A full theory of representations would account for the different kinds of information that can be contained in control states, and the different roles they can play. This can be broken down further into: the different forms in which the information can be expressed (syntax), the different kinds of things that can be referred to or expressed (semantics), the different operations that can be applied to the information (generalised inference) and the different uses to which the information states can be put (pragmatics). (These notions are all elaborated below.)
- Which concepts are applicable to a particular system will depend on its architecture: in a simple architecture such as that of a thermostat a sub-state (e.g curvature of a bi-metallic strip) can be said to contain information about the ambient temperature, but there is no differentiation of the environment into things with properties and relationships, and in particular no representation of any *thing* as having a temperature. Moreover, the use to which the information can be put is very limited and unvarying. By contrast, in a robot with a visual system whose architecture is similar to that sketched below, there is information about the environment at various levels of abstraction, based on considerable amounts of interpretation in the light of prior information, and several potential uses to which the same information can be put. For instance, the same item of information could on some occasions be used to check whether a goal has been achieved, to trigger a new goal, to confirm a hypothesis, to interpret new sensory data, or to derive new factual information via a process of reasoning.
- We must distinguish the representational needs of a designer concerned with building, modifying, maintaining or describing a system, and the needs of the system itself. Computer scientists, and some AI theorists concentrate more on notations for use by researchers and designers. I am more concerned with run-time representation in control states of working systems, not design-time or compile-time formalisms, nor external description. (Of course, where the working system is itself a designer or engineer, or attempts to understand itself, both sorts of representation will play important roles.)
- I believe that when we fully understand design principles for sophisticated control systems, a great many philosophical problems about the nature of mind, language, representation, etc. will be transformed: though not necessarily solved, for in many cases the transformation will consist in the demonstration that the original questions were based on confused or false assumptions, made use of muddled concepts, etc. The transformed questions will then be capable of being answered on the basis of either theoretical (e.g. mathematical or logical) analysis or empirical investigation. (This does not imply that the original questions were of no value: muddled ideas can drive advances that lead to their own replacement.)
- In particular, I expect that we shall discover that many current disputes about the relative merits of "symbolic" and "connectionist" models are as muddled as old investigations attempting to find out which combinations of earth, air, fire and water constituted different known substances.
- When we have a good overview of design options and the different functional roles that sub-mechanisms and control states can have, this will provide a new basis for a conceptual taxonomy for talking about types of representations, which will be much deeper and richer than current naive distinctions such as declarative/procedural, verbal/pictorial, symbolic/connectionist.

From this standpoint the questions that lead to deeper understanding are *design* questions: questions about how to design intelligent, sentient, autonomous agents, with their own desires, goals, and so on. Design questions need to be addressed in the context of *requirements* (Sloman 1993c). And at present we understand little about the requirements driving the design of intelligent agents, whether in the laboratory, or in biological evolutionary processes.

(5) Avoid simplistic dichotomies

People who study representations tend to examine only a small range of cases, and as a result they produce shallow or simplistic distinctions: verbal/visual, analog/digital, symbolic/sub-symbolic, procedural/declarative, explicit/implicit etc. An examination of the history of mathematics, science, engineering, and culture reveals a much greater diversity of types, including many ad-hoc notations, such as the standard arabic notation for numbers in which concatenation representations multiplication by 10 and addition, a type of rule that has no application outside of arithmetic. From a design standpoint we need to understand why diverse notations are to be expected: the answer lies in the variety of functions fulfilled by different information-bearing sub-states in a sophisticated control system (see also Sloman 1985a).

There are many different trade-offs in designing and selecting types of information-rich control states (i.e. representations) for "run-time" use, and these lead to a wide variety of design options, far more than the common categorisations reflect. For example, so-called verbal or symbolic notations often use "pictorial" relationships, such as ordering, to represent similarly structured relationships. "She shot him and drove away" usually implies a different temporal ordering from "She drove away and shot him", and to that extent such verbal forms include a pictorial or iconic function, undermining the common distinction between the two.

Moreover, many programming languages that are normally described as "procedural" make it possible to use data-structures as information stores that are interpreted, just like "facts" in allegedly declarative languages. At the same time, supposedly declarative languages like Prolog usually provide means for expressing not only what is to be done but also in what order various steps should be followed, and to that extent it is as procedural as any other language, except that the mappings from syntactic ordering to process ordering is complicated by backtracking. The terminological confusion is not helped by those who apply the word "declarative" to functional programming languages like Scheme.

A first step towards getting a deeper understanding of the role of representations in intelligent (and non-intelligent) behaving systems is to understand the requirements they may need to satisfy. Some requirements concern their syntactic richness, others concern their manipulability, others concern speed with which they can be created, changed or used, and so on. To illustrate the variety, below are some examples of systems that would not normally be described as intelligent and some that would be. (I am not suggesting that this is a clear distinction based on generally agreed criteria.)

(6) Examples of information states in control systems

The following would normally be described as components of non-intelligent systems:-

1. The state of a temperature sensor in a thermostat.
2. The state of the temperature control in a thermostat.
3. The state of the accelerator, choke, steering wheel or brakes, in a car.
4. The /etc/rc and /etc/rc.local start-up scripts and many other system configuration files in a typical Unix operating system.
5. The internal, constantly changing, tables and processes in a running operating system.
6. Information about the state of network links in a networked communication system.
7. A staff payroll database.

The following would normally be described as involving intelligence:-

1. Current visual (or other) percepts in a person or other animal.

2. Particular memories of previous situations, things, etc. in a person or robot.
3. General beliefs (about the environment, oneself, etc.), e.g. that unsupported objects fall, or that people who smoke tend to smell unpleasant.
4. Motives, desires, preferences, attitudes (affective states at various levels of abstraction, and different degrees of generality and persistence).
5. Personality (a collection of very high level dispositions).
6. Intentions, plans (long term, short term, current, future).
7. Information underlying motor skills (e.g. violin playing).
8. Information underlying linguistic skills (phonological, syntactic, lexical information, etc.).
9. Current information in various motor-control feedback systems, e.g. posture control, hand-movements.
10. Internal self-monitoring states (internal records of bodily states, mental states, dispositions, etc.).

These are merely illustrative of the variety of types of information states that we need to understand. Some are short term, some enduring long term states, some mostly passive (changed by other things), some active (i.e. *sources* of change), others mediators or modifiers of processes that they do not themselves initiate, some directly manipulable by other mechanisms and others only indirectly modifiable (e.g. by training), some consciously accessible, some not. In human beings the vast majority of information states are not consciously accessible.

I am not assuming that there is a sharp and well-understood distinction between intelligent and non-intelligent systems, and I shall say nothing about how that distinction might be clarified, for I think it will eventually be replaced by a whole family of different distinctions concerned with presence of absence of different collections of capabilities: taxonomies are usually more useful than dichotomies.

(7) What is a form of representation?

Information of all those kinds can be embodied in states of complex systems using different notations, or “forms of representation”. Confusion can follow from the common practice of referring to a notation or formalism, such as predicate calculus, or algebra, as a “representation”, as well as individual instances. I shall try to avoid this ambiguity by restricting the word “representation” to *instances*, and will use the expression “formalism”, or “notation” or “form of representation” to refer to the general forms that individual representations are instances of. In this terminology, predicate calculus would be a formalism or notation but not a representation, whereas a particular predicate calculus expression would be a representation. The word “symbol” is also used to refer to particular instances, usually those that lack meaningful internal structure, while “symbolism” is sometimes used as a synonym for “notation” or “formalism”. The word “language” is fairly close to what is here described as a notation or formalism, though it is often thought of as restricted to external forms of communication between complete intelligent agents, whereas what I am talking about could be used internally for communication between parts of a behaving system, and not necessarily an intelligent system.

What I am here calling a “formalism”, “language”, or “notation” will normally have syntax, semantics, pragmatics, and inference rules which determine the consequences of transforming one state into another. Not all notations need have all these features: e.g. a notation used entirely in control signals need not include any inference capabilities. In some simple cases there may be no clear distinction between pragmatics and semantics.

We must take care to interpret these ideas in a sufficiently general way. Features of “external” notations can mislead us into adopting over-simplified theories of representation. For example, notations in a computer or brain may use a topology that cannot be directly represented on paper or any other two dimensional surface. We can aim for full generality by considering how to design behaving systems. From this “design” perspective, a notation can be thought of as: <u>a set of possible information states that can occur in control systems.</u>

It is possible for different notations to be used by different sub-states of the same complex system: I shall try to show that intelligent systems typically require many different kinds of notations or forms of representation, corresponding to different functional roles played by different sub-states.

The relationship between representation and underlying physical structure can be quite unobvious. Often a representation or symbol will be implemented using a perceivable physical pattern or structure, but the physical object does not uniquely determine the representation or symbol. In different contexts the same physical pattern can be an instance of different notations, like the joke utterances that can be interpreted as saying entirely different things in different languages.

Below I shall try to show how the familiar concepts of grammar, pragmatics, semantics, and inference can be generalised to correspond to this general notion of a representation, though not all examples will exhibit the full richness of these concepts. It is common in connection with natural languages to contrast syntax (grammar), semantics (meaning) and pragmatics. There are considerable difficulties in making these notions very precise in their full generality. However, attempting precision at this stage would be premature: we first need a full theory of information-bearing substates and a survey of their possible structures and uses. Within such a theoretical framework it should be possible to give more precise and systematic taxonomies and definitions than we can offer now. Nevertheless we can already offer an approximate first-draft analysis.

(8) The syntax (or grammatical form) of a control state

A notation in this general sense corresponds to a “set of possible sub-states” of a behaving system, just as a conventional grammar determines a set of possible sentences. In some cases, like the temperature-representing sub-state of a thermostat, the set of possible states has a very simple structure, e.g. a linear continuum. In other cases it is far more complex, like the set of possible states of the “start-up” files of an operating system, the set of programs permitted by a programming language, or the set of sentences in a natural language. In this context it is useful to generalise the notion of “grammar” to refer to the set of structures of such states. This is a natural generalisation of the normal linguistic notion of grammar as determining structures of possible sentences, each of which has a specific grammatical form, or syntax.

One difference between our standpoint and that of linguistics is that linguists tend to think of sentences as static entities whereas inside a behaving system representational states need to change, for instance during reasoning, learning, planning, perceiving, as well as during control processes with feedback. So actual variability of representations is important, as opposed to the mere variety of possibilities for static structures. The *kind* of variability that is permitted depends on the grammar. (For now we leave open how to define the boundary between structure and content of a state. It is not very important for us whether grammar of a language incorporates all the individual lexical items or only classes of such items. In the latter case, the actual words in a sentence would not be part of its syntax, only their lexical classes would be. Similarly, it is not very important, for present purposes, to say whether two states of a thermostat corresponding to different temperatures have different syntax, or the same syntax but different content.)

