Agents' Capabilities

Dr S Srinivasan*

Abstract

The objective of this paper is to provide some of important capabilities possessed by the Agents. Twelve architectures have been used for this preliminary analysis representing a wide range of current architectures in artificial intelligence (AI). The aim of the paper is to understand the various capabilities that agents should possess generally. Also because of the design of the architecture of the agents that we have taken in to consideration, the capabilities also vary from one agent to another.

Keywords : Subsumption, Theo, Atlantis, Prodigy, ICARUS, Adaptive Intelligent System, Meta Reasoning Architecture, Homer, SOAR, RALPH-MEA, TETON, Entropy Reduction System.

1. Introduction

A complete functioning agent, whether biological, or simulated in software, or implemented in the form of a robot, needs an integrated collection of diverse but interrelated capabilities. At present, most work in AI and Cognitive Science addresses only components of such an architecture (e.g. vision, speech understanding, concept formation, rule learning, planning, motor control, etc.) or mechanisms and forms of representation and inference (logic engines, condition-action rules, neural nets, genetic algorithms) which might be used by many components. While such studies can make useful contributions it is important to ask, from time to time, how everything can be put together, and that requires the study of architectures. [6] Besides differences in levels of abstraction or implementation, there are differences in types of functionality. A human-like agent needs to be able to perform a large and diverse collection of tasks, both externally (finding and consuming food, avoiding predators, building shelters, making tools, etc.) and internally (interpreting sensory data, generating motives, evaluating motives, selecting motives, creating plans, storing information for future use, making inferences from new or old information, detecting inconsistencies, monitoring plan execution, monitoring various kinds of internal processing, noticing resemblances, creating new concepts and theories, discovering new rules, noticing new possibilities, etc.).

*Department of Computer Applications, PDM College of Engineering, Bahadurgarh, M D University, Haryana, India

2.1 In this paper we list out majority of the capabilities and finally in the form of a table we indicate which agent among the twelve agents we consider possess what kind of capabilities.

The twelve agents we take into consideration , are Subsumption Architecture, ATLANTIS, Theo, Prodigy, ICARUS, Adaptive Intelligent Systems (AIS), A Meta-reasoning Architecture for 'X' (MAX), Homer, Soar, Teton, RALPH-MEA, Entropy Reduction Engine (ERE). [1] and their features are shown in the following table – 1.

