TOWARDS A GENERAL THEORY OF REPRESENTATIONS[1]

Aaron Sloman
School of Computer Science and Cognitive Science Research Centre
The University of Birmingham

Abstract

This position paper presents the beginnings of a general theory of representations 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. 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).

(1) Introduction

It is often assumed that in order to use a representation one must be conscious that one is doing so. I suggest that there is a more general notion of representation, which covers all states or structures that store or contain information used to control internal or external behaviour, whether in humans or other natural or artificial behaving systems. There is a huge variety of types of control states, whose properties depend on the sophistication of the architecture in which they are embedded and their functional role within that architecture. At one extreme this includes information states of simple homeostatic devices, like thermostats. Far more complex representations are involved in the control of internal or external behaviour in a human brain, including many information structures of which we are totally unaware, such as those encoding information about the grammar of our language, the social norms that we follow unconsciously, and all the skills that we use in solving problems, taking decisions, forming judgements, and controlling our actions. Much of our competence depends on stored information — about the world, about our language, about our society, and about how to do things — and we are unconscious of most of it, which is why conceptual analysis and theoretical linguistics are so hard. Discovering what information is used by other animals, and how that information is represented, is at least as hard. We can expect that intelligent robots will also need to have access to vast amounts of information of many kinds, used in different ways. For humans, other animals, and robots some of the information is undoubtedly stored in the environment, like the trail blazed in the forest by a hunter who cannot remember all the details of his route, but there are also *internal* stores which, in part, constitute the explanation of the different capabilities of different agents in the same physical environment.

We need a label to cover all the various kinds of information stores, irrespective of what their structures are, or how they are created, or whether we are aware of using them or not. The word ‘representation’ seems to me to come closest to meeting this requirement, even if some people normally use it in more restricted ways (just as non-mathematicians often use the word ‘ellipse’ in a restricted manner that excludes circles). A study of the general principles of intelligence (Sloman 1994a) must include a survey of different forms of representation (‘notations’, ‘languages’), theUses

1. To appear in D.M.Peterson (ed) *Forms of Representation* Intellect Press, 1994 or 5

architectures in which they are embedded, and the mechanisms that support and manipulate them. Such a study cannot be encumbered by such commonplace assumptions as that representations are external, are primarily used for communication between agents, and can only be used consciously.

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. Within this framework, we can generalise notions like 'syntax', 'semantics', 'pragmatics' and 'inference', as shown below.

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.)

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, for instance during infant development, and possibly even in adult life when new skills are learnt or new connections formed between old skills. Furthermore, within intelligent 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 to be useful for describing and analysing both natural and artificial behaving systems, in terms 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

The analysis of representations as sub-states in control systems 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. This is because many representing structures 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 are 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.

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. These machines are *emergent* in the sense defined in Sloman (1994b). 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.) 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, which is itself an abstract machine 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, is the inverse of the philosopher's concept of 'supervenience', as explained in Sloman (1994b). 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. I shall also assume that causal relations can hold within abstract machines, and between physical and abstract machines: an assumption without which software engineering would be impossible (as explained in Sloman 1994b). In other words, causal influences can be non-physical and can cross implementation boundaries. There is nothing mysterious or mystical about this: it happens all the time in complex computing systems.

I am not saying that it is *only* abstract machines that have information-bearing sub-states. Even a simple thermostat controlling a room heater has a physical sub-state that can be thought of as representing the current ambient temperature and another 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 temperature control system that behaves as a whole. Although the control systems that are of most interest in the study of mind have far more complex architectures than this, it can still be illuminating for a survey of control systems to include the simpler systems. Limiting cases can be part of a concept even though they are very different from most instances.

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, an information-bearing 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'.

