ARTIFICIAL INTELLIGENCE

OVERVIEW

The field of Artificial Intelligence (AI for short) attempts to understand the principles of intelligence. Unlike philosophy and psychology, which are also concerned with intelligence, the goal of AI is to *synthesize* intelligent entities, not only analyze them. The traditional way to define the research topic is based on intuition:

> *AI is the art of creating machines that perform functions that require intelligence when performed by people* (36).

This definition for AI reduces to the question of what *intelligence* is. There is a feel of indefiniteness in this concept: whenever one exactly knows how something is done, the aura of intelligence vanishes. Indeed, in AI, the goals seem to escape — as soon as an AI problem has been solved, it no more feels like *real* AI. A good example is the game of chess: originally, the goal was to construct a program that could play a reasonable end-game; as this was achieved, the goal was to achieve still better functionality; but, even though the computer can today beat the World Chess Champion, intuitively, it seems that there is the essence of intelligence still missing. Intelligence can be seen as an *emergent phenomenon* defying exact definitions — and, by the above definition, this applies also to AI.

The most concrete view of AI is found when one looks at what the AI researchers do. AI currently encompasses a huge variety of subfields, from general-purpose areas such as perception and logical reasoning, to specific tasks such as playing games, proving mathematical theorems, writing poetry, and diagnosing diseases. Often, scientists in other fields facing complex problems move gradually towards AI, where they find the tools and the vocabulary to systemize and automate their studies. Similarly, workers in AI can choose to apply their methods to any area of human intellectual endeavor. In this sense, AI is truly a universal field.

AI is not only universal — it seems to be a *paradigm between paradigms,* hosting studies that cannot yet be classified in existing categories. AI is an interdisciplinary framework between cognitive science, philosophy, and computer science, and it has an important role as the source of new intuition. It is a framework also for heuristic studies of more or less playful nature that have not yet reached the level of standard science; ideas that are too wild can first only be discussed in this framework. Only after new techniques have been firmly established, they can become approved in other disciplines — but, on the other hand, something that has been established, when it is just a matter of mechanical implementation, is no more so interesting from the AI point of view.

As the mystery of what intelligence is will assumedly always remain unsolved, AI will always be in turmoil. The methods will change, and one cannot foresee what kind of transformations take place in this field in the future. Today, AI already has a long history.

In many universities the computer, or the "electronic brain", was first studied in connection with electrical engineering. It was only natural that the engineering-like explorations in the brain functions were also started in these faculties. This historical connection between AI and electrical engineering motivates a lengthy introduction into AI also in this Encyclopedia.

Brief history

From the engineering point of view, most technical decisions are rather pragmatic, that is, the decisions whether to apply a specific technique or not are based on strictly objective criteria. In AI, however, there are opinions and prejudices that are not always objective. It is a rather new field with peculiar, heuristics-based methods. Much too optimistic predictions have been much too common. There have been promises, expectations, often followed by disappointments. The term AI today seems to be loaded with connotations. To have some perspective, a short introduction to the background of the modern AI methodologies is in place.

Early years. The nature and the principles of human thinking has been the subject of intense philosophical interest for centuries, and also visions of a "mechanical man" were presented a long time ago. But everything from the pre-computer era belongs to the pre-history of AI. It is the Darthmouth College in 1956 that can be called the birthplace of modern AI — in that summer, enthusiasts collected together, most notably from MIT (Massachusetts Institute of Technology), CMU (Carnegie Mellon University), Stanford University, and IBM. In the early years, the AI research was centered in these institutions. What was perhaps the most long-lasting contribution, during that workshop the name "artificial intelligence" was adopted.

The early years of AI until the late sixties were full of success — in a limited way: this was the time when everything that seemed even remotely intelligent was seen as success. There were plenty of distinct applications for demonstrating that specific human tasks, on the neuronal level and on the higher cognitive levels, could really be implemented in software. In addition to different kinds of demonstrations, the general AI tools were also developed: John McCarthy introduced the high-level programming language Lisp in 1958. This language, being tailormade for symbol manipulation tasks, became the standard in AI programming.

From the point of view of knowledge engineering, one of the most prominent early applications was the Logic Theorist (*LT*) by Allan Newell and Herbert Simon: given the axioms, it was capable of proving mathematical theorems. Later, they introduced the General Problem Solver (*GPS*) that was programmed to carry out human-like problem solving strategies (49). It was characteristic to these *first generation systems* that intelligence was assumed to be based on clever *search strategies* and *syntactic manipulations* of data structures.

At this early era, the potential of the new machines and methodologies seemed unlimited, and the optimism was

overwhelming. There were fundamental problems, though, and the proposed methodologies were successful only in very limited "toy domains". The huge, exponentially growing sizes of the search spaces were regarded as a problem that would be solved with the more efficient hardware. However, it turned not to be so, and by the end of 1960's, practically all U.S. government funding for academic translation projects was cancelled — this started a depression period in the AI field. This marked the end of early AI enthusiasm.

Role of knowledge. In the *second generation systems* the significance of *knowledge* was recognized. It was no more the special-purpose data structures that were assumed to account for intelligence; the inference engine was the same in different problems, it was the corpus of knowledge that was now responsible for intelligent behavior.

The application of the AI methodologies to real-world problems caused an increase in the demands for workable, general knowledge representation schemes. A large number of representation formalisms were developed — most notably, perhaps, the Prolog language that was based on first-order predicate logic. Other researchers emphasized the need of more structural knowledge representations. in addition to *rules,* in more complex applications *frames* or *scripts* were employed.

In the seventies the AI applications in the form of *knowledge engineering* and *expert systems* became commercially realistic. Later in the eighties, it also became good business: it was the Japanese *Fifth Generation* project that specially boosted the research activity and common interest worldwide. The industrial scale applications made it necessary to combine the AI methodologies in the standard-style environments. In *embedded systems* the knowledge-based unit is an integrated part of the system, so that the straightforward data processing tasks are carried out by traditionally realized program modules. The expert systems were applied to many engineering problems, for example, *expert control* became a hot topic.

New possibilities and challenges to knowledge representation and processing have been offered by the World Wide Web. Today, it is the Semantic Web where the main emphasis seems to be: the "intelligent internet" would facilitate extension of the AI techniques from local to global environments. In such systems, the problem of automating the association of concepts has been changed into the problem of implementation of *ontologies.* There already exist efficient tools for information search and distribution, and new applications are introduced at a brisk pace.

Connectionism. When knowledge-based systems were implemented, the *brittleness* and the unpredictability of the symbolic rule-based inference became a major problem in engineering applications. The *fuzzy systems* were introduced as a solution, so that the originally linguistic rules can be transformed in a consistent way into a numeric form. The "numerical rules" make it possible to implement efficient and fault-tolerant inference systems, and it is also easier to verify the *integrity* of the knowledge base.

But numeric approaches are not only an implementation technique. Another approach to synthesizing intelligent systems is to proceed bottom-up, starting from the low-level functional entities; the starting point is the observation that mind has to be based on the real nerve cells. Indeed, such *connectionistic* approaches have practically as long history as the computational ones.

In 1940's and 1950's, the basic observations underlying the current *artificial neural networks* were made. Donald Hebb found a simple mathematical formulation for neural adaptation; Warren McGulloch and Walter Pitts showed how first-order logic can be implemented by neural systems. Marvin Minsky and Seymour Papert introduced the *perceptron* in 1969. The real boost in *parallel distributed processing* came in the 1980's, largely due to James L. McLelland, David E. Rumelhart, and their PDP research group. There exist many different approaches to connectionism — for example, the *self-organizing maps* of Teuvo Kohonen offer an alternative to the perceptron networks.

The subsymbolic approaches seem to offer new tools for attacking the deficiencies of the symbolic knowledge systems: one of the primary promises is the enhanced capability of *machine learning.*

The dichotomy between the computational top-down approaches and the connectionistic bottom-up approaches is a fundamental one: it seems that the qualitative and quantitative worlds cannot be easily connected. In this respect, AI is related to other areas of current activity; in *complexity theory* one tries to find the simple underlying principles beyond the observed surface-level complexity. In this perspective, the symbols can be seen as *emergent patterns* reflecting *attractors of dynamic processes* in an environment. Intelligence is in the functions, not in the machinery.

AI megatrends. New applications of AI techniques are introduced ever faster. The evolving, more and more complex computer network environments have lately become one of the driving forces in AI. Distribution of intelligence and applications of intelligent agents are coming also to the shop-floor level in modern automation systems.

Below the surface, some fundamental changes are taking place. In recent years both the content and the methodology of research in AI has changed. It is now more common to build on existing theories than to propose brand new ones; claims are based on rigorous theorems or hard experimental evidence rather than on intuition, and real-world problems have substituted the toy examples. In a way, the AI field has matured. The research is more professional and serious. It is generally recognized that the problems of AI are much more difficult than was assumed in the past.

Regardless of the lessons learned, it still seems that the fluctuation between enthusiasm and despair continues in AI in some 20 year cycles. The new generations of researchers do not necessarily recognize that regardless of the new terminologies, the age-old AI challenges are still largely the same. For example, about 20 years ago it was the expert systems with the declarative knowledge representations that were proposed as the solution to the knowledge management problem — today, it is the semantic web with the ontologies that is proposed for the same purpose. And, just as 20 years ago, it again seems that maintainability of such systems is the key challenge.

Still, something has changed — developments in AI are not a cycle but a spiral. Due to technological advances, the tools are now different, and so is the view of the world. Whereas the old expert systems were monolithic entities, the new semantic web applications are truly distributed, knowledge being delocalized in the net. Perhaps the problems with knowledge maintenance can be circumvented as the system updates are carried out by distributed agents?

One of the most characteristic developments in contemporary AI in general is this transition from centralized solutions to distributed ones. It has even been said that AI today stands for "agent intelligence" or "ambient intelligence". As can be seen in (58), a shift in this direction has already been taken. However, regardless of the unifying agent framework, conceptual and practical tools to efficiently functionalize the decentralization schemes are still not yet there.

AI debate

As discussed in (58), the question "can machines think" can be compared to two other formally analogical questions: "can machines fly" and "can machines swim". Most people would agree that the answer to the first question is *yes* (airplanes can fly), but the answer to the second question is *no:* even if boats and submarines go through water, but this act is normally not called swimming — only creatures with limbs, etc., "really" swim. Similarly, the act of thinking is intuitively limited to humans.

Largely due to the semantically loaded concepts, there is an ongoing debate about the limits of AI. In the other extreme, some think that "the human mind can never understand its own functioning"; the positivists, on the other hand, are sure that "within 20 years the computer is so fast that it necessarily outperforms human". But perhaps the essence is not in the machinery but in the functioning?

Strong vs. weak AI. One of the most original contributions in AI (before the name was coined) is the *Turing test,* also known as the "imitation game" (70). This test changes the problem of intelligence into a concrete, measurable form — the experiment setup can be summarized in a slightly modified form as follows:

> An interrogator (a human) communicates with another party via a teletype. The interrogator can ask in writing whatever questions he likes, and the replies are given to him also in a written form. The computer passes the test if the interrogator cannot say whether there is a human or a computer at the other end.

This is a purely behavioral view of AI. It does not matter what are the internal processes, as long as the input-output mapping operates sufficiently. This is called the *weak AI* approach. In engineering applications, like in expert systems where the reasoning rules can explicitly be stated, the weak view of AI has been enough to reach useful results in practice. It is possible to mimic intelligent behavior simply by implementing the rule processing mechanisms. What comes to practical applications, nobody doubts the value of this pragmatic view of AI.

Intuitively, however, the above definition for intelligence is not quite satisfactory. This has parallels in psychology: the behavioristic research tradition has been substituted by cognitivism and constructivism, where the mental processes play a vital role. As it seems that the "easy wins" in AI have already been exhausted, the simple methodologies having been experimented in almost all applications, perhaps it is time to proceed towards "artificial constructivism"? To reach added value and new functionality in the AI systems, the internal structure of the knowledge representations and the inference mechanisms needs to be studied closer.