TABLE - 1

Sl No	Architecture	Properties
1	Subsumption	Complicated Intelligent behavior Into Simple behavior modules Organized into layers
2	THEO	Plan-Then-Complie By this Integrating learning , planning and knowledge representation
3	ICARUS	Specific representation of long term memory .It uses 3 independent asynchronous modules responsible for 1.Perception 2.Planning 3.Effecting
4	PRODIGY	Storing the knowledge in a form of first order predicate logic (FOPL) called Prodigy Description language (PDL) . Has a modular architecture that stores the knowledge symbolically.
5	ATLANTIS	Integrating planning and reacting in a heterogeneous asynchronous architecture for mobile agents . It consists of 3 layers: 1.Control layer 2.sequencing layer 3.deliberative layer
6	Adaptive Intelligent System (AIS)	To reason about and interact with other dynamic entities in real time . --- problem solving techniques --- when encountering un-expected situation , decides whether to and how to respond . --- focus attention on the most critical aspects of current situation . --- operating continuously without rebooting . --- able to coordinate with external agent . (more or less similar to human being)
7	Meta Reasoning (MAX)	Many ideas in MAX may traced to Prodigy. --- rule-based forward – chaining engine that operates on productions . --- is designed to support to modular agents. --- they are used to respond to a dynamic environment in a timely manner . --- modules are categorized in to Behavior and monitor . --- Some of the modules are: 1.attention focusing 2.multiple problem solving strategies 3.execution monitoring 4.goal-directed exploration 5.explantion-based learning 6.process interruption 7.intelligent resumption
8	HOMER	--- Is not designed for general intelligence . --- underlying philosophy is to synthesize several key areas of AI to form one complete system . (like planning, learning , natural language understanding , robotic navigation) . HOMER answers questions posed by users and carries out instructions given by users . --- is a modular structure. It consistes of : 1.memory 2.planner 3.natural language interpreter and generator . 4.reflective processes 5.plan executer
9	SOAR	--- origally known as STATE OPERATOR AND RESULT . --- mail goal is that fuill range of capabilities to be handled by an intelligent agent from highly routines to extremely difficult open-problems --- the underlying SOAR architecture is the view that symbolic system is necessary and sufficient condition for general intelligence . This is known as Physical Symbolic system Hypothesis (PSSH) --- ultimate aim is to get general intellgenct agent --- is based on a production system ie. It uses explicit production rules to govern its behaviors .
10	Teton	--- is a problem solver --- uses two memory areas 1.Short-Term memory 2. Long-Term memory --- like human beings , interruption are allowed . --- it has a feature called Execution Cycle which always look for what to do next .
11	RALPH-MEA	--- is a multiple execution architecture --- like human being , selecting best one from the environment --- RALPH – MEA uses Execution Architecture (EA) to select from one state to best one . --- it uses the following : 1.Condition action 2.Action utility 3.Goal – based 4.Decision Theoretic
12	Entropy Reduction Engine (ERE)	--- focuses on problems that require planning , scheduling and control --- uses many different problem solving methods such as : 1.problem reduction 2.temporal projection 3.rule-based execution

2.2 Learning Capabilities

We discuss here about various learning capabilities. Also we briefly point out against each such learning capability. [4]

2.2.1 Single Learning Method

A system is said to learn if it is capable of acquiring new knowledge from its environment. Learning may also enable the ability to perform new tasks without having to be redesigned or reprogrammed,especially when accompanied by generalization. Learning is most often accomplished in a system that supports symbolic abstraction. This type of learning is separated from the acquisition of knowledge through direct programming by the designer.

2.2.2 Multi-Method Learning

As a capability, learning is often thought of as one of the necessary conditions for intelligence in an agent. Some systems extend this requirement by including a plethora of mechanisms for learning in order to obtain as much as possible from the

system, or to allow various components of their system to learn in their own ways (depending on the modularity, representation, etc., of each). Additionally, multiple methods are included in a system in order to gauge the performance of one method against that of another.

2.2.3 Caching
Caching can be seen as rote learning, but can also be seen as a form of explanation-based learning. This is simply storing a computed value to avoid having to compute it in the future. Caching vastly reduces the high cost of relying on meta-knowledge and the necessary retrieval and application.

2.2.4 Learning by Instruction
An agent that is given information about the environment, domain knowledge, or how to accomplish a particular task on-line (that is in real-time as opposed to some off-line programming) is said to be able to learn from instruction. Some instruction is completely uni-directional: a teacher simply gives the agent the knowledge in a sequential series of instructions. Other learning is interactive: the teacher is prepared to instruct the agent when the agent lacks knowledge and requests it. This last method supports experiential learning in which a teacher may act as both a guide (when called upon) and as an authority (when the agent is placing itself in danger or making a critical mistake).

2.2.5 Learning from Experimentation
Learning from experimentation, also called discovery, involves the use of domain knowledge, along with observations made about the environment, to extend and refine an agent's domain knowledge. The more systematic an agent manipulates its environment to determine new information, the more its behavior seems to follow traditional scientific experimental paradigms. However, the agent's action need not be so planned to produce new behavior.

2.2.6 Learning by Analogy
Reasoning by analogy generally involves abstracting details from a a particular set of problems and resolving structural similarities between previously distinct problems. Analogical reasoning refers to this process of recognition and then applying the solution from the known problem to the new problem. Such a technique is often identified as case-based reasoning. Analogical learning generally involves developing a set of mappings between features of two instances. Paul Thagard and Keith Holyoak have developed a computational theory of analogical reasoning that is consistent with the outline above, provided that abstraction rules are provided to the model.