Although I use the words “grammar” and “syntax” almost interchangeably I shall endeavour to restrict “grammar” to refer to the characterisation of a whole class of possible structures, and “syntax” to refer to the structure of an instance. In this sense a *language* has a grammar, whereas a *sentence*, or *phrase* or *clause* has a syntax. The syntax of an instance and its grammatical form or grammatical category are the same thing. The grammar of a language is the same thing as the set of possible syntactic forms that can occur in the language.

Syntax is not a physical property of a representing structure, but depends on the capabilities of the system that uses that structure and other structures. As previously remarked, physically identical states need not implement representations with the same syntax. A particular string of characters in a language, e.g. "**x+y**" does not uniquely determine the underlying grammar of the language: the same string can occur in different languages with different grammars. In many programming languages that would be a string of three symbols, e.g. two variables separated by an infix operator, whereas in space-delimited languages like LISP it could be a single symbol composed of three characters. Similarly, you cannot infer the underlying notation, or representational formalism, by inspecting the physical form of a particular representational state. What determines the syntax (or grammar) of an information-rich sub-state is what kinds of variation in that state are possible and significant in relation to controlling internal or external behaviour. (This is one reason why introspection concerning the nature of thought processes or mental images can be totally misleading: inspection does not show what variation is possible, nor what the effects are.)

Typically what the syntax of some information state is will be determined by whatever mechanism actually uses those states and the variety of possibilities that it can cope with. The system need not use any *explicit* formalisation of the grammar. (When there is an explicit grammar, e.g. in a grammar-driven parsing program, the syntactic specification will generally use a different notation, with a different syntax from that which it defines: e.g. a formalism other than English can be used by linguists to specify the grammar of English.)

A further complication is that any particular information-bearing sub-state will generally have a syntax at more than one level of abstraction. English sentences have a syntax that is independent of whether they are written or spoken. However, at a lower level, spoken English has a structure in terms of phonemes (or possibly other units of sound-structure) whereas written English has a structure in terms of sequences of letters and spaces and punctuation marks. The lower level syntax is rich in generative power, in that there are far more potential words composed of sequences of letters, or sequences of phonemes, than actually exist in the language. This is not a trivial detail: it is part of what accounts for the ability of a language to evolve. At a still lower level it is arguable that there is yet another form of syntactic structure corresponding to the fact that certain stroke patterns are available for forming letters, and in principle additional letters could be constructed as needed. At that level the syntax is different for different font styles.

Within a computer there are similar levels of structure. For example, lists trees and networks can be constructed out of lower level units that are made of sequences of "bits". In a system with error-correcting memory the bit patterns themselves may be made out of more complex sequences of bits in terms of which they are encoded.

I hope that all this shows that linguistic grammars and the sentences they generate are only special cases of a more general concept. There are very many distinct notations, or representational formalisms, each with its own sets of instances, its own properties, potential uses, etc. For example, conventional grammatical notations cannot capture the syntax of forms of representation that allow *continuous* variability, or structures with mutual containment (A contains B and B contains A), as can happen with lists in a computer.

I have previously (Sloman 1971, Sloman 1975) analysed some of the syntactic differences between applicative or logical representations (where the key syntactic relationship involves "application" of a function sign to argument expressions) and what I then called analogical representations, in which they key syntactic relationship is the holding of a relation within the representing medium. Applicative syntax allows a very rich variety of forms of semantic interpretation, since arbitrary procedures can be associated with function names to compute denotation of complete applicative expressions on the basis of denotations of their components, for instance computing the truth values of the sentences “Mary is richer than Tom”, “Mary is taller than Tom”, “Mary is cleverer than Tom”. In the simplest analogical representations syntactic relationships within the representing medium represent relations in the domain depicted, but this does not imply that there is any isomorphism, as 2-D relationships can depict 3-D relationships. In some cases the mapping from relations in the representation to represented relationships is deterministic (e.g. in most conventional maps). In other cases the representing relations are locally ambiguous and therefore the mapping is highly context sensitive and finding a consistent interpretation requires problem solving or search. (A lot of AI vision work has been based on this fact.) Nothing said so far should be taken to imply that there are only applicative and analogical forms of representations. Many others exist, and moreover, these two can be combined, for instance when the order of sentences depicts the order of events.

Sometimes properties of the representing medium constrain what can be represented because of the relatively direct semantic relationships used. Thus what can be depicted in a 2-D surface using an analogical representation will be limited by the variety of relationships available in that surface, relationships such as neighbourhood, direction, distance, containment, connectivity, and so on. A still wider variety of relationships is available within the class of 3-D structures, but in general it is more difficult for us to create and change them, and to grasp their structure (for instance one part of the structure can obscure another). The use of datastructures containing pointers to other datastructures in a computer provides some of the benefits of pictorial syntax since pointers can be allocated classes, giving analogues of closeness, direction and other spatial properties, without the constraints of 2-D or 3-D spaces, though at the price of loss of continuity (since datastructures are essentially discrete), and the need for specialised procedures for traversing and manipulating links. At this stage it is not clear to what extent human brains use this kind of virtual machine. They are particularly useful for processes that require temporary construction of trees or networks. I have also previously suggested that some aspects of the development of children’s counting skills could be explained by the construction of networks of datastructures. (Chapter 8, in Sloman 1978).

(9) Pragmatics of a control state

The concept of pragmatics used in connection with natural language is somewhat ambiguous, as there are both narrow technical notions and broader more intuitive notions (see Gazdar 1979). Some of the technical linguistic concepts of pragmatics are closely bound up with the notion of linguistic context within which speaker and hearer share presuppositions that can provide a framework for disambiguation of referring expressions, for instance pronouns such as “it” or “she”. Although these problems of “indexicals” are normally associated with communication in an external language between intelligent agents, analogous situations can occur within components of a machine, for example when the instruction address register in a computer is to be interpreted as *relative* to the beginning of the currently executing procedure, whose address is in a different register.

We can generalize further by adopting a notion of pragmatics which is concerned with the *purposes* for which utterances are used, or their functional roles within the larger system. Thus questions, assertions, requests, commands and promises all have different communicative purposes, and the general notion of pragmatics of a notation can be defined as the set of possible purposes for which it is used, together with the rules that link forms of expression to purposes. These rules are often very loose in natural languages, since, for example, utterances in the form of assertions (“Someone left the door open”) and questions (“Can you reach the door handle?”) are often intended, and interpreted, as requests or commands (“Please shut the door”). As such examples show, pragmatic roles of linguistic expressions depend not only on their syntactic form but also on context, the source of the utterance, and relationships between speaker and hearers. The broad study of pragmatics includes analysing the way in which syntactic forms, context, the capabilities and relationships of utterers and receivers help to determine which uses are possible and under which conditions they occur.

We can further generalise this notion of pragmatics to cover the different functional roles that information-bearing sub-states of a complex control system can have, without assuming that autonomous intelligent language-using agents are involved. This general notion includes effects on other sub-states, and the ways in which differences in syntactic forms and other factors contribute to those effects. The notion of "purpose" or "function" that I am using does not require a human designer or any conscious intention. It assumes only that we are talking about an integrated system in which the various components somehow co-operate in a systematic way to produce behaviour. The function, purpose, or role of a component is then defined in terms of how it contributes towards the overall effect, whether directly or indirectly. This seems to be how the concept of anatomical or physiological function is used biology, though in some cases biologists make the (in my view unnecessary) claim that their usage is justified by evolutionary selection of a design. Although I shall not argue over this here, I regard that justification as spurious since what happens now is sufficient justification for talking about the function of the heart or liver, no matter how organisms came to have hearts and livers. In exactly that sense we can talk about the function of the sun in relation to life on earth or weather patterns.

Effects that are relevant to the pragmatics of a control subsystem include such functions as:

- initiating a new process,
- terminating an existing process,
- suspending or interrupting processes,
- modifying processes,
- altering the information stores that control processes,

and many more. Within traditional control theory, notions such as positive feedback, negative feedback, amplification and damping are all concerned with this generalised conception of pragmatics. If we allow a richer variety of control states, and a richer variety of causal interactions then our notion of pragmatics will be correspondingly richer. For example, in a typical control mechanism made up of a fixed set of measurable states each of which admits continuous one-dimensional variation, where all the interactions can be expressed in a set of partial differential equations, the notion of one subsystem asking another a question makes little sense. However, where sub-systems include factual databases that can be interrogated by, or given new information by, other sub-systems we can find analogues of assertions and questions. Simplified analogues of commands are available in many mechanisms where a signal sent by one component can initiate behaviour in another. More complex sorts of commands, closer to linguistic commands, are possible where the control signals have internal structure determining semantic content and can vary in complexity, as sentences can.

On the basis of this simple introduction, readers familiar with modern computing systems, especially networked systems, will easily think of a wide variety of functional roles that could be used to provide a taxonomy of types of pragmatic roles for sub-states in complex behaving systems, including analogues of the problems of disambiguating ambiguous signals on the basis of shared presuppositions, the use of "indexical" expressions similar to "here", "now", "it" and similar things. One of the characteristics of computing systems, unlike previously known mechanisms, is that they were originally designed to take over some human mental tasks, and therefore it is not at all surprising that they should provide much richer illustrations (and potential explanatory models) of familiar concepts of representations than other sorts of mechanisms do.

(10) Semantics of a control state

Analysing the notion of meaning, or semantics, is a notoriously difficult philosophical problem, partly because there are several different concepts with quite complex relations between them. For instance Frege (following Mill and others) attempted to make a distinction between sense (connotation) and reference (denotation), but it turned out a far more slippery task than he at first thought (as he found out when he later began to analyse various tricky special cases, such as the use of the pronoun "I"). This is no place to attempt a full analysis of possible notions of meaning and reference. However, I claim that there are no well-defined concepts that correspond to our normal use of these words. Rather there are many different features to be found in the contexts in which we ordinarily talk about meaning, and different subsets of those features can occur in connection with control states of various kinds of machines. There is no "correct" set of features that uniquely justifies the attribution of meaning. Rather there are different subsets with interestingly different properties, and instead of arguing about which features correspond to *the* concept of "having a meaning" we should explore the similarities and differences implied by different subsets.

I have previously tried to show (Sloman 1985b) that a significant subset of features characteristic of uses of symbols with meaning can be found even in the way in which a computer "understands" its machine language (i.e. without the need for any AI programming). For example, in typical machine languages certain bit patterns are used (by the machine) to refer to locations in the machine's virtual address space, others are used to specify instructions to be performed and others are used as if to refer to numbers, for example when they are incremented during repeated operations. Instead of arguing over whether these are *real* instances of meaning, or reference, we should accept that they are in some ways like and in some ways unlike other instances, and we should try to analyse the implications of these similarities and differences. (The example illustrates that the same structure can have different semantic roles in different pragmatic contexts in a control system.)