One of the most concrete criticisms against the weak AI interpretation is the "Chinese room" argument due to John Searle (60):

> The system consists of a human equipped with a rule book and various stacks of paper, some blank, some with indecipherable inscriptions. The system is in a room with a small opening to the outside. Slips of paper appear in the room through the opening. The paper slips are filled with indecipherable symbols, and the role of the human is to go through the rule book, searching there for matching symbols, and respond accordingly: writing symbols on the paper slips, selecting symbols from the stacks, and rearranging them. Eventually, a paper filled with symbols will be handed through the opening to the outside world.
>
> It turns out that the symbols are Chinese characters, and the rule book contains rules for carrying out a conversation. In the sense of Turing, this system passes the intelligence test. However, the human in the room does not understand Chinese, and the rule book and the stacks of paper do certainly not understand Chinese. Can the system be called intelligent if there is no understanding going on?

According to this argument, running a right program does not generate understanding.

In the other extreme, some researchers claim that there cannot exist intelligence without *consciousness.* An intelligent entity has to be aware of what it is doing — preprogrammed behavior does not fit our understanding of what intelligence is all about. This view is called *strong AI.* Even if the researchers and application engineers could leave the question of consciousness to philosophers, there is an on-going, heated discussion that cannot be avoided when speaking of AI. So, can the computer ever be truly intelligent?

Objections against AI. The main arguments against AI can loosely be grouped in three categories that are closely related:

1. The *argument of intentionality* is the formalization of the above Chinese room example: in an artificial system there can be no understanding (60).
2. The *argument from informality* states that the human behavior is much too complex to be captured by

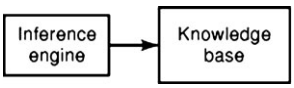

Figure 1. The basic structure of knowledge-oriented systems: the domain-specific knowledge is separated from the general inference mechanisms

any simple set of rules (12).

3. The *mathematical objection* against AI is that, as shown by Gödel, a consistent formal system is necessarily incomplete: all true facts cannot be deduced using mechanical rules. Machines are inferior to humans, who can always "step outside" the limiting logic (41).

According to Searle, consciousness is an *emergent property* of appropriately arranged combinations of neurons only. Roger Penrose goes even deeper: intelligence and "free will" emerges from stochastic quantum processes that cannot be emulated in software (52). A commentary on the informality argument is available in (67).

Without trying to answer the above questions, the *brain prosthesis* experiment illustrates the opposite *functionalist viewpoint* — intelligence cannot be bound to the physical level implementation (48):

> Assume that neurophysiology has been developed to the point where the behavior and connectivity of all the neurons in the brain are perfectly understood. Further assume that there are techniques for replacing the neurons, one by one, with identically behaving electronic devices, without interrupting the operation of the brain as a whole.
>
> By definition of the experiment, the subject's external behavior must remain unchanged compared to what would have been observed if the operation were not carried out. Is it then not also the consciousness that remains, despite the non-biological substrate?

Clearly, following one's own intuitions, one easily ends in paradoxes. In what follows, the approach is pragmatic: the goal of an AI system is to *do thinking-like things.*

Closer look at AI techniques

The viewpoint of AI in this article is rather technical, concentrating on methodologies that have proved useful. The focus is on *knowledge.* The very basic structure underlying the knowledge based systems is shown in Fig. 1. There are two parts — first, the *knowledge base* contains the domain-area expertise in an explicit, declarative rather than procedural form, and, second, the *inference engine* is the rather simple general-purpose machinery that processes the declarative knowledge according to predefined principles (to be precise, the term "inference" refers to a subclass of knowledge oriented activities; in this context "reasoning engine" would perhaps be a more appropriate name).

What is then inside the boxes; how is the knowledge represented, and how is it processed? This is studied in the following section.

About this article. This article only discusses AI from the point of view of *how to present knowledge, how to utilize that knowledge,* and *how to make the machine automatically acquire new knowledge.* The following section discusses knowledge representation formalisms, different logical systems, semantic nets, etc. Planning and problem solving based on different search strategies is studied. After that, reasoning mechanisms are studied — in addition to traditional rule-based inference, probabilistic reasoning as well. The next section is about machine learning. Finally, the experiences are summarized, and a framework is presented where knowledge representation, reasoning, and machine learning can be combined in a plausible way.

Important fields of AI, like pattern recognition and natural language processing, are discussed in separate articles in this Encyclopedia. The approach in this article is mainly *symbolic* — again, there are separate articles on fuzzy systems and connectionistic approaches like different neural network structures to be found elsewhere in this Encyclopedia. Knowledge engineering and expert systems are covered in detail in other articles, and information about practical AI tools and applications are presented in those articles.

Further information. There are dozens of books on AI available — for example, see (5). A classic textbook on AI is (72), while a more up-to-date, exhaustive presentation of the field is given in (58). The philosophical and practical questions of AI are discussed in (27). A specialized encyclopedia on AI has also been published (63). Interesting personal views and visions about AI are presented, for example, in (3), (20), and specially in (65) and (47). Collections of some of the classic publications in AI, either from the philosophical or from the cognitive science point of view also exist (see (4) and (10), respectively).

Most recent results appear in the proceedings of the major AI conferences: the largest one is the biennial *International Joint Conference on AI* (*IJCAI*), and other main meetings include the *National Conference on AI* (*AAAI*) and the *European Conference on AI* (*ECAI*). The major journals in the field of general AI are *Artificial Intelligence* and *Computational Intelligence,* and the electronic *Journal of Artificial Intelligence Research.* Various more specialized journals also exist on AI techniques and applications.

KNOWLEDGE REPRESENTATION

Computer solutions to many problems in AI depend more on the availability of a large amount of knowledge than on sophisticated algorithms. To make the large knowledge bases easily accessible, to assure fast retrieval and inference, the format of the knowledge is an important issue. The *conceptual efficiency* is also a good motivation for clever organization of knowledge in AI systems: the structure of the knowledge representation has to aid in understanding the body of knowledge to facilitate its maintenance. Compressed, domain-oriented formalisms are the key to abstracting the problem field, and they help in mastering

the knowledge intuitively. Various knowledge representation schemes have been proposed to achieve benefits in different application domains.

Representation in logic

In the community of traditional AI, there is a rather wide consensus about the nature of knowledge and reasoning: knowledge representation is *logic,* and reasoning is a form of *automatic theorem proving.* The starting point in all logic formalisms is the *first-order predicate logic* (54).

First-order predicate logic. First order logic is *universal* in the sense that it can express anything that can be programmed — in a more or flexible way. It is also by far the most studied and best understood logic scheme yet devised. Generally speaking, other proposals involving additional capabilities are still debated and only partially understood. Other proposals that are a subset of the first-order logic are useful only in limited domains. There exist special logic programming languages, like Prolog.

The basic elements in first-order predicate logic are *terms* that can be *constants, variables,* or *functions.* Another elementary concept is the *predicate* that stands for a *relation.* These building elements are used in *sentences:* whereas the terms refer to objects in the domain universe, the sentences are used to state *facts* concerning the objects. For example, the following atomic sentence (employing the predicate "Canary" and the constant "Tweety") says that "Tweety is a canary":

```
Canary (Tweety).
```

To construct more complicated sentences, logical *connectives* from the propositional calculus are available: disjunctions and conjunctions ($\vee$ for "or", and $\wedge$ for "and"), negations ($\neg$), and implications ($\rightarrow$). The following compound sentence states that "Bill is the owner of Tweety" and **"Bill** is a boy".

```
Owner(Tweety,Bill) ∧ Boy(Bill).
```

The *quantifiers* are the key to the expressional power of the first-order predicate calculus. There are the *universal* (V) and the *existential* (3) quantifiers, having the meaning "for all" and "for some" (at least one), respectively. The following sentence means that "all canaries are yellow":

```
∀x: Canary(x) → Yellow(x).
```

The quantifiers can be combined, and the order of them is significant. The following sentence means that "all canaries have (some) canary parents":

```
∀x ∃y: Canary(x) → (Canary(y) ∧ Parent(x, y)),
```

whereas the following sentence means that "there is some boy that likes all canaries":

```
∃x ∀y: Canary(y) → (Boy(x) ∧ Likes(x, y)).
```

To achieve understandability and easy maintainability, it may be preferable to leave the logical relationships in the original form that resembles natural language sentences. However, to implement efficient reasoning mechanisms, the knowledge has to be expressed in a compact form.

Transformations of expressions. There are various logical equivalences that can be used for modifying logical expressions. First, assuming that P and Q are arbitrary sentences, one can utilize the *de Morgan rules:*

```
¬(P ∧ Q) ≡ ¬P ∨ ¬Q  and ¬(P ∨ Q) ≡ ¬P ∧ ¬Q.
```

Because the quantifiers have close connection to the connectives (universal quantifier is the "conjunction over the universe of objects", for example), there also holds

```
¬∀x: P(x) ≡ ∃x: ¬P(x)    and ¬∃x: P(x) ≡ ∀x: ¬P(x).
```

Applying this kind of modifications, the sentences can be transformed into *normal forms.* After the transformations, the knowledge base may finally look like (all sentences universally quantified)

$$P_{1,1} \wedge P_{1,2} \wedge \ldots \rightarrow Q_{1,1} \wedge Q_{1,2} \wedge \ldots \;.\; P_{n,1} \wedge P_{n,2} \wedge \ldots \rightarrow Q_{n,1} \wedge Q_{n,2} \wedge \ldots,$$

so that new facts ($Q_{i,j}$) can be deduced from known facts ($P_{i,j}$) in a straightforward manner (see next section). Another form that is specially useful only contains sentences of the form

$$P_{1,1} \wedge P_{1,2} \wedge \ldots \rightarrow Q_1 \;.\; P_{m,1} \wedge P_{m,2} \wedge \ldots \rightarrow Q_m.$$

Alternatively, the above *Horn clauses* are often (for example, in Prolog) presented in another form, as

$$Q_1 :- P_{1,1}, P_{1,2} \ldots \;.\; Q_m :- P_{m,1}, P_{m,2} \ldots.$$

All these formulations make it possible to implement efficient inference mechanisms.

In first order logic, one can quantify only over objects, whereas *higher-order logics* allow one to quantify over functions and relations also. For example, in higher-order logic it is possible to say that two objects are equal if and only if all properties applied to them are equal:

```
∀x ∀y: (x = y) ↔ (∀p: p(x) ↔ p(y)).
```

Higher-order logics are essentially more powerful than first-order logic. However, very little is known of how to reason effectively with this kind of sentences — and the general problem is known to be undecidable. In practice, the first-order logic usually suffices: it is well understood and still very expressive.

Extended logic formalisms. In addition to the first-order logic, various other formalisms have been developed, either to match the "natural" way of thinking better, or to cope with practical applications. Different kinds of *intensional logics* exist for special purposes.

To act in an intelligent way, AI systems may need mechanisms to cope with concepts like *possibility* and *necessity* — it is *modal logic* that studies facts that are not only true but *necessarily* true. It is possible to construct full logic calculus based on first-order predicate logic and augmented with these concepts: note that if it is so that a fact A does

not necessarily hold, it is possible that the negation of A applies, or, written formally, $\neg\Box A \equiv \Diamond\neg A$. Another extension is the *autoepistemic logic* that formalizes the concepts of *knowing* and *believing*. It has been shown that many of the logic extensions are equal to each other, what comes to their expressional power.