I shall not attempt to define 'information-bearing' state here. This is a concept that is best analysed within the framework of the design-based approach to the study of mind (Sloman 1993b, 1993c, 1994b). The design-based approach attempts to understand how different kinds of architectures, notations and mechanisms are able to underpin and explain different sorts of capabilities, human and non-human. It contrasts with phenomena-based investigations, which merely collect empirical facts, and the semantics-based investigations of some philosophers and cognitive scientists which aim to analyse our existing concepts, as opposed to forging new powerful explanatory concepts in the context of deep theories of how things work. Design-based concepts enrich and extend ordinary concepts of mind, knowledge and representation, just as new theories of the architecture of matter enriched and extended our concepts of kinds of stuff, via the periodic table of the elements and theories of chemical composition.

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* and mappings between designs and requirements (Sloman 1993c, 1994a). At present we understand little about the requirements driving the design of intelligent agents, whether in the laboratory, or in biological evolutionary processes.

(3) 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 of representations, including many

ad-hoc notations, such as the standard arabic notation for numbers in which concatenation representations multiplication by 10 plus 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 1971, 1975, 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. Likewise, both maps on paper and arrays of spatially organised information in a computer can include verbal descriptions or labels and other non-spatial information items.

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 (e.g. using the ordering of rules, the order of subgoals in a rule, the order of arguments to a functional expression), and to that extent it is as procedural as any other programming language, and partly pictorial in its interpretation, 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 or ML.

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 (including both richness of structure of individual representations and the diversity in the class of representations), others concern their manipulability, others concern speed with which they can be created, changed or used, and so on.

Among 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.

(4) What is a form of representation?

Information of many 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.

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. 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 in architectures for behaving systems, 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.

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: *the set of states that is possible for a specific (information-bearing) part of a control system.* (For now I shall leave the notion of 'information-bearing part' undefined.) Insofar as each set of possible states includes a variety of forms or structures I shall say that the notation involves a *grammar* or *syntax.*

It is possible for different notations with different grammars to be used by different sub-states of the same complex system. Intelligent systems typically require many different kinds of notations or forms of representation, corresponding to different functional roles played by different sub-states. For example Sloman (1993b) claims that different components of a visual sub-system, or different components of a motivational system, may need different forms of representation.

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, a common source of confusion for students learning a new programming language.

Below I shall try to show how the familiar concepts of syntax (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. For instance, 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.

(5) 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, the set of possible states has a very simple structure, e.g. a linear continuum, like the temperature-representing sub-state of a thermostat. 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. A neural net has a collection of possible control states corresponding to the states of activation of the neurons, and another collection of states corresponding to the possible

weights on connections between neurons. In both cases the grammar corresponds to the set of possible locations in a high dimensional vector space of fixed structure. In a brain where new cells and new links between cells can be grown, the grammar would allow states of varying complexity.

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. Physically identical states need not implement representations with the same syntax. A particular string of characters, 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 both permitted by the underlying mechanisms and significant in relation to controlling internal or external behaviour. This will typically depend on whatever mechanism actually uses those states and the variety of possibilities that it can cope with.

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. At a still lower level 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 computing system there are also usually several different levels of 'grammatical' 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.

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 (as is common in pictures, music and analog computers), 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 what I called *applicative* or *Fregean* representations (where the key syntactic relationship involves 'application' of a function sign to argument expressions, as in logical notations) and what I then called *analogical* representations, in which the key syntactic relationship is the holding of a relation within the representing medium, where different syntactic relations have different semantic interpretations. 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.

An example is computing the truth values of sentences with very similar syntax but very different semantics, or truth-conditions, such as 'Mary is richer than Tom', 'Mary is taller than Tom', 'Mary is cleverer than Tom'. By contrast if one sub-image is above another in a picture the possible interpretations of that relationship are more limited. (Notice, as pointed out in Sloman (1971) that the interpretation need not be rigidly fixed: in 2-D pictures of 3-D scenes, interpretations are generally context-sensitive). 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 some maps). In other cases the representing relations are locally ambiguous and therefore the semantic relations are highly context sensitive, and finding a consistent interpretation requires problem solving or search. This is the source of much difficulty in designing vision systems.

