Lecture 1

Introduction to knowledge-base intelligent systems

- **Intelligent machines, or what machines can do**
- **The history of artificial intelligence or from the "Dark Ages" to knowledge-based systems**
- **Summary**

Intelligent machines, or what machines can do

■ Philosophers have been trying for over 2000 years to understand and resolve two *Big Questions* of the Universe: **How does a human mind work, and Can non-humans have minds?** These questions are still unanswered.

■ *Intelligence* is the ability to understand and learn things. 2 *Intelligence* is the ability to think and understand instead of doing things by instinct or automatically.
(*Essential English Dictionary*, Collins, London, 2008)

- In order to think, some*one* or some*thing* has to have a brain, or an organ that enables some*one* or some*thing* to learn and understand things, to solve problems and to make decisions. So we can define intelligence as *the ability to learn and understand, to solve problems and to make decisions*.

- The goal of *artificial intelligence* (AI) as a science is to make machines do things that would require intelligence if done by humans. Therefore, the answer to the question *Can Machines Think?* was vitally important to the discipline.

- The answer is not a simple "*Yes*" or "*No*".

■ Some people are smarter in some ways than others. Sometimes we make very intelligent decisions but sometimes we also make very silly mistakes. Some of us deal with complex mathematical and engineering problems but are moronic in philosophy and history. Some people are good at making money, while others are better at spending it. As humans, we all have the ability to learn and understand, to solve problems and to make decisions; however, our abilities are not equal and lie in different areas. Therefore, we should expect that if machines can think, some of them might be smarter than others in some ways.

- One of the most significant papers on machine intelligence, **"*Computing Machinery and Intelligence*"**, was written by the British mathematician ***Alan Turing*** over fifty years ago . However, it still stands up well under the test of time, and the Turing's approach remains universal.

- He asked: *Is there thought without experience? Is there mind without communication? Is there language without living? Is there intelligence without life?* All these questions, as you can see, are just variations on the fundamental question of artificial intelligence, *Can machines think?*

- Turing did not provide definitions of machines and thinking, he just avoided semantic arguments by inventing a game, the *Turing Imitation Game*.

- The imitation game originally included two phases. In the first phase, the interrogator, a man and a woman are each placed in separate rooms. The interrogator's objective is to work out who is the man and who is the woman by questioning them. The man should attempt to deceive the interrogator that *he* is the woman, while the woman has to convince the interrogator that *she* is the woman.