One of the formalisms that have been developed to cope with real-life problems is *temporal logic*. When modeling dynamic processes using AI techniques, for example, it is very natural to model phenomena in terms of causal relationships, and to accomplish this the notion of time is essential. The basis for many temporal representations is *situation calculus* (*SC*) (42). A situation is a snapshot of the universe at a given moment, and actions are the means of transforming one situation into another. Situation calculus makes several strong commitments: the first is about discreteness of time, so that continuous processes cannot be modeled; the second assumption is about contiguity of cause and effect, so that the effects of an action are manifested immediately. The framework does not either allow concurrent actions.

Despite the limitations of the modeling based on logical, non-quantitative measures, in *qualitative physics* models are constructed also for real-life continuous systems. Another approach to capturing the complexity of real world into compact dependency structures is proposed in *system dynamics* (62).

Structured representations

There are various special knowledge architectures that are meant for special tasks to find domain-oriented, optimized representations. However, there is a dilemma: the more structured the representations are, the more complex processing mechanisms are also needed. One has to compromise between generality and conceptual efficiency. In the following, two "frame works" are presented that can be regarded as extensions of the basic logic formalisms into practical, complex applications.

Frame theories. Marvin Minsky's proposal (46) that knowledge could be represented by a set of *frames* sparked a new generation of knowledge representations. The primary goal is to capture the real-world (or "common sense") knowledge in the representations in a practical way. A frame is a complex data structure representing a prototypical situation. For example, the concept "bird" could be represented by

```
frame     BIRD size:     NUMBER constraint size > 0.0
flies:    LOGICAL default = yes
```

The contents of the frame is a list of slots that define relationships to other frames playing various roles in the definition. The above example illustrates how the value types and ranges in the slots can be constrained; and in the absence of other information, what is the default value to assume.

An essential feature of the modern frame formalisms is the inclusion of *ISA hierarchies:* structures can be shared between frames. For example, when one defines that a "bird" is an "animal", all properties of an animal are inherited by a bird. The more general frame defines the defaults, even if these default values can be redefined in more specialized frames. Another extension in the frame systems are the *procedures* that allow one to associate more sophisticated operations to frames and slots.

The idea of using frames to capture structured representations is a rather universal one, and this idea has many reincarnations. In Prolog, for example, structured representations are implemented in the form of hierarchical list structures. In special contexts, different names are used for frame-like structures — these alternative names include "scripts" and "schemata". Also more general names like "concept", "unit", and "class" have been used. As a matter of fact, one can see that this idea of classes lives on in the general-purpose *object-oriented programming languages*. The objective is to capture the essence of real-life entities in a natural way — actually, it is no wonder that the idea of concept hierarchies can be traced back to the Aristotelian taxonomies.

Network formalisms. A *semantic network* is a structure for representing knowledge as a pattern of interconnected nodes and arcs. Most versions of semantic nets are oriented toward the special features of natural languages, but the more recent versions have grown in power and flexibility so that they can be regarded as general knowledge representation formalisms. In practice, the implementation of semantic nets and frame systems can be identical; there is not very big difference between these two. As in the frame systems, also in the semantic net formalisms different kinds of extensions have been proposed to enhance the expressiveness — for example, procedural routines can be included in the nodes, and approaches towards incorporating quantifiers in the logical structure have been proposed.

The knowledge representation in predicate logic has the relations between entities as building blocks, so that knowledge about one entity is often distributed between various representational units. In frame-based representations, the relevant relations concerning an entity are collected together, and, further, in semantic nets the relations that are relevant to an entity are shown in a two-dimensional, visually understandable "mind map" form. Perhaps this understandability of semantic nets is their main advantage over the other representations.

In Fig. 2, a very simple semantic network is shown. Even if one of the deficiencies of semantic networks is the lack of common agreement on the notations, there are some practices: in the figure, the subsets and set members are connected to the "parent" sets by arcs, and arcs are also used for representing other relations (the name of the relation is given above the arc). Relations with quantifiers have the relation name in a box (like in *"all* canaries are yellow" in the figure). If all Bill's relations, etc., were written out, a real network would result; and in practical applications, to deal with exceptions (like birds that do not fly), etc., more sophisticated notations are needed.

Representing uncertainty

The knowledge that is available is often uncertain, imprecise, erroneous, or incomplete. Mechanizing the reasoning

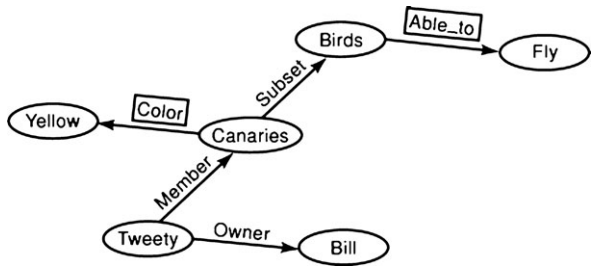

Figure 2. A simple semantic network

that is based on uncertain knowledge has proved difficult, because the classical methods that are based on logic presuppose exact facts. The management of uncertainty in expert systems has usually been left to *ad-hoc* solutions lacking sound theory, but on the theoretical side, various well-founded approaches exist (64).

There are different ways to characterize methods to handle uncertainty. One can classify the approaches to *extensional* and *intensional* ones. In extensional systems, classical logic is used as a starting point, and the uncertainty is expressed as "numerical truth values". The uncertainty of the conclusion is defined as a straightforward function of the component uncertainties. The extensional methods include multiple-valued logics (and also fuzzy logic systems), and the different certainty factor implementations that have been used in the traditional rule-based expert systems.

In intensional systems, however, more theoretical approaches are preferred; uncertainty is integrated in the "possible worlds" underlying the facts and rules, so that uncertainty propagation is based on rigid semantic models rather that on syntactical structures. Whereas the extensional approaches are pragmatic and easily implemented, the intensional ones are semantically more powerful. Reasoning principles that are based on intensional models are explained in more detail later.

PLANNING AND PROBLEM SOLVING

In *planning* and in *problem solving* the objective is to find an action sequence that would lead from the initial state to a goal state. Planning and problem solving are discussed here together, even if these subjects differ from each other in detail: the representations for goals, states, and actions are different. Planning is usual terminology in robotics, production automation, and in study of autonomous vehicles, whereas problem solving is more theoretically oriented. The real-world planning tasks are often reducible, so that the problem can naturally be divided in subtasks — the problem solving cases usually cannot.

The knowledge representations that are used for describing the problem field are those that were reviewed in the previous section (specially, *situation calculus* and the *frames* are often used as frameworks for structuring the search space and mastering the state transitions). The implementation of actions based on the knowledge representations is studied in the next section.

Problem spaces

The idea of *problem spaces* is a useful abstraction; it is a tool to reach a more unified view of AI problems. A problem space is the environment in which the search for the problem solution takes place. Generally, a problem space consists of the set of *states* and the set of *operators* that can be applied to move from a state to another. A state can be expressed as a set (vector) of variable bindings that uniquely determine the situation at hand; the operators modify the state variable values. A *problem instance* is a problem space together with an initial state and a goal state; the goal state can be given explicitly, or if there does not exist a unique goal, certain properties are defined that have to be satisfied by the goal state. Usually various subtasks need to be completed before the goal state can be reached — the planning or problem solving task is to find a sequence of operators that map the initial state into the goal state. In a textual form, the primitive actions, or the state transitions can be presented in the rule form

<Preconditions> → <Postconditions>,

so that if the preconditions are fulfilled, the postconditions are used to modify the current state.

The transitions from a state to another are conveniently presented as *graphs.* The graphs are often expressed in a simple form as *trees,* where the path from the initial state or *root node* to any of the other nodes seems to be unique. However, usually there are various alternative paths, and the tree-form representation becomes redundant, various nodes in the tree representing essentially the same state.

In simple cases, the problem space reduces to a *state space.* In state space the nodes represent actual configurations of the problem to be solved, and the edges represent primitive actions. The *problem-reduction space* is another type of a problem space: in problem-reduction spaces the nodes represent complete subproblems that can be solved by single primitive actions, and the edges represent problem-reduction operators which decompose the given problem into a set of subproblems.

Search strategies

Whatever is the structure of the problem space, to find the path from the initial state to the goal state, efficient *search methods* are needed.

Brute-force methods. The brute-force search algorithms are the most general methods for search, because no domain-specific knowledge is needed. The basic brute-force search methods are *breadth-first search* and *depth-first search* (see Fig. 3; the search spaces are presented in a tree form).

Breadth-first search starts by generating all the successors of the root node (this is known as "expanding" a node). Next, all the successor nodes are expanded, generating all the nodes at level two in the search tree. Breadth-first search continues by expanding one complete level of the tree at a time until a solution is found. Because this method never expands a node before all the nodes at shallower levels are expanded, the path that is finally found is necessarily the shortest one. The main drawback of breadth-

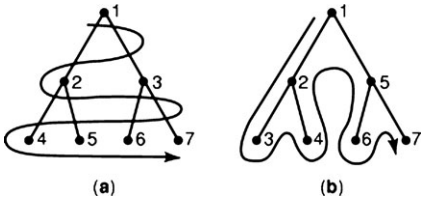

Figure 3. The two elementary search strategies: *breadth-first* (a) and *depth-first* (b)

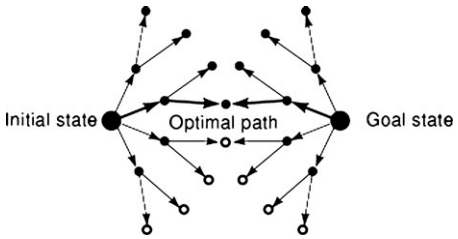

Figure 4. Bidirectional search, starting simultaneously from the initial state and the goal state

first search is its memory requirement: each level of the tree must be saved in its entirety in order to generate the next level. The complexity of this method is related to the *branching factor,* or the number of successors of the nodes.

Depth-first search avoids these space limitations by proceeding only along one search path at a time. Only one of the successors of the root node is expanded first, and after that one of its successors, etc., until the search terminates naturally or it is cut off at certain depth. If the solution has not been found along this path, the algorithm backtracks recursively, expanding the next successor of the previous level node, and so on. In general, depth-first search expands nodes in a last-generated, first-expanded order. The disadvantage of depth-first search is that in general it requires a fixed cutoff depth in order to terminate: if this cutoff limit is too low the solution will never be found, and if it is too high a large price is paid in terms of execution time.

Bidirectional search is yet another brute-force method. The main idea of bidirectional search is that instead of blindly searching forward from the initial state until the goal is reached, one can start the search simultaneously from the goal state, so that the two breadth-first search processes finally meet in the middle (see Fig. 4). However, bidirectional search requires an explicit goal state rather than a goal criterion only; furthermore, the operators in the search space must be invertible, so that *backward-chaining* can be accomplished.

Heuristic search. The brute-force methods suffer from the fact that they are essentially blind searches. The idea of heuristic search, on the other hand, is to utilize the additional information there may be available. This additional information, or the *heuristic* is often given as an evaluation function that calculates the (approximate) cost of the present situation, estimating the distance from the current state to the goal state. A number of different algorithms make use of heuristic evaluation functions — here, the simplest ones, namely *hill-climbing, best-first search,* and *A* algorithm* are briefly reviewed.

The idea of the hill-climbing algorithm is straightforward: assuming that the maximum value of the evaluation function should be found, always proceed towards the maximum increase in the function value. That is, when the current state is expanded, the evaluation function is applied to all of its successors, and the successor state with the highest value is selected as the next state. Unfortunately, there are a number of problems with this approach — first, if the sequence of states is not stored, the same states may be visited over and over again, so that the algorithm may never terminate. The second problem is typical to all steepest ascent (descent) methods: the search may get stuck in a local maximum. This problem is remedied by the best-first search algorithm.