2.2.7 Inductive Learning and Concept Acquisition
In contrast to abstraction, concept acquisition refers to the ability of an agent to identify the discriminating properties of objects in the world, to generate labels for the objects and to use the labels in the condition list of operators, thereby associating operations with the concept. Concept acquisition normally proceeds from a set of positive and negative instances of some concept (or group of segregated concepts). With the presentation of the instances, the underlying algorithm makes correlations between the feature of the instances and their classification. The problem with this technique as it is described here is that it requires the specification of both relevant features and the possible concepts. In general, as an inductive technique, concept acquisition should be able to generate new concepts spontaneously and to recognize the relevant features over the entire input domain.

2.2.8 Abstraction
Contrasted with concept acquisition, abstraction is the ability to detect the relevant -- or critical -- information and action for a particular problem. Abstraction is often used in planning and problem solving in order to form a condition list for operators that lead from one complex state to another based on the criticality of the precondition.

For instance, in an office environment, a robot with a master key can effectively ignore doors if it knows how to open doors in general. Thus, the problem of considering doors in a larger plan may be abstracted from the problem solving. This can be performed by the agent repeatedly to obtain the most general result. Some architectures limit abstraction to avoid the problem of over-generalization, resulting in mistaken applications of the erroneously abstracted operator.

2.2.9 Explanation-Based Learning
When an agent can utilize a worked example of a problem as a problem-solving method, the agent is said to have the capability of explanation-based learning (EBL). This is a type of analytic learning. The advantage of explanation-based learning is that, as a deductive mechanism, it requires only a single training example (inductive learning methods often require many training examples). However, to utilize just a single example most EBL algorithms require all of the following:

- The training example
- A Goal Concept
- An Operationality Criteria
- A Domain Theory

From the training example, the EBL algorithm computes a generalization of the example that is consistent with the goal concept and that meets the operationality criteria (a description of the appropriate form of the final concept). One criticism of EBL is that the required domain theory needs to be complete and consistent. Additionally, the utility of learned information is an issue when learning proceeds indiscriminately. Other forms of learning that are based on EBL are knowledge compilation, caching and macro-ops.

2.2.10 Transfer of Learning
A capability that comes from generalization and is related to learning by analogy. Learned information can be applied to other problem instances and possibly even other instances. Three specific types of learning transfer are normally identified:

- **Within-Trial :** Learning applies immediately to the current situation.
- **Within-Task :** Learning is general enough that it may apply to different problem instances in the same domain.
- **Across-Task :** Learning applies to different domains. Examples here include some types of concept acquisition in which a concept learned in one domain (e.g., blocks) can be related to other concepts (e.g., bricks) through similarities (e.g., stackable). Across-task learning is then strongly analogical.

Now we discuss about various other impotant capabilities that an is expected to possess.

2.3 Planning

Planning is arguably one of the most important capabilities for an intelligent agent to possess. In almost all cases, the tasks which these agents must carry out are expressed as goals to be achieved; the agent must then develop a series of actions designed to achieve this goal.

The ability to plan is closely linked to the agent's representation of the world. It seems that effective planning requires that 1) knowledge of the world is available to the planner, and since most worlds in which we are interested are reasonably complex, this is a strong motivation for implementing 2) a symbolic representation of knowledge. Typically, this knowledge contains information about possible actions in the world, which is then used by the planner in constructing a sequence of actions.