Turing Imitation Game: Phase 1
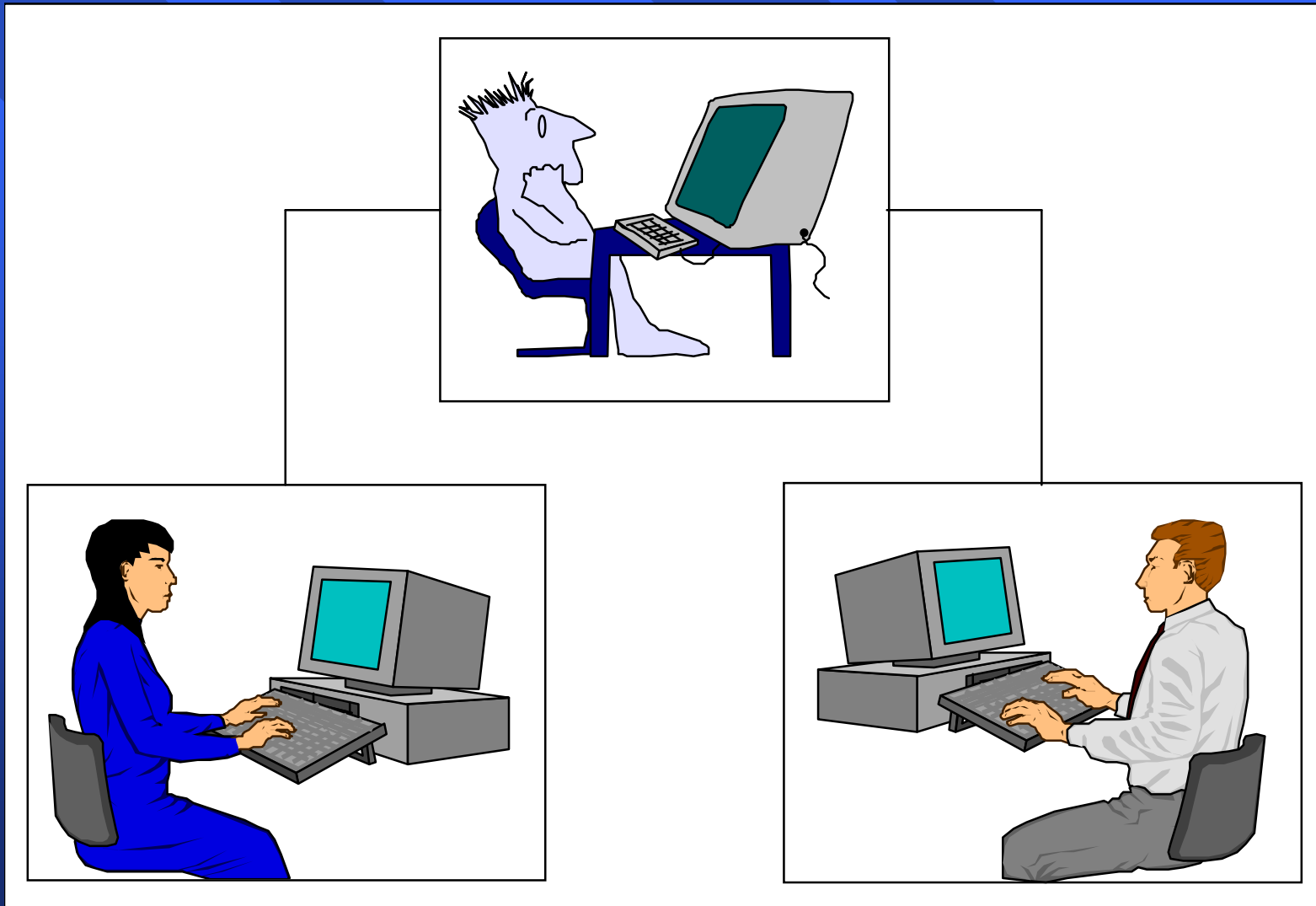
Turing Imitation Game: Phase 2

- In the second phase of the game, the man is replaced by a computer programmed to deceive the interrogator as the man did. It would even be programmed to make mistakes and provide fuzzy answers in the way a human would. If the computer can fool the interrogator as often as the man did, we may say this computer has passed the intelligent behaviour test.