One of the characteristics of typically human uses of symbols is that they occur within an architecture that supports not only states and processes involving notions of truth and falsity, such as believing, perceiving, inferring, predicting and planning but also motivational or affective states such as wanting, hoping, fearing, enjoying and disliking. Fully human-like uses of symbols or representations with meanings will not be possible in machines or other organisms whose architecture is not rich enough to provide the required context for those uses. In other words, human-like semantics presupposes human-like pragmatics, and that requires a human-like functional differentiation in the control-system architecture. (The actual physical architecture could be completely different, however.)

At this stage it is too early to say exactly what the architectural requirements for all those states are, though this is the topic of work currently being done at the University of Birmingham (e.g. Sloman 1993b, Beaudoin and Sloman 1993). In any case, there are many different types of systems that we need to study as part of the general study of "design space", the space of possible designs for interesting behaving systems.

We can make some negative points against over-simple theories of semantics. For example, it would a mistake to require all semantic relations to involve causal connections between representations and what they represent. This would rule out representations of non-existent or impossible states of affairs: any planning system needs to be able to create representations of alternative possible situations most of which will never actually exist. Non-existent objects cannot enter into causal relations. Another mistake would require all representations to share structure with what they represent: many counter-examples are to be found in natural languages, including the fact that the very same thing can be referred to either by a very simple pronoun, or by a variety of phrases of differing structure. A less obvious mistake is to require *every* case of meaning or representation to accord with a general system or convention: for that would rule out the possibility of temporarily using an object arbitrarily to represent another for a short time, as sometimes happens, for example when choosing a pepper pot, a match box and a mark on a table to represent the configuration of three vehicles at the time of a road accident. These selections need not involve any regular association, systematic relationship or social conventions, and the semantic relations might last only for a few seconds until a different event is discussed. If this kind of arbitrary and temporary semantic relationship can be useful in external representations we should expect similar phenomena in internal representations, especially those involving current percepts which can involve unknown objects encountered for only a short time. Any general theory of semantics should allow this.

A more obvious semantic requirement is that where representation is context-free the variability of a sub-state should at least match quantitatively the variety of things that it is required to represent. A set of eight binary switches is capable of only 256 possible states, and so would be inadequate to represent states of a chess board, since far more configurations are possible in chess. However, the set of possible sub-states of a collection of switches grows very rapidly as the number of switches increases, so that a relatively small collection of switches can cover a huge variety of possible states. This is one of the characteristics of computing mechanisms which accounts for their generality and power.

Where the representing medium is the state of a virtual machine, and the syntax allows indefinite structural extension, as in the case of natural languages, predicate calculus, and many programming languages, the semantics may allow an infinite variety of different situations to be differently represented. This implies that the semantic rules mapping representing structures to things represented cannot be full explicit: they must themselves include generative power. This is obviously a requirement for many forms of creativity, in mathematics, science, art and everyday life, when we cope effectively with novel situations.

Of course, the physical mechanisms in the system may impose a limit on that variety, as memory limits in a computer can limit the size of programs or datastructures. But the potential variability inherent in the virtual machine may be far greater than the actual implementation allows. In the world of computing this manifests itself in programming languages that allow programs and data to be created that current computers cannot accommodate, driving the development of new forms of hardware, including memory management systems with larger address spaces. When a computer becomes available with a new bigger memory system it may be possible for the same programs as before to run in it, but tackling larger, more complex, problems. Similarly, it is possible that once animal brains started to use virtual machines that were not inherently limited by the physical structures available, this could increase the evolutionary pressure towards bigger brains to enable those virtual machines to be used more extensively. (This is one way of looking at the old distinction in linguistics between “competence” and “performance”, where the latter is limited by implementation details, but not the former.)

Additional considerations from the design standpoint, include: how quickly required structures can be created, how quickly they can be changed as needed, how quickly the required substructure can be found in a very large database of structures, and how well a collection of permitted structures fit the purposes for which they are to be used. (Compare the criteria for adequacy of representations in McCarthy and Hayes 1969.) Experienced software designers learn a vast stock of representational schemata and are able to select appropriate ones for new design tasks with far greater ease than equally clever individuals who merely know the programming language concerned. Explicit codification of the knowledge of such experienced designers would be of enormous help in formulating a general account of semantic capabilities of different forms of representation, but is likely to be a difficult task, like all attempts to articulate complex intuitive knowledge.

(11) Inferences (reasoning) in control states

Inference (reasoning) is normally defined as derivation of new propositions from old ones subject to the constraint that it is impossible for the old ones to be true and the new ones false simultaneously. We can generalize the concept of "inference" to include all manipulation of information-bearing structures to obtain new structures that enable something to be done that could not be done directly with the originals. (A similar though slightly less general proposal, based on the notion of "denotation" was made in Sloman 1971.) Examples of this generalized notion of inference include such processes as:

- creating an interpretation of a sensory array (see discussion of vision, below),
- deriving a parse-tree from a sentence and a set of syntactic rules,
- forming a plan,
- translating (or compiling) a high level plan (or program) into a more detailed lower-level language,
- searching an abstract space for solution to a problem (e.g. searching for a plan in a space of possible plans)
- using a neural net to transform a vector representing a desired configuration of limbs into a vector representing a set of control signals to be sent to muscles or motors in order to produce that configuration.,

and many more. It should be clear that the variety and power of the inference mechanisms available will be intimately related to the variability of syntactic forms.

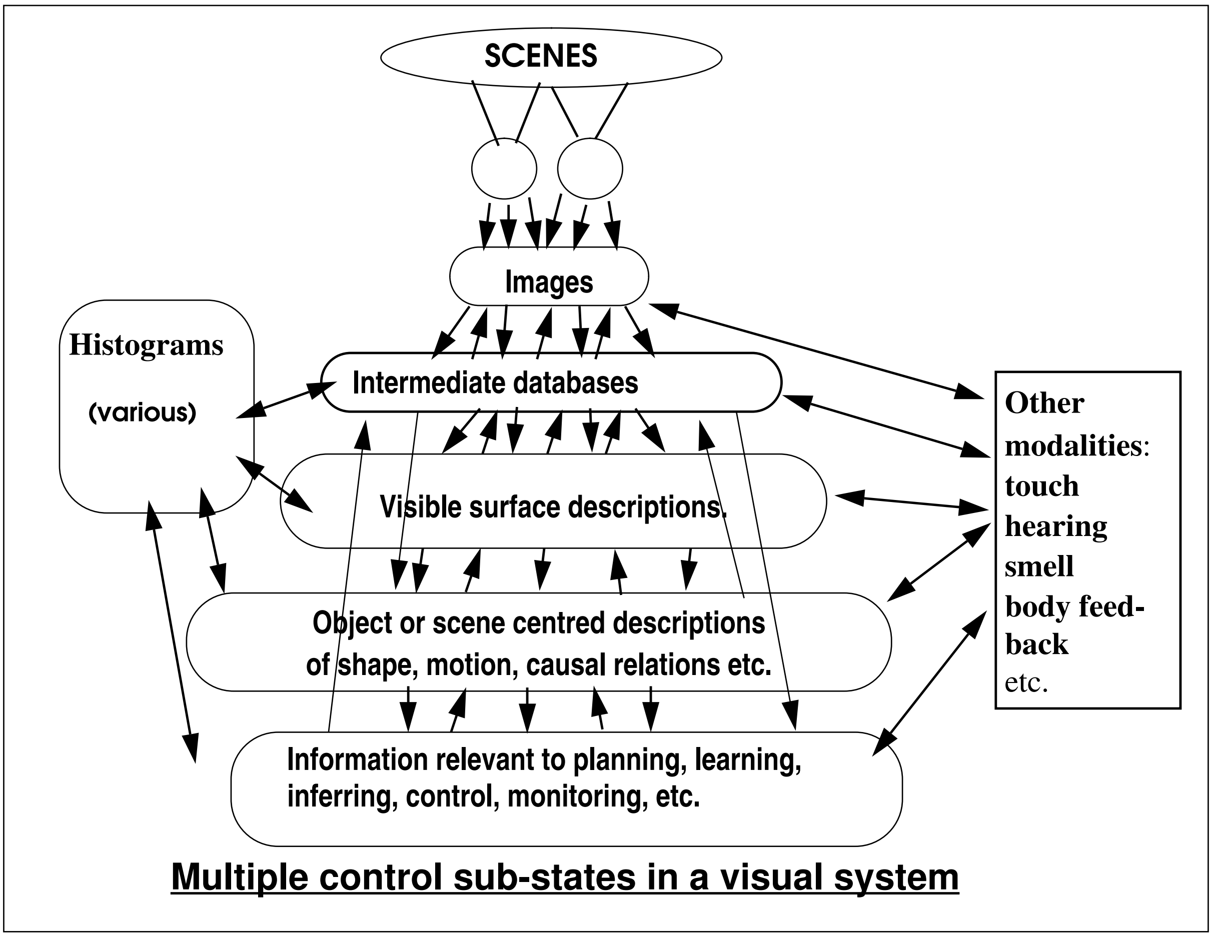
This general notion is not restricted to manipulation of propositions, or symbols that represent propositions, such as are traditionally studied in logic. A very common form of inference in this sense is use of a map to find a good route, or to constrain search for some item satisfying geographical constraints (e.g. finding a railway station close to a particular town), which might be a much slower process if all the available information were in the form of lists of propositions about the locations and features of various towns, roads, rivers, and segments of railways. (Of course, propositions can be organised in such a way as to facilitate searching. In particular, if they are stored in the form of a virtual 2-D array then sets of propositions can replicate some of the functions of a map.)

In this generalized sense almost any transformation of one representing sub-state into another, or construction of a new sub-state under the influence of others, can be regarded as an example of reasoning or inference, provided that the process fulfils some useful function within the total system. A full survey of types of inference would require a survey of architectures and the varieties of functions that could be fulfilled by sub-mechanisms. Taxonomies of forms of syntax, types of pragmatics, kinds of semantics and forms of inference would all need to be closely related.

External representations used by humans vary enormously in their structure, the kinds of variability they allow, and their functions (Sloman 1985a). Similarly, within a complex system that has multiple independently variable components there can be sub-systems with information states that differ widely in syntax, semantics, pragmatics and forms of inference. When the internal representations are based on virtual machines rather than physical structures the variety of possibilities is even greater, since there is a wider variety of types of structures and manipulations of structures within virtual machines than within physical mechanisms.

(12) A Partial view of a visual architecture

We can illustrate some of this variety by considering an over-simplified view of the architecture of a visual sub-system of an intelligent agent. Visual systems provide a great deal of scope for combining different sorts of notations, as they have many intermediate information states with different representational forms and different functions and causal links. The diagram below gives a crude (and partly conjectural) indication of this variety. Some intermediate representations will be very close in



form to the original optic array, preserving neighbourhood relationships and other spatial relationships, whereas others lose spatial structure (e.g. histograms showing relative global amounts of different features) and some will include relatively abstract information (e.g. descriptions of causal and functional properties of perceived items). Some forms of "abstract maps" may need to combine the different types of information, e.g. for indexing purposes and controlling visual search. One of the many interesting questions is to what extent visual systems use virtual structures with unlimited complexity, and to what extent they are limited by fixed size structures, like retinal maps. There seems to be no limit in principle to the structural complexity of scenes that can be perceived (including more and more complex diagrams for example). But there may be limits to what can be done with such scenes, for instance the number of items that can be directly operated on simultaneously by high level processes.