Planning itself is a prerequisite for several other capabilities that are often instantiated in intelligent agents. Certainly, problem solving relies heavily on planning, as most approaches to problem solving consist of incremental movements toward a solution; planning is integral to assembling these steps. Learning and planning have a reciprocal relationship wherein planning creates a new method for carrying out a task, which can then be learned for future use by the planner. [5]

2.3.1 Problem Solving

It may seem that all agents must solve problems, as indeed they must, but problem solving in the technical sense is the ability to consider and attain goals in particular domains using domain-independent techniques (such as the weak methods) as well as domain knowledge. Problem Solving includes the capability to acquire and reason about knowledge, although the level to which such capability is supported differs between architectures. Problem solving, especially human problem solving, has been characterized as deliberate movement through a problem space. A problem space defines the states that are possible for a particular problem instance and the operators available to transform one state to another. In this formulation, problem solving is search through the state space by applying operators until a recognizable goal state is reached. [2]

2.3.2 Replanning

Intelligent agents operating in dynamic environments often find it necessary to modify or completely rebuild plans in response to changes in their environment. There are several situations in which an agent should re-plan.

An intelligent agent should update its plan when it learns new information which helps it accomplish its current goal more quickly. For instance, it may be the case that in the process of satisfying one goal the agent also satisfies one or more of its other goals. The agent should recognize when it has already satisfied a goal and change its plan accordingly.

In addition, an agent should replan when facts about the world upon which its current plan are based change. This is important when in the act of achieving one goal the agent undoes another. The agent must realize this and update its plan to satisfy both goals.

Re-planning is a capability that arises from other capabilities, namely planning and interruptability.

2.3.3 Support of Multiple, Simultaneous Goals

Taskable agents can support the achievement of externally specified top-level goals. Some of these agents can support the achievement of many top-level goals at once. This is usually performed in conjunction with planning such that the goals are sequenced in some rational way.

2.4 Self Reflection

Systems which are capable of self reflection are able to examine their own internal processing mechanisms. They can use this capability to explain their behavior, and modify their processing methods to improve performance. Such systems must have some form of Meta-Knowledge available, and in addition, they must actively apply the Meta-Knowledge to some task. The list below explains the common uses of self reflection.

2.4.1 Performance Fine Tuning

Performance can be fine tuned by gathering statistics on the efficiency of various problem solving methods. These statistics are then examined to determine which problem solving methods are most efficient for certain classes of problems. This is closely related to the learning capability described above.

2.4.2 Explanation

Systems can use self reflection to explain their behavior to an outside observer. This action is often performed by examining traces of the problem solution and reporting key aspects of it.

2.4.3 Episodic Recall

Self Reflection can take the form of reporting past experiences to an outside observer. This is usually accomplished through some form of episodic memory, where experiences are stamped with indications of when they occurred.

There are several different mechanisms that can be included in an architecture to help facilitate self reflection. These are explained below.

"Glass Box" Knowledge Representation

If knowledge if uniformly represented and completely open to examination throughout the system then it is easier to add functionality which can examine this knowledge. Only one form of knowledge needs to be examined, and the knowledge itself is easily obtained. The other common approach to knowledge representation is called the "black box" approach, where knowledge is localized and hidden within the various modules of the system. This makes it difficult to extract the knowledge to be reflected upon, and may require the use of several different methods of looking at the knowledge once it is obtained.

2.4.4 Episodic Memory

Episodic memory is directly applicable to episodic recall. This type of memory is often costly, however, both in terms of space and time. As the agent's experiences grow the size of the memory space to store these experiences must grow as well. In addition, searching through past experiences for some specific detail is often too time consuming to be practical.

2.4.5 Problem Solving Traces
Problem solving traces are used by many of the learning mechanisms. In addition they can be used to explain the behavior of the system. Problem solving traces are usually only kept around for a short period of time, and are often tied to the specific learning or explanation functions that use them. Once the system learns from them (or proceeds to another task) the trace is discarded. Key aspects of it may be saved in an episodic memory if the system has one.