Turing Imitation Game: Phase 2
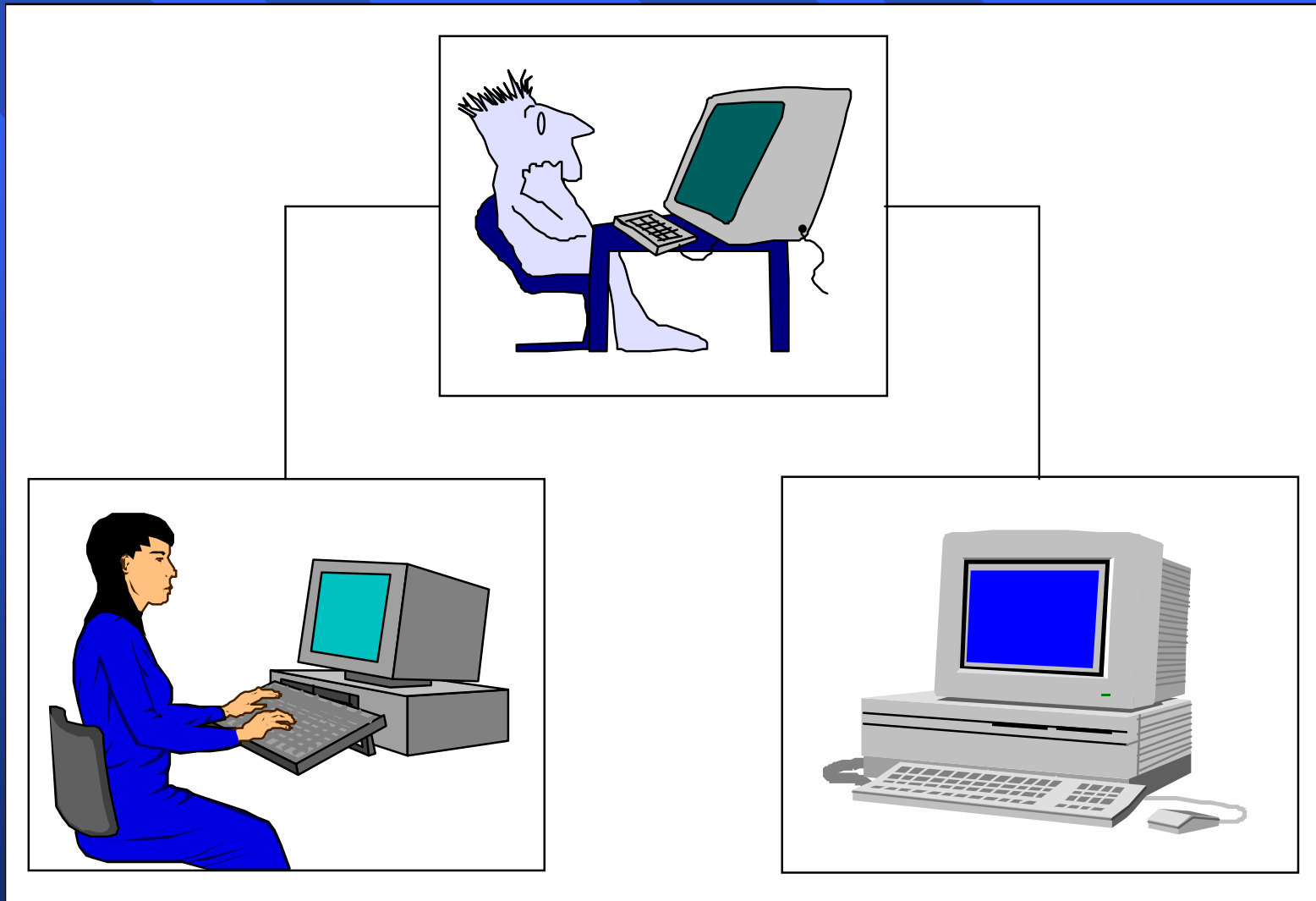
The Turing test has two remarkable qualities that make it really universal.

- By maintaining communication between the human and the machine via terminals, the test gives us an objective standard view on intelligence.

- The test itself is quite independent from the details of the experiment. It can be conducted as a two-phase game, or even as a single-phase game when the interrogator needs to choose between the human and the machine from the beginning of the test.

- Turing believed that by the end of the 20th century it would be possible to program a digital computer to play the imitation game. Although modern computers still cannot pass the Turing test, it provides a basis for the verification and validation of knowledge-based systems.

- **A program thought intelligent in some narrow area of expertise is evaluated by comparing its performance with the performance of a human expert.**

- To build an intelligent computer system, we have to capture, organise and use human expert knowledge in some narrow area of expertise.

The history of artificial intelligence

The birth of artificial intelligence (1943 – 1956)

- The first work recognised in the field of AI was presented by **Warren McCulloch** and **Walter Pitts** in 1943. They proposed a model of an artificial neural network and demonstrated that simple network structures could learn.

- McCulloch, the second "founding father" of AI after Alan Turing, had created the corner stone of neural computing and *Artificial Neural Networks* (ANN).