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. 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 to 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, which is particularly useful for processes that require rapid construction and reorganisation of temporary trees or networks. I have 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).

(6) Pragmatics of a control state

Information-bearing control states have a pragmatic role within a control system. This is defined in terms of their function or purpose within the architecture. This notion of 'purpose' or 'function' 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 pragmatic 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 in biology, though in some cases biologists make the (in my view unnecessary) claim that their usage is justified by evolutionary selection of a design. Examples of pragmatic roles include the following:

- storing information that can be accessed by other sub-systems,
- initiating new processes,
- terminating existing processes,
- 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, 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 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.

(7) 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'). I claim that there are no precisely definable concepts that correspond to our normal use of expressions like 'meaning' or 'information content'. There are clusters of features to be found in the contexts in which we ordinarily talk about meaning, and different sub-clusters 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.

One of the characteristics of typically human uses of symbols is that they occur within an architecture that supports 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 very different, however.) At this stage it is too early to say exactly what the architectural requirements for all those states are, though some ideas are reported in Sloman 1993b, Beaudoin and Sloman 1993, Beaudoin 1994).

We can make some negative points against over-simple theories of semantics. For example, it would be 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. Similarly, 2-D images can depict 3-D scenes and many of the scene relationships have no direct pictorial analogues. For example a visible edge of a cube can join a visible

face and an invisible face on the far side, whereas a picture cannot include invisible objects or ‘far sides’.

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 (indeed exponentially) 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 indefinite variety of different situations to be differently represented. This is obviously a requirement for many forms of creativity, in mathematics, science, art and everyday life, when we cope effectively with novel situations.

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 (in principle) 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 this increased 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 the permitted structures fit the purposes for which they are to be used. (Compare the criteria for adequacy of representations in McCarthy and Hayes 1969.)

(8) 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. (Cf. Sloman 1971.) Examples of this generalized notion of inference include such processes as:

- creating an interpretation of a sensory array, e.g. by a visual system,
- 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 a 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.,
- generating a new goal on the basis of new percepts and old attitudes (e.g. creating the goal to help someone seen to be in distress),

and many more. The variety and power of the inference mechanisms available will be intimately related to the variability of syntactic forms.

This general notion of inference is not restricted to manipulation of symbols that represent propositions. A common form of inference in this sense is use of a map to find a good route, which could 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. The use of diagrams or visual images is also very common in mathematical reasoning. (I discussed some of the reasons for this in Sloman 1971, 1985a)

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. Similarly, within a complex system that has multiple independently variable components there can be internal 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 the variety of possibilities is even greater.

There is no uniquely optimal way of representing any given set of information. Often representational forms have properties that are tailored to particular problems or purposes, even though the same information may be better represented in a different way for other purposes. This can be illustrated by analysing a well known example of a change of representation.

(9) 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, consistent with 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 pictorial 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 remaining five. This shows that the equations as they stand share some pictorial structure with the diagram, whose properties are helpful in finding a solution, as I'll now explain.

The square has 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

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

therefore also be three kinds of numbers 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 for each number we check 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 combined with **1** 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: (**2 5 8**), (**3 5 7**), (**4 5 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, so assign **6** to **a** and **8** to **c**). As **a** and **e** are now fixed and add up to **11**, their collinear companion **i** must be **4.** 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. It seems that the visible structures in a 3 by 3 array form patterns that are instances of general schemata that our visual systems have frequently encountered, so that properties of such patterns are readily accessed and used. (How exactly this happens remains a hard research problem: until that is solved we have only a very sketchy theory.) 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.

Unfortunately, the speed and introspective simplicity of these visual experiences hides very complex processing, which we still do not understand, and which it is too easy to take for granted as if they provided explanations: from the design standpoint they merely help to define what needs to be explained. These are special uses of sophisticated general human visual capabilities not yet matched by AI vision systems nor explained by brain scientists. Only when we have a good theory of vision and spatial inference can we hope to have a deep account of how people combine different forms of representation in solving such problems. In the mean time we cannot trust introspection. (See also the papers by Reisberg and Slezak in Narayanan 1993).