2.5 Meta Reasoning

Reasoning about reasoning, or meta-reasoning is a critical capability for agents attempting to display general intelligence. Generally intelligent agents must be capable of constantly improving skills, adapting to changes in the world, and learning new information. Meta-reasoning can be deployed implicitly through mechanisms such as domain-independent learning, or explicitly using, for example, declarative knowledge which the agent can interpret and manipulate. The domain-independent approaches seem the most successful so far.

Other aspects of meta-reasoning include the consideration of computational costs of processing, leading to the issues such as focused processing and real-time performance.

2.5.1 Inductive and Deductive Reasoning
Deductive reasoning can be described as reasoning of the form if A then B. Deduction is in some sense the direct application of knowledge in the production of new knowledge. However, this new knowledge does not represent any new semantic information: the rule represents the knowledge as completely as the added knowledge since any time the assertions (A) are true then the conclusion B is true as well. Purely deductive learning includes methods such as caching, building macro-operators, and explanation-based learning.

In contrast to this, inductive reasoning results in the addition of semantic information. There are a great many ways in which inductive inference has been characterized but most are similar to those specified by the philosopher John Stuart Mill (1843). Basically, in this paradigm, positive instances of some phenomena that have a common trait identify that trait as indicating some larger commonality. Similarly, negative instances that differ for some trait from the positive instances are also indicative of a crucial feature. This methodology is at the center of concept acquisition programs and plays a key role in many AI systems. In general, induction is more difficult than deduction because of both the addition of new semantic information and because the inferred concept may not be the correct one. In induction, assertions do not necessarily lead to true conclusions.

Combinations of inductive and/or deductive reasoning are present in most cognitive architectures that utilize a symbolic world model and are described in the individual architecture document with more specific capabilities such as planning and learning.

2.5.2 Prediction
Our use of the term prediction refers to an architecture's ability to predict what the state of the world is or might be, what things might happen in the outside world, and what other things might happen as a consequence of the agent's actions. It should be clear that, for an architecture to be able to predict it needs to have a fairly good and consistent model of the outside world. In fact, architectures with no such model are unable to do prediction.

2.5.3 Query Answering and Providing Explanations for Decisions
Query answering is the ability to query the agent about things like past episodes ("Where were you last night?"), or the current state of the world ("Are your fingernails clean?"). If not posed in natural language, some of these queries are quite simple if the agent simply has episodic or state information immediately available. While a number of architecture discussions omitted query answering, many have general problem-solving ability that could be applied in this direction.

It is also often desirable that an agent provide explanations of its actions. For instance, supervisors may monitor a system's performance, possibly because of the sensitivity of the domain, in which case the agent must provide justifications of its choices. More commonly, a system may make mistakes which must be corrected. Because of the complexity of most architectures, such debugging is difficult, if not intractable. A trace of the agent's processing, in the form of a decision explanation, would provide valuable information to the system designers in trying to amend the situation.

2.5.4 Navigational Strategies
Agents constructed under the hypothesis of situated action often have rudimentary reactions built into the architecture. These built-in reactions give rise to the strategy that the agent will take under certain environmental conditions. Reactive agents, such as the Brooksian agents, have emergent navigational strategies. Other agents augment emergent strategies with a degree of explicit planning.

2.5.5 Natural Language Understanding
Natural language understanding and generation abilities are required to communicate with other agents, particularly with people. Natural language understanding corresponds to receiving words from outside world, and natural language generation corresponds to sending words that may be compiled internal deliberation of the agent itself, to external world.

2.5.6 Perception
Perception refers to the extraction of knowledge (usually in the form of signals) from the environment. One characteristic of perception is that it may integrate sensory information from different modalities. For example, in humans the modalities of perception correspond to the five senses: taste, touch, sight, sound, and smell.

Agents that sense the world and generate knowledge accessible to processes that reason are said to perceive the world. Perception drives a continuum of behaviors that extend from the simplicity of a thermostat which simply measures the temperature to the assumption used by some agents that objects containing all relevant information about things in the world get inserted into the agent's knowledge.