If all this is correct, understanding how vision works must be an important step towards a general understanding of how to build systems making use of multiple forms of representation. (For a more detailed analysis or requirements for visual perception see Sloman 1989).

The internal information structures within a perceptual system depend not only on the nature of the environment but also on the agent's general needs and current purposes, and conceptual apparatus. Although house-flies and humans share the same physical environment it is likely that their perceptual systems store very different kinds of information and use it for very different purposes. There will also be diversity among humans, depending on cultural background and individual learning and interests: for example, within two people the process of looking at the same Chinese text could produce very different intermediate visual databases if only one of them is fluent reader of Chinese. Even within a single agent different intermediate databases in a visual system may be put to very different kinds of use. For example, compare recognition of word-shapes whilst reading and perception of 3-D structure in the environment. Or contrast the use of optical flow to control posture and the use of optical flow in perceiving the shapes of objects. The former requires very rapidly computed global information about flow patterns in order to determine whether forward or backward falling motion needs to be counteracted by contracting muscles, whereas the latter requires far more intricate detail.

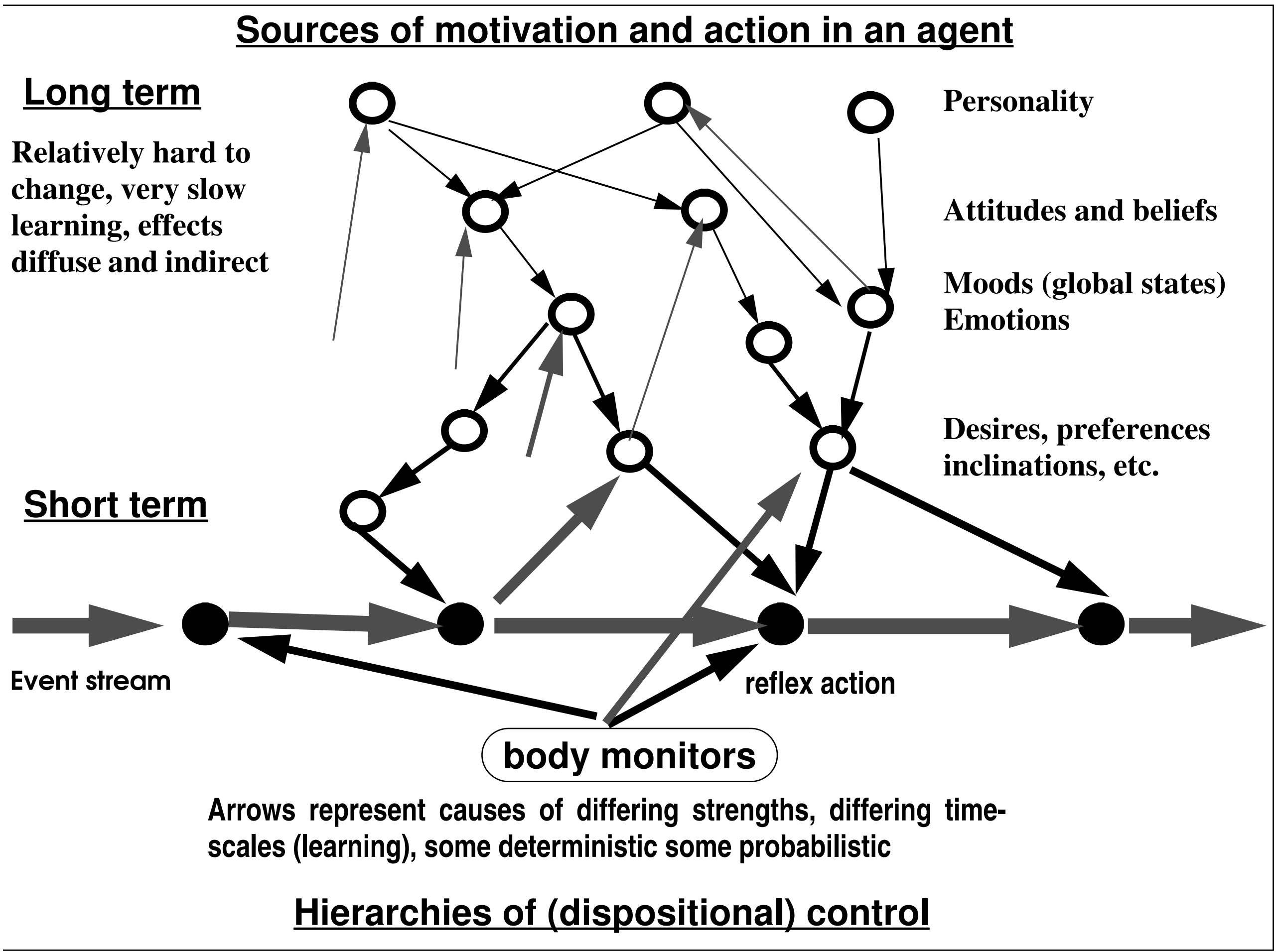
The need for a variety of different notations, or representational forms, in different sub-databases in a visual system is a special case of a general requirement for sensory systems to use different representing sub-states for different purposes, e.g. in auditory, tactile and other forms of sensory input. Different output stages in motor-control subsystems are also likely to require different notations depending on whether they are concerned with relatively high level plan information or collections of detailed and rapidly changing low level control signals to motors or muscles. For internal monitoring of internal states, and control of internal processes, such as reasoning, learning or planning, yet more forms of representation may be needed. For the purpose of acquiring and storing vast amounts of content-addressable information, something like a connectionist network containing a lot of distributed superimposed information could be very useful whereas using the information to solve an algebraic problem is more likely to use something more like a conventional symbolic representation.

Which notations are good for which purposes are engineering design questions, to be answered not by armchair philosophical discussions about abstract requirements for rationality or thought, or intelligence but by detailed analysis of design problems, along with experiments using test implementations to find out where they work and where they go wrong.

Which notations are actually used by different organisms is a different sort of question, i.e. an empirical question, to be answered by empirical investigations that need to be informed by the engineering design considerations. There may be links between the empirical question and the design question insofar as evolutionary pressures favour good design solutions. However, evolution is constrained by what has previously evolved and a failure to take that into account can lead to incorrect theories about how humans and other animals work.

(13) Hierarchies of dispositions (desire-like states).

Overall control of behaviour is another domain in which intelligent systems are likely to use multiple forms of representation, as shown, approximately, in the conjectural diagram below, which is mainly concerned with motivational and affective states. Different levels of control correspond to different layers of information states. Some of them correspond to dispositions that persist over a long time, but are "activated" only under certain conditions, which may occur rarely, or never. Some long term dispositions are very general in their effects and are hard to change (e.g. personality, attitudes). Other control states more episodic and transient (e.g. desires, beliefs, intentions, moods). Many are complex, richly-structured, sub-states, e.g. political attitudes, involving several different general beliefs, preferences, desires and principles.



Control states in such a system differ in various ways, e.g. according to such factors as:

- how generally applicable they are (e.g. a generous temperament will influence behaviour towards a larger class of individuals than the attitude of loyalty to ones friends),
- how long they persist (desires, moods and emotional states can be short-lived),
- how easy they are to change, and what changes them,
- how directly they influence actual (internal or external) behaviour (e.g. some ethical principles correspond to states that do not directly motivate any sort of behaviour but control the selection when two specific sorts of motivation come into conflict: e.g. a principle that patriotic duty should come before family loyalty, or *vice versa*)
- how much of the system they affect at any given time (certain alarm states, and moods, seem to have very global effects on mental and physical processes, even though they are transient),
- whether they produce only transient effects or whether they feed back into long term states (a form of learning).

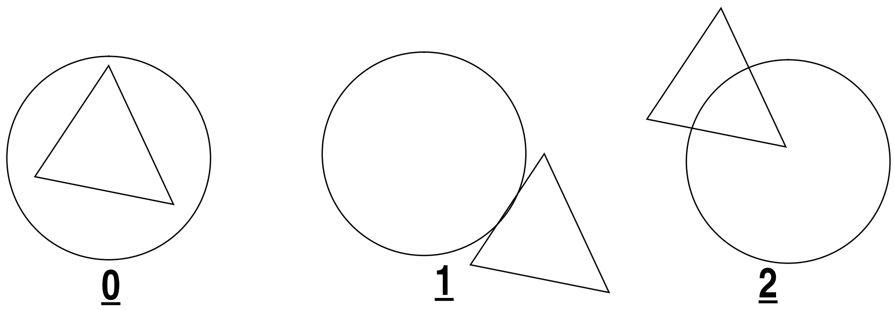
Our ordinary language concepts are not rich and precise enough for describing the full richness and variety of control states. Only when we know more about possible underlying mechanisms and have a good theory about how different overall architectures meet different design requirements, will we be able to generate a good taxonomy of mental states (as the theory of the structure of matter generated the periodic table of the elements).

Different levels of control may use different notations or forms of representation. Some may be "symbolic", some neural or "connectionist". It remains an open question which forms of representation are most suitable for which of these different control states. A hierarchy of control states might be composed of a hierarchy of production systems, where each system includes rules for modifying lower level systems. Alternatively, each node might be a neural net which responds to its inputs by modifying weights in a lower level neural net. Both of these implementations may be capable of meeting the same general design requirements, differing only in some of their detailed consequences, which might turn out to be relatively unimportant and hard to detect "from outside". In order to decide between these options in explaining human capabilities we need both empirical evidence and a design-based theory about the trade-offs.

(14) People use many kinds of information states in solving problems

It seems that when solving problems, people often use "mixed" notations, both internally and externally, as illustrated below. Asking people how many different intersection points are possible between a circle and triangle in the same plane normally causes people to start imagining various continuous transformations of a circle and triangle relative to each other. Somehow, by doing this many people (though not all) are able to discover that there are seven possible numbers of intersections 0,1,2,3,4,5 and 6. Some of the numbers can be achieved in more than one way, e.g. using

In how many ways can a circle and a triangle intersect? I.e. how many different numbers of intersections are there? Below are three examples with 0 intersections, 1 intersection, and 2 intersections. How many other possible numbers are there?

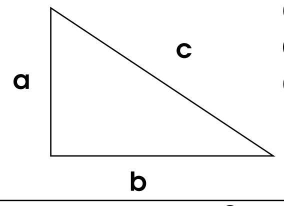


Most people claim to "visualise" the arrangements, imagine various changes, and count the resulting intersection points.