- The third founder of AI was **John von Neumann**, the brilliant Hungarian-born mathematician. In 1930, he joined the Princeton University, lecturing in mathematical physics. He was an adviser for the Electronic Numerical Integrator and Calculator project at the University of Pennsylvania and helped to design the **Electronic Discrete Variable Calculator**. He was influenced by McCulloch and Pitts's neural network model. When **Marvin Minsky** and **Dean Edmonds**, two graduate students in the Princeton mathematics department, built the first neural network computer in 1951, von Neumann encouraged and supported them.

■ Another of the first generation researchers was **Claude Shannon**. He graduated from MIT and joined Bell Telephone Laboratories in 1941. Shannon shared Alan Turing's ideas on the possibility of machine intelligence. In 1950, he published a paper on chess-playing machines, which pointed out that a typical chess game involved about 10^{120} possible moves (Shannon, 1950). Even if the new von Neumann-type computer could examine one move per microsecond, it would take 3×10^{106} years to make its first move. Thus Shannon demonstrated the need to use heuristics in the search for the solution.

■ In 1956, **John McCarthy**, **Martin Minsky** and **Claude Shannon** organised a summer workshop at Dartmouth College. They brought together researchers interested in the study of machine intelligence, artificial neural nets and automata theory. Although there were just ten researchers, this workshop gave birth to a new science called *artificial intelligence*.

The rise of artificial intelligence, or the era of great expectations (1956 – late 1960s)

■ The early works on neural computing and artificial neural networks started by McCulloch and Pitts was continued. Learning methods were improved and **Frank Rosenblatt** proved the *perceptron convergence theorem*, demonstrating that his learning algorithm could adjust the connection strengths of a perceptron.

- One of the most ambitious projects of the era of great expectations was the **General Problem Solver (GPS)**. **Allen Newell** and **Herbert Simon** from the Carnegie Mellon University developed a general-purpose program to simulate human-solving methods.

- Newell and Simon postulated that a problem to be solved could be defined in terms of *states*. They used the mean-end analysis to determine a difference between the current and desirable or *goal state* of the problem, and to choose and apply *operators* to reach the goal state. The set of operators determined the solution plan.

- However, GPS failed to solve complex problems. The program was based on formal logic and could generate an infinite number of possible operators. The amount of computer time and memory that GPS required to solve real-world problems led to the project being abandoned.

- In the sixties, AI researchers attempted to simulate the thinking process by inventing *general methods* for solving *broad classes of problems*. They used the general-purpose search mechanism to find a solution to the problem. Such approaches, now referred to as *weak methods*, applied weak information about the problem domain.

■ By 1970, the euphoria about AI was gone, and most government funding for AI projects was cancelled. AI was still a relatively new field, academic in nature, with few practical applications apart from playing games. So, to the outsider, the achieved results would be seen as toys, as no AI system at that time could manage real-world problems.

Unfulfilled promises, or the impact of reality (late 1960s – early 1970s)

The main difficulties for AI in the late 1960s were:

■ Because AI researchers were developing general methods for broad classes of problems, early programs contained little or even no knowledge about a problem domain. To solve problems, programs applied a search strategy by trying out different combinations of small steps, until the right one was found. This approach was quite feasible for simple **toy problems**, so it seemed reasonable that, if the programs could be "scaled up" to solve large problems, they would finally succeed.

- Many of the problems that AI attempted to solve were **too broad and too difficult**. A typical task for early AI was machine translation. For example, the National Research Council, USA, funded the translation of Russian scientific papers after the launch of the first artificial satellite (Sputnik) in 1957. Initially, the project team tried simply replacing Russian words with English, using an electronic dictionary. However, it was soon found that translation requires a general understanding of the subject to choose the correct words. This task was too difficult. In 1966, all translation projects funded by the US government were cancelled.

- In 1971, the British government also suspended support for AI research. Sir **James Lighthill** had been commissioned by the Science Research Council of Great Britain to review the current state of AI. He did not find any major or even significant results from AI research, and therefore saw no need to have a separate science called "artificial intelligence".