In best-first search, the list of visited states is stored to avoid looping forever, and, additionally, also the list of states that have been generated but not yet expanded is stored. In this latter list, the available search tree branches are ordered according to their evaluated value, and the assumedly best state in this list is always selected as the next state to expand. This strategy gives the algorithm the ability to resume a path that was temporarily abandoned in favor of a path that appeared more promising. The best-first search is guaranteed to eventually find the global optimum (assuming that there are no space or time limitations), but it usually does not find the shortest path to the goal. The reason for this is that only the assumed cost from the current state to the goal state is weighted, and the cost from the initial state to the current state is ignored. To fix this problem, the A^* algorithm is needed.

The A^* algorithm is a best-first search algorithm in which the figure of merit associated with a state is not just a heuristic estimate, but rather has the form $f(n) = g(n) + h(n)$, where $g(n)$ is the cost of the path from the initial state to the state n, and $h(n)$ is the heuristic estimate of the cost from the state n to the goal state. In other words, $f(n)$ is an estimate of the total cost of the cheapest solution path going through the state n. At each point of the algorithm the state with the lowest value of f is chosen to be expanded next. An important theorem concerning the A^* algorithm is that it will always not only find the goal state, but also find the optimal route there (assuming that the heuristic function $h(n)$ never overestimates the actual distance to the goal). The main drawback of this algorithm is its memory requirement; there are more sophisticated algorithms to avoid this problem, for example, *beam search* and *iterative-deepening-A**.

About games

Games have always been an important application field of AI research. One of the reasons for this is that most games are based on rigid rules that can easily be formalized, even if successful playing seems to be quite an intellectual challenge.

There are classes of games where explicit winning strategies exist; however, normally the only approach that is available is search. A *game tree* is an explicit representation of all possible plays of the game, the root node being the initial position and the terminal nodes representing win, loss, or draw. In non-trivial games, the search space is

so huge that heuristic search methods are needed. When constructing a game-playing algorithm, the basic search procedures often need to be modified — for example, in two-player games, the possible actions of the opponent have to be processed in a different way compared to the own ones. This problem setting can often be represented as an *AND/OR graph:* whereas only one of the own moves (hopefully, the best one) needs to be expanded, *all* of the opponents replies need to be taken care of. The nodes at every other level in the search tree can be called OR links and the other ones are AND links, respectively.

A practical way to formalize many two-player games is to use *minimax algorithms* that have a close connection to the AND/OR graphs: these algorithms find the path through the search space so that the own best actions with the highest profit are selected, whereas the opponent's worst actions (from the own point of view) are selected. In practice, clever search tree pruning methods are additionally needed to achieve acceptable performance of the game-playing algorithms.

Chess is perhaps the most widely studied application example, it has even been used as a benchmark problem for AI methodologies, and a little closer look at it is taken here. In principle, chess is a two-player game, and the above guidelines can be applied: the legal moves define the transitions between states, spanning the game tree. The size of the game tree is immense: it has been estimated that there are about 10^{120} different games possible, and it is clear that heuristic search methods are needed. There are different criteria that need to be weighted when evaluating the chess board configuration; for example:

- Material balance. Different pieces have different importance, and nu meric values can be defined accordingly (queen 9, pawn 1, etc.). The sum of these piecewise values gives a crude estimate of the material power.
- Mobility of the pieces. The gross mobility can be defined as the total number of moves that are possible in the given configuration.
- Positional advantage. The center of the board is usually the key to governing the game; an isolated pawn is vulnerable to attack, etc.

All such factors should be evaluated in some way; defining the evaluation function is a difficult task, and the human pattern analysis capability has proven to outperform its algorithmic counterparts. It is the processing capacity of the computer that is needed to compensate for this handicap — and, lately, one of the ultimate AI goals was achieved: the "Deeper Blue" program was able to beat the world champion in chess.

REASONING AND INFERENCE

Reasoning in general refers to several different kinds of activities, like making assumptions about a situation, diagnosing possible causes for some condition, and analyzing and organizing the facts and data concerning some problem. All these tasks can be carried out by AI programs. A specially important subclass of reasoning activities is drawing conclusions from facts and suppositions, or *inference.* Methods to achieve automated inference will be concentrated on in this section.

Inference principles

Knowledge representations that are based on logic can efficiently be utilized in an automated inference process. Programs that perform in this way are often called *inference engines* — these inference engines range from user-guided, interactive systems to fully autonomous ones. In classical AI the processing of knowledge is centralized, but also distributed control mechanisms have been proposed — for example, there are the *blackboard techniques,* etc.

Logical inference. The prototypical reasoning method in propositional logic is the *modus ponens* inference rule:

$$P \quad P \rightarrow Q \Rightarrow Q.$$

More generally, various facts can be substituted simultaneously using the *generalized modus ponens* rule:

$$P_1, \ldots, P_n \quad P_1 \wedge \ldots \wedge P_n \rightarrow Q \Rightarrow Q.$$

Compared to the basic modus ponens rule, the generalized version takes "longer" and more sensible reasoning steps, even though there is no difference in expressional power. However, an inference system that is based on the modus ponens rules alone is clearly *not complete:* not all true sentences can be derived within that framework, and something more general is needed. Another age-old reasoning rule is the simple *syllogism:*

$$P \rightarrow Q \quad Q \rightarrow R \Rightarrow P \rightarrow R.$$

For example, if it is known that "canary is a bird" and "bird flies", one can deduce "canary flies". The *generalized resolution* principle is an extension that can efficiently be applied for knowledge bases that consist of Horn clauses:

$$P_1 \wedge \ldots \wedge P_m \rightarrow Q_k \quad Q_1 \wedge \ldots \wedge Q_{k-1} \wedge Q_k \wedge Q_{k+1} \wedge \ldots \wedge Q_n \rightarrow R \Rightarrow Q_1 \wedge \ldots \wedge Q_{k-1} \wedge Q_{k+1} \wedge \ldots \wedge Q_n \wedge P_1 \wedge \ldots \wedge P_m \rightarrow R.$$

Surprisingly, the resolution principle alone suffices, no other inference mechanisms are needed. In logic programming (in Prolog, for example), problem solving and inference tasks are presented in a theorem proving framework, that is, the axioms are written as Horn clauses, and the goal is to find the inference steps that are needed to deduce the given theorem. In practice, rather that showing that the theorem is consistent with the axioms, it is shown that the *negated* theorem contradicts the axioms in the knowledge base. It can be shown that resolution by *refutation* (or *proof by contradiction,* or *reductio ad absurdum)* is complete (or, rather, *semidecidable* so that true (derivable) sentences can be derived from the premises, but one cannot always show whether a sentence is false or not derivable).

The above deduction rules were written for clauses in propositional logic with no variables. The situation becomes slightly more complicated in first-order predicate logic, where the terms containing variables first have to be *unified.*

Because the sentences in the canonical forms only contain universal quantifiers, it is possible to substitute the variables with arbitrary constants; that is, $\forall x$: P(x) → P(a). Without going into details, the goal is the find the *most general unifiers* that make the corresponding arguments look the same. For example, if the terms Owns(Bill, x) and Owns(y, z) have to be unified, the variable binding that makes the least commitment about the variables is $x = z$ and $y =$ Bill (before unification the conflicting variable names have to be *standardized apart;* see (8)).

Production systems. Most logical inference systems (like Prolog) are *backward-chaining,* that is, they search for a constructive "proof" that satisfies the query. An alternative is a *forward-chaining* approach, where inference rules are applied to the knowledge base, yielding new assertions until no more rules *fire* (or some other criterion is fulfilled). The forward-chaining approach is specially useful in real-time applications, where the intelligent agent receives new information on each operation cycle, adding the new percepts to the knowledge base, and reacting accordingly.

A typical production system contains a *working memory* where the atomic variables defining the state are stored. The *rule memory* contains the set of inference rules — the inference rules are of the familiar form <conditions> → <actions>, where the actions typically modify the contents of the working memory. During each operation cycle, the system computes the subset of rules whose left-hand side is satisfied by the current contents of the working memory (this unification task can be computationally very expensive; there are efficient methods to relieve this problem, for example the *rete algorithm).* From this *conflict set* one of the rules is selected to be executed. There are various heuristics that can be applied to this *conflict resolution* task:

- No duplication. The same rule is not executed twice on the same ar guments.
- Recency. Execute rules that refer to recently created working memory elements (in effect, this results in depth-first operation; in an interactive system, this refinement strategy "behaves reasonably").
- Specificity. Prefer rules that are more specific (this principle makes it possible to implement *default reasoning:* if no special cases have been defined, the most general default action is taken).
- Operation priority. Some rules may be more important than the others; these rules should be preferred.

Forward-chaining production systems were the foundation of much influential work in AI. In particular, the XCON system (originally called R1) was built on this kind of an architecture (44). This system contained several thousand rules for designing configurations of computer components, and it was one of the first clear commercial successes in the field of expert systems.

It seems that production systems are also popular in *cognitive architectures* that try to model human reasoning. For example, in ACT−R (1) and in SOAR (37), the working memory of the production system is used to model the human short-term memory, and the productions are part of long-term memory.

Connections to problem solving. It is often instructive to try and see a problem in a wider framework — for example, it may be helpful to recognize the differences and similarities between logical inference and problem solving (as discussed in the previous section). It turns out that the previously presented concept of state space makes it possible to have a unified view (in the case of monotonic reasoning; see below).

The knowledge base that is written as first-order predicate logic sentences defines a (possibly infinite-dimensional) *Herbrand universe* spanned by the ground clauses (for example, see (8)). This Herbrand universe can in principle be identified with a state space. In problem solving one tries to find a sequence of transformations to reach another state, and in inference one tries to find out *what is the current state:* in which state all the known facts are consistent? In problem solving the problem states are simple and the transformations may be complicated, whereas in inference tasks the state space itself is complex and high dimensional. In problem solving the actions define the state transformations, whereas in inference the rules define interrelations between the state variables, constraining the feasible solutions. This kind of state-space approach to knowledge representation is further elaborated on and concretized in the final section.

In principle, augmenting the state space appropriately, the problem solving tasks can be expressed as logical inference tasks. However, the representations become less understandable — and in AI, one of the basic ideas is to use domain-oriented representations. The problem solving tasks and the production systems are better managed when the unknown state components can be hidden and ignored. Additionally, the more limited languages that are used in production systems can provide greater efficiency because the branching factor is reduced.

Problems with logic representations

The validity of the two-valued "crisp" logic has been questioned. It has been proposed that *fuzzy logic* would be better suited for modeling real expertise, and a heated discussion continues. In this section two basic problems are studied that impair the operation of knowledge-based systems that are based on traditional techniques — problems that become evident in systems that are large enough.

Nonmonotonic reasoning. Many knowledge systems differ from classical logic in a very fundamental way: the reasoning in them is *nonmonotonic,* so that new information can refute earlier conclusions. In classical logic no such phenomenon emerges; what is the reason for the operational difference? The answer is the *closed world assumption:* if there is no affirmative statement in the knowledge base, a proposition is assumed to be false. A new piece of information (the proposition is true after all) can then dramatically change the overall reasoning results. Whereas strictly speaking no logical inference is possible in case of

missing information, in the practical knowledge systems a pragmatic compromise has been adopted to avoid the deadlocks in reasoning. For example, Prolog language is based on this closed world assumption.

More generally, there are various kinds of *default logics* and formalisms (compare to the frame structures) that extend this idea of "typical" values if no more accurate knowledge is available. One of the classic approaches is *circumscription* (43). Nonmonotonic logic systems are reviewed in detail in (22).

The nonmonotonic nature of the knowledge representations causes surprises in reasoning; what is perhaps more acute, *belief revision* becomes a difficult problem. When the knowledge base is being updated and new facts are added, old inference sequences may become outdated; there are no efficient methods for maintaining the integrity.