In this later case, the amount of perceptional information at any

one time may be overwhelm the agent's processing abilities. One way to circumvent this problem in real domains is to include a system for focusing attention on relevant percepts. In this case, the architecture makes a deliberate decision to concentrate on particular environmental percepts and must be forced (perhaps by a high priority stimulus) to move its attention elsewhere.

In addition to attentional mechanisms, perception may also be corrupted by faulty transducers or some other problem with accurately sensing the environment. In some cases, the architectures are then supplied with the ability to support and recover from inaccurate sensing.

2.5.7 Support for Inaccurate Sensing
Sensors provide incomplete information and the state of the agent is always behind the state of the external environment. Some agents account for this in the architecture while others make tacit or explicit assumptions (or requirements) that sensors be perfect. Several architectures support inaccuracies and delays in sensing. Others assume or require that sensors be perfect.

2.6 Real-Time Execution

While speed is an issue in all architectures to varying degrees, the ability to guarantee real-time performance places a tighter restriction on the speed requirements of the system. Real-time performance means that the agent is guaranteed to behave within certain time constraints as specified by the task and the environment. This is especially challenging in a dynamic environment because it provides an very tight time constraint on performance. Perfect rationality is perhaps impossible to guarantee when operating under a real-time constraint and thus some architectures will satisfy with bounded rationality to achieve this goal.

2.7 Focused Behavior and Processing/Selective Attention

The designers of most intelligent agents intend their agents to be operative in a complex, dynamic environments, usually the "real world" or some subset thereof. This, however, often causes significant practical problems: the real world provides a literally overwhelming amount of information to the agent; if the agent were to attempt to sense and process all this information, there would be very few computational resources remaining for other processes such as planning or learning.

One way in which this problem is overcome is by incorporating some form of focusing mechanism, whereby the agent determines what sort of information it needs to attack the current problem. It looks for and processes all relevant information it can, but (more or less) ignores other extraneous data. By focusing all of its processes only on the problem at hand, the combinatorial explosion of information from the world can be sidestepped.

2.8 Goal Reconstruction

Goal reconstruction is the ability of an agent to exploit short-cuts to return to a problem where it was last left off, even when the memory in which the problem was stored has been used for other purposes. This capability is implicit in some architectures and explicit in others. Kurt VanLehn argues that goal reconstruction is critical to mimic the human capability of quickly restarting a problem after being indefinitely interrupted. Teton employs goal reconstruction explicitly using two mechanisms in order to balance efficiency and speed with robustness.

2.9 Responding Intelligently to Interrupts and Failures

The ability to respond intelligently to interrupts is extremely important for agents that must operate in a dynamic environment. In particular, interrupt ability may be an important feature that supports reactivity but neither property implies the other.

Architectures that tend to focus their attention on a particular activity, such as planning, at a particular moment in time may have some difficulty incorporating external, high priority perceptions into its behavior patterns. An architecture could simply treat the new situation as a standard goal and handle it in the normal course of cognitive processing. But, if the agent's success or survival depends on the timely handling of such a situation, it may be more appropriate to interrupt the current behavior according to the priority of the situation. This may simply involve stopping the current cognitive process in lieu of a more important process. Of course this raises the question of "clean up", that is whether the current process should be stopped immediately or put in a state such that the agent could return to it effectively after dealing with the interrupt.

2.10 Human-like Math Capability

Humans often solve arithmetic problems the "long way". The optimal bit-based methods of the computer are not natural and, as such, not employed by humans. Several psychological experiments have been performed showing that, not only are the arithmetic operations used by humans not optimal, but the long-hand algorithms can be suboptimal and sometimes inconsistent. Some humans classify problems before approaching them (even the classifications can be inconsistent) and use a personal method that varies consistently with the class of problem.