The technology of expert systems, or the key to success (early 1970s – mid-1980s)

- Probably the most important development in the seventies was the realisation that the domain for intelligent machines had to be sufficiently restricted. Previously, AI researchers had believed that clever search algorithms and reasoning techniques could be invented to emulate general, human-like, problem-solving methods. A general-purpose search mechanism could rely on elementary reasoning steps to find complete solutions and could use weak knowledge about domain.

- When weak methods failed, researchers finally realised that the only way to deliver practical results was to solve typical cases in narrow areas of expertise, making large reasoning steps.

DENDRAL

- DENDRAL was developed at Stanford University to determine the molecular structure of Martian soil, based on the mass spectral data provided by a mass spectrometer. The project was supported by NASA. Edward Feigenbaum, Bruce Buchanan (a computer scientist) and Joshua Lederberg (a Nobel prize winner in genetics) formed a team.

- There was no scientific algorithm for mapping the mass spectrum into its molecular structure. Feigenbaum's job was to incorporate the expertise of Lederberg into a computer program to make it perform at a human expert level. Such programs were later called *expert systems*.

- DENDRAL marked a major "paradigm shift" in AI: a shift from general-purpose, knowledge-sparse weak methods to domain-specific, knowledge-intensive techniques.

- The aim of the project was to develop a computer program to attain the level of performance of an experienced human chemist. Using heuristics in the form of high-quality specific rules, rules-of-thumb , the DENDRAL team proved that computers could equal an expert in narrow, well defined, problem areas.

- The DENDRAL project originated the fundamental idea of expert systems – *knowledge engineering*, which encompassed techniques of capturing, analysing and expressing in rules an expert's "know-how".

MYCIN

- MYCIN was a rule-based expert system for the diagnosis of infectious blood diseases. It also provided a doctor with therapeutic advice in a convenient, user-friendly manner.

- MYCIN's knowledge consisted of about 450 rules derived from human knowledge in a narrow domain through extensive interviewing of experts.

- The knowledge incorporated in the form of rules was clearly separated from the reasoning mechanism. The system developer could easily manipulate knowledge in the system by inserting or deleting some rules. For example, a domain-independent version of MYCIN called EMYCIN (Empty MYCIN) was later produced.

PROSPECTOR

■ PROSPECTOR was an expert system for mineral exploration developed by the Stanford Research Institute. Nine experts contributed their knowledge and expertise. PROSPECTOR used a combined structure that incorporated rules and a semantic network. PROSPECTOR had over 1000 rules.

■ The user, an exploration geologist, was asked to input the characteristics of a suspected deposit: the geological setting, structures, kinds of rocks and minerals. PROSPECTOR compared these characteristics with models of ore deposits and made an assessment of the suspected mineral deposit. It could also explain the steps it used to reach the conclusion.

■ A 1986 survey reported a remarkable number of successful expert system applications in different areas: chemistry, electronics, engineering, geology, management, medicine, process control and military science (Waterman, 1986). Although Waterman found nearly 200 expert systems, most of the applications were in the field of medical diagnosis. Seven years later a similar survey reported over 2500 developed expert systems (Durkin, 1994). The new growing area was business and manufacturing, which accounted for about 60% of the applications. Expert system technology had clearly matured.

However:

■ Expert systems are restricted to a very narrow domain of expertise. For example, MYCIN, which was developed for the diagnosis of infectious blood diseases, lacks any real knowledge of human physiology. If a patient has more than one disease, we cannot rely on MYCIN. In fact, therapy prescribed for the blood disease might even be harmful because of the other disease.

■ Expert systems can show the sequence of the rules they applied to reach a solution, but cannot relate accumulated, heuristic knowledge to any deeper understanding of the problem domain.

- Expert systems have difficulty in recognising domain boundaries. When given a task different from the typical problems, an expert system might attempt to solve it and fail in rather unpredictable ways.