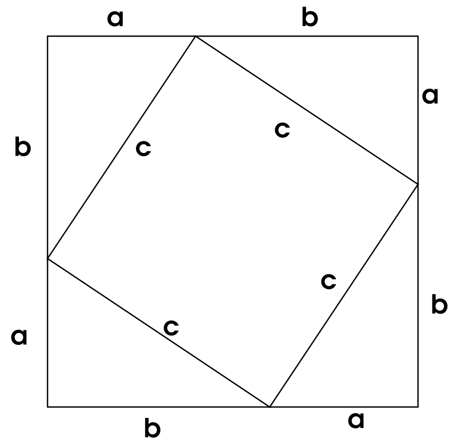
Reasoning about intersections in a plane

tangents or incidence of vertices on the circle. When answering this question many people seem to use an internal representation that admits transformations like sliding and rotating, rather than being restricted to the kind of applicative syntax that allows only insertion, deletion or substitution. However, it is clear that simply transforming images is not enough: the images need to be linked to the terms of the question and the intersection points identified and counted, showing the need to combine different forms of representation.

A more subtle requirement is the ability to reason about the transformations that are possible so as to be sure that all significantly different cases have been covered. This is often a requirement for mathematical proofs using diagrams, and seems to require the ability to relate diagrammatic representations to propositions stating general facts about those representations. Another example is the Chinese proof of Pythagoras’ theorem., summarised below.



The figure on the right shows clearly that the square of (a + b) is equal to: the sum of the square of c plus four triangles of base b and height a.



$$(a+b)^2 = c^2 + 4x1/2(ab)$$

$$a^2+\cancel{2ab}+b^2 = c^2 + \cancel{2ab}$$

$$a^2 + b^2 = c^2$$

<u>The Chinese proof of Pythagoras’ theorem: a “hybrid” proof</u>

In principle the theorem can be stated entirely in terms of geometrical concepts (including addition of areas). It is also possible in principle to prove the theorem by using only logical and algebraic notations, starting from some set of axioms for geometry, though it is arguable that insofar as any such axioms are supposed to be non-arbitrary they must be derived from some non-axiomatic knowledge of spatial structure, whose precise nature remains to be investigated. Moreover, most people find purely logical and algebraic proofs hard to understand and seem to need the aid of pictures or diagrams. Often thinking of the right picture transforms the problem so that it is much easier to solve. For example the Chinese proof illustrated transforms the problem of proving the theorem into that of showing that the area of a large square is composed of the area of a smaller square and four triangles, which, together with a little algebra, proves the theorem easily, even though the squares on the shorter sides of the triangles are nowhere depicted. Again the use of this mixture of representations seems to require the ability to see that a *particular* diagram captures all the essential features of a whole range of cases, including all forms of right-angled triangles. A still deeper problem is how to encode knowledge about shapes and configurations in such a way as to enable one to think of a good transformation when presented with a difficult problem. There seems to be wide variety between people as regards such capabilities. How many different forms of representation mathematicians actually use when thinking about such problems remains an open question.

I suspect that this use of images or diagrams in mathematical reasoning is very closely related to everyday human capabilities, for example when a child looking at a mechanical device is able to “see” how it works (turning this knob moves that lever which....). How such abilities develop in children, what the architectural requirements are, which mechanisms are involved, and how they change, remains unexplained.

It is worth noting that people who claim to use images to solve problems could be deceiving themselves. Introspection is not a reliable way of telling what is going on: self-perception is no more reliable than object perception! In both, the reality is mostly hidden: internal and external perceptual systems evolved to give us information that is useful for practical purposes, rather than to tell us what “really” exists, which is why the researches of scientists often produce results that extend or contradict what we perceive. There is no more reason to believe that introspection can tell us the syntax of notations we use internally than there is to believe that external perception can tell us the true physical structure of physical objects. Thus we can only form conjectures and try to test them in the normal way that scientific conjectures are tested: by exploring their implications, seeing if they have the consequences they are alleged to have, and seeing whether their consequences fit other facts and theories. We also need to allow the possibility that there are several layers of implementation, using quite different notations in high level and low level virtual machines.

There is no uniquely optimal way of representing any given set of information. Often representational forms have properties that suit them well to particular problems, even though the same information may be better represented in a different way for other purposes. This can be illustrated with a slightly modified version of a well known example.

(15) The magic square example

Algebraic and arithmetical notations are extremely useful for solving many problems relating to numbers. For example, if you are given the values of six variables, **a**, **b**, **d**, **e**, **g**, and **h** and the task of computing three more, **c**, **f** and **i**, then the following three equations would make the task very easy, using simple algebraic transformations:

```
a + b + c = 15
d + e + f = 15
g + h + i = 15
```

Moreover, checking subsequently that the values all satisfied the following equations would also be very easy:

```
a + d + g = 15
b + e + h = 15
c + f + i = 15
a + e + i = 15
c + e + g = 15
```

By contrast, a different task based on the same set of relationships, expressed in the same set of equations, is much more tedious, namely the task of finding all the possible one-to-one mappings, satisfying the above equations, from the numbers:

1,2,3,4,5,6,7,8,9

onto the letters:

a,b,c,d,e,f,g,h,i

This can be done by exhaustive search through all possible ways of mapping the numbers onto the letters, but the search space has factorial 9, i.e. 362880, nodes. This would be very simple for a computer program, but is not easy for people: it would take a lot of time and be error prone.

It turns out very much easier for some people to perform the task if they switch to a non-algebraic notation that reveals important features of the problem that enable the solution to be found in far fewer steps. The crucial point is that the problem of finding a mapping that satisfies the equations is isomorphic to the problem of creating a 3 by 3 "magic square", in which all rows columns and diagonals add up to 15, using only the numbers **1** to **9**. The correspondence between the two tasks can be seen by labelling the locations in the square with the variables **a** to **i**, as in the diagram below. It is also necessary to check the equations against the collinear triples in the square. That task would be made harder if the letters within each equation were re-ordered, and the first three equations were mixed up with the others. This shows that the equations as they stand share some pictorial structure with the diagram, whose properties are very helpful in finding a solution, as I'll now explain.

The visually evident structure of the square shows that there are three different kinds of locations with different roles in the solution, four mid-edge locations each occurring in two collinear triples, four corner locations each occurring in three triples, and the centre location, which occurs in four triples. In any solution to the problem there must therefore also be three kinds of numbers

a	b	c
d	e	f
g	h	i

There are three kinds of locations (mid-edge, corner, and centre) so there must be three kinds of numbers to go into them:
(1) four numbers in two triples (mid-edge)
(2) four numbers in three triples (corner)
(3) one number in four triples (centre)

Using visually obvious differences between locations in a square

corresponding to three kinds of locations: four numbers occurring in two equations, four in three equations, and one in four equations. This is easily checked: **a**, for example occurs in three equations, so it must correspond to a corner location, as shown, and **e** occurs in four, so it must be mapped to the centre.

If we check for each number how many triples adding up to 15 contain it, we can use the size of the set to assign the number to a particular category, which constrains the mappings onto letters, considerably reducing the search space by eliminating inappropriate mappings, leaving a total of factorial 4 x factorial 4 = 576 mappings to consider. However, consideration of most of those can be avoided by propagating partial solutions over the square, using the observation that symmetry implies that the first assignment can go to *any* location of the appropriate sort. This can be used to cut down the search enormously. After a few numbers are in, the rest are fixed, and remaining solutions are found by reflecting and rotating the original solution.

To illustrate how the search is cut down, start with the number **1** and consider which pairs can be combined with it. Because **15−1=14**, the numbers in each pair must add up to **14**. Since the largest other number is **9** and **14−9=5**, that rules out the numbers **2**, **3** and **4** as too small to be combined with **1**. The only remaining candidate triples are quickly found to be (**1 5 9**) and (**1 6 8**). Thus the number **1** occurs in only two triples and must therefore be a “mid-edge” number. It could be mapped onto locations labelled **b d f** or **h**, and at this stage, with no other constraints, we could arbitrarily choose **b**. A similar check shows that **5** can occur in three more triples, with (**2 8**) (**3 7**) (**4 6**) Since it occurs in four triples **5** must go in the middle, and be mapped onto **e**. Since the triple (**1 5 9**) starts with a mid-edge location and goes through the middle, the diagram shows that it must end with a mid-edge location, and that means that **9** must be mapped onto **h**. From this it follows that the other pair of numbers combining with **1**, i.e. **6** and **8** must lie in an edge triple, and therefore they must be in the adjacent corners, i.e. **a** and **c** (at this stage either mapping would do, as the situation is still symmetrical). As **a** and **e** are now fixed and add up to **14**, their companion **i** must be **4**; similarly completing the other triples can be done without any further search. Another solution is obtained by reflecting about the middle vertical axis. All remaining solutions can be found by rotation of those two solutions through 90 degrees. (Is it obvious that no other solutions are possible?)

Perceiving the symmetries in the whole structure and detecting the differences in types of locations both use human abilities to detect spatial structures and spatial relations. I do not pretend to have an any explanation of how this is done, though no doubt it has something to do with the fact that the visible structures in a 3 by 3 array form patterns that are instances of general schemata that the visual system has frequently encountered, so that properties of such patterns are well known. In part the very syntax of the representation makes certain properties easier to detect than corresponding properties in the equation format. For example finding how many equations contain the letter **e** requires a search through the nine equations for occurrences of the letter, whereas finding how many collinear triples go through the central location does not require such a search as there is only one occurrence of the location, and for someone accustomed to such structures it is easy to cycle through the lines counting them. However, the apparently simplicity of the experience hides very complex processing.

These are special cases of sophisticated general human visual capabilities whose underlying mechanisms we do not at present understand, although studies of vision by researchers in psychology and AI have explored many different mechanisms and forms of representation, all with only limited success. Only when we have a good theory of vision and spatial inference can be hope to have a deep account of how people combine different forms of representation in solving such problems. Up to a point people can do these things both with external representations and with internal ones, though controlled experiments sometimes show that an individual's confidence regarding manipulation of internal images is misplaced. For instance people who claim to be able to visualize printed words on a blank wall cannot read them backwards nearly as fast as they can read them forwards, whereas with real printed text the speed difference is considerably reduced. (See also the papers by Reisberg and Slezak in Narayanan 1993).

It is worth noting that in the case of the magic square the representational sophistication required for finding the solution without search is far greater than that required to understand the original problem and test a solution. In particular, the "short-cut" requires use of a richer description language, including descriptions of different sorts of triples (e.g. "diagonal"), and different sorts of numbers, or different sorts of locations ("mid-edge", "corner", "centre"). This extra capability is not required for the task of finding the solution using a systematic generate and test algorithm, which would be trivial to program, by comparison, especially in a typical AI programming language. It is far easier to write a program to do the exhaustive search than to write one that can go through the search-avoiding manipulations illustrated above. This is a general phenomenon: finding a solution without "brute-force" combinatorial searching often requires more sophisticated, and often more varied, representational abilities than are required for finding *some* solution or checking any proposed solution. In this sense, intellectual laziness requires intelligence. In fact, a major component of intelligence as we know it could be characterised as "productive laziness", though how this is achieved still remains to be explained. In contrast with all this representational versatility the syntactic manipulations of logical and arithmetical notations are relatively easy to model on a computer. (For further discussion of transformations of representations see Peterson 1994).