There is another aspect that is rather closely related to the nonmonotonicity problem; that is *frame problem.*

Frame problems. A classic problem in AI is how to present *common sense knowledge,* that is, how to capture all the nuances of everyday life. In technical terms, this problem often emerges in the frame systems, and is there called the *frame problem:* the number of frame axioms that would be needed to handle all special situations is unrealistically high. The incomplete coverage of all possible variations causes *brittleness* in reasoning: the same reasoning tools that normally work fine may collapse altogether when there is a seemingly irrelevant change in the situation.

There are different kinds of efforts to attack these common sense problems. An example of the most ambitious approaches is the CYC project (39), where the basic nature of the abstract yet fundamental common sense concepts is explicitly represented. This huge corpus of knowledge is proposed as an underlying structure below standard knowledge-based systems, so that when the specialized reasoning system fails, the underlying knowledge base would be utilized instead.

A simpler approach to the common sense problem is called *case-based reasoning:* problem situations are stored, and when facing a new situation, a similar, previously solved case is taken as the prototype which is then modified according to the current situation. Promising results have been achieved (40).

Probabilistic reasoning

There are different views of what is the source of uncertainty in the knowledge systems, and, correspondingly, there are various approaches to automate reasoning that is based on uncertain information. The models that are based on symbolic representations are mostly designed to handle the uncertainty that is caused by the *incompleteness* of the information (like the default logic formalisms), and they are usually inadequate to handle the case of *imprecise* information because they lack any measure to quantify the levels of *confidence.* A well-established approach to utilizing pieces of information with varying degrees of belief is known as *probabilistic* or *plausible* reasoning (51). In more recent work, like in the *Dempster-Shafer theory,* a more ambitious view of uncertainty is adopted (for example, see (64)). Only probabilistic reasoning is discussed in what follows.

The heart of probabilistic reasoning is the *Bayes rule* that combines the probabilities of hypotheses and the corresponding evidence:

$$P(\mathrm{H}|\mathrm{e}) = \frac{P(\mathrm{e}|\mathrm{H}) \cdot P(\mathrm{H})}{P(\mathrm{e})}.$$

This means that the *a posteriori* belief of the hypothesis H after receiving the evidence e, denoted $P(\mathrm{H}|\mathrm{e})$, can be calculated when one knows the *a priori* probabilities $P(\mathrm{H})$ and $P(\mathrm{e})$, and the likelihood $P(\mathrm{H}|\mathrm{e})$ (the probability that e will materialize assuming H is true). The denominator $P(\mathrm{e})$ in the formula is constant, because it is assumed that $P(\mathrm{H}|\mathrm{e})$ and $P(\mathrm{H}|\neg\,\mathrm{e})$ sum to unity.

Whereas the above formula is, mathematically speaking, a straightforward identity stemming from the definition on conditional probabilities,

$$P(\mathrm{A}|\mathrm{B}) = \frac{P(\mathrm{A},\mathrm{B})}{P(\mathrm{B})} \quad \text{and} \quad P(\mathrm{B}|\mathrm{A}) = \frac{P(\mathrm{A},\mathrm{B})}{P(\mathrm{A})},$$

in probabilistic reasoning it is regarded as a normative rule for updating beliefs in response to evidence. The basic rule can easily be extended — for example, assume that there are alternative hypotheses $\mathrm{H}_1,\ldots,\mathrm{H}_\mathrm{n}$ and various pieces of evidence $\mathrm{e}_1,\ldots,\mathrm{e}_m$. Then the Bayes rule can be written for each hypothesis H_i as

$$P(\mathrm{H}_i|\mathrm{e}_1,\ldots,\mathrm{e}_m) = \frac{P(\mathrm{e}_1,\ldots,\mathrm{e}_m|\mathrm{H}_i) \cdot P(\mathrm{H}_i)}{\sum_{j=1}^{n}(P(\mathrm{e}_1,\ldots,\mathrm{e}_m|\mathrm{H}_j) \cdot P(\mathrm{H}_j))},$$

assuming that the following conditions are fulfilled:

- Each hypothesis H_i is mutually exclusive with any other hypothesis, and the set of n hypotheses is exhaustive, or

$$P(\mathrm{H}_i, \mathrm{H}_j) = 0 \text{ for } i \neq j, \text{ and } \sum_{i=1}^{n} \mathrm{P}_i = 1.$$

- Each piece of evidence e_j is conditionally independent under each hypothesis, that is,

$$P(\mathrm{e}_1,\ldots,\mathrm{e}_m|\mathrm{H}_i) = \prod_{j=1}^{m} P(\mathrm{e}_j|\mathrm{H}_i).$$

The Bayesian approach requires a large amount of data to determine the estimates for the prior and conditional probabilities. Such a requirement becomes better manageable when the problem can be decomposed, or when it can be represented as a sparse *Bayesian network.* In Bayesian networks nodes (or variables) representing hypotheses and pieces of evidence alike, are connected to other nodes only if there is a causal dependency between these variables. The formed directed graph defines a hierarchy between the nodes. In these networks the dependencies among variables are known and only the explicitly defined conditional probabilities between the variables have to be obtained experimentally.

When applied to complicated networks, the Bayesian approach has various shortcomings. The assumptions on

which this method is based are not easily satisfied — for example, if the network contains multiple paths linking some evidence to the same hypotheses, the independence assumption is violated: the pieces of evidence that are used for inference, or the values of the variables along the parallel paths, are bound together. Similarly, the claims of mutual exclusiveness and exhaustiveness of the hypotheses are not very realistic: the strict exclusiveness (only one hypothesis can apply at a time) means the same limitations to expressive power as the single-fault assumption in the simplest diagnosing systems does. The exhaustiveness assumption (all possible hypotheses are known *a priori)* is violated if the problem domain is not suitable to the closed world assumption.

Another source of difficulties is caused by the knowledge representation with no "fine structure": for example, disjunctive clauses and other *nonsin-gletons* cannot be assigned probability values. This implies the requirement for a uniform granularity of evidence. Also, there are no mechanisms to detect conflictive information. However, some researchers think that the most severe limitation of the Bayesian approach is its inability to represent *ignorance* and other flavors of everyday uncertainty.

MACHINE LEARNING

One of the most characteristic capabilities of an intelligent system is its ability to *learn.* However, to look intelligent, the system must not only store old experiences — in this context, it will be assumed that a learning system has to be able to find new structures that can represent the knowledge in a better, more polished way. The connectionistic approaches that outperform the traditional AI methodologies what comes to *adaptation,* or fine-tuning of the parameters, are no panacea when the task is to *optimize structures.* In this section, only those machine learning methods are discussed that are based on structural, symbolic representations. Machine learning is discussed in more detail, for example, in (19).

Classification of approaches

The machine learning algorithms are often characterized according to how independently they operate:

- In *supervised learning* the correct behavior is explicitly given to the learning system together with the input example data.
- In *reinforcement learning* an outside critic evaluates the system behavior according to some cost criterion, and the learning mechanism tries to enhance the system behavior using some kind of search strategy for cost minimization.
- In *unsupervised learning* no "correct" results exist; the system just tries to find some kind of a model for the input data.

From the point of view of strong AI, perhaps the most interesting of the above classes is unsupervised learning: when the system is not told how it should manage the input data, something unanticipated and truly "intelligent-looking" behavior can emerge without explicit preprogramming. What is essential is that new structures are constructed based on the earlier ones *incrementally:* illusion of intelligence (sometimes) emerges when simple constructs are stacked together.

Another classification of machine learning algorithms can be done as defined in (7): the algorithm is either based on *learning of rules* or *learning of concepts.* Put informally, the concepts are the symbols, or the names of the primary objects and their relations in the logical sentences, whereas the rules are the logical sentences themselves. These two alternatives are discussed separately in what follows.

Learning of rules

Inductive inference is the process of hypothesizing a general rule from examples. Inductive inference is based on some kind of *background theory* that defines the concepts that are available, the possible constraints, and the class of examples that are used in the learning process. Positive and negative examples are needed to *generalize* and to *specialize* rules, respectively.

There are different ways to modify rules to match the examples, and, simultaneously, simplify the set of rules. Typical generalization operations include:

- Elimination of a conjunct. One or more constraints can be eliminated from the set of preconditions; for example, the rule

  $\texttt{Flies}(x) \wedge \texttt{LaysEggs}(x) \wedge \texttt{Feathered}(x) \rightarrow \texttt{Bird}(x)$

  can be generalized (assuming that the property "to have feathers" is redundant) as

  $\texttt{Flies}(x) \wedge \texttt{LaysEggs}(x) \rightarrow \texttt{Bird}(x).$

- Addition of a disjunct. Rules can be relaxed by adding disjunctions to the set of preconditions; for example,

  $\texttt{Flies}(x) \rightarrow \texttt{Bird}(x)$

  can be extended as

  $\texttt{Flies}(x) \vee \texttt{Penguin}(x) \rightarrow \texttt{Bird}(x).$

How this kind of modifications are carried out, is a matter of heuristic search strategies — in the absence of smoothly behaving cost criterion, no analytical optimization methods exist. For example, *genetic algorithms* have been proposed and experimented in this rule optimization task.

One of the other basic keywords in the field of rule learning is *learning by analogies,* or *similarity-based learning.* The idea is to find analogous rule structures, and extend the existing rules to apply to new types of objects.

Another discipline that has lately become popular is *learning by discovery,* where the human process of inventive acquisition of new rules is imitated.

Concept formation: examples

Concepts are the means of structuring a complex domain. Automatic concept formation is an active area of research (16), but it is also highly complicated: what characterizes "good" concepts — this question is closely connected to the

mostly undiscovered principles of the human mental processes. It seems that good concepts are tailor-made for each application domain, capturing the domain-specific dependency structures in an optimal way. In what follows, examples of some influential systems are briefly presented.

The "Automatic Mathematician" program AM is a classic system showing that some aspects of creative research can be effectively modeled as heuristic search (38). The system starts with only a very elementary knowledge of the basic ideas in number theory, but it creates new, more sophisticated concepts. The "goodness" of a new concept is evaluated using a heuristic criterion that measures its *aesthetic* appearance, or the assumed "interestingness" of the concept. The operating principle of AM differs very much from the theorem provers: it generates hypotheses rather than proofs, relying on the heuristic criterion rather than on proof procedures. It should be noted that AM was capable of deriving something as advanced as the famous *Goldbach conjecture.* However, the extended version of the program, called EURISKO, showed that the same idea did not to work as well in non-mathematical application domains.

The EPAM model is one of the most significant general models of concept formation (15). Even if it was originally designed as a model of rote learning behavior, it has since been used to model a varied set of phenomena, including word recognition and memorization of chess patterns. EPAM constructs a *decision tree* in an unsupervised, incremental manner. Each leaf in the tree represents a "concept", or a pattern of features that matches a subset of observations. When a new observation enters, it is classified down a path of matching arcs in the tree to some leaf node. If the observation matches the concept that is stored at the leaf exactly (that means, all features are equal), then a process *of familiarization* occurs: the leaf is specialized further by adding a feature that is present in the new observation but not yet present in the leaf node. On the other hand, if there is a mismatch of one or more features, the tree is expanded by increasing either the breadth or the depth of the tree. In sum, leaves begin as general patterns that are gradually specialized by familiarization as subsequent observations are classified; because of the unsupervised manner of the concept learning, the emerging concepts are unlabeled, characterized solely by their special features. The structure of EPAM is quite simple, but it still accounts for a significant amount of experimental data — in particular, the model simulates nicely the mental memorization of stimuli as a general-to-specific search process. It was the first computational model of concept formation, and it has had a significant effect on the systems that have been developed later.

The CLUSTER/2 algorithm (45) was the first system doing *conceptual clustering.* It forms categories that have "good" conjunctive expressions of features that are common to all (or at least most) category members. The criteria for category goodness in CLUSTER/2 are conflicting: on the other hand, the *simplicity* criterion prefers short expressions, while the *fit* criterion prefers detailed characterizations — these criteria are used to guide an iterative optimization procedure. What is important in CLUSTER/2, it was the first attempt towards coupling *characterization* and *clustering.* In conceptual clustering methods, the categorization and the interpretation of these categories are tightly coupled coroutines (as compared to the various *numerical taxonomy* methods for analysis of data where the classification of the created clusters is largely manual).