- Heuristic rules represent knowledge in abstract form and lack even basic understanding of the domain area. It makes the task of identifying incorrect, incomplete or inconsistent knowledge difficult.

- Expert systems, especially the first generation, have little or no ability to learn from their experience. Expert systems are built individually and cannot be developed fast. Complex systems can take over 30 person-years to build.

How to make a machine learn, or the rebirth of neural networks (mid-1980s – onwards)

■ In the mid-eighties, researchers, engineers and experts found that building an expert system required much more than just buying a reasoning system or expert system shell and putting enough rules in it. Disillusions about the applicability of expert system technology even led to people predicting an *AI "winter"* with severely squeezed funding for AI projects. AI researchers decided to have a new look at neural networks.

- By the late sixties, most of the basic ideas and concepts necessary for neural computing had already been formulated. However, only in the mid-eighties did the solution emerge. The major reason for the delay was technological: there were no PCs or powerful workstations to model and experiment with artificial neural networks.

- In the eighties, because of the need for brain-like information processing, as well as the advances in computer technology and progress in neuroscience, the field of neural networks experienced a dramatic resurgence. Major contributions to both theory and design were made on several fronts.

- Grossberg established a new principle of self-organisation (*adaptive resonance theory*), which provided the basis for a new class of neural networks (Grossberg, 1980).

- Hopfield introduced neural networks with feedback – *Hopfield networks*, which attracted much attention in the eighties (Hopfield, 1982).

- Kohonen published a paper on *self-organising maps* (Kohonen, 1982).

- Barto, Sutton and Anderson published their work on *reinforcement learning* and its application in control (Barto et al., 1983).

- But the real breakthrough came in 1986 when the *back-propagation learning algorithm*, first introduced by Bryson and Ho in 1969 (Bryson & Ho, 1969), was reinvented by Rumelhart and McClelland in *Parallel Distributed Processing* (1986).

- Artificial neural networks have come a long way from the early models of McCulloch and Pitts to an interdisciplinary subject with roots in neuroscience, psychology, mathematics and engineering, and will continue to develop in both theory and practical applications.

Evolutionary computation, or learning by doing (early 1970s – onwards)

- Natural intelligence is a product of evolution. Therefore, by simulating biological evolution, we might expect to discover how living systems are propelled towards high-level intelligence.

- Nature learns by doing; biological systems are not told how to adapt to a specific environment – they simply compete for survival.

- The evolutionary approach to AI is based on the computational models of natural selection and genetics.

- Evolutionary computation works by simulating a population of individuals, evaluating their performance, generating a new population, and repeating this process a number of times.

- Evolutionary computation combines three main techniques: **genetic algorithms**, **evolutionary strategies** and **genetic programming**.

- The concept of **genetic algorithms** was introduced by John Holland in the early 1970s. He developed an algorithm for manipulating artificial 'chromosomes' (strings of binary digits), using such genetic operations as selection, crossover and mutation. Genetic algorithms are based on a solid theoretical foundation of the Schema Theorem.

- In the early 1960s, Ingo Rechenberg and Hans-Paul Schwefel, students of the Technical University of Berlin, proposed a new optimisation method called **evolutionary strategies**. They suggested using random changes in the parameters, as happens in natural mutation.

- **Genetic programming** represents an application of the genetic model of learning to programming. Genetic programming generates computer programs as the solution.

- The interest in genetic programming was greatly stimulated by John Koza in the 1990s. He used genetic operations to manipulate symbolic code representing LISP programs.

- Genetic algorithms, evolutionary strategies and genetic programming represent rapidly growing areas of AI, and have great potential.

The new era of knowledge engineering, or computing with words (late 1980s – onwards)

■ Neural network technology offers more natural interaction with the real world than do systems based on symbolic reasoning. Neural networks can learn, adapt to changes in a problem's environment, establish patterns in situations where rules are not known, and deal with fuzzy or incomplete information.