(10) Simpler processing requires more sophisticated concepts

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 language for describing and manipulating the information in the problem, 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 much easier to implement. 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.

A system with human-like intelligence would also need *general* information about different types of representations and their uses, in order to enable it to determine when to switch 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 humans this ability is far from perfect. 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?

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

It is often claimed that for certain problems analogical or pictorial representations are more efficient than applicative (e.g. logical) ones. However, 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'. 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 (which may be a virtual machine level)?

(b) For problem P, using implementation machine M and a particular implementation on 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, are pointless. (Pat Hayes made similar comments in his 1974 paper criticising my 1971 paper. See also Hayes 1993.)

(12) 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, functions, 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 continuously varying optic array from a particular viewpoint. Many maps 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? As remarked in the previous section, questions about relative efficiency can be deeply ambiguous.

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 (many of which are obvious to AI researchers though not to people who do not adopt the design stance):

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, but made it hard to represent or compute some potentially useful relationships, such as that a person's finger tip was on his nose, because computing the relationship between finger and nose required computing a sequence of coordinate transformations from nose to head, head to torso, torso to arm, and so on.

Ability to be tailored to requirements of particular applications. Some ways of storing information to specify a computer interface are far more rigid than others. Some databases assume fixed numbers of information slots for each item.

Use as a substratum for other 'virtual machines'. Many programming languages and abstract machine languages are designed to have this capability. An open question is the extent to which neural nets provide a useful substratum for implementing totally different kinds of 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 in encoding or decoding messages, as is commonplace in natural language. Ambiguity that might seem to be a flaw in a language can sometimes support reduction of bandwidth required where shared knowledge is used for disambiguation.

Robustness. E.g. do minor deformations during storage or transmission lead to undetectable errors?

and no doubt many more!

These properties need 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.

Choice of formalism can involve subtle trade-offs. Features that are 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 and other mathematically elegant programming languages) 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, such a s a human being, though it is no problem for computing systems with a very large stack.

(13) Conclusion

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. We cannot ask what the steps might have been or what the

evolutionary pressures were, without having a good theory of types of representations. I have tried to outline a theory by distinguishing dimensions in which representations can vary, including syntax, semantics, inference and pragmatics.

- Unfortunately internal representations used by animals don't leave fossil records. So evidence is necessarily very indirect and difficult to interpret.
- A major gap in our current knowledge 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. Representations currently employed in CAD, computer vision, expert systems seem to lack the richness and flexibility required for human and animal vision.
- 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 *possibilities for* and *restrictions of* motion, deformation, change, etc., as claimed in Sloman (1989) Intelligent agents need a way of generating all this information very quickly from structural information in images, and making it accessible for *many* different uses, including control of movement, planning, recognition, etc.

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 or motor control 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 environments. Similarly we don't yet know what the design requirements are underlying human motivational and affective states (Beaudoin and Sloman 1993). In humans there appear to be many levels of control with subtle differences corresponding to such notions as personality, character, attitudes, preferences, tastes, moods, desires, emotions, and so on. We still know little about such control hierarchies are implemented (Sloman 1993b).

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 is not the thing referred to. 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 non-propositional 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, the functional roles of such states within the architecture and their relations to actual and possible environmental states. Philosophy needs to be closely allied with 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. (submitted 1994), *A design-based study of autonomous agents*, PhD thesis, School of Computer Science The University of Birmingham.

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,

SLOMAN, A (1994a). Explorations in design space, in *Proc ECAI94, 11th European Conference on Artificial Intelligence* Edited by A.G.Cohn, John Wiley, pp 578-582.

SLOMAN, A (1994b). Semantics in an intelligent control system, in *Proc. British Academy and Royal Society Conference: Artificial Intelligence and The Mind: New Breakthroughs Or Dead Ends?* to appear in *Philosophical Transactions of the Royal Society.*