In the following section, the automatic formation of categories is further elaborated on.

FUTURE PERSPECTIVES

The traditional knowledge engineering approaches work well if the application area is narrow enough, if there is an expert that can define the inference rules explicitly, and if the knowledge base needs not be updated. However, as the potential of "good old-fashioned AI" (GOFAI for short; see (27)), has been evaluated in practical applications, it seems that new needs are emerging. One problem is how to assure *integrity:* how to update the knowledge base in a consistent manner, and how to assure relevance of the rules so that outdated rules are eliminated? There is more and more information available, but there are fewer and fewer human experts available to implement rules — the AI systems should autonomously adapt.

The current AI methodologies give no tools for attacking the new challenges. To avoid the deadlock, it is necessary to widen horizons.

In what follows, a very concrete, engineering-like approach to AI will be studied. Rather than discussing whether the weak or the strong view of AI should be adopted, take an application oriented view, simply defining intelligence as

> *ability to adapt in a previously unknown environment.*

This working hypothesis leads to a new problem setting — the goal now is to find ways to *modeling of the environment.* The problem field should be *abstracted* somehow, autonomously, without external supervision. In this framework, questions of "consciousness", etc., can be ignored, and many of the age-old arguments in AI can be avoided.

Related disciplines

What this "intelligence modeling" actually means — to have some perspective in this question, an interdisciplinary approach is necessary. In this section the findings from various related fields are first summarized, and a new approach combining these results is sketched thereafter.

Cognitive psychology. It is reasonable to start from the assumption that the behavior of the models for intelligence should share the same properties as does the only known truly intelligent system, the human brain. *Cognitive psychology* (for example, see (68) or (34)) studies the mental phenomena on the systemic level. In cognitive science a wealth of concrete facts have been found that are in conflict with mainstream knowledge engineering methodologies (28); the following points will be concentrated on below:

- The knowledge representation and manipulation structures in the contemporary knowledge systems

differ from the functions of the human memory. For example, the operational differences between the long-term memory (*LTM*) and short-term (working) memory (*STM*) is not addressed (26). Even if this strict dichotomy has been questioned, it still offers a firm basis for concrete analysis.

- The *fuzziness* of categories (57) has not been embedded as an integral entity in the knowledge representations. The "crispness" of the concepts is one reason for the brittleness of the inference systems, and the gap between the symbolic and the subsymbolic levels makes machine learning a difficult task.
- A striking difference between the artificial and the real intelligent systems is visible in the *shift from novice to expert* (13).

In expertise research it has been noticed that the expert reasoning becomes faster and more automated compared to the novice reasoning. This is known as the *speedup principle* (9). The traditional way to enhance knowledge bases is through growing the number or sophistication of the rules (for example, see the CYC project (39)) — but how could the expert make the conclusions faster than the novice if the rules were more complex?

Very different motoric and mental skills can be regarded as manifestations of special expertise. There are attempts towards a general theory of expertise (14); however, without deeper analysis it is impossible to see the general underlying laws.

Neurophysiology. Another source of evidence is *neurophysiology* that studies the mental phenomena on the neuronal level. The gap between the lowest physical level and the level of cognitive processes is so huge that any conclusions must be made with extreme care. However, in the analysis of sensory signals, specially in the field of vision, interesting results have been obtained: it seems plausible that on the visual cortex a special kind of pattern reconstruction takes place, so that the visual image is presented in terms of elementary *features.* The cortical cells constitute a reservoir of feature prototypes — at any instant only a subset of all available features is utilized. The active cells define the *coding* of the observed image; because only a few of the cells are active, the code becomes *sparse* (50).

The potential of sparse representations has been studied earlier (33), and also in a distributed production system architecture (69). Using simulations it has been shown that sparse representations can be obtained autonomously, for example, using *anti-Hebbian learning* (18) or *Independent Component Analysis* (*ICA*) (32).

As motivated in (1), there cannot exist separate faculties for different mental capabilities in the brain — the same principles are responsible for all different kinds of high-level activities. This uniformity principle can be extended further: it can be assumed that the above mentioned *sparse coding of features* is characteristic to *all levels of mental behavior.*

There are various frameworks available for modeling of mental phenomena, most notably perhaps ACT-R (1) and SOAR (37). The validity of a mental model cannot be rigorously proven — by extending a model, additional phenomena can be explained *ad infinitum.* The only problem is that the model structure becomes increasingly complex, thus making the model intuitively less appealing.

Robotics. Artificial intelligence differs from cognitive science because the emphasis is on synthesis rather than only analysis. The best proof of theories is practice, and there are lessons learned in the field of modern robotics.

One of the cornerstones of early AI was *cybernetics* (71). It is the control structures that intuitively contain the kernel of intelligent behavior. After the early years, developments in robotics proceeded in the direction of symbolic approaches and explicit problem solving tasks, alongside mainstream AI — however, these robots are restricted by their narrow worlds. It seems that it is time to look back to basic principles, or the ideas of control.

The today's emphasis on the environment and survival is manifested in the "subsumption architectures" (6): intelligence is seen as a holistic adaptability in one's environment. The robot systems are again very simple. But perhaps there is no need to go back to the simplest control systems — control theory has matured during the decades, too. In modern control theory one emphasizes the role of *model-based control* — what is this model like now?

System theory. System theory is the theory about how to construct models. Additional challenges to traditional system theories are caused by the intuition that the mental model should not be centralized but distributed. It seems that the framework of dynamic systems offers tools also for understanding tensions caused by the environment: there is a *dynamic equilibrium,* determined by the *attractor structures* of the environmental properties. The *emergent patterns* can be observed as the asymptotic behaviors converge to some fixed state (30).

How to compare models, or how to make structurally different models mutually comparable? System theory that offers tools also for defining concrete criteria also for mental models. Information theoretic analyses give a rather firm basis for constructing "optimal" models (for example, see the *Minimum Description Length principle* (*MDL*) as defined in (55)). Without going into details, the idea is to weigh the *complexity* of the model against its *representational power:* the more one needs degrees of freedom, or free parameters, to adjust the model to stand for a specific application, the lower the model "goodness" is (in fact, this is the modern formulation of the *Occam's razor:* simple explanations should be preferred).

Behaving as well as possible given some limited resources — this kind of an optimization viewpoint is becoming an important new approach in AI research, known as *rationality* and *bounded optimality* (59). However, in a complex environment it is difficult to explicitly define the constraints, and the resulting analyses seem to be cumbersome. To avoid the complexities, the analysis is in this context started from the first principles. How to construct a mental representation so that the capacity requirements are minimized?

Ontology and epistemology. The fundamental role of the mental representations is to convey *semantics,* the *meaning* of constructs: what is the link between the internal mental structures and the outside world? To evaluate the representational power of a mental model, to compare different models reasonably, and to find tailor-made models to match the observed data, one is facing deep questions. The nature of semantics has been one of the main tracks in philosophical discussion over the years (17).

The strong AI goal of "self-consciousness" can be slightly relaxed: the machine only has to "understand" the *observations* in some sense. Only then, intelligent-looking behavior can be reached without explicit preprogramming.

So, what is the nature of the observation data, or what is its underlying structure? In other words, what is the data *ontology?*

In philosophy, *empirism* offers a fruitful framework for AI, as contrasted to the metaphysical paradigms like *dualism* or *rationalism.* Rather than assuming that there were predestinated *a priori* mental structures that could not be automatically reconstructed, in empirism one assumes that knowledge emerges from observations.

Along the extremely reductionistic and simplistic lines of reasoning that are here adopted, the goal is to find the data structures starting from *tabula rasa:* no specialized data structures exist. This starting point parallels with the approach of David Hume: all concepts and mental constructs must have direct correspondence with observations, either directly or indirectly. As concluded by Immanuel Kant, the only pre-existing thing is the structure construction mechanism.

This view results in the *naturalistic* semantics: "temperature" is a concept that is directly defined in terms of thermometer reading, etc. The semantics of more complex concepts is defined by their *context:* how a concept is related to other ones dictates its contents. This means that rather than speaking of a meaning of an individual symbol one has to study *semantic nets* as basic entities. The naturalistic interpretation makes it possible to avoid the problem of *infinite regress,* where concepts are defined in terms of other concepts *ad infinitum.*

When the contents of the semantic universe are learned empirically, starting from scratch, the observed data directly dictates the contents of the mental representations. This leads to the *epistemic* problem setting: what is *knowledge,* and what is *truth* in the first place? Things that have been observed together many times, become coupled together — it is *relevance* that is of primary importance rather than "truth", determining what are the "beliefs" of the system.

In connectionism, these questions are discussed as *computational* or *procedural* semantics (31). Perhaps a better name would be *associative semantics,* to emphasize the need of parallel processing.

Mathematics and statistics. There does not exist a language for presenting parallel phenomena. Mathematical formalism helps in circumventing this problem — additionally, in mathematical framework the theories can be easily analyzed and tested using simulations.

In practice, when modeling complex environments, the data structures become so huge that efficient data processing machinery is needed, and that is why the most powerful mathematical tools are applied in what follows — this means *statistical analysis* and *linear systems theory.* Linearity property makes it also possible to utilize the "divide and conquer" idea, so that a simple substructure can be analyzed separately and later be included in the overall system. It is assumed that the illusion of intelligence emerges when large numbers of simple operations are combined.

Summarizing and concretizing the previous discussion, now define the *pattern vector f* containing the observation data. The dimension of *f* is such that *all independent observation units* will have an entry of their own; these observation units will be called *variables* henceforth (see Sec. 6.2 for examples). As was assumed, the observation pattern can be decomposed into *features* θ_i, where $1 \leq i \leq N_{\mathrm{LTM}}$. This means that the long term memory capacity of N_{LTM} units is assumed to be filled with feature prototypes that are somehow optimally defined to minimize the reconstruction error. Because of the linearity assumption, f can be expressed as a weighted sum of the features:

$$f = \phi_1(f)\theta_1 + \cdot s + \phi_{N_{\mathrm{LTM}}}(f)\theta_{N_{\mathrm{LTM}}}.$$

Using the terminology from linear algebra, the feature vectors span *subspaces* in the high-dimensional "observation space". The nonlinearity that undoubtedly is needed in the representations results from the *sparsity assumption,* that means, only $\mathrm{N_{STM}}$ of the weighting factors $\phi_i(f)$ can be simultaneously non-zero.

How the feature prototypes θ_i can be optimized is not elaborated on here (see (29) for more discussion and a practical algorithm). It only needs to be noted that the features are *dependency structures* that are reflected in the correlations between the variables. The role of the model is to compress the high-dimensional space of interdependent variables into a set of maximally independent components. A closely related statistical method for detection of the underlying structure in the data is called *factor analysis* (2).

Experiences from artificial neural networks research are not included in this list of related disciplines as a separate item. The reason for this is that their operation can be explained in terms of mathematics — they are just another tool to implement statistical data manipulations. What is more, the connectionist methods are usually used so that only the input-output relation is realized, and no internal structure emerges. In complex modeling tasks finding the underlying structural properties of the domain area is of utmost importance. It can be claimed that in the perceptron networks, for example, the *ontological assumption fails:*

> It is hypothesized that because these networks can implement *any function,* they can be used to model the mental functions as well, without the need of studying the properties of these functions beforehand. However, having no prior restrictions to the class of functions to be searched for, immense amounts of data are needed for training. To make the network adaptation feasible in practical applications, the search is limited only to *smooth* functions.