What is sometimes forgotten is that for the purpose of designing a system with human-like intelligence it is not sufficient to have mechanisms that can use the new representation of the problem in order to solve it in the transformed form. The system would also need *general* information about different types of representations and their uses, in order to enable it to think of switching from one notation to another. How is that general information acquired? How is it stored? How is it used? (Do any other animals have that sort of "meta-level" representational capability?) In my own case, I was aware of the magic square problem for many years before I noticed that the distinction between three types of location in the square could be used drastically to reduce the search space. Why did it take so long? Why is it not blindingly obvious immediately the problem is posed? The answer may have something to do with the way in which the meta-level representational capabilities are deployed. Perhaps some individuals deploy them far more effectively than others?

(16) Efficiency of a representation is relative to a virtual machine

Talking about efficiency requires care. Often the question "Which solution requires fewer steps?" does not have a unique answer. It depends on which implementation machine is being considered. For example, on a machine that is well suited to generating sets of numbers and adding them up, the exhaustive search through combinations of numbers may require fewer "machine instructions" than any attempt to simulate, on the same machine, the human visual capabilities that enable spatial and topological features to be detected very quickly and easily (compare Glasgow 1993, Sloman 1993a). It may turn out that this human capability depends on massive neural networks, whose simulation on a standard computer would be very expensive, and which inherently require more storage space and more "atomic" operations than the exhaustive combinatorial search. Thus what appears efficient and effortless for us may actually, in some deep sense, be very inefficient, though very quick because of the parallelism. This illustrates a general point: biological designs, produced by evolution, often trade huge amounts of storage space and processing power for speed and versatility. The costs are sometimes forgotten by those who advocate spatial representations on computers as "more efficient". In summary, the following questions need to be distinguished:

(a) For problem P, does formalism F1 or formalism F2 enable solutions to be found in fewer steps at the level of the formalism?

(b) For problem P, using implementation machine M, does formalism F1 or formalism F2 enable solutions to be found in fewer steps at the level of machine M?

(c) For problem P, which combination of formalisms and implementations produces the fastest solution in real time using available physical mechanisms?

The answer to (a) may be F1 and the answer to (b) F2, and the best answer to (c) may be something totally different.

Thus asking in the abstract which formalism is better for a problem can be a silly exercise. A full analysis of this problem would require investigation of the relative efficiency of implementing different kinds of abstract formalisms in different kinds of implementation machines, which may also be virtual machines concerning which similar questions arise relative to lower level implementation machines, and so on. Until we know a lot more about these issues, debates about which representations are best in the abstract, like debates between symbolists and connectionists, are pointless. (Pat Hayes made similar comments in his 1974 paper criticising my 1971 paper. See also Hayes 1993.)

(17) Information-bearing control states

In the light of discussion so far we can now see that information-bearing control states involve the following, in varying degrees of sophistication.

1. An underlying medium and a mechanism that manipulates it.

The medium and the mechanism determine the kind of variability supported (the syntax of the information states), the kinds of inferences available and possible ways of constructing, changing, storing, or searching structures.

- Some formalisms are sufficiently abstract to allow a very wide variety of media and mechanisms to support them. (The same language can be spoken, written, put into morse code, etc.)
- Often the medium in which a formalism is implemented supports a richer form of variation than the formalism requires: e.g. sequences of characters on a 2-D surface use only a subset of the potential inherent in the surface. 2-D structure in tables makes more use of the medium than ordinary prose. One form of creativity is extending a formalism to use more of the power of the medium, as has

happened often in the history of mathematics. The result is sometimes an extensions that works in one medium but not another, e.g. written mathematical notations for which there is no spoken equivalent. Some poets have also used 2-D format.

- The main medium available in computers is essentially a very large one dimensional array of bits with a set of operations that can be used to implement a very wide variety of additional virtual media, including, for example, N-dimensional arrays of bit-patterns for N = 2, 3, etc. Although such mechanisms are very general indeed they are not totally general and in particular cannot support continuous variation, or arbitrary rotations, though such things can be approximated. It is not clear what the implications of these limitations are.
- The mechanism may support distributed or superimposed states (like superimposed wave forms in multiplexed signals). Superimposed structures may be completely separable, or not; i.e. with cross-talk or without. Sometimes cross-talk between structures provides a useful form of generalisation: storing information about a particular case can change the stored information about similar cases.
- The medium may support direct jumps between arbitrary states, or only transitions via many intermediate states. Conventional computers have the disadvantage that transitions between arbitrary states generally require large numbers of bits in memory to be changed, and this requires many intermediate states in which different portions of memory are changed. The intermediate states may be meaningless relative to the control-function of the mechanism. An implementation using large numbers of neurons, or large numbers of computers, can support a much larger number direct state transitions, as well as being more robust. Nevertheless if the computers are fast enough they can match the functionality of the parallel implementation, insofar as they can achieve all the same state-transitions in the same times. However, if neurons support continuous variation that can only be approximately matched.

2. Levels of syntax (allowable forms of variation)

- There can be different levels of structure within a formalism. E.g. at one level the syntax of written English is a set of short strokes. At another level there is a sequence of letters, punctuation marks and spaces. At another level there are nouns, verbs, phrases, clauses sentences, etc. Discourse grammars and story grammars encode attempts to characterise still higher levels of structure.
- Whether the variability supported at a particular syntactic level is significant for the control state or is a mere implementation detail depends on whether that variability has a pragmatic or semantic role. For example in many contexts the thickness of strokes forming printed characters is irrelevant, but in some contexts it can be used for emphasis, or to indicate switching between different notations with different interpretations (as in many printed manuals). Often 2-D layout of text is used to help the reader parse what has been written, i.e. the layout has a pragmatic but not a semantic role.
- Each syntactic level defines a set of possible structures and a sort of topology defined by which transitions between permitted structures are minimal transitions. Each item has a set of neighbours reachable by minimal transitions. Where there is continuous variation there are no minimal transitions. In some cases more than one ordering of the same set of structures can be useful for a particular application: for example there is an ordering of pairs of integers based on their sum, e.g.

  (1 1) (1 2) (2 1) (2 2) (1 3) (3 1) (2 3) (3 2) (1 4) (4 1) (3 3) etc

  and a different ordering based on their ratios, e.g.:

  ... (1 32) ... (5 16) ... (3 8) ... (1 2) ... (5 8) ... (13 16) ... (31 32) ...

  the latter maps onto the natural ordering of the set of rational numbers and admits no nearest neighbours.
- Whether a particular topology is useful will depend on the pragmatic and semantic roles to be associated with the states.

- An important feature of the lower level syntax is its generative power for producing new notational elements (e.g. words). Contrast the generative power of sequences of letters, and collections of 2-D vectors, or bitmaps. For a given level of generative power in a sequence of symbols there is a trade-off between the number of primitive symbols available and the permitted sequence lengths. Human languages might have used a two-letter alphabet but the words would have been much longer and processing requirements on brains (e.g. short-term memory requirements) quite different.
- The syntax of a notation may be more or less closely related to the structures of the things represented, e.g. 2-D pictures of 3-D scenes. It is important not to confuse structural relations with isomorphism. Many people wrongly think that pictures are isomorphic with what they represent. Yet it is clear that a 2-D picture cannot be isomorphic with a 3-D object that it depicts.
- Syntax is relative to mechanisms and procedures: it's not *physical* structure that matters, but *used* or *interpreted* structure.

3. Procedures for operating on and transforming syntactic structures.

The mechanisms that support the representing medium will provide a set of operations that can be performed on states, in order to create or modify representing structures.

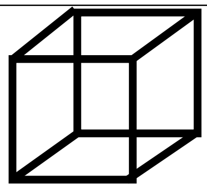
- Syntax depends on 'permitted' operations e.g. substitution, combination, translation, stretching, rotation, insertion, searching, matching, etc. (E.g. allowing superscripts changes the syntax of numeric expressions.) Operations can be discrete or continuous, quantitative or structural, destructive or constructive, etc. E.g. conventional vs digital thermostats.
- Procedures are used for creating instances, storing instances, searching for instances, matching instances, transforming instances, describing instances (e.g. building parse trees) interpreting instances. Syntactic and semantic operations may overlap. E.g. finding edges in images is syntactic insofar as it involves analysis of image structures and semantic insofar as it is part of a process that maps pixel arrays onto configurations of (possibly continuous) curves.
- Significant syntax need not map onto underlying physical structure. E.g. as previously remarked, sparse array can have <u>far</u> more components than the transistors implementing it. Compare lazily evaluated infinite datastructures and dynamic lists in Pop2 and Pop11.
- Some mechanisms support only local operations, e.g. use of a pencil on paper or most computer operations on datastructures. In those cases global operations (e.g. changing the stroke thickness of all the text in a particular page, or replacing all occurrences of one symbol with another) may be relatively cumbersome and time-consuming. Some control systems (brains?) may need both global changes and intricate local changes to be made simultaneously, e.g. a pianist smoothly changing speed and dynamics at the same time as playing many notes.

4. Semantics and meta-semantics.

By "meta-semantics" I mean a set of principles for using notations to refer to, denote, express, store information about, various things. Reasoning based on such principles could justify the selection of particular formalisms for particular tasks.

- An arbitrary formula in predicate calculus, such as "P(a,b,c)" does not have a semantics, but there is a meta-semantics which specifies the sorts of things the symbols can refer to, i.e. properties, relationships, individual objects, etc.
- Similarly, the meta-semantics of 2-D line drawings requires syntactic relations in the pictures to express some geometrical or topological relationships between things depicted. But the meta-semantics does not require isomorphism: 2-D can represent 3-D. Moreover, in pictures like the following, there need not be any unique decomposition of either picture of thing depicted into

corresponding parts, whereas most logical notations require a unique decomposition of the relevant bit of the world.



The number of line segments and junctions seen here will differ according to whether it is seen as a 2-D pattern or as a picture of a wire-frame cube.

- Some syntax/semantics relations are general purpose (e.g. logic) some ad-hoc (e.g. using digit concatenation in arabic numerals to mean “multiply by 10 and add”). Musical notations and many maps use a mixture of arbitrary and principled relationships.
- Choice of syntax and semantics is determined not only by what is represented, but how the information is to be used. (Both maps and lists of towns with their coordinates are useful in atlases, though they are used for different purposes.)