■ However, they lack explanation facilities and usually act as a black box. The process of training neural networks is slow, and frequent retraining can cause serious difficulties.

- Classic expert systems are especially good for closed-system applications with precise inputs and logical outputs. They use expert knowledge in the form of rules and, if required, can interact with the user to establish a particular fact.

- A major drawback is that human experts cannot always express their knowledge in terms of rules or explain the line of their reasoning. This can prevent the expert system from accumulating the necessary knowledge, and consequently lead to its failure.

- Very important technology dealing with vague, imprecise and uncertain knowledge and data is **fuzzy logic**.

- Human experts do not usually think in probability values, but in such terms as *often*, *generally*, *sometimes*, *occasionally* and *rarely*. Fuzzy logic is concerned with capturing the meaning of words, human reasoning and decision making. Fuzzy logic provides the way to break through the computational bottlenecks of traditional expert systems.

- At the heart of fuzzy logic lies the concept of a **linguistic variable**. The values of the linguistic variable are words rather than numbers.

- Fuzzy logic or *fuzzy set theory* was introduced by Professor **Lotfi Zadeh**, Berkeley's electrical engineering department chairman, in 1965. It provided a means of computing with words. However, acceptance of fuzzy set theory by the technical community was slow and difficult. Part of the problem was the provocative name – "fuzzy" – it seemed too light-hearted to be taken seriously. Eventually, fuzzy theory, ignored in the West, was taken seriously in the East – by the Japanese. It has been used successfully since 1987 in Japanese-designed dishwashers, washing machines, air conditioners, television sets, copiers, and even cars.

Benefits derived from the application of fuzzy logic models in knowledge-based and decision-support systems can be summarised as follows:

- **Improved computational power:** Fuzzy rule-based systems perform faster than conventional expert systems and require fewer rules. A fuzzy expert system merges the rules, making them more powerful. Lotfi Zadeh believes that in a few years most expert systems will use fuzzy logic to solve highly nonlinear and computationally difficult problems.

■ **Improved cognitive modelling:** Fuzzy systems allow the encoding of knowledge in a form that reflects the way experts think about a complex problem. They usually think in such imprecise terms as *high* and *low*, *fast* and *slow*, *heavy* and *light*. In order to build conventional rules, we need to define the crisp boundaries for these terms by breaking down the expertise into fragments. This fragmentation leads to the poor performance of conventional expert systems when they deal with complex problems. In contrast, fuzzy expert systems model imprecise information, capturing expertise similar to the way it is represented in the expert mind, and thus improve cognitive modelling of the problem.

■ **The ability to represent multiple experts:**
Conventional expert systems are built for a narrow domain. It makes the system's performance fully dependent on the right choice of experts. When a more complex expert system is being built or when expertise is not well defined, *multiple experts* might be needed. However, multiple experts seldom reach close agreements; there are often differences in opinions and even conflicts. This is especially true in areas, such as business and management, where no simple solution exists and conflicting views should be taken into account. Fuzzy expert systems can help to represent the expertise of multiple experts when they have opposing views.

- Although fuzzy systems allow expression of expert knowledge in a more natural way, they still depend on the rules extracted from the experts, and thus might be smart or dumb. Some experts can provide very clever fuzzy rules – but some just guess and may even get them wrong. Therefore, all rules must be tested and tuned, which can be a prolonged and tedious process. For example, it took Hitachi engineers several years to test and tune only 54 fuzzy rules to guide the Sendal Subway System.

■ In recent years, several methods based on neural network technology have been used to search numerical data for fuzzy rules. Adaptive or neural fuzzy systems can find new fuzzy rules, or change and tune existing ones based on the data provided. In other words, data in – rules out, or experience in – common sense out.

Summary

- Expert, neural and fuzzy systems have now matured and been applied to a broad range of different problems, mainly in engineering, medicine, finance, business and management.

- Each technology handles the uncertainty and ambiguity of human knowledge differently, and each technology has found its place in knowledge engineering. They no longer compete; rather they complement each other.