The function smoothness is an additional assumption that has now sneaked in — and when looking at the operation of the brain, this assumption is hardly justified (sometimes rather abrupt switching between categories seems to take place in practice).

As was seen, the problems of AI and their possible solutions have been discussed in many separate disciplines. An interdisciplinary approach is needed to combine them.

Example: implementing a "mental imagery"

The idea of *mental images* is a powerful concept. Originally, mental imagery was studied exclusively in the context of concrete visual scenes, and different kinds of representations for spatial images were proposed (see (35); also (24)). However, the nature of mental imagery is not agreed upon (53), and parallel "mental views" seem like a good approach to discuss expertise as well — the expert has internalized a sophisticated set of mental images governing the problem area. As presented below, the specialized imagery consisting of the domain-specific features constitutes a "filter" that preprocesses the observation data, creating a compact internal representation of the situation at hand.

The case illustrated below is an extension of the idea of *conceptual spaces* (21).

Pattern matching. To decompose the observation vector into features, or to represent f as a sum of vectors θ_i various options exist. A heuristic algorithm is presented below:

1. Select the prototype vector with index c, where $1 \leq c \leq N_{\mathrm{LTM}}$, that "best explains" the vector f. Usually the vector having the highest (weighted, unnormalized) correlation with the vector f is selected:

$$c = \arg\max_{1 \leq i \leq N_{\mathrm{STM}}} \{|\theta_i^T \; W \;\; f|\}.$$

2. The "loading" of the feature c is now

$$\phi_c = \theta_c^T \; W \;\; f.$$

3. Eliminate the contribution of the feature c by setting

$$f \leftarrow \; f - \frac{\phi_c}{\theta_c^T W \theta_c} \cdot \theta_c.$$

4. If the iteration limit N_{STM} has not yet been reached, go back to Step 1.

The sequence of c indices and the corresponding weights ϕ_c stands for the internal representation of the observation vector f. The associative reconstruction of f is the sum of these features. Running the algorithm once constitutes *one reasoning step,* completing the variables that have not been directly observed. The matrix W is used for *attention control:* it distinguishes between the *explicitly known* and *unknown* variables (examples are given shortly).

In Step 1, a more complicated criterion for selecting the contributing features might be in place. The above option results in "forward-chaining", whereas goal-directed operation can also be obtained, as presented in (29).

Because of the non-orthogonality of the feature vectors, the presented algorithm does not give theoretically correct loadings for the features vectors. Assume that Θ is a *feature matrix* whose columns are the selected features; then the vector $\Phi(f)$ containing the corresponding theoretically correct loadings can be found as

$$\Phi(\; f) = (\Theta^T \; W \; \Theta)^{-1} \cdot \Theta \; W \cdot \;\; f.$$

The associative reconstruction of the vector f can be expressed in the matrix form

$$\overset{a}{f} = \Theta \Phi(\; f),$$

as derived in (29).

Declarative knowledge. The question where the features come from will now be elaborated on. As long as the LTM memory limits have not been reached, the examples or rules can directly be transformed into feature vectors. Assume that the following set of propositions is to be coded:

```
CANARY is a BIRD. BIRD is a CREATURE that FLIES. TWEETY
is a CANARY.
```

The capitalized words are the variables, or the observation units, each spanning a new dimension in the observation space. It need not be known whether the variables represent actual input channels or connections to lower-level subsystems carrying out more elementary categorization tasks. The variables can also be *outputs* to some motoric unit.

Assuming that at least three long-term memory units are available, the following three vectors corresponding to the above rules can be stored in LTM (empty slots denote "no connection", represented by zeros in the mathematical formulation):

$$\begin{array}{ll} \texttt{CANARY} & \rightarrow \\ \texttt{BIRD} & \rightarrow \\ \texttt{CREATURE} & \rightarrow \\ \texttt{FLIES} & \rightarrow \\ \texttt{TWEETY} & \rightarrow \end{array} \left(\begin{array}{c|c|c} 1 & & 1 \\ 1 & 1 & \\ & 1 & \\ & 1 & \\ & & 1 \end{array} \right)$$

Examples and simple explicit rules can be represented as independent features, if no memory constraints exist. As can be seen, the logical structure of the knowledge representations is very constrained: only simple propositions with static, associative dependencies can be modeled. It can be argued that in many cases this suffices — the assumed causality in the rule-form representations is just an illusion. However, the significance of rules is discussed in (66). The presented representation formalism can be interpreted as an "extended" form of propositional logic (or a "restricted" version of first-order predicate logic — for more discussion, see (29)).

Combining quantitative and qualitative information is straightforward in this framework. For example, take the following additional piece of information:

```
...CANARY has SIZE "SMALL".
```

When the linguistic value is fuzzified (by scaling it appropriately, so that the maximum value is 1.0 and the min-

imum is 0.0, for example), the new feature can be included in the knowledge base:

$$\begin{array}{lcl} \texttt{CANARY} & \rightarrow & \left(\begin{array}{c|c} \cdots & 1 \\ & \vdots \\ & 0.1 \end{array}\right) \\ \vdots & & \\ \texttt{SIZE} & \rightarrow & \end{array}$$

Associative reasoning. Reasoning based on the feature representation is implemented as associative reconstruction of incomplete information.

To apply the feature-form knowledge base for reasoning, all evidence is input simultaneously in the vector f. Assume that the only "fact" that is known is TWEETY. This also means that only this variable should be observed in the feature matching process, and the weighting matrix W (the "attention control") is defined accordingly:

$$W = \begin{pmatrix} \varepsilon & & & & \\ & \varepsilon & & & \\ & & \varepsilon & & \\ & & & \varepsilon & \\ & & & & 1 \\ & & & & & \varepsilon \end{pmatrix},$$

where $\varepsilon << 1$. Now, using the presented algorithm, the "spread of activation" proceeds first to produce the internal image of TWEETY − CANARY (working memory size 1 suffices):

$$f = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} \rightarrow \overset{a}{f} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}.$$

It needs to be noted that only very simple rules can be implemented directly in this formalism — for example, there is problem with the connectives (how to distinguish between disjunction and conjunction?). This approach resembles the methodology proposed in (23), and, as presented therein, a more sophisticated reasoning algorithm can be implemented based on this kind of knowledge representation directly. However, even if further reasoning steps would activate new facts, in this framework the selection of the "conflict set" and the control of rule firing is not simpler than in traditional production systems.

The problem is that the novice knowledge representation is not optimized. In technical terms, the question is of *tangled subspaces* spanned by the unpolished feature vectors.

Expert data structures. In (11), it is hypothesized that there are *five stages* in the process of acquiring expertise, starting from traditional AI-like rules and ending with the ability to select correct responses instantaneously. It is proposed there that the expert *know-how* would be based on "some kind of holographic memory" or something else as unimaginable. It can be shown that the data structure that was presented above implements the instantaneous responses in a simple way.

When there is a plenty of data available, most observations will not have individual prototypes in the long-term memory. This scarcity of memory resources is the *key to more sophisticated knowledge representations.* The actual feature optimization process is not studied here (see (29)), but the resulting (slightly streamlined) data structure is shown:

$$\begin{array}{lcl} \texttt{CANARY} & \rightarrow & \left(\begin{array}{c|c} 0.1 & -0.3 \\ 0.1 & 0.4 \end{array}\right) \\ \texttt{EAGLE} & \rightarrow & \\ \vdots & & \\ \texttt{BIRD} & \rightarrow & \left(\begin{array}{c|c} 0.5 & \\ 0.1 & \\ 0.4 & \\ 0.1 & 0.2 \end{array}\right) \\ \texttt{CREATURE} & \rightarrow & \\ \texttt{FLIES} & \rightarrow & \\ \texttt{SIZE} & \rightarrow & \\ \vdots & & \\ \texttt{TWEETY} & \rightarrow & \left(\begin{array}{c|c} 0.1 & \end{array}\right) \\ \vdots & & \end{array}$$

Above, it is assumed that LTM has capacity of only 2 memory elements, even if various examples of birds have been presented. The memory cannot store all of the information, and redundancies have been utilized for optimizing the memory contents. For example, Tweety, being a single, seldom encountered instance, has not deserved a feature of its own. However, Tweety has not vanished altogether: there is still a trace of its existence in the general bird concept (the lengths of the feature vectors have been normalized).

The reasoning is still based on the presented algorithm. Now it turns out that just *one run of the algorithm suffices* to complete a consistent mental view of the observations (assuming that the set of observations can be matched with the feature model).

It needs to be noted that the variable "BIRD" is just an ordinary input signal (perhaps from an auditory or text analysis subsystem). However, this input seems to contribute very much in θ_1, and the whole category may be labeled accordingly. What is important is that there are a plenty of other connections to other signals, so that this computational "bird concept" does not remain empty — it has *meaning* or default connotations in this context. The overall "birdness" of an observation f is given by $\phi_1(f)$, and this signal can be utilized in subsequent classification phases. Similarly, the feature θ_2 could be paraphrased as "being big": the bigger a bird is, the higher is the weight of this feature. Note that as the bird prototype describes an average bird, the features that are used to modify it can have negative as well as positive weights. Very much differing examples (like birds that do not fly) may be better modeled as separate prototypes rather than using cumulative features.

Discussion. It seems that after adaptation one of the features usually dominates over the other ones, defining the "category center" that is modified by the less significant features. Depending on the point of view, the categories may be called *concepts, symbols,* or *clusters.* Correspondingly, regardless of the different manifestations, the features may can be interpreted as *attributes* or *chunks* (9). More traditional views of concept formation are presented in (16). As compared with conceptual clustering (compare to CLUSTER/2 in the previous section), the operation of the cate-

gorization in the above schema is totally unsupervised — this often results in subsymbolic, non-linguistic categories.

When to use features and when to create a new category prototype? At some point quantitative changes must result in qualitative differences in the structure. In the adaptation algorithm (29), the control between these two extremes is based on *self-organization.* Comparing to other approaches that are based on self-organization (see (56) for "semantic maps" and (61) for self-organizing concept acquisition), the approach where features rather than the prototypes only are mapped, allows finer structure of categories to be modeled.

Comparing the proposed feature representation of knowledge with the rule based representation, one key difference is the *graceful degradation* of the reasoning capability due to the fuzzy-like categorization. Comparing with probabilistic reasoning, the idea of representing the high-dimensional probability distribution of the observations using only the variables that are the most significant is actually similar in both cases — however, now the coordinate axes that the observations are projected onto are hidden variables (the optimized, many-faceted features), rather than the original observation variables themselves. The main difference in the operation when compared with these approaches, however, is the *associative, undirected,* and usually *one-step* nature of reasoning based on a pattern matching process. Expertise as a pattern matching process has been proposed, for example, also in (25) with the name *template theory.*

Conclusions

The field of AI is in turmoil, largely due to the impact from connectionism. The general principles of the "third generation" knowledge systems have not yet been found. Also in the above analysis, parallels between analogical phenomena in different disciplines were drawn, even if there not yet exists general agreement. However, there seems to be very much potential.

One of the main benefits in the reviewed methodology is that it makes it possible to cross the gap between symbolic and subsymbolic representations. It is the same data structure for the (qualitative) category prototypes and the (quantitative) features that are used to modify the prototype. The sparse coding of the internal representations facilitates abrupt changes in structure. This possibility to structural changes eliminates the paradox between the two goals: the tailor-made and domain-oriented nature of the representations, and the universality and generality of them.

It needs to be recognized that it is not only the traditional expert system applications that one should be thinking of. There are potential applications of the new AI based modeling tools, for example, in *data mining,* in *information refinement,* and in *modeling of unstructured systems.* In these fields, the actual logical form of the problem is often simple, and the complexity is caused by the high dimensionality and low quality of the available data. In a technical environment where the sensors differ very much from our senses, a human simply cannot figure out how to utilize the measurements optimally. Until now the scarcity of experts has been one of the main problems in knowledge engineering — in the future, *there does not exist a single human expert* that would master the information in the new increasingly complex environments. Hopefully, the AI based modeling tools can take care of the information preprocessing in a clever way.