(18) Dimensions of information states

We can now provisionally characterise information states in a behaving system in many different dimensions, including the following:

- Their structure or syntax, which depends on the kinds of variation they can support, such as:
  - continuous or discrete variability,
  - one or many dimensions of variation,
  - whether the variation is structural, allowing changes in complexity, or simply variation in a fixed number of linear dimensions,
  - whether the complexity of syntax has an upper bound or not,
  - whether the space of possibilities is homogeneous (e.g. the space of N dimensional vectors is homogeneous, whereas the space of unbounded binary trees is not, since at some locations in the latter space there are far more neighbours, that is trees that can be accessed by a single change, than at other locations in the space.)
  - whether the Fregean function/argument syntactic form is supported
  - whether higher order function forms are supported (e.g. quantifiers, lambda operators),
  - etc.
- What functional role they fulfil:
  - recording input (at various levels of abstraction)
  - controlling behaviour (e.g. representing goals, plans, control-signals)
  - whether the control function is direct or indirect, short term or long term, etc.
  - long term vs short term information stores
  - high level vs low level control of behaviour, etc.
  - storing general vs specific ‘facts’ about some domain
  - representing suppositions or possibilities as opposed to facts
  - performing inferences
  - expressing preferences, goals, instructions, plans, partial plans, etc.
- Among the functions required in intelligent agents are:
  - <u>Desire-like</u> control states, which tend to initiate changes (under certain conditions). These could include include a wide variety of types of motivational states, including general principles and ideals, as well as specific desires and goals.

- Belief-like control states, which tend to be changed under ‘external’ influences, and which also determine how desire-like states produce changes. This would include general beliefs about classes of objects and events as well as specific beliefs, e.g. about a particular property of a particular object.
- Supposition-like control states, i.e. states that are merely used to represent possibilities without any commitment to their being actual (such states are important in the conditions of conditional instructions or rules, and in the process of creating plans by exploring possible consequences of different actions).
- Plan-like control states, i.e. states that store information about steps required to achieve certain ends, or types of ends. These could include very detailed stored plans (like the pianist who has memorised a Beethoven sonata) or very abstract schematic plans whose details need to be filled in during execution.
- Association stores, i.e. stored mappings of associations between patterns of various kinds. Neural nets are one time of mechanism that can store associations though there are many others that have been explored in computer science and AI, including simple linear association lists, indexed association tables to reduce the need for searching, including hash-coded association tables.
- Reflex control states, i.e. states that get triggered by detection of some pattern in sensory data and then automatically send signals to motors to perform some action. The mechanisms required for such control states must be able to store associations between patterns. It seems very likely that some internal processes also require the use of innate or learned reflexes: cognitive reflexes.

(A deeper analysis would produce a more detailed taxonomy of causal roles.)

- Whether they have a semantic function and if so what sort
  - whether there’s a generative semantic relation or a fixed set of mappings
  - what kind of mapping is used, e.g. applicative semantics, analogical, systematic, ad-hoc, mixed, etc.
  - the ontology of the domain represented
  - whether only particular positive facts or also general, negative, disjunctive, implicational propositions can be expressed,
  - whether nonexistent objects and states of affairs can be represented,
  - in the case of procedural semantics, which kinds of procedures can be specified (e.g. conditionals, loops, recursion, etc.)
- What kinds of processes they support, e.g. creating representations, changing them, comparing them, storing them, searching for them in various ways, transforming them (inference), etc. This includes such things as:
  - How they are created (contrast: perception, reasoning, motive generation, planning, fine tuning in motor control, outputs of neural nets, short term memory in a production system, etc.)
  - How they interact with other states (changing them, modifying their influences, deleting them, etc.)
- Whether different items of information are separately represented or superimposed and distributed over many implementation units. This distinction covers a variety of different cases. For example in a logical database all the theorems are distributed over the axioms needed for their derivation in a manner that is different from the distribution of information over weights in a neural net. In the two co-ordinates of a chess board location, **rank** and **file** there is also information distributed about which two diagonals the location lies on, one computed as **rank+file** the other as **rank-file**. There are many similar forms of distributed representation in conventional computing systems.
- How all this is achieved

- what mechanisms are used to implement the relevant virtual machine
- how semantic mappings are set up
- which algorithms are used in the virtual machines

• Other dimensions correspond to engineering requirements (e.g. speed, robustness, completeness, soundness of inferences, etc.): usually not all can be met simultaneously, so that trade-offs have to be assessed by designers.

Accessibility of information is important in intelligent systems. If nothing can access the information represented in a sub-state, it might as well not be there. Thus every object potentially has complete information about itself embedded in itself: but usually most of that information is totally inaccessible to the system: there is no part of the system that can use the state information to vary its own behaviour. Contrast the way in which a CPU in a computer can access the states of bit-patterns in the virtual memory, or the way in which a system containing a neural net can implicitly access information stored in the weights in the connections by feeding in a vector and seeing what vector comes out.

Any physical state can in principle be treated as an information store, e.g. a store of information about the system itself (many animals use the environment as a representation of itself - e.g. trail-blazing as an alternative to memorising a route). But often the information will not be accessible unless there are specific mechanisms that enable interaction with the structure. (There are different kinds of access: updating, interrogating, changing, obeying, comparing, etc.) Access can be continuous or discrete, high-bandwidth or low-bandwidth, serial or parallel, etc. Access to computer memories is discrete, (relatively) low bandwidth, parallel (insofar as several bits of information can be retrieved simultaneously).

Explicitness *vs* implicitness. This is a supposed distinction that can easily cause great confusion, since what is clearly explicit in a virtual machine may be implicit in a lower level medium. For example for a system that treats sparse arrays just like ordinary arrays the contents of the sparse array are explicit: they can be examined or changed in exactly the same way as the contents of any other array. At the implementation level there is a difference insofar as most of the locations in the sparse array are not explicitly represented, and their values are computed as required. In this case the access time is bounded and the difference in access times between sparse and ordinary arrays may be so small as not to be noticeable. If a logical database is used some of whose contents are explicitly represented while others are derived on demand things are more complex. If all the items accessed over a period of time are so quickly derivable that it is difficult to distinguish the database from one in which they are all pre-stored, then at the level of a machine that uses the database there need be no functional distinction between the stored and the derived items. If the amount of search required varies then there can be gradual degradation as harder and harder questions are asked of the database. Here, instead of a sharp functional distinction between what is explicit and what is implicit we find a difference in degree of accessibility. However this can be something that varies dynamically, for instance if derived items are explicitly stored in a cache for a time: then access times will vary depending on how recently an item was last requested. If the cache stores not only items explicitly sought but "neighbours" based on some heuristic neighbourhood metric, and if the cache includes a gradual decay mechanism then explicit/implicit distinction is even less relevant. A full overview of useful forms of representation and implementation mechanisms would show that there is far more variety than common-sense distinctions can cope with. Unfortunately many people who discuss representations, including many philosophers and psychologists, have not been exposed to this variety.

(19) Ontological, epistemological and heuristic requirements.

Different kinds of representations have a syntax and semantics that implicitly assume different ontologies. E.g. predicate calculus presupposes an ontological carving up of the world into objects, properties, relations, etc. plus a pair of truth values. Whether a pictorial notation does this depends on the detailed form and use of the notation. For example, use of a 2-D array each location of which can contain only a list of names of objects (as in Glasgow 1993) presupposes a different ontology from the use of an array of pixels representing a sampling of intensity values in the optic array from a particular viewpoint (Sloman, 1989). Many maps and atlases use a mixed ontology, with names and other symbols representing distinct objects superimposed on analogical depictions of continuously varying contours, coast-lines, rivers, roads, etc.

As McCarthy and Hayes pointed out in 1969, there are several different questions that can be asked in relation to a representational formalism and the world it is used to represent. For example there are questions about:

- Metaphysical (or ontological) scope/adequacy relative to a world: can the notation represent everything that can exist in the world?
- Epistemological scope/adequacy relative to an agent: can the notation represent everything the agent needs to know about the world, or everything the agent needs to be able to suppose or ask about the world?
- Heuristic scope/adequacy relative to an agent, a set of purposes, in a particular world: does the notation (with its associated mechanisms) allow the agent to solve problems in order to fulfil those purposes and do so in a reasonable time, with reasonable accuracy?

In addition to their questions we can ask others from the standpoint of biology or cognitive science:

- Could the notation have evolved from earlier forms?
- Could it be extended as required to cope with new developments in the environment and new purposes?

These can be broken down into further sub-questions. For example questions about heuristic adequacy can break down into questions concerning the following:

- Learnability: could the notation be learnt? Does it support learning of information that it expresses?
- Storage, searching, matching: e.g. what are the space or time costs and trade-offs
- Size of search spaces: does the notation allow irrelevant nodes to be generated in searching so that they have to be explicitly rejected, or does it inherently constrain searching to what is relevant (Sloman 1971).
- Expressive power relative to solving new problems. E.g. does the notation make it easy to match a new problem against stored information about previously solved problems? Does it make it easy to explore different representations of the same problem to find one that facilitates solution? (“Easy” may itself be relative to a mechanism, as previously explained, rather than being an intrinsic property of the formalism.)
- Sensitivity to changes in the ‘world’: can fine differences between objects, or between viewpoints be expressed? (E.g. without this kind of sensitivity in visual images, binocular disparity could not be a cue to depth.)
- Stability: does the representation remain invariant when the context or object depicted changes in a way that is irrelevant to current purposes (e.g. an object centred coordinate frame can provide a representation that is invariant with respect to viewing direction and distance). (Note that stability and sensitivity can be in opposition.)

- Ease of construction, and ease of derivation from originally available data. E.g. enhanced images are easy to derive from visual data by filtering processes, whereas descriptions of 3-D scene structure can be very hard to derive.
- Flexibility and multiplicity of use. E.g. Marr's hierarchical representations of 3-D structure simplified the task of recognition, made it hard to represent or compute some potentially useful relationships, such as that a person's finger tip was on his nose.
- Ability to be tailored to requirements of particular applications
- Use as a substratum for other 'virtual machines'
- Extendability: this depends on whether the formalism uses some medium that has more richness than the formalism already exploits. For example, English does not use all the sets of possible character sequences corresponding to normal word lengths. E.g. "glutterzank" remains unused.
- Use for transmitting information This involves such things as bandwidth required, ease of parsing/decoding/interpreting by the receiver, ability to make use of shared knowledge, etc. (But communication is not the sole or primary function of presentation.)
- Robustness E.g. do minor deformations during storage or transmission lead to undetectable errors.

and no doubt many more!

Contrary to how such issues are sometimes discussed, these properties may not be inherent in the syntax of a formalism, but may be relative both to a class of problems and also to an implementation mechanism.

It is important to note that features that may be nice from a mathematical point of view, e.g. economy of basic symbols and syntactic simplicity or elegance, may be undesirable from other points of view. Features that are desirable from the point of view of a designer (at compile time) may not necessarily be desirable at run time. For example, economy of primitives leads to complexity of representations, complexity of matching and complexity of searching for solutions to problems. (Compare building an aeroplane out of general-purpose Lego and building one out of a Lego aeroplane kit.) Syntactic simplicity of a notation (as in Lisp) can imply considerable local ambiguity that needs global context for disambiguation, and therefore make the parsing and interpreting process difficult for an agent with limited short term memory. This is no problem for computing systems with a very large stack.