Kurt Van Lehn argues that a non-Last-In, First-Out (LIFO) goal reconstruction technique can reproduce this behavior. An essential component to the reproduction of this behavior is that goals cannot be managed by a LIFO stack. VanLehn's Teton architecture was designed specifically to model these types of behaviors. Additionally, the Soar architecture has also been applied to the cognitively-plausible solution of math problems.

In the following table-2 the rows represent the capabilities and the column corresponds to agents with specific architecture . In the cells ‘Y’ indicates that agent corresponds to the column possess the capability represented by the row .

3. Conclusion

The problem of AI is to describe and build agents that receive percepts from the environment and perform actions, and each such agent is implemented by a function that maps percepts to actions. It explains the role of learning as extending the reach of the designer into unknown environments, and shows how it constrains agent design, favoring explicit knowledge representation and reasoning.

Table 2

Capability	Subsumption	ATLANTIS	THEO	PRODIGY	ICARUS	ADAPTIVE INTELLIGENT SYSTEMS	META REASONING ARCHITECTURE	HOMER	SOAR	TETON	RALPH-MEA	ENTROPY REDUCTION ENGINE
Single Learning Method					Y			Y	Y			
Multi-Method Learning			Y	Y			Y		Y			Y
Caching			Y						Y			
Learning by Instruction				Y					Y			
Learning from Experimentation				Y			Y					
Learning by Analogy				Y					Y			
Inductive Learning and Concept Acquisition			Y		Y				Y			Y
Abstarction				Y					Y			
Explanation-based learning			Y	Y			Y		Y			Y
Transfer of Learning									Y			
Planning		Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Problem Solving				Y		Y		Y	Y	Y		Y
Replanning		Y			Y	Y	Y	Y	Y		Y	
Support of Multiple , Simultaneous Goals		Y	Y			Y		Y		Y	Y	
Self Reflection			Y	Y	Y	Y	Y	Y				
Meta-Reasoning			Y	Y		Y	Y		Y		Y	
Expert System Capability						Y			Y			
Inductive and deductive Reasoning												
Prediction			Y	Y	Y	Y					Y	Y
Query Answering and providing Explanations for Decisions			Y			Y		Y				
Navigational Strategies	Y	Y	Y		Y	Y		Y				
Natural Language Understanding								Y	Y			
Perception	Y	Y	Y			Y		Y	Y			Y
Support for Inaccurate sensing		Y				Y					Y	
Robotic Tasks	Y	Y	Y			Y		Y	Y		Y	
Real-time Execution												
Focussed Behavior and Processing/Selective Attentions	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	
Goal Recontruction										Y		
Responding Intelligently to Interrupts and Failures		Y		Y	Y	Y	Y	Y	Y		Y	
Human-like Math Capability									Y			

It analyzes basic techniques for addressing complexity. The idea is to Integrate state-of-the art AI techniques into intelligent agent designs, using examples from twelve agents to full knowledge-based agents with natural language capabilities and so on. This leads to the study of Multi Agent systems and its applications. In depth analysis of various Agent architectures and their capabilities is to build a Multi Agent System that will be suitable for our future work on Supply Chain Management.

4. References

1. *Anderson, J. (1991). Cognitive Architectures in a rational analysis. In K. VanLehn (ed.), Architectures for Intelligence, pp. 1-24, Lawrence Erlbaum Associates, Hillsdale, N.J.*
2. *Fikes, R., Nilsson, N. (1971). STRIPS: A new approach to the application of theorem proving to problem solving. Artificial Intelligence, 2, pp. 189-203.*
3. *Newell, A. (1982). The knowledge level. Artificial Intelligence. 18(1), 87-127.*
4. *Rich, E., Knight, K. (1991). Artificial Intelligence, 2nd Edition. McGraw-Hill, New York, New York.*
5. *Schoppers, M. (1987). Universal Plans for Reactive Robots in Unpredictable Environments. Proc. of the IEEE. Vol. 77, No. 1. (January).pp. 81-98.*
6. *Simon, H. (1957). Models of Man. Wiley, New York.*