■ A synergy of expert systems with fuzzy logic and neural computing improves adaptability, robustness, fault-tolerance and speed of knowledge-based systems. Besides, computing with words makes them more "human". It is now common practice to build intelligent systems using existing theories rather than to propose new ones, and to apply these systems to real-world problems rather than to "toy" problems.

Main events in the history of AI

Period	Key Events
The birth of Artificial Intelligence (1943–1956)	McCulloch and Pitts, *A Logical Calculus of the Ideas Immanent in Nervous Activity*, 1943 Turing, *Computing Machinery and Intelligence*, 1950 The Electronic Numerical Integrator and Calculator project (von Neumann) Shannon, *Programming a Computer for Playing Chess*, 1950 The Dartmouth College summer workshop on machine intelligence, artificial neural nets and automata theory, 1956

Period	Key Events
The rise of artificial intelligence (1956–late 1960s)	*LISP* (McCarthy) The General Problem Solver (GPR) project (Newell and Simon) Newell and Simon, *Human Problem Solving*, 1972 Minsky, *A Framework for Representing Knowledge*, 1975
The disillusionment in artificial intelligence (late 1960s–early 1970s)	Cook, *The Complexity of Theorem Proving Procedures*, 1971 Karp, *Reducibility Among Combinatorial Problems*, 1972 The Lighthill Report, 1971

Period	Key Events
The discovery of expert systems (early 1970s–mid-1980s)	DENDRAL (Feigenbaum, Buchanan and Lederberg, Stanford University)
	MYCIN (Feigenbaum and Shortliffe, Stanford University)
	PROSPECTOR (Stanford Research Institute)
	PROLOG - *a logic programming language* (Colmerauer, Roussel and Kowalski, France)
	EMYCIN (Stanford University)
	Waterman, *A Guide to Expert Systems*, 1986

Period	Key Events
The rebirth of artificial neural networks (mid-1980s onwards)	Hopfield, *Neural Networks and Physical Systems with Emergent Collective Computational Abilities*, 1982 Kohonen, *Self-Organized Formation of Topologically Correct Feature Maps*, 1982 Rumelhart and McClelland, *Parallel Distributed Processing*, 1986 The First IEEE International Conference on Neural Networks, 1987 Haykin, *Neural Networks*, 1994 Neural Network, MATLAB Application Toolbox (The MathWork, Inc.)

Period	Key Events
Evolutionary computation (early 1970s onwards)	Rechenberg, *Evolutionsstrategien - Optimierung Technischer Systeme Nach Prinzipien der Biologischen Information*, 1973
	Holland, *Adaptation in Natural and Artificial Systems*, 1975.
	Koza, *Genetic Programming: On the Programming of the Computers by Means of Natural Selection*, 1992.
	Schwefel, *Evolution and Optimum Seeking*, 1995
	Fogel, *Evolutionary Computation –Towards a New Philosophy of Machine Intelligence*, 1995.

Period	Key Events
Computing with Words (late 1980s onwards)	Zadeh, *Fuzzy Sets*, 1965
	Zadeh, *Fuzzy Algorithms*, 1969
	Mamdani, *Application of Fuzzy Logic to Approximate Reasoning Using Linguistic Synthesis*, 1977
	Sugeno, *Fuzzy Theory*, 1983
	Japanese "fuzzy" consumer products (dishwashers, washing machines, air conditioners, television sets, copiers)
	Sendai Subway System (Hitachi, Japan), 1986
	The First IEEE International Conference on Fuzzy Systems, 1992
	Kosko, *Fuzzy Thinking*, 1993
	Cox, *The Fuzzy Systems Handbook*, 1994
	Zadeh, *Computing with Words - A Paradigm Shift*, 1996
	Fuzzy Logic, MATLAB Toolbox (The MathWork)
	Neural Networks, MATLAB Toolbox (The MathWorks)
	Berkeley Initiative in Soft Computing (BISC) http://www-bisc.cs.berkeley.edu