(20) What sort of underlying engine is needed?

Much of this is neutral as to what mechanisms are used to implement the various kinds of substates and causal linkages. They might be neural mechanisms or some other kind. As in circuit design, global properties of the architecture are more important than which particular mechanisms are used: when the design is right architecture dominates mechanism. The detailed mechanisms make only marginal differences as long as they provide the following:

- sufficient structural variability to meet task requirements (e.g. a representational system that is incapable of taking on more than N distinct states cannot distinguish more than N different states of the environment).
- sufficient architectural richness in the notation and the mechanisms using it
  - number of independently variable components
  - functional differentiation of components
  - variety of causal linkages
- sufficient speed of operation

"Virtual" machines in computers seem to have many of these features for a wide range of tasks. This is one of the reasons why general purpose computers are used so widely.

But we don't yet know enough about requirements, nor about available mechanisms, to be able to say with any confidence which infrastructure could and which could not work for implementing human-like systems. E.g. it could turn out that, in our universe, only a mixture of electrical pathways and chemical soup could provide the right combination of fine-grained control, structural variability and global control to support human-like intelligence. If that were so, we might not be able to base human-like intelligence on systems built from computers as we currently know them. Of course if additional mechanisms are required they will be added to computers of the future, and that will change what computers can and cannot do! (Such changes may not all preserve the mathematical equivalence between computers and Turing Machines. Compare Penrose 1989.)

We need new thinking tools to help us grasp all this complexity. AI has provided many detailed concepts for thinking about and modelling processes in perception, planning, reasoning, learning. But we still lack more "global ideas" to help us think about the architecture of the whole mind: how the bits fit together, and this paper is intended as a small contribution to filling that need.

Our key notion that the mind is a very complex control system, whose functioning depends crucially on the creation and manipulation of information-rich states, using a variety of forms of representation requires us to generalise standard notions of control systems and their states, as already mentioned. In particular, it is important to allow not only "atomic" states in a phase space, but also "molecular" states which have their own internal, changing, structure. (For more on this see Sloman 1993b.) The latter seems to be needed for understanding intelligence. A full theory would need to include a more detailed survey of types of uses of information-rich control states in intelligent systems.

A full survey would require an analysis of the types of architectures available and the functional roles they allocate to different kinds of components and their states. Although there is a rich body of mathematics of control systems, most of it appears to be incapable of coping with the requirements listed here. For example, no fixed set of differential equations can cope with structural change, either local or global. Such equations do not represent changes in sets of equations. Some other kind of mathematics is needed for the study of systems whose topology can change.

(21) Conjectures

I'll end with a few conjectures and some unanswered questions. first the conjectures.

- Major steps in evolution involved the development of new representational formalisms and mechanisms for manipulating them.
- Unfortunately such things don't leave fossil records. So evidence is going to be very indirect and only on the basis of a good theoretical framework can we hope to interpret it.
- A major gap in our ability concerns representation of spatial structure adequate to support the antics of birds, squirrels, and other animals that interact in a quick and fluent manner with an intricate, changing, collection of spatial structures.
- It may turn out that the effective representation of shape is not a representation of structure but of "affordances" (Gibson 1979). So a cube is not (just) a collection of edges and surfaces (like CAD representations), but a collection of <u>possibilities for</u> and <u>restrictions of</u> motion, deformation, change, etc. (Sloman 1989) Intelligent agents need a way of generating all this information very quickly from structural information in images, and making it accessible for <u>many</u> different uses, including control of movement, planning, recognition, etc.
- If this is right, representation of <u>flat</u> surfaces, <u>straight</u> edges, etc. may use notations required for <u>arbitrary</u> curves and shapes, and representation of <u>static</u> structure may use notations required for

arbitrary motion. Many computer models of visual processing assume it's the other way round, i.e. the more complex items are represented in terms of the less complex.

- Many aspects of human experience, for instance emotional states, are not concerned so much with what we can represent and what inferences or algorithms we can apply, as with what the global architecture is and what sorts of causal interactions can occur. For example, many of the states we call emotional have a characteristic aspect that includes partial loss of control of one's thought processes and attention.

There is much that we still don't understand about the representational powers of human beings and other animals. That is partly because we do not yet have a good understanding of the requirements, including, for example, the requirements for a human-like visual system. For instance: what are the affordances in a piece of crumpled paper? How arbitrary shapes and motions should be represented is still an unanswered question. Yet many animals must have solutions implicit in their design, since they cope so well with varied and richly structured spatial environment. Similarly we don't yet know what the design requirements are underlying human motivational and affective states (Beaudoin and Sloman 1993).

(22) Unanswered questions

Apart from denotational or model-theoretic semantics how many other kinds are there? In particular, what should be said about 'intensional' semantics, according to which knowing the meaning of something need not involve knowing what it refers to but does involve knowing how to determine whether something is or isn't referred to. Perhaps a clue lies in the fact that besides the association between representations and what they refer to or depict there is also an association between (simple or complex) symbols and procedures for associating those symbols with other things (objects, properties, relations, actions, questions, goals, etc.). The objects denoted (the extension) would be distinct from the procedures and mechanisms that define the association (the intension). It is not clear whether, and to what extent, such distinctions can also be applied to pictorial representations, information states distributed over the weights in a neural net, and other implicit information stores.

I suspect that contrary to the standard approach to analysing properties of a language, like its semantic and pragmatic properties, by simply writing down sentences and formulas relating that language to things in the world, a truly general theory would have to define semantics in terms of the architecture of the control system that makes use of the particular information-rich state, and the functional roles of such states within the architecture.

Philosophy needs to become a branch of engineering design.

References

The following items illustrate ways in which topics mentioned in this paper can be developed. This is not an exhaustive bibliography.

BEAUDOIN, L.P. and A.SLOMAN, (1993) 'A study of motive processing and attention', in A.Sloman, D.Hogg, G.Humphreys, D. Partridge, A. Ramsay (eds) *Prospects for Artificial Intelligence*, IOS Press, Amsterdam, pp 229-238

BOBROW, D.G. (1975) Dimensions of representation in D.G.Bobrow and A Collins, (eds) *Representation and Understanding* New York: Academic Press, pp1-34

DENNETT, D.C., (1978) *Brainstorms* Bradford Books and Harvester Press.

FUNT, B. V. (1983). A parallel-process model of mental rotation.*Cognitive Science,* 7(1): 67-93.

GAZDAR, G.J.M. (1979) *Pragmatics: Implicature, Presupposition and Logical Form,* New York, London: Academic Press.

GIBSON, J. J. (1979). *The Ecological Approach to Visual Perception*. Lawrence Erlbaum Associates, Hillsdale, NJ, reprinted 1986.

GLASGOW, J.I (1993) The Imagery Debate Revisited: A Computational Perspective. *Computational Intelligence* Vol. 9 No 4, pp 300-435

HAYES, P. J. (1974). Some problems and non-problems in representation theory. In *Proceedings of the AISB 1974 Summer Conference*, University of Sussex. Reprinted in *Readings in knowledge representation*. Edited by R.J. Brachman and H.J. Levesque. Morgan Kaufmann, Los Altos, CA, 1985, pp. 4-21.

HAYES, P.J. (1993) I Can't Quite See What You Mean, *Computational Intelligence*, Special issue on Computational Imagery, Vol. 9 No 4, pp 381-386

MARR, D. (1982). *Vision.* W.H. Freeman, San Francisco, CA.

MCCARTHY, J. and P.J. HAYES. (1969). Some philosophical problems from the standpoint of artificial intelligence. In *Machine Intelligence 4*. Edited by B. Meltzer and D. Michie. Edinburgh University Press.

NARAYANAN. N.H. (editor) (1993) *Taking Issue/Forum: Computational Intelligence* Vol. 9 No 4, Nov. 1993, pp 300-435 Blackwell.

NEWELL Allen, and H.A.Simon, (1981) 'Computer science as empirical enquiry: Symbols and Search' in J. Haugeland (ed) *Mind Design: Philosophy, Psychology, Artificial Intelligence*, Cambridge Mass: Bradford Books, MIT Press. pp 35-66

PENROSE, R*, The Emperor's New Mind* Oxford: Oxford University Press, 1989

PETERSON, D.M., (1994) "Re-representation and Emergent Information in Three Cases of Problem Solving", in *Artificial Intelligence and Creativity*, T. Dartnall ed., Kluwer [forthcoming 1994]

SLOMAN, A. (1971). Interactions between philosophy and A.I.: the role of intuition and non-logical reasoning in intelligence. In *Proceedings of the Second International Joint Conference on Artificial Intelligence*.Reprinted in *Artificial Intelligence* 1 pp 209-225, 1971, and in I*mages, perception, and knowledge*. Edited by J.M. Nicholas. Reidel, Dordrecht-Holland, 1977.

SLOMAN, A. (1975). Obituary notice: analogical representations. *AISB Quarterly.* Reprinted as 'Afterthoughts on Analogical Representation,' in R. Schank and B. Nash-Webber (eds), *Theoretical Issues in Natural Language Processing* Proceedings of TINLAP conference held at MIT in June 1975, and In *Readings in knowledge representation*. R.J. Brachman and H.J. Levesque (eds) Morgan Kaufmann, Los Altos, CA, 1985, pp. 432-439.

SLOMAN, A. (1978). *The Computer Revolution in Philosophy: Philosophy of Science and Models of Mind*. Harvester Press and Humanities Press

SLOMAN, A. (1985a). Why we need many knowledge representation formalisms. In *Research and Development in Expert Systems*. Ed M. Bramer. Cambridge University Press, pp. 163-183.

SLOMAN, A. (1985b), What enables a machine to understand? in *Proceedings 9th International Joint Conference on AI,* pp 995-1001, Los Angeles, August 1985

SLOMAN, A. (1989). On designing a visual system: towards a Gibsonian computational model of vision. *Journal of Experimental and Theoretical Artificial Intelligence,* 1(4): 289-337.

SLOMAN, A. (1993a) 'Varieties of formalisms for knowledge representation' in *Computational Intelligence*, Special issue on Computational Imagery, Vol. 9, No. 4, November 1993

SLOMAN, A (1993b) 'The Mind as a Control System' in *Philosophy and the Cognitive Sciences,* (eds) C. Hookway and D. Peterson, Cambridge University Press, pp 69-110 1993

SLOMAN, A. (1993c). 'Prospects for AI as the general science of intelligence' in A.Sloman, D.Hogg, G.Humphreys, D. Partridge, A. Ramsay (eds) *Prospects for Artificial Intelligence,* IOS Press, Amsterdam, pp1-10