Whether this is possible or not is crucially dependent of the validity of the ontological assumptions that are made. The questions of the essence of the observation data remain to be solved. It can only be demonstrated that the hypothesis of the data structure that was presented here seems to give interesting results in very differing branches of engineering work (29).

If the natural mental machinery obeys the same principles, or if the ontological assumption is correct and if the human perception and reasoning mechanisms have evolved to observe the environment in an optimal way, then there should be a correspondence between the neural structures and the computational ones after adaptation (when the input data is identical in both systems). In that case, it does not matter what are the underlying mechanisms, and it suffices to study phenomena on the systemic level. Mental phenomena can be studied in the computer — maybe AI in a stronger sense *is* possible, after all.

BIBLIOGRAPHY

1. J.R. Anderson, *The Architecture of Cognition*. Cambridge, Massachusetts: Harvard University Press, 1983.
2. A. Basilevsky, *Statistical Factor Analysis and Related Methods*. New York: John Wiley & Sons, 1994.
3. D.G. Bobrow (ed.), Special volume "Artificial Intelligence in Perspective" of *Artificial Intelligence*, **59**, (1–2), 1993.
4. M.A. Boden (ed.), *The Philosophy of Artificial Intelligence*. Oxford, UK: Oxford University Press, 1990.
5. M.A. Boden (ed.), *Artificial Intelligence*. San Diego, California: Academic Press, 1996.
6. R.A. Brooks, *Cambrian Intelligence*. Cambridge, Massachusetts: MIT Press, 1999.
7. A. Bundy, B. Silver, and D. Plummer, An analytical comparison of some rule-learning programs. *Artificial Intelligence*, **27**(2): 137–181, 1985.
8. C.-L. Chang and R. Lee, *Symbolic Logic and Mechanical Theorem Proving*. New York: Academic Press, 1973.
9. W.G. Chase and K.A. Ericsson, Skill and working memory.InG.H. Bower (ed.), *The Psychology of Learning and Motivation*. New York: Academic Press, 1982.
10. A. Collins andE.E. Smith (eds.), *Readings in Cognitive Science—A Perspective from Psychology and Artificial Intelligence*. San Mateo, California: Morgan Kaufmann Publishers, 1988.
11. H.L. Dreyfus and S.E. Dreyfus, *Mind over Machine: The Power of Human Intuition and Expertise in the Era of the Computer*. Oxford, UK: Blackwell, 1986.
12. H.L. Dreyfus, *What Computers Still Can't Do: A Critique of Artificial Reason*. Cambridge, Massachusetts: MIT Press, 1992.
13. R. Elio and P.B. Scharf, Modeling novice-to-expert shifts in problem-solving strategy and knowledge organization. *Cognitive Science*, **14**: 579–639, 1990.

14. K.A. Ericsson andJ. Smith (eds.), *Towards a General Theory of Expertise*. New York: Cambridge University Press, 1991.
15. E.A. Feigenbaum and H.A. Simon, EPAM-like models of recognition and learning. *Cognitive Science*, **8**: 305–336, 1984.
16. D.H. Fisher, Jr., M.J. Pazzani, andP. Langley (eds.), *Concept Formation: Knowledge and Experience in Unsupervised Learning*. San Mateo, California: Morgan Kaufmann Publishers, 1991.
17. A. Flew (ed.), *A Dictionary of Philosophy*, 2nd ed. London, UK: Picador, 1984.
18. P. Földiák, Forming sparse representations by local anti-Hebbian learning. *Biological Cybernetics*, **64**(2): 165–170, 1990.
19. R. Forsyth (ed.), *Machine Learning—Principles and Techniques*. London, UK: Chapman and Hall, 1989.
20. S. Franklin, *Artificial Minds*. Cambridge, Massachusetts: MIT Press, 1995.
21. P. Gärdenfors, *Conceptual Spaces*. Cambridge, Massachusetts: MIT Press, 2000.
22. M. Ginsberg (ed.), *Readings in Nonmonotonic Reasoning*. Los Altos, California: Morgan Kaufmann Publishers, 1987.
23. C.G. Giraud-Carrier and T.R. Martinez, An integrated framework for learning and reasoning. *Journal of Artificial Intelligence Research*, **3**: 147–185, 1995.
24. J. Glasgow and D. Papadias, Computational imagery. *Cognitive Science*, **16**: 355–394, 1992.
25. F. Gobet, Roles of pattern recognition and search in expert problem solving. *Thinking and Reasoning*, **3**: 291–313, 1997.
26. J.A. Groeger, *Memory and Remembering*. Hong Kong: Longman, 1997.
27. J. Haugeland (ed.), *Artificial Intelligence: The Very Idea*. Cambridge, Massachusetts: MIT Press, 1985.
28. K.J. Holyoak, Symbolic connectionism: toward third-generation the ories of expertise.In K.A. Ericsson andJ. Smith (eds.), *Towards a General Theory of Expertise*. New York: Cambridge University Press, 1991.
29. H. Hyötyniemi, *Mental Imagery: Unified Framework for Associative Representations*.Helsinki University of Technology, Control Engineering Laboratory, Report 111, 1998.Available at http://saato014.hut.fi/hyotyniemi/publications.
30. H. Hyötyniemi, *Neocybernetics in Biological Systems*.Helsinki University of Technology, Control Engineering Laboratory, 2006.
31. S.A. Jackson, *Connectionism and Meaning: From Truth Conditions to Weight Representations*. New Jersey: Ablex Publishing, 1996.
32. C. Jutten and J. Herault, Blind separation of sources, part 1: An adaptive algorithm based on neuromimetic architecture. *Signal Processing*, **24**: 1–10, 1991.
33. P. Kanerva, *Sparse Distributed Memory*. Cambridge, Massachusetts: MIT Press, 1988.
34. R.T. Kellogg, *Cognitive Psychology*. London, UK: SAGE Publications, 1995.
35. S.M. Kosslyn, *Image and Mind*. Cambridge, Massachusetts: Harvard University Press, 1980.
36. R. Kurzweil, *The Age of Intelligent Machines*. Cambridge, Massachusetts: MIT Press, 1990.
37. J.E. Laird, P.S. Rosenbloom, and A. Newell, Chunking in SOAR: the anatomy of a general learning mechanism. *Machine Learning*, **1**: 11–46, 1986.
38. D.B. Lenat, AM: Discovery in mathematics as heuristic search.InR. Davis andD.B. Lenat (eds.), *Knowledge-Based Systems in Artificial Intelligence*. New York: McGraw-Hill, 1980.
39. D.B. Lenat and R.V. Guha, *Building Large Knowledge-Based Systems: Representation and Inference in the CYC Project*. Reading, Massachusetts: Addison-Wesley, 1990.
40. T.-P. Liang andE. Turban (eds.), *Expert Systems with Applications*, **6** (1), 1993.Special issue on case-based reasoning and its applications.
41. J.R. Lucas, Minds, machines, and Gödel. *Philosophy*, **36**, 1961.
42. J.M. McCarthy and P.J. Hayes, Some philosophical problems from the standpoint of artificial intelligence. *Machine Intelligence*, **4**: 463–502, 1969.
43. J.M. McCarthy, Circumscription—a form of non-monotonic reasoning. *Artificial Intelligence*, **13**: 27–39, 1980.
44. J. McDermott, R1: A rule-based configurer of computer systems. *Artificial Intelligence*, **19**(1): 39–88, 1982.
45. R.S. Michalski, Knowledge acquisition through conceptual clustering: A theoretical framework and algorithm for partitioning data into conjunctive concepts. *International Journal of Policy Analysis and Information Systems*, **4**: 219–243, 1980.
46. M. Minsky, A Framework for representing knowledge.In P.H. Winston (ed.), *The Psychology of Computer Vision*. New York: McGraw-Hill, 1975.
47. M. Minsky, *The Society of Mind*. New York: Simon & Schuster, 1985.
48. H. Moravec, *Mind Children: The Future of Robot and Human Intelligence*. Cambridge, Massachusetts: MIT Press, 1988.
49. A. Newell and H.A. Simon, GPS, a program that simulates human thought.In E.A. Feigenbaum andJ. Feldman (eds.), *Computers and Thought*. New York: McGraw-Hill, 1963.
50. B.A. Olshausen and D.J. Field, Sparse coding with an overcomplete basis set: a strategy employed by V1? *Vision Research*, **37**: 3311–3325, 1997.
51. J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. San Mateo, California: Morgan Kaufmann Publishers, 1988.
52. R. Penrose, The emperor's new mind: concerning computers, minds, and the laws of physics. *Behavioral and Brain Sciences*, **13**(4): 643–654.
53. Z. Pylyshyn, The imagery debate: analogue media versus tacit knowledge. *Psychological Review*, **88**: 16–45, 1981.
54. W.V. Quine, *Methods of Logic*, 4th ed. Cambridge, Massachusetts: Harvard University Press, 1982.
55. J. Rissanen, Stochastic complexity and modeling. *Annals of Statistics*, **14**: 1080–1100, 1986.
56. H. Ritter and T. Kohonen, Self-organizing semantic maps. *Biological Cybernetics*, **61**: 241–254, 1989.
57. E. Rosch, Principles of categorization.InE. Rosch andB.B. Lloyd (eds.), *Cognition and Categorization*, Hillsdale, New Jersey: Erlbaum, 1978
58. S. Russell and P. Norvig, *Artificial Intelligence—A Modern Approach*. Englewood Cliffs, New Jersey: Prentice-Hall International, 1995.
59. S.J. Russell, Rationality and intelligence. *Artificial Intelligence*, **94**(1–2): 57–77, 1997.
60. J.R. Searle, Minds, brains and programs. *Behavioral and Brain Sciences*, **3**: 417–424, 1980.
61. P.G. Schyns, A modular neural network model of concept acquisition. *Cognitive Science*, **15**: 461–508, 1991.

62. J. Sterman, *Business Dynamics: Systems Thinking and Modeling for a Complex World*. New York: Irwin/McGraw-Hill, 2000.

63. S.C. Shapiro (ed.), *Encyclopedia of Artificial Intelligence, Vols. 1–2*. New York: John Wiley & Sons, 1987.
64. G. Shafer andJ. Pearl (eds.), *Uncertain Reasoning*. Los Altos, California: Morgan Kaufmann Publishers, 1990.
65. H.A. Simon, *The Sciences of the Artificial*, 3rd ed. Cambridge, Massachusetts: MIT Press, 1996.
66. E.E. Smith, C. Langston, and R.E. Nisbett, The case of rules in reasoning. *Cognitive Science*, **16**: 1–40, 1992.
67. M. Stefik andS.W. Smoliar (eds.), *What Computers Still Can't Do: five reviews and a response. Artificial Intelligence*, **80** (1): 95–191, 1996.
68. N.A. Stillings, S.E. Weisler, C.H. Chase, M.H. Feinstein, J.L. Garfield, andE.L. Rissland (eds.), *Cognitive Science—An Introduction*, 2nd ed., Cambridge, Massachusetts: MIT Press, 1995.

69. D.S. Touretzky and G.E. Hinton, A distributed connectionist production system. *Cognitive Science*, **12**: 423–466, 1988.
70. A. Turing, Computing Machinery and intelligence. *Mind*, **59**: 433–460, 1950.
71. N. Wiener, *Cybernetics: Or Control and Communication in the Animal and the Machine*. New York: John Wiley & Sons, 1948.

72. P. Winston, *Artificial Intelligence*. Reading, Massachusetts: Addison-Wesley, 1984.

HEIKKI HYÖTYNIEMI
Helsinki University of
Technology, Helsinki,
Finland